

JELGAVAS TEHNIKUMS

ROBERTS JANSONS

**Mājas lapas pilot versija uzņēmuma iekšējai
stāvvietu reģistrācijas sistēmai “*ParkingPRO*”**

**Kvalifikācijas darbs kvalifikācijas ieguvei programmēšanas
tehnikā**

Darba izpildītājs: 410. gr. izgl. R. Jansons
paraksts, datums

Jelgava 2025

ANOTĀCIJA

**Jansons R. Mājas lapa uzņēmuma iekšējai stāvvietu sistēmai “*ParkingPRO*”:
kvalifikācijas darbs. Jelgava: JT, 2025. 136. lpp., 36. att., 10. tab. un 37. pielikumi.**

Darbs ir izstrādāts, lai nodrošinātu klientu transportlīdzekļu reģistrēšanu priekš uzņēmuma iekšējas lietošanas, lai varētu reģistrēt mašīnas, kas tiek atstātas ilgtermiņa vai īslaicīgas mašīnas atstāšanu drošībā, kamēr pats cilvēks ir devies ārpus Latvijas vai cita iemesla dēļ atstāj mašīnu.

Kvalifikācijas darba centrālais uzdevums ir izstrādāt iekšēju sistēmu, kas nodrošina vienkāršu klientu transportlīdzekļu reģistrāciju un pārvaldību, iekļaujot tādos parametrus kā klienta dati (vārds, uzvārds, kontaktinformācija), transportlīdzekļa informācija (reģistrācijas numurs, modelis, stāvvietas numurs) un novietošanas laiks.

Darba mērķis ir izveidot ērtu un drošu risinājumu, kas optimizē klientu apkalpošanu un stāvvietas pārvaldību uzņēmumam.

Šāda sistēma uzlabos darba efektivitāti, samazinot manuālās datu apstrādes nepieciešamību, kā arī veicinās organizētu un caurskatāmu klientu transportlīdzekļu novietošanas pārvaldību.

Darbs tiek izstrādāts individuāli.

ANNOTATION

Jansons R. Website for the company's internal parking system "ParkingPRO": qualification thesis. Jelgava: JT, 2025. 136. p., 36. fig., 10. tab. and 37. appendices.

The work has been developed to provide registration of customer vehicles for internal use, to be able to register cars that are left in the safety of a long-term or short-term vehicle while the person themselves is out of the country or leaves the car for another reason.

The central task of the qualification work is to develop an internal system that allows easy registration and management of customer vehicles, including parameters such as customer data (name, contact details), vehicle information (registration number, model, parking number) and parking time.

The aim of the work is to develop a convenient and secure solution that optimises customer service and parking management for the company.

Such a system will improve work efficiency by reducing the need for manual data processing, and will facilitate an organised and transparent management of customer parking.

The work is developed individually.

SATURS

IEVADS	9
AKRONĪMI UN SAĪSINĀJUMI.....	11
1. UZDEVUMA NOSTĀDNE	13
1.1. IEVADS	13
1.2. LĪDZĪGU PROGRAMMATŪRU ANALĪZE.....	13
1.2.1. Mobilā lietotne “Mobilly”	13
1.2.2. Mobilā lietotne “EuroPark”	16
1.2.3. Mājaslapa “ParkSmart”	16
2. PRASĪBU SPECIFIKĀCIJA	18
2.1. IEVADS	18
2.1.1. Dokumenta nolūks.....	18
2.1.2. Darbības sfēra	18
2.1.3. Definīcijas	19
2.1.4. Dokumenta pārskats.....	19
2.2. VISPĀRĪGS APRAKSTS	20
2.2.1. Produkta perspektīva	20
2.2.2. Produkta funkcijas	20
2.2.3. Lietotāja raksturojums.....	21
2.2.4. Vispārējie ierobežojumi	21
2.2.5. Projekta plānošana un plānu pārskatīšana	22
2.2.6. Programmatūras gaitas apraksts.....	23
2.2.7. Līmeņu izkārtojuma apraksts.....	24
2.3. FUNKCIONĀLĀS PRASĪBAS	24
2.3.1. Lietotāja pieslēgšanās.....	24
2.3.2. Transportlīdzekļa reģistrācija	25
2.3.3. Transportlīdzekļa meklēšana	27
2.3.4. Transportlīdzekļa filtrēšana.....	27
2.3.5. Transportlīdzekļa datu rediģēšana	28
2.3.6. Transportlīdzekļa izņemšana/dzēšana no stāvvietas	29
2.3.7. Konta pievienošana.....	30
2.4. NEFUNKCIONĀLĀS PRASĪBAS	31
2.4.1. Programmas logi	31
2.4.2. Standarta pogas	34
2.4.3. Ievadlauki.....	35
2.4.4. Drošība un datu aizsardzība.....	35
2.4.5. Pieejamība un veiktspēja	35
2.4.6. Valodas prasības.....	35
3. UZDEVUMA RISINĀŠANAS LĪDZEKĻU PAMATOJUMS	36
3.1. KODA REDAKTORS	36
3.1.1. Visual Studio Code 1.95.2 (Universal)	36
3.2. DATUBĀZES PROJEKTĒŠANAS RĪKS.....	36
3.2.1. TablePlus 5.9.2	36
3.3. LOKĀLĀ SERVERA IZVEIDES RĪKS	37
3.3.1. XAMPP 8.2.12	37
3.4. BLOKSHĒMU IZVEIDES RĪKS.....	37
3.4.1. Lucidchart (premium)	37
3.5. WIREFRAME IZVEIDES RĪKS	38
3.5.1. Canva (free)	38
3.6. SKRIPTU, STILA UN KODA PROGRAMMĒŠANAS VALODAS	38
3.6.1. HTML (HyperText Markup Language).....	38
3.6.2. JavaScript	38
3.6.3. CSS (Cascading Style Sheets)	39
3.6.4. PHP (Hypertext Preprocessor).....	39

4.	SISTĒMAS STRUKTŪRAS MODELIS	40
4.1.	SISTĒMAS STRUKTŪRA	40
4.2.	ALGORITMA SHĒMA AR IZVĒLĒTĀS RISINĀŠANAS METODES APRAKSTU	41
4.3.	KLAŠU DIAGRAMMAS	41
4.4.	ER MODELIS	42
4.4.1.	“users” tabula	42
4.4.2.	“cars” tabula	43
5.	FUNKCIONĀLAIS UN DINAMISKAIS SISTĒMAS MODELIS	44
5.1.	DATU PLŪSMAS DIAGRAMMAS	44
5.2.	STĀVOKĻA DIAGRAMMA	45
5.3.	SCENĀRIJS – TRANSPORTLĪDZEKĻA REĢISTRĀCIJA	45
5.4.	SCENĀRIJS – TRANSPORTLĪDZEKĻA MEKLĒŠANA SISTĒMĀ	47
5.5.	SCENĀRIJS – LIETOTĀJA REĢISTRĒŠANA SISTĒMĀ	48
6.	DATU STRUKTŪRU APRAKSTS	51
6.1.	DATU BĀZES APRAKSTS	51
6.2.	OBJEKTU KLASES	51
6.3.	DATU PLŪSMAS PROCESS SISTĒMĀ	52
7.	TESTĒŠANA	53
7.1.	TESTĒŠANAS KRITĒRIJI	53
7.2.	TESTĒŠANAS RĪKI UN VEIDI	53
7.3.	TESTPIEMĒRI	53
7.3.1.	Konta reģistrācijas funkcijas testēšana	53
7.3.2.	Pieteikšanās funkcijas testēšana	54
7.3.3.	Transportlīdzekļa pievienošanas funkcijas testēšana	55
7.3.4.	Transportlīdzekļa meklēšanas funkcijas testēšana	55
8.	LIETOTĀJA CEĻVEDIS	57
8.1.	SISTĒMAS PIEKĻUVE UN PIESLĒGŠANĀS	57
8.1.1.	Sistēmas atvēršana	57
8.1.2.	Lietotāja pieslēgšanās	57
8.2.	TRANSPORTLĪDZEKĻA REĢISTRĒŠANA	58
8.2.1.	Jauna transportlīdzekļa reģistrēšana	58
8.2.2.	Reģistrācijas datu ievade	59
8.2.3.	Pārbaudes un kļūdas ziņojumi	60
8.2.4.	Reģistrēšanas pabeigšana	60
8.3.	TRANSPORTLĪDZEKĻA DATU REDIGĒŠANA UN DZĒŠANA	61
8.3.1.	Kā rediģēt transportlīdzekļa datus	61
8.3.2.	Kā dzēst transportlīdzekli no sistēmas	61
8.4.	TRANSPORTLĪDZEKĻU MEKLĒŠANA	62
8.5.	FILIĀLES PĀRVALDĪBA UN IESTATĪJUMI	62
8.5.1.	Kā pievienot jaunu filiāli	62
8.5.2.	Kā rediģēt filiāles datus	63
8.6.	DROŠĪBA UN DATU AIZSARDZĪBA	64
8.6.1.	Sistēmas drošība un datu aizsardzība	64
8.7.	SISTĒMAS INTERFEISS UN DARBĪBA	64
8.7.1.	Lietotāja interfeiss	64
8.7.2.	Navigācijas ērtības	64
8.7.3.	Atsaucīga dizaina iespējas	64
	SECINĀJUMI UN PRIEKŠLIKUMI	65
	IZMANTOTO AVOTU SARAKSTS	66
	PIELIKUMI	68
1.	PIELIKUMS INDEX.PHP	69
2.	PIELIKUMS INDEX.JS	70
3.	PIELIKUMS LOGIN.CSS	71
4.	PIELIKUMS LOGIN.PHP	73

5.	PIELIKUMS ADMIN.CSS	74
6.	PIELIKUMS ADMIN.HTML	76
7.	PIELIKUMS REGISTERADMIN.PHP	77
8.	PIELIKUMS SAVECAR.PHP	80
9.	PIELIKUMS REG.CSS	82
10.	PIELIKUMS VIEWADMIN.PHP	84
11.	PIELIKUMS VIEWCARS.CSS	87
12.	PIELIKUMS VIEWADMIN.JS	93
13.	PIELIKUMS DELETE.PHP	95
14.	PIELIKUMS UPDATECAR.PHP	96
15.	PIELIKUMS EDITCAR.PHP	98
16.	PIELIKUMS SETTINGS.HTML	101
17.	PIELIKUMS REGISTERBRANCH.PHP	102
18.	PIELIKUMS ADDBRANCH.HTML	105
19.	PIELIKUMS ADDBRANCH.JS	106
20.	PIELIKUMS ADDBRANCH.CSS	107
21.	PIELIKUMS SAVEBRANCH.PHP	109
22.	PIELIKUMS LOOKBRANCH.PHP	110
23.	PIELIKUMS LOOKBRANCH.JS	112
24.	PIELIKUMS DELETEBRANCH.PHP	113
25.	PIELIKUMS EDITBRANCH.PHP	114
26.	PIELIKUMS EDITBRANCHPROCESS.PHP	115
27.	PIELIKUMS BRANCH.PHP	116
28.	PIELIKUMS STYLES.CSS	117
29.	PIELIKUMS SAVECARBRANCH.PHP	121
30.	PIELIKUMS ADDBRANCH.PHP	123
31.	PIELIKUMS VIEWBRANCH.PHP	124
32.	PIELIKUMS VIEWBRANCH.JS	128
33.	PIELIKUMS DELETE.PHP	129
34.	PIELIKUMS EDITCARBR.PHP	130
35.	PIELIKUMS UPDATECARBR.PHP	133
36.	PIELIKUMS LOGOUT.PHP	135
37.	PIELIKUMS DB.PHP	136

ATTĒLU SARAKSTS

1. ATT. EKRĀNKOPIJA NO <i>MOBILLY</i> TUVĀKO STĀVVIETU MEKLĒTĀJU	14
2. ATT. EKRĀNKOPIJA NO <i>MOBILLY</i> REZERVĀCIJAS LOGA.....	14
3. ATT. EKRĀNKOPIJA NO <i>MOBILLY</i> PAPILDUS FUNKCIJĀM.....	15
4. ATT. <i>MOBILLY WEB</i> LAPAS IZSKATS[7]	15
5. ATT. EKRĀNKOPIJA NO “ <i>EUROPARK</i> ” APLIKĀCIJAS	16
6. ATT. “ <i>PARKSMART</i> ” <i>WEB</i> LAPAS IZSKATS [8]	17
7. ATT. LIETOTĀJA PIESLĒGŠANĀS FUNKCIJAS BLOKSHĒMA	25
8. ATT. TRANSPORTLĪDZEKĻA REĢISTRĒŠANAS FUNKCIJAS BLOKSHĒMA	26
9. ATT. TRANSPORTLĪDZEKĻA MEKLĒŠANAS FUNKCIJAS BLOKSHĒMA	27
10. ATT. TRANSPORTLĪDZEKĻA REDIGĒŠANAS FUNKCIJAS BLOKSHĒMA	29
11. ATT. TRANSPORTLĪDZEKĻA DZĒŠANAS FUNKCIJAS BLOKSHĒMA	30
12. ATT. JAUNA KONTA REĢISTRĒŠANAS FUNKCIJAS BLOKSHĒMA	31
13. ATT. PIETEIKŠANĀS LOGA MAKETS	32
14. ATT. PROGRAMMAS SĀKUMEKRĀNA MAKETS.....	33
15. ATT. DATU IEVADES LOGA MAKETS	33
16. ATT. DATU APSKATES LOGA MAKETS	34
17. ATT. POGAS MAKETS	34
18. ATT. IEVADLAUKA MAKETS.....	35
19. ATT. SISTĒMAS STRUKTŪRAS PĀRVALDĪBU [MĀKSLĪGĀ INTELEKTA ĢENERĒTS ATTĒLS] ..	40
20. ATT. ENTĪTIJAS ATTIECĪBU DIAGRAMMA KURA IZVEIDOTA IZMANTOJOT <i>WEB</i> LAPU[3]	42
21. ATT. LIETOTĀJA AUTENTIFIKĀCIJAS SECĪBA [MĀKSLĪGĀ INTELEKTA ĢENERĒTS ATTĒLS]..	44
22. ATT. TRANSPORTLĪDZEKĻA REĢISTRĀCIJAS PROCESS [MĀKSLĪGĀ INTELEKTA ĢENERĒTS ATTĒLS].....	44
23. ATT. TRANSPORTLĪDZEKĻA MEKLĒŠANAS UN PĀRVALDĪBAS SECĪBA [MĀKSLĪGĀ INTELEKTA ĢENERĒTS ATTĒLS]	45
24. ATT. PĀRLŪKPROGRAMMAS MEKLĒŠANAS JOSLA	57
25. ATT. PIESLĒGŠANĀS LOGS (SKAT. 1. PIELIKUMĀ)	57
26. ATT. ADMINISTRATORA SĀKUMA LOGS (SKAT. 6. PIELIKUMĀ)	58
27. ATT. FILIĀLES SĀKUMA LOGS (SKAT. 27. PIELIKUMĀ).....	58
28. ATT. TRANSPORTLĪDZEKĻA REĢISTRĀCIJAS LOGS (SKAT. 7., 8., 29. UN 30. PIELIKUMĀ)	59
29. ATT. KĻŪDAS PAZIŅOJUMS, JA KĀDS NO LAUKIEM NAV AIZPILDĪTS	60
30. ATT. KĻŪDAS PAZIŅOJUMS, JA KĀDS NO LAIKIEM IR IEVADĪTS NEPAREIZI.....	60
31. ATT. PAZIŅOPJUM, KA DATI IR SAGLABĀTI DATUBĀZĒ	60
32. ATT. TRANSPORTLĪDZEKĻA APSKATĪŠANAS/MEKLĒŠANAS/REDIGĒŠANAS/DZĒŠANAS LOGS (SKAT. 10., 12., 31. UN 32. PIELIKUMĀ)	61
33. ATT. DZĒŠANAS PAZIŅOJUMS	62
34. ATT. IESTATĪJUMU LOGS (SKAT. 16. PIELIKUMĀ)	62
35. ATT. FILIĀLES KONTA PIEVIENOŠANA (SKAT. 18. PIELIKUMĀ)	63
36. ATT. PROFILU APSKATES/MEKLĒŠANAS/REDIGĒŠANAS/DZĒŠANAS LOGS (SKAT. 22. PIELIKUMĀ).....	63

TABULU SARAKSTS

1. TABULA "USERS" SATUR INFORMĀCIJU PAR SISTĒMAS LIETOTĀJIEM (ADMINISTRATORI UN DATU IEVADES SPECIĀLISTI).....	42
2. TABULA "CARS" SATUR INFORMĀCIJU PAR KLIENTU TRANSPORTLĪDZEKĻIEM UN NOVIETOŠANAS DATUMIEM.....	43
3. TABULA "USER" KLASES PĀRVALDA LIETOTĀJA DATUS UN FUNKCIONALITĀTI	51
4. TABULA "USER" KLASES PĀRVALDA LIETOTĀJA DATUS UN FUNKCIONALITĀTI	51
5. TABULA "CAR" KLASES PĀRVALDA TRANSPORTLĪDZEKĻU REĢISTRĀCIJU UN MEKLĒŠANU	51
6. TABULA "CAR" KLASES PĀRVALDA TRANSPORTLĪDZEKĻU REĢISTRĀCIJU UN MEKLĒŠANU	52
7. TABULA REĢISTRĀCIJAS FUNKCIJAS TESTPIEMĒRI.....	54
8. TABULA PIETEIKŠANĀS FUNKCIJAS TESTPIEMĒRI.....	54
9. TABULA TRANSPORTLĪDZEKĻA PIEVIENOŠANAS TESTPIEMĒRI	55
10. TABULA TRANSPORTLĪDZEKĻA MEKLĒŠANAS TESTPIEMĒRI.....	56

IEVADS

Automašīnas bieži vien ir nozīmīgs īpašums, kuru cilvēki vēlas uzticēt drošām rokām, ja uz ilgāku laiku nepieciešams doties ārpus Latvijas vai arī, ja rodas citi iemesli, kad transportlīdzekļa glabāšana ir jāuztic uzņēmumam. Šādās situācijās ir svarīgi, lai uzņēmuma darbinieki varētu vienkārši un efektīvi pārvaldīt klientu un transportlīdzekļu informāciju, nodrošinot pārskatāmu un precīzu transportlīdzekļa novietošanas un uzraudzības procesu. Šis projekts piedāvā efektīvu un lietotājam draudzīgu sistēmu, kas ļaus uzņēmumam ātri un ērti reģistrēt, uzglabāt un pārvaldīt informāciju par klientiem un viņu transportlīdzekļiem.

Sistēma nodrošinās darbiniekiem pārredzamību par klientu automašīnām – tās novietojumu, novietošanas ilgumu, kā arī klientu kontaktinformāciju. Tā kalpos kā uzticama platforma, kas ļaus izvairīties no neskaidrībām par automašīnu atrašanās vietu un vienkāršos datu piekļuvi. Automātiskā datu apstrāde samazinās kļūdu iespējamību un atbrīvos uzņēmuma resursus, ļaujot darbiniekiem vairāk laika veltīt citiem uzdevumiem.

Iemesls, kāpēc šāda sistēma ir svarīga, ir vienkāršs - tā uzlabo uzņēmuma darba procesu efektivitāti un precizitāti. Izmantojot šo sistēmu, klienti var justies pārliecināti par savu automašīnu drošību, ziņot, ka viņu transportlīdzekļi tiek uzraudzīti pārredzamā un drošā procesā. Sistēmas strukturēta datu uzskaitē padara autostāvvietas pārvaldību ātrāku un caurspīdīgāku, ļaujot darbiniekiem viegli piekļūt informācijai un ātri reaģēt uz klientu jautājumiem vai lūgumiem.

Projekta mērķis

Izstrādāt sistēmu, kas ļauj uzņēmumam pārvaldīt datus par klientiem un viņu transportlīdzekļiem, nodrošinot efektīvu, drošu un ērtu automašīnu uzskaiti uzņēmuma stāvvietā.

Darba uzdevumi

1. Esošo sistēmu izpēte – analizēt līdzīgu transportlīdzekļu uzskaites un novietošanas sistēmu funkcionalitāti un risinājumus, identificējot priekšrocības un trūkumus.
2. Sistēmas arhitektūras izstrāde – definēt sistēmas komponentes, to savstarpējo saistību un datu plūsmu.
3. Datubāzes izveide – veidot datubāzi klientu un transportlīdzekļu datu glabāšanai, nodrošinot ātru piekļuvi un efektīvu datu pārvaldību.

4. Lietotāja reģistrācijas un pieteikšanās izstrāde – izveidot ērtu veidu, kā darbinieki var piekļūt sistēmai, autentificējoties drošā veidā.

5. Lietotāja profila un transportlīdzekļa uzskaites moduļa izveide – izstrādāt ērtu interfeisu, kur darbinieki var ievadīt, rediģēt un apskatīt klientu un automašīnu informāciju, piemēram, reģistrācijas numurus, modeļus un novietošanas laikus.

6. Stāvvietas pārvaldības funkcionalitātes ieviešana – izveidot sistēmu, kas ļauj darbiniekiem ātri pārskatīt, kurā stāvvietā konkrētais transportlīdzeklis ir novietots.

7. Atskaites un meklēšanas funkcijas izstrāde – nodrošināt, ka sistēma spēj ģenerēt pārskatus un veikt meklēšanu pēc automašīnas reģistrācijas numura.

8. Sistēmas testēšana un uzlabošana – veikt pilnīgu sistēmas testēšanu, novērst kļūdas un pielāgot funkcionalitāti atbilstoši lietotāju atsauksmēm.

9. Tehniskās dokumentācijas un lietotāja rokasgrāmatas izstrāde – sagatavot detalizētu dokumentāciju sistēmas uzturēšanai un darbinieku apmācībai.

AKRONĪMI UN SAĪSINĀJUMI

<i>Android</i>	Operētājsistēma mobilajām ierīcēm, kuru izstrādājusi <i>Google</i>
<i>Apache</i>	Atvērtā koda tīmekļa servera programmatūra
<i>Apple Pay</i>	<i>Apple</i> izstrādāts mobilo maksājumu un digitālā maka pakalpojums
Att.	Saīsinājums vārdam attēls
<i>Authentication</i>	Lietotāja identitātes pārbaude
<i>Canva</i>	Grafiskā dizaina platforma, kas ļauj izveidot vizuālus prototipus un maketus
<i>CSS (Cascading Style Sheets)</i>	Stilizēšanas valoda tīmekļa lapu dizainam un izkārtojumam
<i>CSV</i>	Datu formāts, kur vērtības atdalītas ar komatu
Dec.	Saīsinājums vārdam decembris
<i>ERD</i>	Entītiņu un attiecību diagramma
<i>EuroPark</i>	Uzņēmums, kas piedāvā autostāvvietu pakalpojumus
<i>GDPR</i>	Vispārīgā datu aizsardzības regula (<i>General Data Protection Regulation</i>)
<i>Google Chrome</i>	<i>Chrome</i> ir <i>Google</i> izstrādāta starp platformu tīmekļa pārlūkprogramma
<i>GPS (Global Positioning System)</i>	Globālās pozicionēšanas sistēma
<i>HeidiSQL</i>	Rīks datubāzes pārvaldībai un struktūras izveidei
<i>Hex.</i>	Heksadecimālais skaitļu formāts
<i>HTML (HyperText Markup Language)</i>	Marķēšanas valoda tīmekļa lapu struktūras izveidei
<i>ID</i>	Identifikators
<i>iOS</i>	<i>Apple</i> izstrādāta operētājsistēma mobilajām ierīcēm
Izgl.	Saīsinājums vārdam izglītojama
Jan.	Saīsinājums vārdam janvāris
Jansons R.	Darba autors Roberts Jansons
<i>JavaScript</i>	Programmēšanas valoda tīmekļa lapu interaktivitātes pievienošanai
<i>JT</i>	Saīsinājums vārdam Jelgavas tehnikums
<i>Lucidchart</i>	Tiešsaistes rīks, kas paredzēts diagrammu un bloکشēmu veidošanai
<i>macOS</i>	<i>Apple</i> izstrādāta operētājsistēma datoriem
<i>MariaDB</i>	Atvērtā koda datubāzes pārvaldības sistēma
<i>Microsoft</i>	Starptautisks tehnoloģiju uzņēmums, kas izstrādā programmatūru un ierīces
<i>Microsoft Edge</i>	<i>Microsoft</i> izstrādāta tīmekļa pārlūkprogramma
<i>Mobily</i>	Mobilās sakaru un digitālo pakalpojumu sniedzējs
<i>Mozilla Firefox</i>	<i>Firefox</i> ir <i>Mozilla Foundation</i> izstrādāta bezmaksas un atvērtā pirmkoda tīmekļa pārlūkprogramma
<i>MySQL</i>	Atvērtā koda datubāzes pārvaldības sistēma
Nov.	Saīsinājums vārdam novembris
Okt.	Saīsinājums vārdam oktobris
<i>ParkingLot</i>	Autostāvvietā
<i>ParkingPro</i>	Transportlīdzekļu pārvaldības sistēmas projekts

<i>ParkSmart</i>	Autostāvvietu pārvaldības pakalpojums pie Rīgas lidostas
<i>Password</i>	Parole
<i>PHP (Hypertext Preprocessor)</i>	Servera puses skriptu valoda dinamisku tīmekļa vietņu izstrādei
<i>Px</i>	Pikseli, attēla vai ekrāna izšķirtspējas vienība
<i>Responsive design</i>	Reaģējošs tīmekļa lapas dizains
<i>Safari</i>	<i>Apple</i> izstrādāta tīmekļa pārlūkprogramma
<i>Sep.</i>	Saīsinājums vārdam septembris
<i>Skat.</i>	Saīsinājums vārdam "skatīties"
<i>TablePlus</i>	Atvērtā koda datubāzes pārvaldības sistēma
<i>URL</i>	Vienotais resursu vietrādis (<i>Uniform Resource Locator</i>)
<i>User</i>	Sistēmas lietotājs
<i>Username</i>	Lietotājvārds
<i>Vehicle</i>	Transportlīdzeklis
<i>Visual Studio Code</i>	Viegls koda redaktors ar plašu funkcionalitāti
<i>WEB</i>	Tīmekļa lapa
<i>Windows</i>	Operētājsistēma, ko izstrādājusi <i>Microsoft</i>
<i>Wireframe</i>	Tīmekļa vai lietotnes vizuālais strukturālais izkārtojuma makets
<i>XAMPP</i>	Komplektēts rīks lokālās tīmekļa izstrādes vides izveidei

1. UZDEVUMA NOSTĀDNE

1.1. Ievads

Kvalifikācijas darbs ir veltīts iekšējās informācijas sistēmas izstrādei, kas palīdz uzņēmumam reģistrēt un pārvaldīt klientu transportlīdzekļus, kas novietoti uzņēmuma drošajā stāvvietā. Projekta mērķis ir nodrošināt efektīvu un pārskatāmu transportlīdzekļu uzskaiti un pārvaldību, uzlabojot klientu apkalpošanu un optimizējot stāvvietas pārvaldību.

Sistēmai jānodrošina šādas funkcionālās prasības:

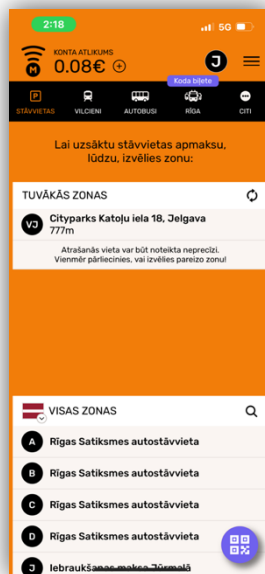
- Klientu un transportlīdzekļu reģistrācija: iespēja pievienot jaunu klientu un viņa transportlīdzekli ar tādiem datiem kā klienta vārds, uzvārds, kontaktinformācija, automašīnas reģistrācijas numurs, modelis un novietošanas laiks.
- Transportlīdzekļu novietošanas pārvaldība: iespēja atzīmēt, kurā stāvvietas sektorā transportlīdzeklis ir novietots, lai viegli atrastu to pēc nepieciešamības.
- Lietotāja autentifikācija: sistēmai jānodrošina droša darbinieku pieteikšanās, lai aizsargātu klientu un transportlīdzekļu datus.
- Pārskatāma informācijas piekļuve: datu struktūra un meklēšanas funkcija, kas ļauj ātri atrast klienta vai automašīnas datus pēc automašīnas reģistrācijas numura.
- Datu rediģēšanas un dzēšanas iespējas: nodrošināt ērtu informācijas atjaunošanas iespēju, piemēram, ja klienta kontaktinformācija mainās vai automašīna tiek izņemta no stāvvietas.
- Tehniskā dokumentācija un lietotāja rokasgrāmata: detalizēts ceļvedis sistēmas lietošanai.

Dokuments sastāv no astoņām galvenajām daļām, kas detalizēti apraksta projektu un tā izstrādes posmus, ietverot ievadu, projekta plānošanu, prasību specifikāciju, sistēmas arhitektūras un datubāzes dizainu, izmantotās tehnoloģijas, produkta izstrādi un testēšanu, kā arī lietotāja rokasgrāmatu.

1.2. Līdzīgu programmatūru analīze

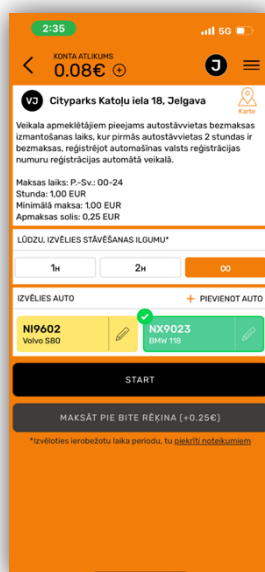
1.2.1. Mobilā lietotne “*Mobilly*”

“*Mobily*” ir mobilā lietotne, kas nodrošina stāvvietu rezervācijas iespējas visā Latvijā. Lietotāji var meklēt tuvākās stāvvietas (skat *1.att.*).



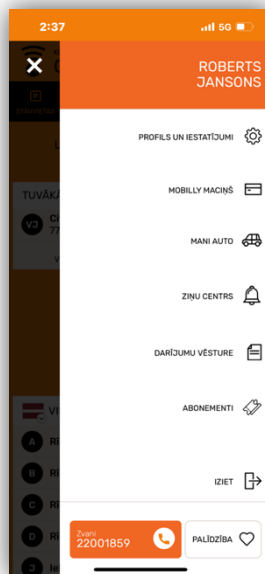
1. att. Ekrānkopija no *Mobilly* tuvāko stāvvietu meklētāju

Salīdzināt cenas un veikt rezervācijas tiešsaistē (skat. 2.att.).



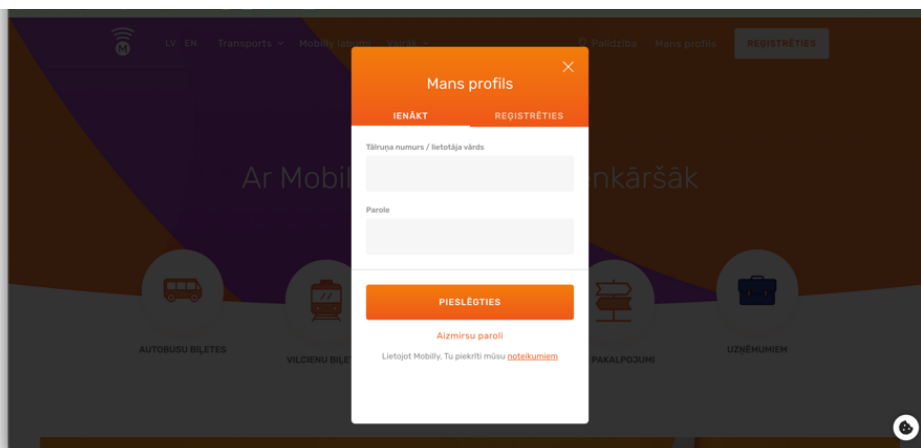
2. att. Ekrānkopija no *Mobilly* rezervācijas loga

Mobily lietotājiem ir pieejama arī informācija par stāvvietu pieejamību reāllaikā, kā arī funkcija, kas ļauj dalīties ar citiem lietotājiem par stāvvietām un dalīties atsauksmēs. Reģistrācija ir nepieciešama, lai piekļūtu visām funkcijām, tostarp personalizētajiem ieteikumiem un vēstures pārskatiem (skat. 3.att.).



3. att. Ekrānkopija no *Mobilly* papildus funkcijām

Lietotne ir pieejama gan *Android*, gan *iOS*, gan *WEB* platformās (skat. 4.att.), un tā ir bezmaksas.



4. att. *Mobilly* *WEB* lapas izskats[7]

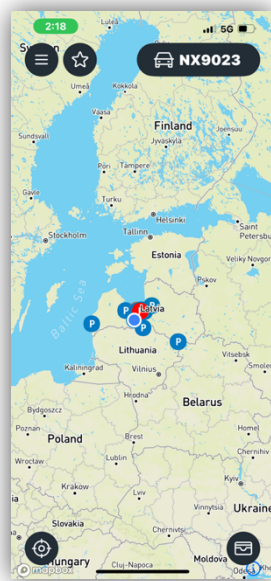
Galvenās priekšrocības:

- Lietotāji var ātri atrast stāvvietas, pamatojoties uz atrašanās vietu.
- Reāllaika informācija par stāvvietu pieejamību un cenām.
- Sociālā integrācija, kur lietotāji var dalīties ar savām pieredzēm un atsauksmēm.

1.2.2. Mobilā lietotne “EuroPark”

EuroPark[4] ir viena no vadošajām stāvvietu pārvaldības platformām, kas piedāvā lietotājiem iespēju atrast un apmaksāt stāvvietas Latvijā, Lietuvā un Igaunijā. Jaunajā *EuroPark* lietotnē ir integrēts automātiskais numuru atpazīšanas sistēma, kas ļauj lietotājiem ērti veikt maksājumus, pat ja automātiskās maksājumu iekārtas nav pieejamas.

Lietotne izmanto *GPS*, lai parādītu tuvākās stāvvietas (skat. 5.att.) un ļautu lietotājiem viegli izvēlēties un apmaksāt stāvvietas.



5. att. Ekrānkopija no “EuroPark” aplikācijas

Papildus tam, *EuroPark* piedāvā reģistrāciju vairākiem automobiļiem, kā arī *Apple Pay* iespēju.

Galvenās priekšrocības:

- Plaša stāvvietu izvēle visā Eiropā.
- Iespēja saņemt atlaides un piedāvājumus.
- Droša un kvalitatīva stāvvietu pārvaldība.

1.2.3. Mājaslapa “ParkSmart”

ParkSmart ir autostāvvietu kompānija, kas paredzētas cilvēkiem, kuri dodas ārpus Latvijas. Autostāvvietā atrodas netālu no lidostas, nodrošinot ērtu un drošu automašīnu novietošanu ceļotājiem, kas ceļotājiem atvieglo tikšanu no autostāvvietas uz lidostu, viņiem ir

ar papildus pakalpojums, ka palīdz saviem klientiem nokļūt no stāvvietas līdz lidostai un atgriežoties Latvijā no lidostas uz stāvvietu.

ParkSmart mājaslapā ir ļoti viegli saprotama visu gadu gājuma cilvēkiem, mājaslapā klients var veikt rezervāciju un veikt apmaksu (skat. 6. att.).

Izvēloties aptuvenos datumus un laikus ir iespējams arī veikt aprēķinu, kad mājaslapa parāda aptuveno summu, ar kādu būtu jāērēķinās atstājot konkrēto periodu mašīnu (skat 6. att.).



6. att. “ParkSmart” WEB lapas izskats [8]

ParkSmart ir īpašs ar savu moderno funkcionalitāti – cilvēks veic rezervāciju *WEB* lapā, un ierodas *ParkSmart* stāvvietā konkrētajā rezervēšanas laikā, braucot iekšā būs kameras ar numurzīmju atpazīšanu un, ja ir veikta rezervācija vārti atvērsies un klients var atstāt mašīnu, jebkurā brīvajā vietā.[9]

Galvenās priekšrocības:

- Stāvvietas netālu no lidostas.
- Ērta mājaslapa visu gadu gājuma cilvēkiem.
- Droša un kvalitatīva stāvvietu pārvaldība.

2. PRASĪBU SPECIFIKĀCIJA

2.1. Ievads

Prasību specifikācija ir galvenais dokuments, kas detalizēti apraksta uzņēmuma iekšējās sistēmas prasības klientu transportlīdzekļu reģistrācijai un pārvaldībai. Šis dokuments nosaka sistēmas galvenās funkcijas, lietotāja interfeisa dizainu un tehniskos parametrus, lai nodrošinātu ērtu un drošu transportlīdzekļu uzskaiti un pārvaldību. Specifikācija kalpo par pamatu sistēmas izstrādei, veicinot tās efektīvu un kvalitatīvu ieviešanu.

2.1.1. Dokumenta nolūks

Programmatūras prasību specifikācijas nolūks ir sniegt detalizētu informāciju par uzņēmuma iekšējās informācijas sistēmas plānošanu un izstrādi transportlīdzekļu reģistrācijai un pārvaldībai. Prasību specifikācija ir paredzēta, lai:

1. Nodrošinātu izstrādes komandai un citiem ieinteresētajiem pilnu izpratni par iekšējās sistēmas projektu.
2. Detalizēti aprakstītu sistēmas struktūru un funkcionalitāti, kas ietver klientu un transportlīdzekļu reģistrāciju, informācijas meklēšanu un datu pārvaldību.
3. Noteiktu pamatprasības sistēmai, nodrošinot skaidru izpratni par tās darbību un lietošanas ērtumu.
4. Sniedz informāciju par izstrādes vides izvēli un izmantotajām tehnoloģijām.
5. Precīzi definētu darba uzdevumus un sistēmas izstrādes gaitu, nodrošinot augstu kvalitāti un efektivitāti projekta īstenošanā.

2.1.2. Darbības sfēra

Projekta iekšējās informācijas sistēmas darbības sfēra ietver transportlīdzekļu reģistrāciju un pārvaldību uzņēmuma drošajā stāvvietā. Šī sistēma nodrošina iespēju efektīvi uzskaitīt un pārvaldīt klientu automašīnas, ļaujot darbiniekiem viegli piekļūt informācijai par transportlīdzekļu novietošanas laiku, stāvvietu un klientu datiem. Sistēma veicina organizētu transportlīdzekļu uzskaiti, uzlabojot klientu apkalpošanu un optimizējot stāvvietas pārvaldību uzņēmumā.

2.1.3. Definīcijas

Lietotājs – Persona(s), kas lieto *WEB* lapu.

Klients – Persona, kam tiek veidota *WEB* lapa.

2.1.4. Dokumenta pārskats

Dokuments satur iekšējās informācijas sistēmas prasību specifikāciju, kas detalizēti apraksta projektu un tā funkcionalitāti transportlīdzekļu reģistrācijai un pārvaldībai uzņēmumā. Šī struktūra sniedz visaptverošu informāciju par projektu un tā izstrādes posmiem.

1. Ievads ietver programmas mērķi un darba uzdevumus, kā arī sniedz pārskatu par tās nozīmi uzņēmuma darbībā.
2. Projekta plānošana un plānu pārskatīšana apraksta, kā sasniegt noteiktos rezultātus, identificē darba uzdevumu izpildītājus un nosaka elastīgu, taču precīzu grafiku, ņemot vērā projekta mērķus un riskus.
3. Programmatūras vides iekārtošana aplūko, kā sagatavot un organizēt izstrādes vidi, tostarp integrētas izstrādes vides, versiju kontroli, datubāžu pārvaldību un drošības pasākumus.
4. Prasību specifikācija satur detalizētu aprakstu par sistēmas plānošanu un izstrādi, kā arī tās prasības, lai nodrošinātu pilnīgu izpratni par projekta mērķiem un funkcionalitāti.
5. Izmantotās tehnoloģijas apskata tehnoloģijas, kas tiks izmantotas projektā, un apraksta izvēles pamatnostādnes, to pielietojumu un priekšrocības.
6. Produkta izstrāde detalizē produkta izstrādes procesu, ietverot klientu reģistrāciju, transportlīdzekļu pievienošanu, informācijas meklēšanu un citas būtiskas funkcijas.
7. Testēšana ietver testēšanas metodes un scenārijus, kas nodrošinās produkta kvalitāti, kā arī apraksta testēšanas gaitas, kas tiks veiktas sistēmas izstrādes laikā.
8. Lietotāja ceļvedis sniedz detalizētu informāciju par sistēmas funkcijām un to izmantošanu, lai nodrošinātu lietotāju ērtību un efektivitāti, lietojot sistēmu.

2.2. Vispārīgs apraksts

2.2.1. Produkta perspektīva

Projekts "*ParkingPro*" ir izstrādāts, lai piedāvātu uzņēmumam efektīvu un organizētu risinājumu klientu transportlīdzekļu reģistrācijai un pārvaldībai. Šī sistēma kalpos uzņēmuma iekšējām vajadzībām, palīdzot uzturēt pārskatāmu informāciju par transportlīdzekļiem, kas tiek atstāti uzņēmuma stāvvietā dažādu iemeslu dēļ, piemēram, klientiem atrodoties ārpus Latvijas, vai vienkārši ir nepieciešamība atstāt transportlīdzekli uz nenoteiktu laiku tuvākajā uzņēmuma stāvvietā.

2.2.2. Produkta funkcijas

- Lietotāju reģistrācija un autorizācija:
 - Iespēja administratīvajiem lietotājiem droši reģistrēties un piekļūt sistēmai.
 - Sistēmas pārvaldība, lai nodrošinātu autorizētu piekļuvi, lai saglabātu datu drošību un uzticamību.
- Transportlīdzekļu reģistrācija:
 - Iespēja reģistrēt klienta transportlīdzekli, norādot transportlīdzekļa datus (reģistrācijas numurs, modelis un novietošanas periods).
 - Transportlīdzekļa informācijas glabāšana un atjaunošana, lai nodrošinātu ātru piekļuvi datiem.
- Klientu informācijas pārvaldība:
 - Klienta datu ievade un pārvaldība, ieskaitot vārdu, uzvārdu un kontaktinformāciju, lai nodrošinātu ātru saziņu un informācijas uzturēšanu.
 - Detalizētu klientu datu pievienošana un atjaunināšana.
- Stāvvietu pārvaldība:
 - Informācijas glabāšana par transportlīdzekļu atrašanās vietu, tostarp stāvvietas numuru, lai nodrošinātu pārskatāmību un operatīvu transportlīdzekļu pārvaldību.
 - Uzraudzības funkcionalitāte, kas ļauj ātri noteikt transportlīdzekļu pieejamību un to stāvvietas statusu.
- Laika uzskaite un statistika:

- Reģistrēšanās laika un novietošanas perioda uzskaitē, lai pārraudzītu, cik ilgi transportlīdzeklis atrodas stāvvietā.
- Datu statistikas analīze, kas palīdz optimizēt resursu plānošanu un noskaidrot ilgtermiņa novietošanas tendences.
- Meklēšanas opcija:
 - Meklē reģistrētos transportlīdzekļus pēc transportlīdzekļa reģistrācijas numura.
 - Darbiniekam vieglāk atrast caur visiem ierakstiem konkrēto transportlīdzekli.

2.2.3. Lietotāja raksturiezīmes

“*ParkingPro*” lietotāji ir iekšējie uzņēmuma darbinieki un administrators, kuru uzdevums ir pārvaldīt un reģistrēt klientu transportlīdzekļu datus un klienta kontaktinformācijas datus, ilgtermiņa vai īslaicīgai novietošanai. Lietotājiem ir pieejamas dažādas funkcijas, kas paredzētas, lai nodrošinātu efektīvu stāvvietas pārvaldību un klientu datu uzskaiti.

Sistēmas lietotājiem ir šādas raksturiezīmes:

- Administratori – lietotāji ar augstāko piekļuves līmeni, kuri var veikt transportlīdzekļu reģistrāciju, rediģēt klientu un transportlīdzekļu informāciju, meklēt ierakstus pēc transportlīdzekļa reģistrācijas numura, kā arī pievienot jaunus kontus, redz visu nodaļu ierakstus un visas reģistrētos kontus.
- Datu ievades speciālisti – darbinieki, kuri galvenokārt atbild par klientu un transportlīdzekļu datu ievadīšanu sistēmā, nodrošinot, ka visi nepieciešamie dati tiek reģistrēti precīzi un ātri, redz tikai savas nodaļas ierakstus, kurus ir iespējams arī labot pēc tam, kā arī ir meklēšanas opcija, kur var meklēt pēc reģistrēto transportlīdzekļu reģistrācijas numuriem caur saviem ierakstiem.

2.2.4. Vispārējie ierobežojumi

- Piekļuves prasības:
 - Lietotājiem jābūt piekļuvei uzņēmuma tīklam, lai izmantotu sistēmu, jo tā ir paredzēta uzņēmuma iekšējai lietošanai.
 - Sistēmai nepieciešama stabila interneta savienojuma pieejamība, lai nodrošinātu drošu un nepārtrauktu datu apmaiņu starp sistēmas komponentēm.
- Platformas saderība:

- Sistēmai jādarbojas uz uzņēmuma izmantotajām ierīcēm un pārlūkprogrammām, kas ietver populārākos interneta pārlūkus, piemēram, “*Chrome*” un “*Firefox*”, un jābūt optimizētai darbam uz galda datoriem un planšetēm.
- Drošības ierobežojumi:
 - Datu aizsardzības prasību ievērošana ir obligāta, īpaši attiecībā uz klientu personīgo un transportlīdzekļu informāciju. Sistēmai jābūt aizsargātai pret nesankcionētu piekļuvi un jānodrošina datu šifrēšana.
- Lietotāja interfeisa vienkāršība:
 - Sistēmai jābūt viegli pārskatāmai un lietotājiem saprotamai, lai nodrošinātu ātru piekļuvi informācijai un samazinātu laiku, kas nepieciešams datu ievadei un rediģēšanai.
- Valodas ierobežojumi:
 - Sistēmas interfeisam un dokumentācijai jābūt pieejamai latviešu valodā lai atvieglotu lietošanu darbiniekiem.

2.2.5. Projekta plānošana un plānu pārskatīšana

Projekta “*ParkingPro*” – transportlīdzekļu autostāvvietas reģistrācijas vadības sistēma, izstrādes plānā ir ņemti vērā svarīgākie aspekti un izvirzīti galvenie uzdevumi, lai nodrošinātu veiksmīgu rezultātu un sistēmas ieviešanu:

- **Rezultāti un ieguvumi:**
 - Projekta galarezultāts ir moderna un lietotājam draudzīga sistēma, kas ļauj reģistrēt un pārvaldīt klientu transportlīdzekļus efektīvā un pārskatāmā veidā.
 - Sistēma sniegs iespēju automatizēt procesus, optimizējot uzņēmuma resursu izmantošanu un uzlabojot klientu apkalpošanas kvalitāti.
- **Projekta realizētājs:**
 - Projektu izstrādā Jelgavas tehnikuma audzēknis Roberts Jansons, kurš izstrādā šo sistēmu kā daļu no sava kvalifikācijas darba, attīstot savas prasmes programmatūras veidošanā.
- **Termiņi un darba plāns:**
 - Projekts tiek plānots noteiktos laika ierobežojumos, un izstrādes noslēgumam jābūt līdz 2025. gada jūnijam.

- Plānojot darbu, tiek paredzēta elastība, lai pielāgotos iespējamām neparedzētām situācijām, vienlaikus piegādāt kvalitatīvu risinājumu.
- **Galvenie riski un to pārvaldība:**
 - Projekta lielākie izaicinājumi ir saistīti ar tehniskiem šķēršļiem, piemēram, sistēmas kļūdām vai datu pārvaldības sarežģījumiem.
 - Lai risinātu šos jautājumus, ir plānots rūpīgi testēt sistēmu un veikt regulāras pārbaudes.
- **Stratēģiskā nozīme:**
 - Projekts nodrošina uzņēmumam modernu digitālu rīku, kas palīdzēs vienkāršot un uzlabot transportlīdzekļu novietošanas pārvaldību.
 - Tā ieviešana sniegs uzņēmumam būtiskas priekšrocības, piemēram, laika un resursu ekonomiju.

Pārdomāts izstrādes plāns un risinājumu stratēģija garantē, ka “*ParkingPro*” sistēma būs uzticama un efektīva, nodrošinot atbilstību uzņēmuma vajadzībām un ilgtermiņa mērķiem.

2.2.6. Programmatūras gaitas apraksts

- **Reģistrācija un Pieslēgšanās:**
 - Lai izmantotu “*ParkingPro*” platformas iespējas, lietotājiem jābūt reģistrētam caur administratora profilu, norādot nepieciešamo informāciju – filiāles iekšējais nosaukums (piemēram, “P1”), lietotājvārdu un paroli. Pēc veiksmīgas reģistrācijas filiāle iegūst piekļuvi savam kontam, kur viņi var pārvaldīt savas rezervācijas.
 - Reģistrētie lietotāji var pieslēgties platformai, izmantojot savu lietotājvārdu un paroli. Pēc seansa beigām lietotājiem ir iespēja izrakstīties no sava konta, nodrošinot datu drošību un aizsardzību.
- **Transportlīdzekļu un klientu kontaktinformācijas reģistrācija un datu labošana:**
 - Katrai filiālei ir savs profils, kurā tiek reģistrēti transportlīdzekļa dati un klienta kontaktinformācija, administrators savā profilā redz visu reģistrētos ierakstus.
 - Katra filiāle var labot un dzēst savus veiktos ierakstus, bet administrators, var labot un dzēst visu veiktos ieraksts, kuri ir reģistrēti datubāzē.
- **Izrakstīšanās:**
 - Pēc lietošanas sesijas beigām lietotāji var droši izrakstīties no platformas, nodrošinot datu drošību un lietotāja sesijas aizsardzību.

2.2.7. Līmeņu izkārtojuma apraksts

- **Līmenis: Filiāles kots**

- Pieeja: filiāles kots ir paredzēts konkrētu filiāļu pārvaldībai un piedāvā piekļuvi informācijai par autostāvvietām, kas saistītas ar šo filiāli. Lietotāji ar šo piekļuves līmeni var meklēt, pievienot, rediģēt vai dzēst stāvvietu datus un rezervācijas informāciju.

- Atšķirības: filiāles kots lietotājiem nav piekļuves globālai sistēmas pārvaldībai vai citām filiālēm. Viņu funkcionalitāte aprobežojas ar konkrēto filiāli, kurā tie ir piešķirti.

- **Līmenis: Administratora kots**

- Pieeja: administratora kots nodrošina pilnīgu piekļuvi visai “*ParkingPro*” sistēmai, ieskaitot visu filiāļu pārvaldību un lietotāju administrēšanu.

- Atšķirības: administratoram ir visplašākās piekļuves un pārvaldības tiesības. Šis līmenis ir piemērots vispārējai sistēmas pārraudzībai un uzturēšanai.

2.3. Funkcionālās prasības

2.3.1. Lietotāja pieslēgšanās

Lietotāja reģistrācija ir nepieciešama, lai nodrošinātu piekļuvi “*ParkingPro*” sistēmas funkcionalitātēm, piemēram, transportlīdzekļu reģistrēšanai un datu pārvaldībai (skat. 1., 2. un 4. pielikumā).

Izsaukšanas elements: Poga “Pieteikties”.

Pieslēgšanās parametri: Lietotājvārds un parole;

Darbība (skat. 7. att.):

Pieteikšanās: Lietotājs noklikšķina uz pogas “Pieslēgties”;

Informācijas nolasīšana: Nolasa laukus: lietotājvārds un parole, pārbauda, vai lauki nav tukši;

Ja kāds no laukiem nav aizpildīts, parādās paziņojums: “*Please fill in this field.*”

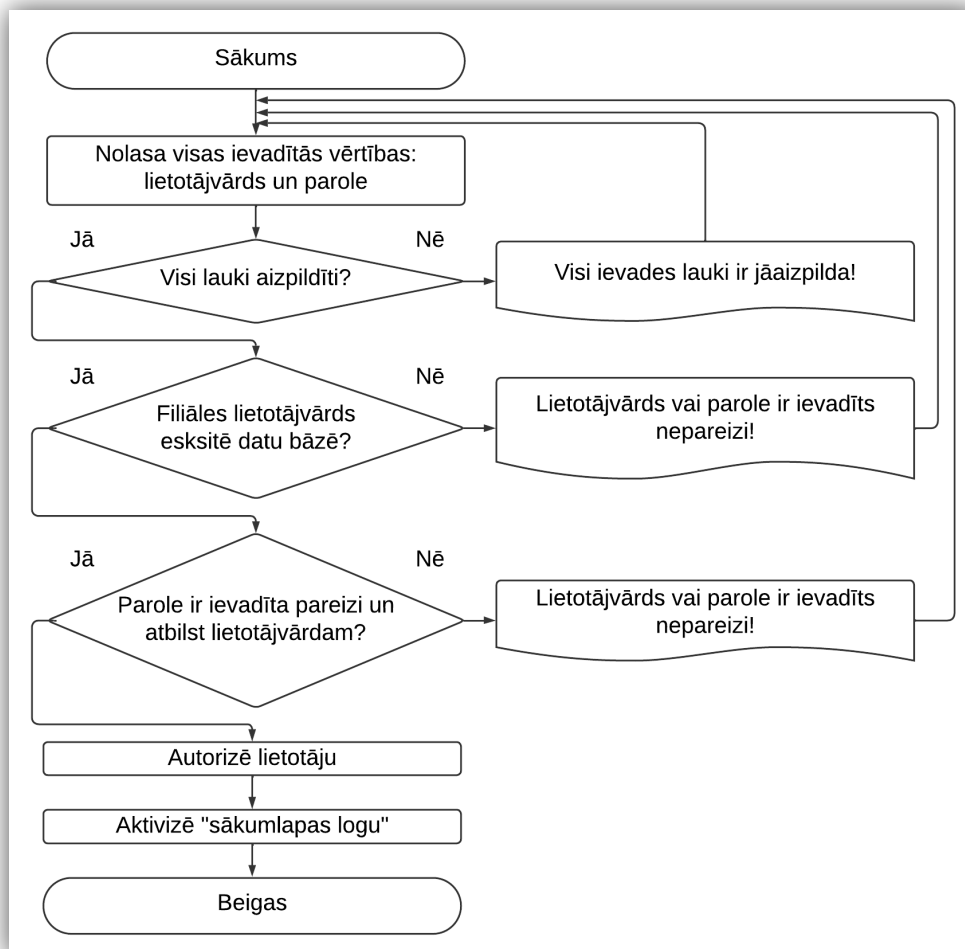
Datu pārbaude: Ievadītie autentifikācijas dati tiek salīdzināti ar saglabātajiem datiem datubāzē. Tiek pārbaudīts vai:

- e-pasts eksistē datubāzē;
- parole ir ievadīta pareizi un atbilst ievadītajam e-pastam;

Ja kāds no laukiem nesakrīt ar datubāzē esošo informāciju parādās paziņojums: “Pieteikšanās kļūda! Nepareizs lietotājvārds un/vai parole! Problēmu gadījumā sazināties ar Administratoru!”.

Pieteikšanās: Ja ievadītie dati sakrīt ar datu bāzē esošajiem, autentifikācija ir veiksmīga un lietotājs tiek pieteikts sistēmā kā autentificēts lietotājs, un viņam tiek piešķirts piekļuves līmenis, lai piekļūtu saviem datiem un citām sistēmas funkcijām.

Loga aktivizēšana: tiek aktivizēts “Sāku lapas logs”



7. att. Lietotāja pieslēgšanās funkcijas blokshēma

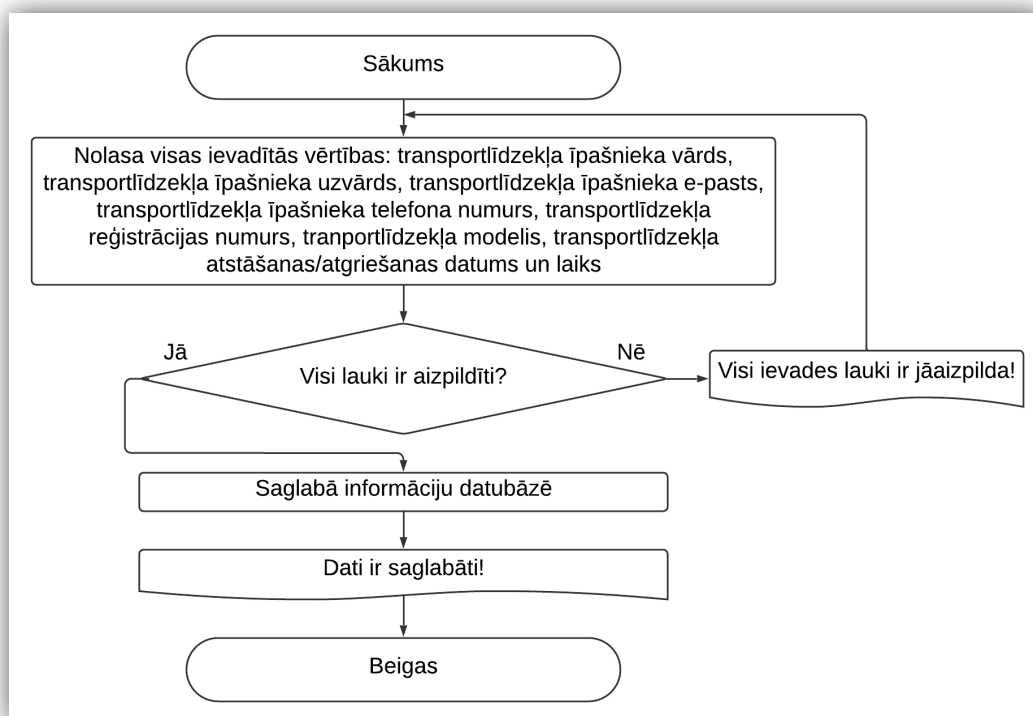
2.3.2. Transportlīdzekļa reģistrācija

Šī funkcija ļauj lietotājiem pievienot transportlīdzekļa datus sistēmā, lai tos reģistrētu stāvvietas uzskaitē (skat. 7., 8., 29. un 30. pielikumā).

Izsaukšanas elements: Poga “Reģistrēt transportlīdzekli”.

Ieejas parametri: Īpašnieka un transportlīdzekļa informācija - vārds, uzvārds, telefona numurs, e-pasts, transportlīdzekļa reģistrācijas numurs, modelis, novietošanas periods.

Darbība (skat. 8. att.):



8. att. Transportlīdzekļa reģistrēšanas funkcijas blokshēma

Informācijas nolasīšana: Sistēma nolasa ievadītos laukus un pārbauda, vai tie ir aizpildīti.

Ja kāds no laukiem nav aizpildīts, tad pie konkrētā lauka parādās paziņojums: “*Please fill in this field.*”.

Ja ievada e-pastu bez “@” simbola, tad pie e-pasta ievades lauka parādās paziņojums “*Please include an '@' in the email address. ‘(parādās teksts ko lietotājs ir ievadījis)’ is missing an '@’.*”.

Ja tiek ievadīts telefona numurs ar kādu simbolu parādās paziņojums “*Kļūda! Telefona numuram jābūt tikai cipariem.*”.

Ja tiek ievadīts reģistrācijas numurs, kas konkrētajā laika periodā ir jau reģistrēts, tad parādās paziņojums “*Kļūda! Šajā laika periodā jau ir reģistrēta šis transporta līdzekļa numurs.*”.

Ja reģistrācijas procesā, laika periodā, laiks no ir norādīts jaunāks par laiku līdz parādās paziņojums “*Atstāšanas laikam līdz jābūt vēlākam par atstāšanas laiku no.*”.

Datu saglabāšana: Ja dati ir derīgi, transportlīdzekļa informācija tiek saglabāta datu bāzē.

Paziņojums: Parādās paziņojums “*Veiksmīgi! Dati veiksmīgi saglabāti!*”.

2.3.3. Transportlīdzekļa meklēšana

Lietotāji var meklēt transportlīdzekļa datus pēc reģistrācijas numura (skat. 10. un 31. pielikumā).

Izsaukšanas elements: Poga “Reģistrētie transportlīdzekļi”.

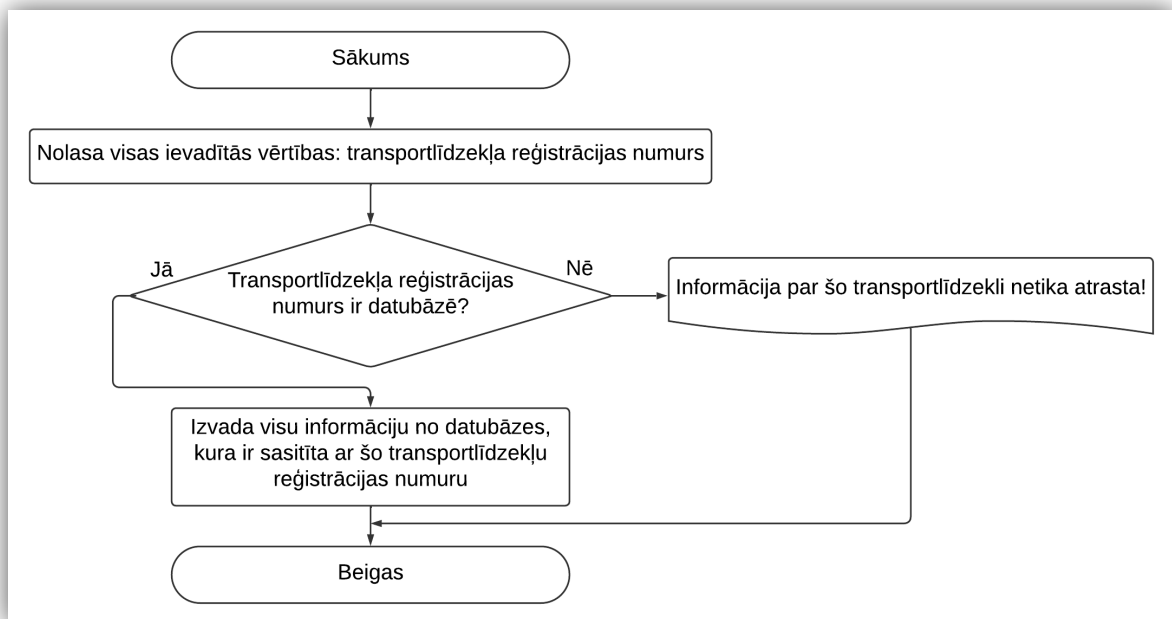
Ieejas parametri: Īpašnieka transportlīdzekļa reģistrācijas numurs.

Darbība (skat. 9. att.):

Meklēšana: Sistēma filtrē transportlīdzekļus, balstoties uz ievadīto kritēriju – transportlīdzekļa reģistrācijas numuru.

Rezultātu attēlošana: Meklēšanas rezultāti tiek attēloti tabulā ar transportlīdzekļu detaļām.

Datu apskates forma: Logā rādās seši ieraksti vienlaicīgi, ja ir vairāk par sešiem ierakstiem, tad parādās vadības josla, kur var pārslēgt ierakstu lapas.



9. att. Transportlīdzekļa meklēšanas funkcijas blokshēma

2.3.4. Transportlīdzekļa filtrēšana

Lietotāji var filtrēt transportlīdzekļa datus pēc ieraksta laika perioda (skat. 10. un 31. pielikumā).

Izsaukšanas elements: Pogas “Šobrīd aktīvs”, “Plānotais”, “Vēsture” un “Viss”.

Rezultātu attēlošana: Meklēšanas rezultāti tiek attēloti tabulā ar transportlīdzekļu detaļām.

Datu filtrēšana: Pēc noklusējuma dati rādās rindas kārtībā pēc ievadīšanas, jaunākie ieraksti pirmie,

Poga “Šobrīd aktīvs” – rādās ieraksti kuru laika periods ir šobrīd aktīvs;

Poga “Plānotais” – rādās ieraksti kuru laika periods vēl nav sācies;

Poga “Vēsture” – rādās ierakstu kuru laika periods ir jau beidzies;

Poga “Viss” – rādās visi veiktie ieraksti.

Datu apskates forma: Logā rādās seši ieraksti vienlaicīgi, ja ir vairāk par sešiem ierakstiem, tad parādās vadības josla, kur var pārslēgt ierakstu lapas.

2.3.5. Transportlīdzekļa datu rediģēšana

Lietotāji var rediģēt transportlīdzekļa datus pēc datu reģistrēšanas (skat. 14., 15., 34. un 35. pielikumā).

Izsaukšanas elements: Poga “Rediģēt”.

Rediģēšana: Lietotājs var izvēlēties transportlīdzekli un atvērt rediģēšanas logu, kurā ir iespējams labot datus.

Darbība (skat. 10.att.)

Informācijas nolasīšana: Sistēma nolasa ievadītos laukus un pārbauda, vai tie ir aizpildīti.

Ja kāds no laukiem nav aizpildīts, tad pie konkrētā lauka parādās paziņojums: “*Please fill in this field.*”.

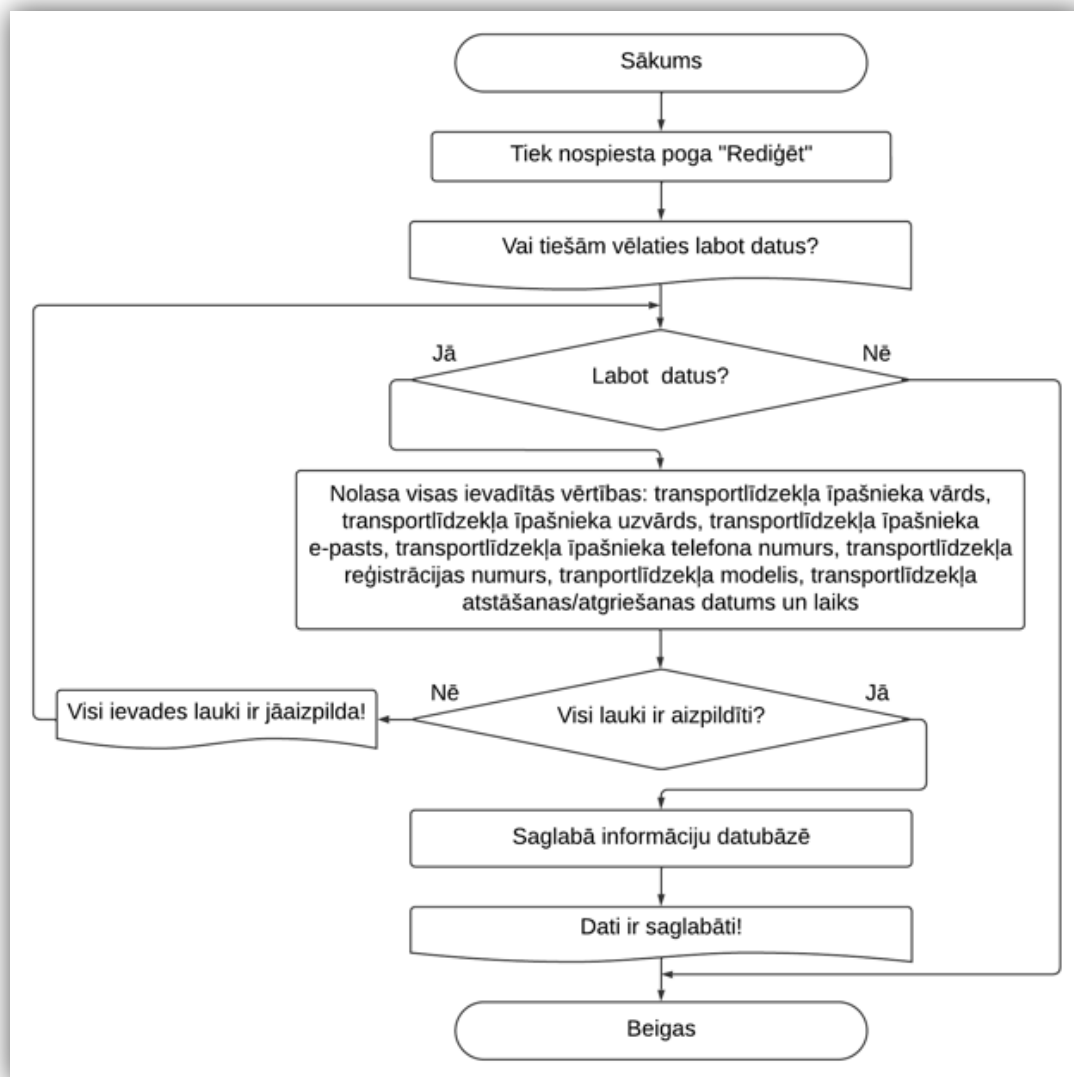
Ja ievada e-pastu bez “@” simbola, tad pie e-pasta ievades lauka parādās paziņojums “*Please include an '@' in the email address. '(parādās teksts ko lietotājs ir ievadījis)' is missing an '@'.*”.

Ja tiek ievadīts telefona numurs ar kādu simbolu parādās paziņojums “Kļūda! Telefona numuram jābūt tikai cipariem.”.

Ja tiek ievadīts reģistrācijas numurs, kas konkrētajā laika periodā ir jau reģistrēts, tad parādās paziņojums “Kļūda! Šajā laika periodā jau ir reģistrēta šis transporta līdzekļa numurs”.

Ja reģistrācijas procesā, laika periodā, laiks no ir norādīts jaunāks par laiku līdz parādās paziņojums “Atstāšanas laikam līdz jābūt vēlākam par atstāšanas laiku no.”.

Datu saglabāšana: Ja dati ir derīgi, transportlīdzekļa informācija tiek labota un saglabāta datu bāzē.



10. att. Transportlīdzekļa rediģēšanas funkcijas blokhēma

2.3.6. Transportlīdzekļa izņemšana/dzēšana no stāvvietas

Lietotāji var dzēst veiktos ierakstus, ieraksti tiek dzēsti neatgriezeniski (skat. 13. un 33. pielikumā).

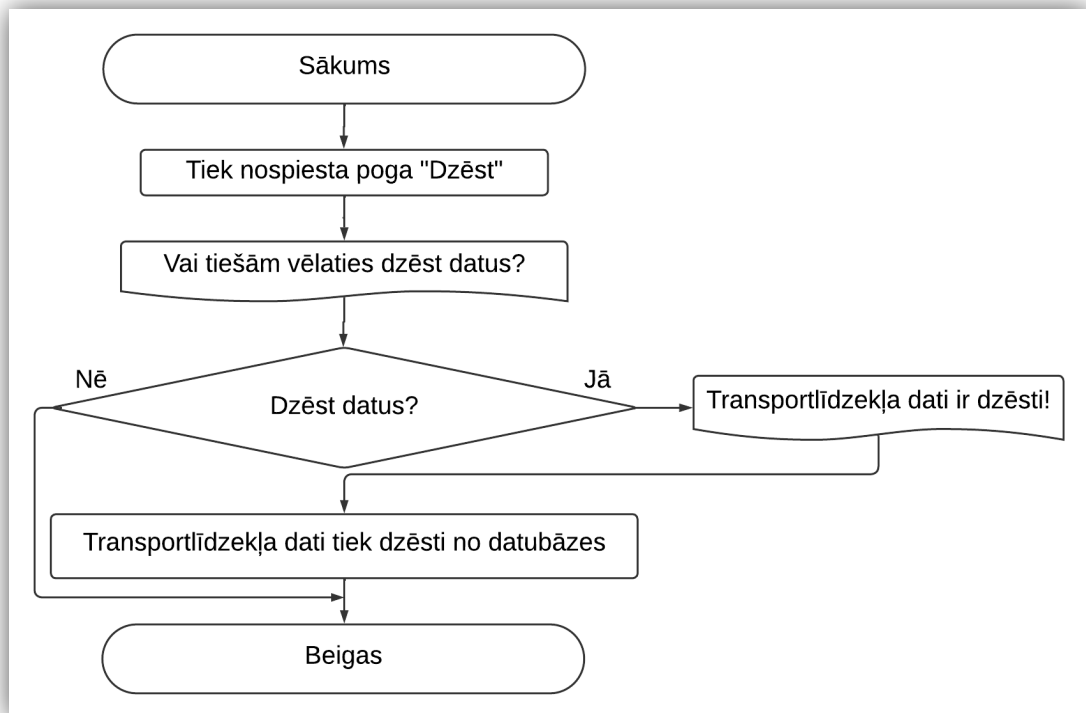
Izsaukšanas elements: Poga “Dzēst”.

Darbība (skat. 11. att.):

Informācijas nolasīšana: Sistēma nolasa transportlīdzekļa reģistrācijas numuru un pārbauda tā esamību datubāzē.

Paziņojums: Parādās paziņojums “Vai tiešām vēlies dzēst? Šo darbību nevar atsaukt!”, parādās divas pogas - “Atcelt” nenotiek dzēšanas operācija un aizver paziņojuma logu, “Jā,

dzēst!", ieraksts tiek neatgriezeniski izdzēsts un parādās paziņojums uz trīs sekundēm "Dati veiksmīgi izdzēsti!".



11. att. Transportlīdzekļa dzēšanas funkcijas blokshēma

2.3.7. Konta pievienošana

Logs kas ir pieejams tikai administratoram, sadaļā "Iestatījumi" (skat. 16., 17., 18., 19., 20., 21., 22., 23., 24., 25. un 26. pielikumā).

Izsaukšanas elements: Poga "Pievienot filiāli".

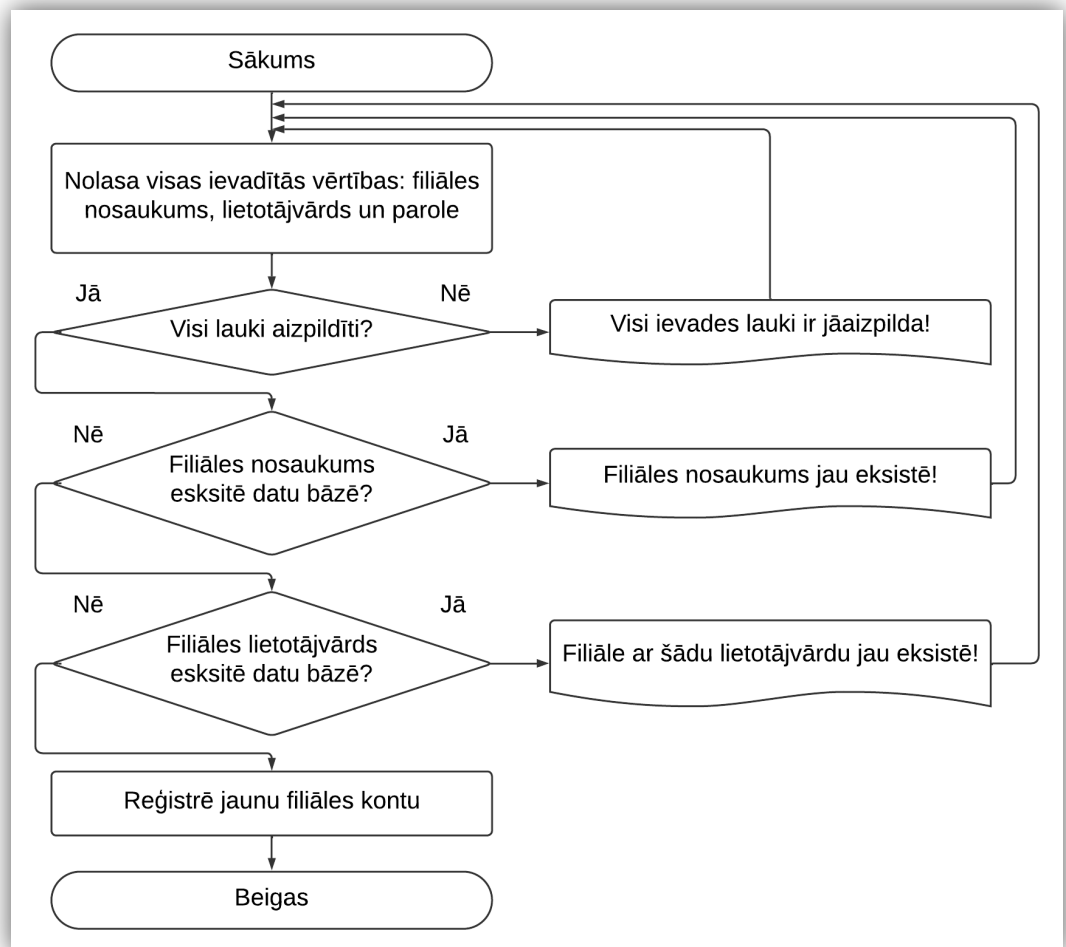
Ieejas parametri: Filiāles nosaukums, lietotājvārds un parole.

Darbība (skat. 12. att.):

Informācijas nolasīšana: Nolasa laukus: filiāles nosaukums, lietotājvārds, parole, pārbauda, vai lauki nav tukši, ja kāds no laukiem nav aizpildīts, tad pie konkrētā lauka parādās paziņojums "Please fill in this field.".

Datu pārbaude: Ievadītie dati tiek pārbaudīti, lai nodrošinātu to pareizību un atbilstību noteiktajiem kritērijiem. Piemēram, tiek pārbaudīts vai:

- lietotājvārds jau eksistē datubāzē, ja ir jau ieraksts ar tādu lietotājvārdu, tad parādās paziņojums: “Kļūda! Lietotājvārds jau eksistē!”.



12. att. Jauna konta reģistrēšanas funkcijas blokhēma

Konta izveide: Kad dati ir veiksmīgi pārbaudīti un apstiprināti, konts tiek izveidots sistēmā, un lietotājs tiek pievienots datubāzē ar unikālu identifikatoru.

Paziņojums par veiksmīgu reģistrāciju: Lietotājam parādās paziņojums “Filiāle veiksmīgi reģistrēta!”.

2.4. Nefunkcionālās prasības

2.4.1. Programmas logi

Vienots dizains: visi programmas logi tiek veidoti, ievērojot vienotu vizuālo identitāti. Tas nodrošina vienmērīgu lietotāja pieredzi un intuitīvu navigāciju (skat. 3., 5., 9., 11., 20. un 28. pielikumā).

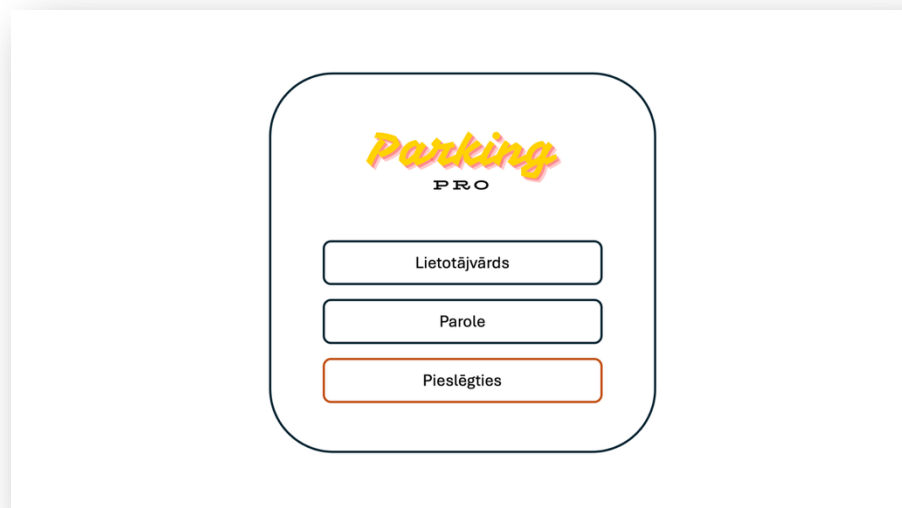
Reagējošs izkārtojums: logu izkārtojumi un elementi automātiski pielāgojas dažādiem ekrāna izmēriem, neatkarīgi no lietotāja ierīces, padarot sistēmu pieejamu gan datoros, gan mobilajās ierīcēs.

Krāsu shēma: baltais fons (*ghost white*, hex code: #F4F4F9) nodrošina tīru, minimālistisku izskatu, kas samazina vizuālo slodzi un uzlabo teksta lasāmību.

- **Pieteikšanās logs (skat. 13. attēlu):**

- **Formas elementu dizains:** pieteikšanās forma satur divus ievades laukus "Lietotājvārds" un "Parole", kā arī pieteikšanās pogu, kas atrodas tieši zem formas. Visi elementi ir centrēti, lai radītu vizuālu līdzsvaru un nodrošinātu ērtu piekļuvi (skat. 3. pielikumā).

- **Funkcionalitāte:** pieteikšanās logs pārbauda ievadīto datu korektumu un atbilstību sistēmā esošajiem lietotāju datiem, pēc nepieciešamības nodrošinot arī kļūdu paziņojumus (piemēram, nepareizs lietotājvārds vai parole).



13. att. Pieteikšanās loga makets

- **Programmas sākumekrāns (skat. 14. attēlu):**

- **Galvenās izvēlnes struktūra:** sākumekrānā ir pieejama administrācijas izvēlne ar četrām pogām: "Reģistrēt transportlīdzekli", "Reģistrētie transportlīdzekļi", "Iestatījumi", un "Atslēgties". Šīs izvēles ir sakārtotas vertikāli un vienmērīgi izvietotas, lai nodrošinātu ātru piekļuvi būtiskākajām funkcijām.

- **Navigācijas vienkāršība:** izvēlnes pogas ir pietiekami lielas, lai nodrošinātu ērtu lietošanu gan ar peli, gan pieskārieniem uz skārienekrāniem.



14. att. Programmas sākumekrāna makets

- **Datu ievades logs (skat. 15. attēlu):**
 - **Datu ievades lauki:** šajā logā lietotājam tiek piedāvāti vairāki ievades lauki transportlīdzekļu informācijas ievadīšanai. Katrā laukā ir paredzēta norāde vai viettura teksts, lai atvieglotu lietotāja datu ievadi (skat. 9. pielikumā).
 - **Verifikācijas mehānismi:** ievades lauki ietver datu validāciju, lai novērstu kļūdainu vai nepilnīgu informāciju, piemēram, kļūdas paziņojumus par neatbilstošu telefona numura formātu.

15. att. Datu ievades loga makets

- **Datu apskates logs (skat. 14. attēlu):**
 - **Datu attēlošanas veids:** šis logs nodrošina tabulas veida izkārtojumu, kur lietotājs var apskatīt visu pieejamo transportlīdzekļu informāciju. Kolonnas tiek skaidri atdalītas, nodrošinot pārskatāmību.
 - **Interaktivitāte:** lietotāji var izmantot papildu funkcijas, piemēram, datu filtrēšanu un kārtošānu, lai ātri atrastu nepieciešamo informāciju.

REĢISTRĒTIE TRANSPORTLĪDZEKĻI									
Šobrīd aktīvs Plānotais Vēsture Viss Numura zīme Meklēt									
Vārds	Uzvārds	E-Pasts	Telefona numurs	Numura zīme	Marka	Atstāšanas laiks NO	Atstāšanas laiks Līdz	Faila	Darības
piemērs	piemērs	piem@pie m.piem	123	Piem.1	piemērs	Yyyy-mm-dd hh:mm:ss	Yyyy-mm-dd hh:mm:ss	Admin	Rediģēt Dzēst
piemērs	piemērs	piem@pie m.piem	123	Piem.2	piemērs	Yyyy-mm-dd hh:mm:ss	Yyyy-mm-dd hh:mm:ss	p1	Rediģēt Dzēst
piemērs	piemērs	piem@pie m.piem	123	Piem.3	piemērs	Yyyy-mm-dd hh:mm:ss	Yyyy-mm-dd hh:mm:ss	p2	Rediģēt Dzēst
piemērs	piemērs	piem@pie m.piem	123	Piem.4	piemērs	Yyyy-mm-dd hh:mm:ss	Yyyy-mm-dd hh:mm:ss	p3	Rediģēt Dzēst
piemērs	piemērs	piem@pie m.piem	123	Piem.5	piemērs	Yyyy-mm-dd hh:mm:ss	Yyyy-mm-dd hh:mm:ss	p4	Rediģēt Dzēst
piemērs	piemērs	piem@pie m.piem	123	Piem.6	piemērs	Yyyy-mm-dd hh:mm:ss	Yyyy-mm-dd hh:mm:ss	p5	Rediģēt Dzēst

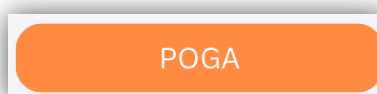
1 2 3 4

Atpakaļ

16. att. Datu apskates loga makets

2.4.2. Standarta pogas

Pogas tiek izmantotas dažādu funkciju izpildei, un to dizains ir vienots visā programmā (skat. 17. att.):



17. att. Pogas makets

- Forma: Taisnstūrveida ar noapaļotiem stūriem.
- Fona krāsa: Oranža (shades of sun, *hex code*: #FF8A41).
- Teksta krāsa: Balta (*white*, *hex code*: #FFFFFF), kas nodrošina skaidru kontrastu uz baltā fona.
- Teksta izmērs: 16 *px*, lai nodrošinātu viegli saskatāmu tekstu uz pogas.
- Teksta pozicionējums: Centrēts horizontāli un vertikāli, lai teksts vienmēr atrastos pogas vidū.

2.4.3. Ievadlauki

Ievadlauki nodrošina vienkāršu un skaidru datu ievadīšanu (skat. **18.** att.):



18. att. Ievadlauka makets

- Fona krāsa: Balta (*white*, *hex code*: #FFFFFF), kas saglabā vienkāršu dizainu.
- Teksta krāsa: Melna (*black*, *hex code*: #000000), nodrošinot skaidru kontrastu ar fona krāsu.
- Rāmja krāsa: Gaiši pelēka (*light grey*, *hex code*: #e8e8e8).

2.4.4. Drošība un datu aizsardzība

Sistēmai jānodrošina atbilstošas drošības prasības, lai aizsargātu klientu un transportlīdzekļu informāciju:

- Datu šifrēšana: Jānodrošina, ka visi klientu dati un paroles tiek šifrēti, lai aizsargātu tos pret nevēlamu piekļuvi.
- Datu dublēšana: Datu dublēšana tiek veikta reizi dienā, lai nodrošinātu datu atgūstamību, ja rodas sistēmas kļūdas vai datu zudumi.

2.4.5. Pieejamība un veiktspēja

Sistēmai jābūt uzticamai un pieejamai visiem lietotājiem:

- Sistēmas pieejamība: Jānodrošina, ka sistēma darbojas 99% no laika, lai lietotājiem būtu nepārtraukta piekļuve.
- Platformas saderība: Sistēmai jābūt saderīgai ar galvenajām interneta pārlūkprogrammām, piemēram, "Google Chrome", "Mozilla Firefox", "Safari" un "Microsoft Edge".

2.4.6. Valodas prasības

- Interfeisa un dokumentācijas valoda ir pieejama latviešu valodā.

3. UZDEVUMA RISINĀŠANAS LĪDZEKĻU PAMATOJUMS

3.1. Koda redaktors

3.1.1. *Visual Studio Code 1.95.2 (Universal)*

Visual Studio Code[13] ir viegla un jaudīga koda rediģēšanas platforma, ko izstrādājusi *Microsoft*, lai nodrošinātu programmētājiem plašu iespēju klāstu. Šī rīka galvenās priekšrocības ir tā spēja atbalstīt vairākas programmēšanas valodas un paplašināšanas iespējas, izmantojot dažādus spraudņus.

Šajā projektā tika izvēlēts *Visual Studio Code*, jo tas piedāvā lietotājam draudzīgu saskarsmi, kurā ērti iespējams pārvaldīt dažādu valodu failus, piemēram, *HTML*, *CSS*, *JavaScript* un *PHP*, kas izmantoti projekta veidošanā.

Tā atbalsta arī integrētas termināļa un atklādošanas iespējas, padarot to par būtisku rīku koda testēšanai un labošanai. *Visual Studio Code* ir platforma, kas darbojas uz dažādām operētājsistēmām, tai skaitā *Windows* un *macOS*, kuru izmanto šī projekta izstrādes laikā.

3.2. Datubāzes projektēšanas rīks

3.2.1. *TablePlus 5.9.2.*

TablePlus[2] ir moderns un intuitīvs datubāzes pārvaldības rīks, kas atbalsta daudzas datubāžu sistēmas, tostarp *MySQL*, *PostgreSQL*, *SQLite* un citas. Tas nodrošina ērti lietojamu interfeisu, kas ļauj izstrādātājiem ātri piekļūt, pārvaldīt un rediģēt datubāzes struktūru un datus.

Šajā projektā *TablePlus* tika izvēlēts kā galvenais rīks darbam ar *MySQL* datubāzi, pateicoties tā funkcionalitātei un lietotājam draudzīgajam dizainam. *TablePlus* piedāvā šādas priekšrocības:

Intuitīvs interfeiss: skaidra un viegli saprotama saskarne, kas ļauj ātri piekļūt nepieciešamajiem datiem un funkcijām.

Vaicājumu izpilde: spēcīgs *SQL* redaktors ar sintakses izcelšanu un automātisku vaicājumu pabeigšanu, kas paātrina darbu ar datubāzēm.

Reāllaika datu rediģēšana: iespēja tieši rediģēt datus tabulās, ievērojot *MySQL* datu integritāti.

Drošība: integrētas drošības funkcijas, piemēram, *SSH* atslēgas un *SSL* sertifikāti, lai nodrošinātu datubāzes savienojumu aizsardzību.

Datu eksportēšana un importēšana: vienkārši rīki datu pārsūtīšanai starp dažādiem formātiem un sistēmām.

Izmantojot *TablePlus*, šajā projektā tika izveidotas, uzturētas un pārvaldītas datubāzes tabulas, kā arī veikta datu pārbaude un uzlabošana. Tā elastība un plašais funkciju klāsts padarīja to par piemērotu izvēli šīs sistēmas izstrādei un testēšanai.

3.3. Lokālā servera izveides rīks

3.3.1. *XAMPP 8.2.12*

XAMPP[14] ir atvērta koda tīmekļa izstrādes komplekts, kas piedāvā visas nepieciešamās tehnoloģijas, lai veidotu un testētu tīmekļa lietojumprogrammas uz lokālās vides. Tas apvieno *Apache* serveri, *MySQL* datubāzi un *PHP*, kā arī citus noderīgus rīkus vienā platformā, padarot to par lielisku izvēli tīmekļa izstrādātājiem.

Šajā projektā tiks izmantots *XAMPP*, jo tas nodrošina vienkāršu un ātru risinājumu lokāla servera izveidei, kas nepieciešams *PHP* un *MySQL* bāzētu lietojumprogrammu testēšanai. Rīks ir viegli instalējams un konfigurējams, ļaujot izstrādātājam koncentrēties uz kodēšanu, nevis uz tehnisko konfigurāciju. *XAMPP* ir būtisks līdzeklis izstrādes un testēšanas posmā, piedāvājot stabilu vidi, kurā simulēt servera darbību un pārlicināties par sistēmas funkcionalitāti.

3.4. Blokslēmu izveides rīks

3.4.1. *Lucidchart (premium)*

Lucidchart[6] ir intuitīvs tīmekļa rīks, kas tiek izmantots diagrammu un blokslēmu veidošanai. Tā lietotājam draudzīgā saskarne un plašais funkciju klāsts ļauj viegli vizualizēt sarežģītas sistēmas un procesus.

Projekta ietvaros tika izvēlēts *Lucidchart*, jo tas piedāvā plašu veidņu un pielāgošanas iespēju klāstu, kas atvieglo gan sistēmas loģikas vizualizāciju, gan komandas saziņu.

Lucidchart ļauj izveidot strukturētas blokslēmas, kas precīzi attēlo projekta funkcionalitāti un datu plūsmas.

3.5. *Wireframe* izveides rīks

3.5.1. *Canva (free)*

Canva[11] ir grafiskā dizaina platforma, kas piedāvā rīkus vienkāršu un profesionālu vizualizāciju izveidei. Šis rīks nodrošina piekļuvi plašai veidņu un elementu bibliotēkai, padarot to ideāli piemērotu sākotnējās lietotāja saskarnes (*wireframe*) izstrādei.

Šajā projektā *Canva* tika izmantota, lai izveidotu tīmekļa saskarnes maketus, kuros skaidri attēlota lapas struktūra, izvietojums un funkcionalitāte.

Canva ir viegli lietojama un ļauj ātri pielāgot izkārtojumu, kā arī pievienot vizuālos elementus, kas nodrošina vienkāršu un efektīvu komunikāciju starp izstrādātājiem un klientiem.

3.6. Skriptu, stila un koda programmēšanas valodas

3.6.1. *HTML (HyperText Markup Language)*

HTML[5] ir pamata marķēšanas valoda, kas tiek izmantota tīmekļa lapu satura un struktūras veidošanai. Tā nosaka, kā elementi, piemēram, virsraksti, paragrāfi, attēli un saites, tiek izvietoti un attēloti lietotāja pārlūkprogrammā.

Šajā projektā tiks izmantots *HTML*, jo tā vienkāršība un saderība ar visām populārākajām pārlūkprogrammām nodrošina stabilu un viegli uzturamu tīmekļa lapu pamatu. *HTML* ļauj veidot pārskatāmu dokumenta struktūru, kas turpmāk atvieglo sadarbību ar citām tehnoloģijām, piemēram, *CSS* un *JavaScript*, kā arī nodrošina elastību projekta paplašināšanai nākotnē.

3.6.2. *JavaScript*

JavaScript[12] ir skriptu valoda, kas tiek izmantota, lai nodrošinātu interaktivitātes un dinamiskas funkcijas tīmekļa lapās. Tā spēj reaģēt uz lietotāju darbībām, validēt formas datus un manipulēt ar *HTML* elementiem, lai uzlabotu lietotāja pieredzi.

Projekta “*ParkingPro*” ietvaros *JavaScript* tika izmantots, lai izveidotu dinamiskas funkcijas, piemēram, meklēšanas laukus un interaktīvus ziņojumus.

Tā nodrošina sistēmas responsivitāti un palīdz apstrādāt lietotāju darbības reālajā laikā, uzlabojot lietotāja pieredzi. *JavaScript* spēja integrēties ar citām tehnoloģijām padara to par neaizstājamu rīku šajā projektā.

3.6.3. *CSS (Cascading Style Sheets)*

CSS[1] tiek izmantots, lai kontrolētu tīmekļa lapas vizuālo izskatu, definējot tādas elementus kā krāsas, fontus, izvietojumu un kopējo dizainu. Tā palīdz nodalīt satura struktūru no stila, kas uzlabo koda pārskatāmību un uzturamību.

Projekta ietvaros CSS tika izmantots, lai izveidotu tīru un profesionālu dizainu “*ParkingPro*” sistēmai. Izmantojot atsaucīgo dizainu (*responsive design*), sistēma ir pielāgota dažādām ierīcēm un ekrānu izmēriem. CSS nodrošināja vienotu izskatu visās sistēmas lapās, kas veicina lietotāja ērtības un uzlabo vizuālo pievilcību.

3.6.4. *PHP (Hypertext Preprocessor)*

PHP[10] ir servera puses skriptu valoda, kas ļauj veidot dinamiskas un datu bāzēs balstītas tīmekļa lietotnes. Tā ir plaši izmantota, pateicoties tās elastībai un vienkāršībai, veidojot servera loģiku, piemēram, datu apstrādi, autentifikāciju un sesiju pārvaldību.

Šajā projektā PHP tika izvēlēts, lai izveidotu servera puses loģiku “*ParkingPro*” sistēmai. PHP tika izmantots datu bāzes pieprasījumu izpildei, lietotāju autentifikācijas nodrošināšanai un sesiju pārvaldībai. Tā spēja efektīvi apstrādāt datu bāzes darbības un integrēties ar *MySQL* padarīja to par lielisku izvēli projekta mērķu sasniegšanai. PHP nodrošina drošu un stabilu sistēmas darbību, kā arī pielāgojamību nākotnes uzlabojumiem.

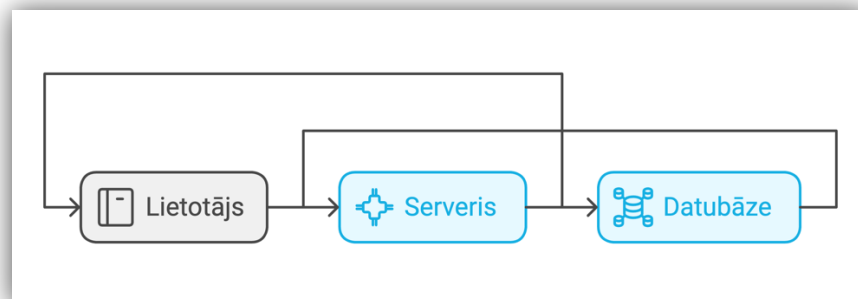
4. SISTĒMAS STRUKTŪRAS MODELIS

4.1. Sistēmas struktūra

Sistēma "*ParkingPro*" ir veidota kā uzņēmuma iekšējās lietošanas rīks transportlīdzekļu reģistrācijai un pārvaldībai.

Sistēma sastāv no šādiem komponentiem (skat. 19. att.):

- **Lietotāja interfeiss**
 - Administrācijas interfeiss, kas ļauj piekļūt visu filiāļu datiem.
 - Filiāļu interfeiss, kas ierobežo piekļuvi konkrētās filiāles datiem.
 - Izmanto *HTML*, *CSS* un *JavaScript* lietotāja pieredzes un funkcionalitātes nodrošināšanai.
- **Servera puse**
 - Serveris, kas apstrādā pieprasījumus un saziņu starp lietotāju un datu bāzi, izmantojot *PHP*.
 - Droša autentifikācija un datu šifrēšana.
- **Datubāze**
 - Izstrādāta, izmantojot *TablePlus*, nodrošinot klientu un transportlīdzekļu datu glabāšanu un efektīvu apstrādi (skat. 36. un 37. pielikumā).
- **Funkcionalitāte**
 - Lietotāju autentifikācija.
 - Transportlīdzekļu reģistrācija, meklēšana un pārvaldība.
 - Stāvvietu pārvaldības funkcionalitāte.



19. att. Sistēmas struktūras pārvaldību [Mākslīgā intelekta ģenerēts attēls]

4.2. Algoritma shēma ar izvēlētās risināšanas metodes aprakstu

Sistēmā izmantoti šādi algoritmi:

- **Autentifikācija:** Lietotājvārds un parole tiek salīdzināti ar datiem datubāzē. Veiksmīgas pārbaudes gadījumā tiek piešķirta piekļuve, citādi lietotājs saņem kļūdas paziņojumu “Pieteikšanās kļūda! Nepareizs lietotājvārds un/vai parole! Problēmu gadījumā sazināties ar Administratoru!”.
- **Transportlīdzekļu reģistrācija:** Ievadītie dati tiek pārbaudīti un validēti, pēc tam saglabāti datubāzē. Nepareizi aizpildītu lauku gadījumā lietotājs saņem paziņojumu par nepieciešamajiem labojumiem, e-pasta, telefona numura, reģistrācijas numura vai perioda norādīšanas laukam.
- **Datu meklēšana:** Meklēšanas kritēriji (reģistrācijas numurs) tiek pārbaudīti, un atbilstošie dati tiek atgriezti no datu bāzes, izmantojot *SQL* vaicājumus.
- **Datu rediģēšana:** Lietotājs var modificēt esošos datus, un izmaiņas tiek atspoguļotas datu bāzē reāllaikā.
- **Datu dzēšana:** Lietotājs var dzēst esošos ierakstus, un izmaiņas nav iespējams atsaukt.

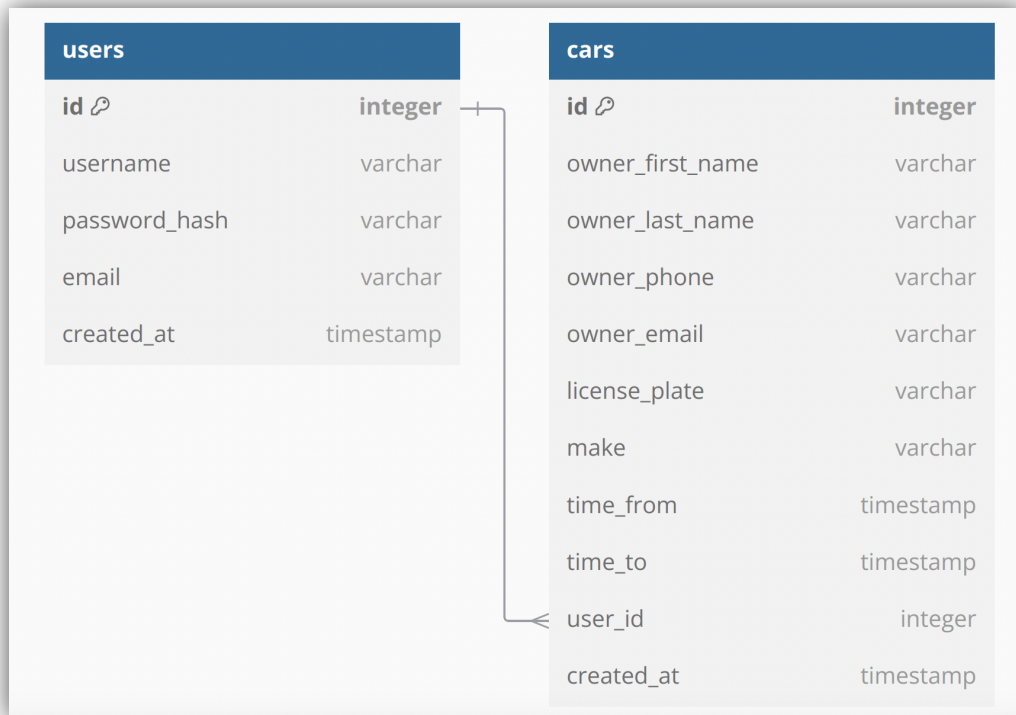
4.3. Klašu diagrammas

Sistēmas arhitektūrā tiek izmantotas šādas galvenās klases:

- **Lietotājs (*User*):**
 - Loma, vārds, uzvārds, kontaktinformācija, piekļuves līmenis.
- **Transportlīdzeklis (*Vehicle*):**
 - Reģistrācijas numurs, modelis, īpašnieka *ID*, novietošanas datumi.
- **Stāvvietā (*ParkingLot*):**
 - Numurs, transportlīdzekļa *ID*, stāvvietas statuss.
- **Autentifikācija (*Authentication*):**
 - Metodes lietotāju datu validācijai un autentifikācijai.

4.4. ER modelis

Datubāze "ParkingPro" ir veidota, lai efektīvi pārvaldītu transportlīdzekļu reģistrāciju un saistītos datus. Tā ietver divas galvenās tabulas: "users" (struktūru skat. 1. tabulā.) un "cars" (struktūru skat. 2. tabulā.). Šīs tabulas ir savstarpēji saistītas ar relāciju, lai nodrošinātu datu integritāti un loģisku saikni starp lietotājiem un transportlīdzekļiem (skat. 20. attēlu).



20. att. Entītijas attiecību diagramma kura izveidota izmantojot *WEB* lapu[3]

4.4.1. "users" tabula

Satur informāciju par sistēmas lietotājiem (administratori un datu ievades speciālisti).

Katrs lietotājs ir unikāli identificēts ar primāro atslēgu *id*.

Tabulā glabājas lietotājvārds (*username*), parole (*password_hash*), e-pasts (*email*) un konta izveides datums (*created_at*), (struktūru skat. 1. tabulā.) (skat. 17. pielikumā).

1. tabula "users" satur informāciju par sistēmas lietotājiem (administratori un datu ievades speciālisti)

Nosaukums	Tips	Atribūti	Komentāri
<i>id</i>	<i>integer</i>	<i>NOT NULL, PRIMARY KEY, AUTO INCREMENT</i>	Lietotāja unikālais identifikators
<i>username</i>	<i>varchar(255)</i>	<i>NOT NULL</i>	Lietotājvārds

Nosaukums	Tips	Atribūti	Komentāri
<i>password_hash</i>	<i>varchar(255)</i>	<i>NOT NULL</i>	Lietotāja šifrētā parole
<i>email</i>	<i>varchar(255)</i>	<i>NOT NULL</i>	Lietotāja e-pasta adrese
<i>created_at</i>	<i>timestamp</i>	<i>NOT NULL</i>	Konta izveides datums un laiks

4.4.2. “cars” tabula

Satur informāciju par klientu transportlīdzekļiem.

Katrs transportlīdzeklis ir unikāli identificēts ar primāro atslēgu *id*.

Tabulā glabājas transportlīdzekļa reģistrācijas numurs (*license_plate*), īpašnieka dati (*owner_first_name*, *owner_last_name*, *owner_phone*, *owner_email*), kā arī transportlīdzekļa novietošanas periods (*time_from*, *time_to*), (struktūru skat. 2. tabulā.) (skat. 8. un 29. pielikumā).

Transportlīdzeklis ir saistīts ar konkrētu lietotāju, izmantojot ārējo atslēgu *user_id*.

2. tabula “cars” satur informāciju par klientu transportlīdzekļiem un novietošanas datumiem

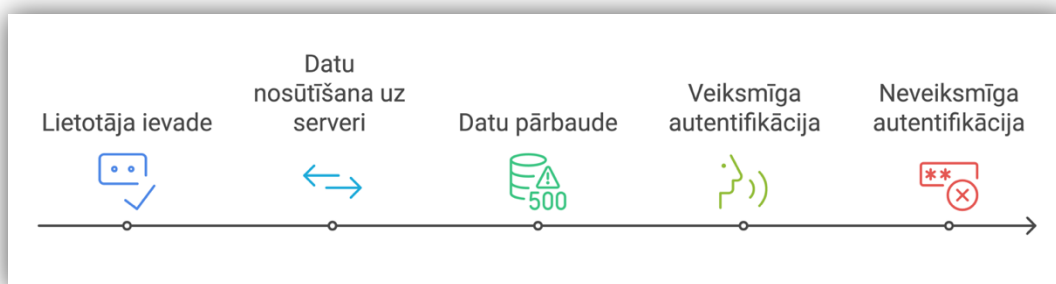
Nosaukums	Tips	Atribūti	Komentāri
<i>id</i>	<i>integer</i>	<i>NOT NULL, PRIMARY KEY, AUTO INCREMENT</i>	Transportlīdzekļa unikālais identifikators
<i>owner_first_name</i>	<i>varchar(255)</i>	<i>NOT NULL</i>	Īpašnieka vārds
<i>owner_last_name</i>	<i>varchar(255)</i>	<i>NOT NULL</i>	Īpašnieka uzvārds
<i>owner_phone</i>	<i>varchar(255)</i>	<i>NOT NULL</i>	Īpašnieka tālruņa numurs
<i>owner_email</i>	<i>varchar(255)</i>	<i>NOT NULL</i>	Īpašnieka e-pasta adrese
<i>license_plate</i>	<i>varchar(255)</i>	<i>NOT NULL</i>	Transportlīdzekļa reģistrācijas numurs
<i>make</i>	<i>varchar(255)</i>	<i>NOT NULL</i>	Transportlīdzekļa marka
<i>time_from</i>	<i>timestamp</i>	<i>NOT NULL</i>	Transportlīdzekļa novietošanas sākuma laiks
<i>time_to</i>	<i>timestamp</i>	<i>NOT NULL</i>	Transportlīdzekļa novietošanas beigu laiks
<i>user_id</i>	<i>integer</i>	<i>NOT NULL, FOREIGN KEY</i>	Saistība ar lietotāju tabulu
<i>created_at</i>	<i>timestamp</i>	<i>NOT NULL</i>	Ieraksta izveides datums un laiks

5. FUNKCIONĀLAIS UN DINAMISKAIS SISTĒMAS MODELIS

5.1. Datu plūsmas diagrammas

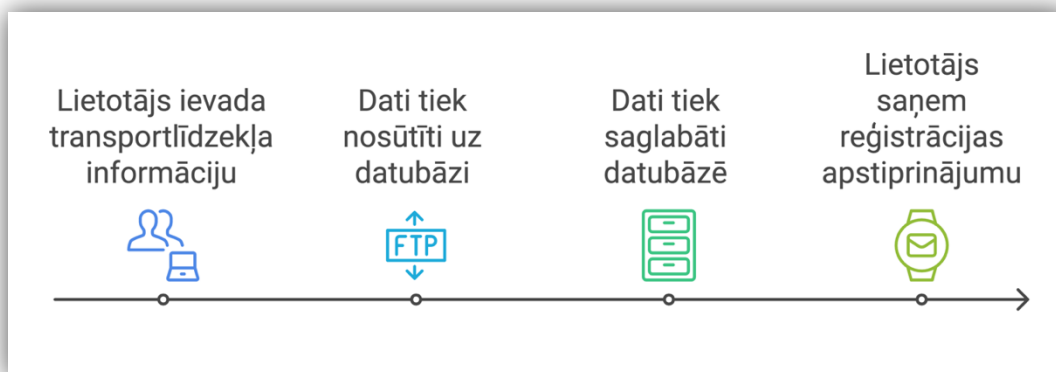
Datu plūsmas diagrammas parāda, kā informācija pārvietojas starp dažādām sistēmas komponentēm un lietotāju. Sistēmai "ParkingPro" būtiskas ir šādas datu plūsmas:

Lietotāja autentifikācija: process sākas, kad lietotājs ievada savu lietotājvārdu un paroli. Šie dati tiek nosūtīti uz sistēmas serveri, kur tie tiek pārbaudīti pret datubāzes ierakstiem. Ja ievadītie dati atbilst kādam no datubāzes ierakstiem, lietotājs tiek autentificēts un iegūst piekļuvi sistēmas funkcionalitātēm. Pretējā gadījumā tiek parādīts kļūdas paziņojums (skat. 21.att.).



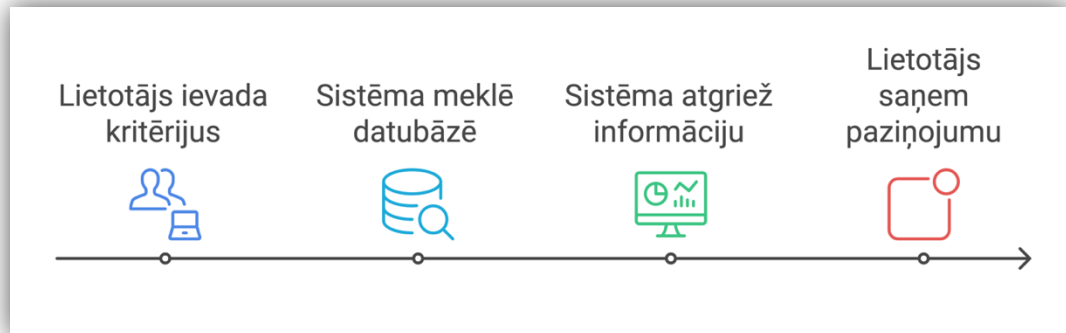
21. att. Lietotāja autentifikācijas secība [Mākslīgā intelekta ģenerēts attēls]

Transportlīdzekļa reģistrācija: process notiek, kad lietotājs ievada nepieciešamo informāciju par savu transportlīdzekli, piemēram, reģistrācijas numuru, marku un modeli. Šie dati tiek nosūtīti uz sistēmas datubāzi, kur tie tiek saglabāti. Pēc veiksmīgas reģistrācijas lietotājs saņem apstiprinājumu par transportlīdzekļa pievienošanu (skat. 22.att.).



22. att. Transportlīdzekļa reģistrācijas process [Mākslīgā intelekta ģenerēts attēls]

Meklēšana un pārvaldība: process sākas, kad lietotājs ievada kritērijus, piemēram, reģistrācijas numuru, lai atrastu konkrētu transportlīdzekli. Sistēma veic meklēšanu datubāzē un atgriež atbilstošo informāciju, ja tāda ir pieejama. Ja transportlīdzeklis netiek atrasts, lietotājs saņem paziņojumu par to, ka meklētā informācija nav pieejama (skat. 23.att.).



23. att. Transportlīdzekļa meklēšanas un pārvaldības secība [Mākslīgā intelekta ģenerēts attēls]

5.2. Stāvokļa diagramma

Stāvokļa diagramma parāda, kā sistēma reaģē uz dažādām lietotāja darbībām un maina stāvokļus. "ParkingPro" būtiskākie stāvokļi ir:

- **Nepieteicies lietotājs:**
 - Pieejams tikai pieteikšanās logs.
 - Stāvokļa maiņa notiek pēc veiksmīgas autentifikācijas.
- **Autentificēts lietotājs:**
 - Lietotājs var veikt transportlīdzekļa reģistrāciju, meklēšanu un rediģēšanu un dzēšanu.
- **Administrators režīms:**
 - Administrators var pārvaldīt lietotājus un redzēt visu transportlīdzekļu ierakstus.

5.3. Scenārijs – transportlīdzekļa reģistrācija

Scenārija mērķis: demonstrēt, kā lietotājs reģistrē transportlīdzekli, izmantojot sistēmu "ParkingPro". Scenārijā ir detalizēti aprakstīti soļi, sistēmas reakcija un datu apstrāde.

- **Sākotnējais stāvoklis:**
 - Lietotājs vēl nav pieteicies sistēmā.
 - Sistēma atrodas sākumlapā, kur pieejama pieteikšanās forma.

- **Turpmākie soļi:**
 - Lietotājs piesakās sistēmā.
 - Lietotājs atver sistēmas sākulapu un ievada savu lietotājvārdu un paroli.
 - Sistēma veic ievadīto datu validāciju un salīdzina tos ar datu bāzē esošajiem datiem.
 - Veiksmīgs gadījums: Sistēma apstiprina lietotāja identitāti un pāradresē uz galveno izvēlni.
 - Neveiksmīgs gadījums: Sistēma parāda kļūdas ziņojumu ("Nepareizs lietotājvārds vai parole") un piedāvā atkārtot mēģinājumu.
- **Lietotājs atver transportlīdzekļa reģistrācijas logu:**
 - Galvenajā izvēlnē lietotājs izvēlas sadaļu "Reģistrēt transportlīdzekli".
 - Sistēma atver transportlīdzekļa reģistrācijas formu.
- **Lietotājs aizpilda reģistrācijas formu ar sekojošo informāciju:**
 - Īpašnieka vārds un uzvārds.
 - Īpašnieka kontaktinformācija (telefona numurs un e-pasts).
 - Transportlīdzekļa reģistrācijas numurs.
 - Transportlīdzekļa marka.
 - Novietošanas sākuma un beigu laiks.
- **Sistēma validē ievadīto informāciju:**
 - Lauki nedrīkst būt tukši.
 - Pārbauda vai e-pasta ievades sakrīt e-pasta formātam, piemēram, piemers@piemers.lv.
 - Pārbauda vai telefona numurs ir tikai cipari.
 - Pārbauda vai konkrētajā laika periodā nav reģistrēts konkrētais transportlīdzekļa reģistrācijas numurs.
 - Laika intervāls tiek pārbaudīts, lai sākuma laiks būtu vecāks par beigu laiku.
- **Sistēma apstrādā datus, ja validācija ir veiksmīga:**
 - Sistēma saglabā ievadīto informāciju tabulā "cars".
 - Transportlīdzekļa ierakstam tiek piešķirts unikāls *ID*, un tā izveides datums tiek reģistrēts.
 - Pārbauda paziņojumu "Veiksmīgi! Dati veiksmīgi reģistrēti!" un atver "Reģistrētie transportlīdzekļi" logu.

- **Ja validācija ir neveiksmīga:**
 - Sistēma atgriež kļūdas ziņojumu ar konkrētu problēmu (piemēram, "Telefona numuram jābūt tikai cipariem!").
- **Sistēma parāda rezultātu lietotājam, ja ir veiksmīga reģistrācija:**
 - Sistēma parāda ziņojumu: "Veiksmīgi! Dati veiksmīgi saglabāti!".
 - Lietotājs tiek pārnests uz "Reģistrētie transportlīdzekļi" logu.
- **Sistēma parāda rezultātu lietotājam, ja ir neveiksmīga reģistrācija:**
 - Sistēma atgriež formu ar kļūdas ziņojumiem un pieprasa veikt labojumus.
- **Beigu stāvoklis:**
 - Lietotāja ievadītie dati ir veiksmīgi saglabāti datubāzē tabulā "cars".
 - Sistēma ir gatava pieņemt jaunus pieprasījumus vai veikt citas darbības.

5.4. Scenārijs – transportlīdzekļa meklēšana sistēmā

Scenārija mērķis: demonstrēt, kā lietotājs veic transportlīdzekļa meklēšanu sistēmā, izmantojot sistēmu "ParkingPro". Scenārijā ir detalizēti aprakstīti soļi, sistēmas reakcija un datu apstrāde

- **Sākotnējais stāvoklis:**
 - Lietotājs vēl nav pieteicies sistēmā.
 - Sistēma atrodas sākumlapā, kur pieejama pieteikšanās forma.
- **Turpmākie soļi:**
 - Lietotājs piesakās sistēmā.
 - Lietotājs atver sistēmas sākumlapu un ievada savu lietotājvārdu un paroli.
 - Sistēma veic ievadīto datu validāciju un salīdzina tos ar datu bāzē esošajiem datiem.
 - Veiksmīgs gadījums: Sistēma apstiprina lietotāja identitāti un pāradresē uz galveno izvēlni.
 - Neveiksmīgs gadījums: Sistēma parāda kļūdas ziņojumu ("Pieteikšanās kļūda! Nepareizs lietotājvārds un/vai parole! Problēmu gadījumā sazināties ar Administratoru! ") un piedāvā atkārtot mēģinājumu.
- **Lietotājs atver reģistrētie transportlīdzekļi logu:**
 - Galvenajā izvēlnē lietotājs izvēlas sadaļu "Reģistrētie transportlīdzekļi".
 - Sistēma atver logu, kur ir reģistrētie transportlīdzekļi.

- Lietotājs meklēšanas logā ievada transportlīdzekļa reģistrācijas numuru un spiež meklēt.
- **Sistēma validē ievadītos meklēšanas kritērijus:**
 - Ja nav ievadīti kritēriji, sistēma nereaģē uz nospiesto pogu.
- **Ja dati ir korekti, sistēma virzās uz nākamo soli:**
 - Sistēma apstrādā pieprasījumu
 - Sistēma izveido *SQL* vaicājumu, lai meklētu datubāzes tabulā "*cars*" atbilstošos ierakstus.
- **Sistēma pārbauda rezultātus:**
 - Ja ieraksti tiek atrasti: sistēma sagatavo un atgriež rezultātu.
 - Ja ieraksti netiek atrasti: sistēma izvada tukšu tabulu.
 - Sistēma parāda rezultātu lietotājam
- Sistēma attēlo meklēšanas rezultātus tabulas formātā, ietverot šādu informāciju:
 - Reģistrācijas numurs.
 - Īpašnieka vārds un uzvārds.
 - Īpašnieka kontaktālrūnis un e-pasts.
 - Transportlīdzekļa marka.
 - Novietošanas sākuma un beigu laiks.
 - Ja nepieciešams, lietotājs var izmantot papildu funkcijas, piemēram:
 - Rediģēt transportlīdzekļa informāciju.
 - Dzēst ierakstu.
- Beigu stāvoklis
 - Sistēma ir veikusi transportlīdzekļa meklēšanas darbību.
 - Lietotājs ir ieguvis nepieciešamo informāciju vai izvēlējies veikt jaunu meklēšanu.

5.5. Scenārijs – lietotāja reģistrēšana sistēmā

Scenārija mērķis: demonstrēt, kā administrators veic jauna lietotāja reģistrāciju, izmantojot sistēmu "*ParkingPro*". Scenārijā ir detalizēti aprakstīti soļi, sistēmas reakcija un datu apstrāde

- **Sākotnējais stāvoklis:**
 - Administrators vēl nav pieteicies sistēmā.
 - Sistēma atrodas sākumlapā, kur pieejama pieteikšanās forma.

- **Turpmākie soļi:**
 - Administrators piesakās sistēmā.
 - Administrators atver sistēmas sākumlapu un ievada savu lietotājvārdu un paroli.
 - Sistēma veic ievadīto datu validāciju un salīdzina tos ar datu bāzē esošajiem datiem.
 - Veiksmīgs gadījums: Sistēma apstiprina lietotāja identitāti un pāradresē uz galveno izvēlni.
 - Neveiksmīgs gadījums: Sistēma parāda kļūdas ziņojumu ("Pieteikšanās kļūda! Nepareizs lietotājvārds un/vai parole! Problēmu gadījumā sazināties ar Administratoru!") un piedāvā atkārtot mēģinājumu.
- **Administrators atver transportlīdzekļa reģistrācijas logu:**
 - Galvenajā izvēlnē administrators izvēlas sadaļu "Iestatījumi" un tad "Pievienot filiāli".
 - Sistēma atver jaunas filiāles reģistrācijas formu.
- **Administrators aizpilda reģistrācijas formu ar sekojošo informāciju:**
 - Filiāles nosaukums.
 - Filiāles lietotājvārds.
 - Filiāles parole.
- **Sistēma validē ievadīto informāciju:**
 - Lauki nedrīkst būt tukši.
 - Pārbauda vai ievadītās filiāles lietotājvārds jau neeksistē sistēmā.
- **Sistēma apstrādā datus, ja validācija ir veiksmīga:**
 - Sistēma saglabā ievadīto informāciju tabulā "users".
 - Filiāles kontam tiek piešķirts unikāls ID, un tā izveides datums tiek reģistrēts.
- **Ja validācija ir neveiksmīga:**
 - Sistēma atgriež kļūdas paziņojumu ar konkrētu problēmu (piemēram, "Kļūda! Lietotājvārds jau eksistē!").
- **Sistēma parāda rezultātu administratoram, ja ir veiksmīga reģistrācija:**
 - Sistēma parāda ziņojumu: "Filiāle ir veiksmīgi reģistrēta!".
- **Sistēma parāda rezultātu lietotājam, ja ir neveiksmīga reģistrācija:**
 - Sistēma atgriež formu ar kļūdas ziņojumiem un pieprasa veikt labojumus.
- **Beigu stāvoklis:**
 - Administratora ievadītie dati ir veiksmīgi saglabāti datubāzē tabulā "users".

- Sistēma ir gatava pieņemt jaunus pieprasījumus vai veikt citas darbības.

6. DATU STRUKTŪRU APRAKSTS

6.1. Datu bāzes apraksts

Datu bāze tika veidota, izmantojot *TablePlus* datu bāzes pārvaldības sistēmu. Tā sastāv no divām galvenajām tabulām: “*users*”(struktūru skat. 1. tabulā.) un “*cars*”(struktūru skat. 2. tabulā.), kas ir savstarpēji saistītas ar ārējās atslēgas palīdzību.

Tabulu relācija: tabulas “*users*” un “*cars*” ir saistītas ar relāciju caur *user_id*, kas nodrošina, ka katram transportlīdzekļa ierakstam ir piesaistīts konkrēts lietotājs.

6.2. Objektu klases

Sistēmā tiek izstrādāta “*user*” klase (struktūru skat. 3. tabulā) un “*user*” metodes (struktūru skat. 4. tabulā), kas pārvalda lietotāja datus un funkcionalitāti *PHP* objektu datu pārvaldībai.

3. tabula “*user*” klases pārvalda lietotāja datus un funkcionalitāti

Atribūti	Atšifrējums
<i>id</i>	Lietotāja unikālais identifikators
<i>username</i>	Lietotāja vārds
<i>password_hash</i>	Lietotāja šifrētā parole
<i>email</i>	Lietotāja e-pasta adrese
<i>created_at</i>	Konta izveides datums

4. tabula “*user*” klases pārvalda lietotāja datus un funkcionalitāti

Metodes	Atšifrējums
<i>createUser()</i> :	Izveido jaunu lietotāju datu bāzē
<i>validateUser()</i> :	Pārbauda lietotāja akreditācijas datus
<i>updateUser()</i> :	Atjauno lietotāja datus

Sistēmā tiek izstrādāta “*car*” klase (struktūru skat. 5. tabulā) un “*car*” metodes (struktūru skat. 6. tabulā), kas pārvalda transportlīdzekļu reģistrāciju un meklēšanu *PHP* objektu datu pārvaldībai.

5. tabula “*car*” klases pārvalda transportlīdzekļu reģistrāciju un meklēšanu

Atribūti	Atšifrējums
<i>id</i>	Transportlīdzekļa unikālais ID
<i>owner_first_name</i>	Īpašnieka vārds
<i>owner_last_name</i>	Īpašnieka uzvārds
<i>license_plate</i>	Transportlīdzekļa reģistrācijas numurs
<i>time_from</i>	Novietošanas sākuma laiks

Atribūti	Atšifrējums
<i>time to</i>	Novietošanas beigu laiks
<i>user id</i>	Lietotāja ID

6. tabula “*car*” klases pārvalda transportlīdzekļu reģistrāciju un meklēšanu

Metodes	Atšifrējums
<i>registerCar()</i> :	Reģistrē jaunu transportlīdzekli
<i>searchCar()</i> :	Atrod transportlīdzekli pēc reģistrācijas numura
<i>updateCar()</i> :	Atjauno transportlīdzekļa datus

6.3. Datu plūsmas process sistēmā

Lietotāja darbība: Sistēma saņem lietotāja ievades datus (reģistrācija, transportlīdzekļa pievienošana).

Datu validācija: Sistēma validē ievades laukus un pārbauda datu precizitāti.

Datu saglabāšana: Apstiprinātie dati tiek saglabāti *TablePlus* datu bāzē.

Datu atjaunināšana un dzēšana: Lietotājs var rediģēt vai dzēst esošos ierakstus.

7. TESTĒŠANA

7.1. Testēšanas kritēriji

Testēšanas kritēriji nosaka prasības, kas jāizpilda, lai uzskatītu sistēmas funkcionalitāti par veiksmīgu. Sistēmas "*ParkingPro*" testēšanas kritēriji ietver šādus aspektus:

Pareiza funkcionalitāte: sistēmas funkcijas darbojas atbilstoši specifikācijām (reģistrācija, pieteikšanās, transportlīdzekļu pievienošana un meklēšana).

Datu validācija: sistēma validē visus lietotāja ievadītos datus un nodrošina atbilstošus kļūdu paziņojumus.

Datu integritāte: ievadītie dati tiek korekti saglabāti datubāzē un netiek bojāti.

Drošība: paroles tiek šifrētas, un lietotāja sensitīvie dati netiek atklāti.

Lietotāja pieredze: sistēma nodrošina skaidru, saprotamu un draudzīgu lietotāja saskarni ar atbilstošiem kļūdu paziņojumiem.

Veiktspēja: sistēma ātri apstrādā pieprasījumus un uzrāda rezultātus.

7.2. Testēšanas rīki un veidi

- **Testēšanai tiek izmantots viens rīks, jeb opcija - manuālā:**
 - Tiek veikta funkcionalitātes manuāla pārbaude, simulējot lietotāja darbības, kā arī tiek pārbaudīta datubāzes integritāte un datu glabāšanas korektums.
- **Funkcionālā testēšana:**
 - Pārbauda, vai visas sistēmas funkcijas darbojas atbilstoši prasībām.
- **Integrācijas testēšana:**
 - Pārbauda mijiedarbību starp sistēmas komponentēm, piemēram, datubāzi un lietotāja saskarni.
- **Validācijas testēšana:**
 - Pārbauda, vai ievades dati tiek korekti validēti (piemēram, e-pasta formāts, telefona numura formāta pārbaude).

7.3. Testpiemēri

7.3.1. Konta reģistrācijas funkcijas testēšana

Reģistrēšanās funkcija nodrošina iespēju administratoriem izveidot kontu sistēmā "*ParkingPro*". Šajā testēšanā tiek pārbaudīta pareiza datu validācija, ievades lauku aizpildes

kontrolē un kļūdu apstrādi. Galvenais uzdevums ir nodrošināt, ka administrators var veiksmīgi reģistrēt ar derīgu informāciju, bet tiek atturēts no reģistrēšanas, ja dati ir kļūdaini vai nepilnīgi.

- **Testa situācijas (struktūru skat. 7. tabulā):**

- Pilnīga un korekta datu ievade.
- Obligāto lauku neaizpildīšana.
- Lietotājvārda dublikāts.

7. tabula Reģistrācijas funkcijas testpiemēri

Testa gadījums	Situācija	Rezultāts
1.	Lietotājs aizpilda visus laukus korekti ar unikālu lietotājvārdu.	Sistēma veiksmīgi reģistrē lietotāju un izveido kontu.
2.	Lietotājs mēģina reģistrēties, neaizpildot kādu no obligātajiem laukiem.	Pie konkrētā lauka parāda kļūdas ziņojumu: <i>"Please fill in this field!"</i> .
3.	Lietotājs ievada lietotājvārdu, kas jau ir reģistrēts datubāzē.	Sistēma parāda kļūdas ziņojumu: <i>"Kļūda! Lietotājvārds jau eksistē!"</i> .

7.3.2. Pieteikšanās funkcijas testēšana

Pieteikšanās funkcija ļauj reģistrētiem lietotājiem piekļūt savam kontam, ievadot lietotājvārdu un paroli. Šajā testēšanā tiek pārbaudīts, vai sistēma veiksmīgi autorizē lietotājus ar pareizu informāciju un pareizi apstrādā kļūdu gadījumus, piemēram, neeksistējošu lietotājvārdu vai nepareizu paroli.

- **Testa situācijas (struktūru skat. 8. tabulā):**

- Pareiza lietotājvārda un paroles ievade.
- Nepareiza parole.
- Neeksistējošs lietotājvārds.

8. tabula Pieteikšanās funkcijas testpiemēri

Testa gadījums	Situācija	Rezultāts
1.	Lietotājs ievada pareizu lietotājvārdu un paroli.	Sistēma veiksmīgi pieslēdz lietotāju sistēmai.
2.	Lietotājs ievada pareizu lietotājvārdu, bet nepareizu paroli.	Sistēma parāda kļūdas ziņojumu: <i>"Pieteikšanās kļūda! Nepareizs lietotājvārds un/vai parole! Problēmu gadījumā sazināties ar Administratoru!"</i> .
3.	Lietotājs ievada lietotājvārdu, kas neeksistē sistēmā.	Sistēma parāda kļūdas ziņojumu: <i>"Pieteikšanās kļūda! Nepareizs lietotājvārds un/vai parole! Problēmu gadījumā sazināties ar Administratoru!"</i> .

7.3.3. Transportlīdzekļa pievienošanas funkcijas testēšana

Šī funkcija ļauj lietotājiem reģistrēt transportlīdzekli sistēmā, aizpildot pieprasīto informāciju (īpašnieka datus, reģistrācijas numuru, transportlīdzekļa marku un novietošanas laiku). Testēšanas mērķis ir nodrošināt, ka sistēma pievieno transportlīdzekli tikai tad, ja dati ir pilnīgi un atbilstoši prasībām, kā arī nodrošina kļūdu apstrādi.

- **Testa situācijas (struktūru skat. 9. tabulā):**
 - Korekta visu datu ievade.
 - Nepilnīga datu ievade.
 - Nepareizs e-pasta formāts.
 - Nepareizs telefona numura formāts.
 - Ievadīts reģistrācijas numurs, kas ir aktīvs konkrētajā laika periodā.
 - Novietošanas laika loģistikas kļūda.

9. tabula Transportlīdzekļa pievienošanas testpiemēri

Testa gadījums	Situācija	Rezultāts
1.	Lietotājs ievada visus nepieciešamos laukus korekti.	Transportlīdzeklis tiek veiksmīgi saglabāts datubāzē.
2.	Lietotājs mēģina pievienot ierakstu ar vienu vai vairākiem neaizpildītiem laukiem.	Sistēma parāda kļūdas ziņojumu pie neaizpildītā lauka: <i>"Please fill in this field"</i> .
3.	Lietotājs ievada e-pastu, kas neatbilst formāta prasībām.	Sistēma parāda kļūdas ziņojumu: <i>"Please include an '@' in the email address."</i> (parādās teksts ko lietotājs ir ievadījis) <i>'is missing an '@'."</i>
4.	Lietotājs ievada ciparus un simbolus vai tikai simbolus telefona numura laukā.	Sistēma parāda kļūdas ziņojumu: <i>"Telefona numuram jābūt tikai cipariem."</i>
5.	Lietotājs ievada transportlīdzekļa reģistrācijas numuru, kas norādītajā laika periodā ir aktīvs.	Sistēma parāda kļūdas ziņojumu: <i>"Transportlīdzekļa reģistrācijas numurs jau ir reģistrēts šajā laika periodā."</i>
6.	Lietotājs vadot laika periodu norāda laiku no jaunāku nekā laiku līdz.	Sistēma parāda kļūdas ziņojumu: <i>"Atstāšanas laikam līdz jābūt vēlākam par atstāšanas laiku no."</i>

7.3.4. Transportlīdzekļa meklēšanas funkcijas testēšana

Meklēšanas funkcija ļauj lietotājiem atrast reģistrētos transportlīdzekļus, ievadot kritērijus, piemēram, reģistrācijas numuru vai īpašnieka datus. Testēšanas mērķis ir pārbaudīt,

vai sistēma atgriež precīzus rezultātus un atbilstošus paziņojumus situācijās, kad dati netiek atrasti vai kritēriji ir kļūdaini.

- **Testa situācijas (struktūru skat. 10. tabulā):**
 - Pareizu datu meklēšana.
 - Meklēšana ar nederīgiem vai tukšiem kritērijiem.

10. tabula Transportlīdzekļa meklēšanas testpiemēri

Testa gadījums	Situācija	Rezultāts
1.	Lietotājs meklē transportlīdzekli, ievadot pareizu reģistrācijas numuru.	Sistēma atgriež meklēto transportlīdzekļa ierakstu.
2.	Lietotājs ievada tukšus vai nederīgus meklēšanas kritērijus.	Sistēma atgriež tukšu tabulu.

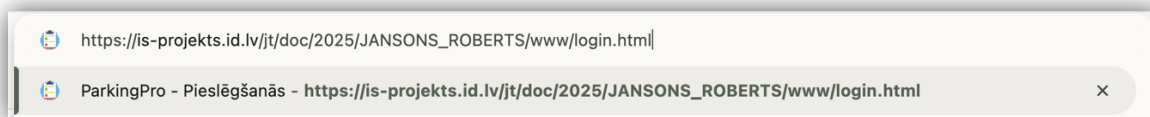
8. LIETOTĀJA CEĻVEDIS

8.1. Sistēmas piekļuve un pieslēgšanās

8.1.1. Sistēmas atvēršana

Lai piekļūtu *ParkingPRO* sistēmai, lietotājiem jābūt pieejamai internetam un sistēmai jādarbojas uzņēmuma iekšējā tīklā. Ja sistēma ir uzstādīta uz vietējā servera, pārliecinieties, ka ir savienojums ar lokālo tīklu.

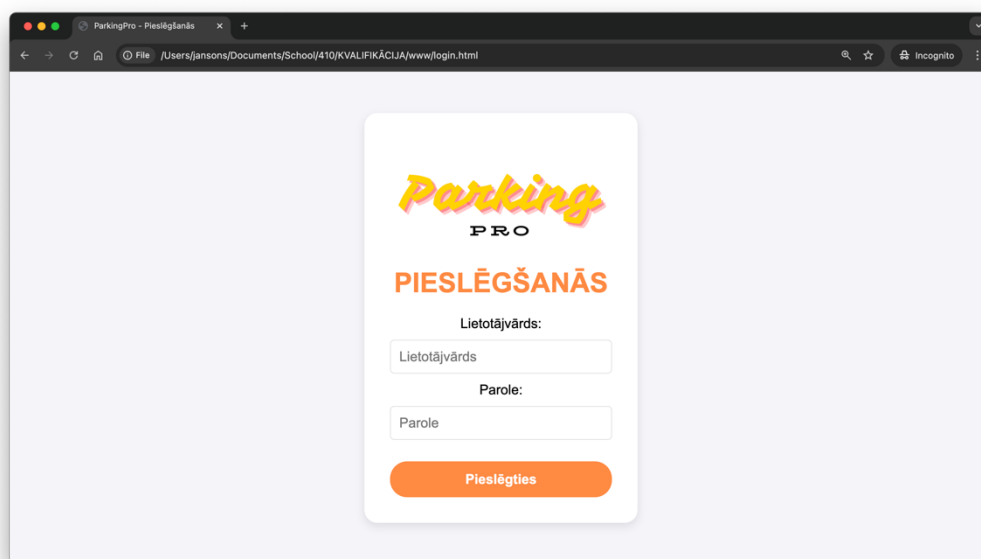
- Atveriet savu tīmekļa pārlūkprogrammu (*Google Chrome*, *Mozilla Firefox* vai citu interneta pārlūkprogrammu).
- Ievadiet pārlūkprogrammas meklēšanas joslā *URL* adresi (skat. 24. att.):
https://is-projekts.id.lv/jt/doc/2025/JANSONS_ROBERTS/www/.



24. att. Pārlūkprogrammas meklēšanas josla

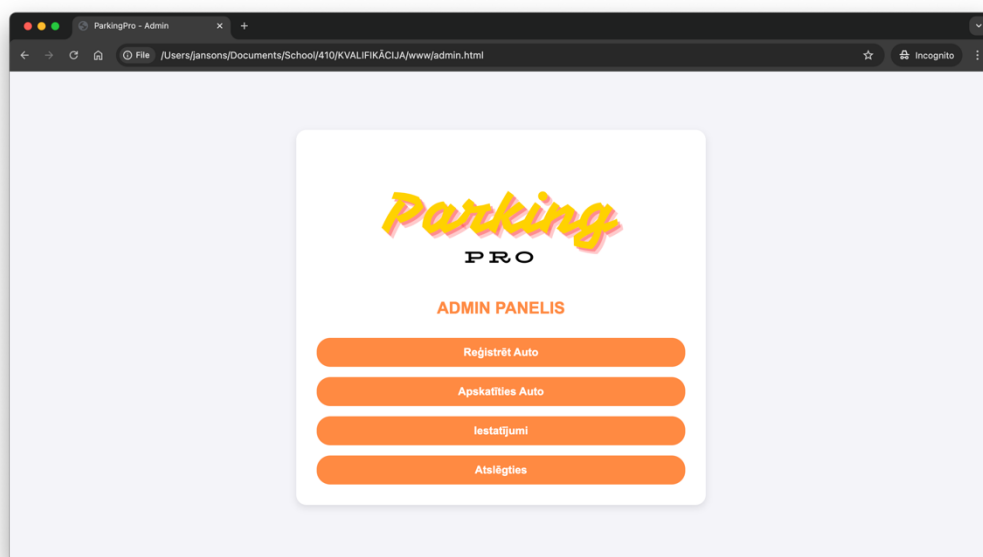
8.1.2. Lietotāja pieslēgšanās

- Ierakstiet savu **Lietotājevārdu** un **Paroli** pieslēgšanās logā (skat. 25.att.).
- Klikšķiniet uz **Pieslēgties**, lai piekļūtu sistēmai.

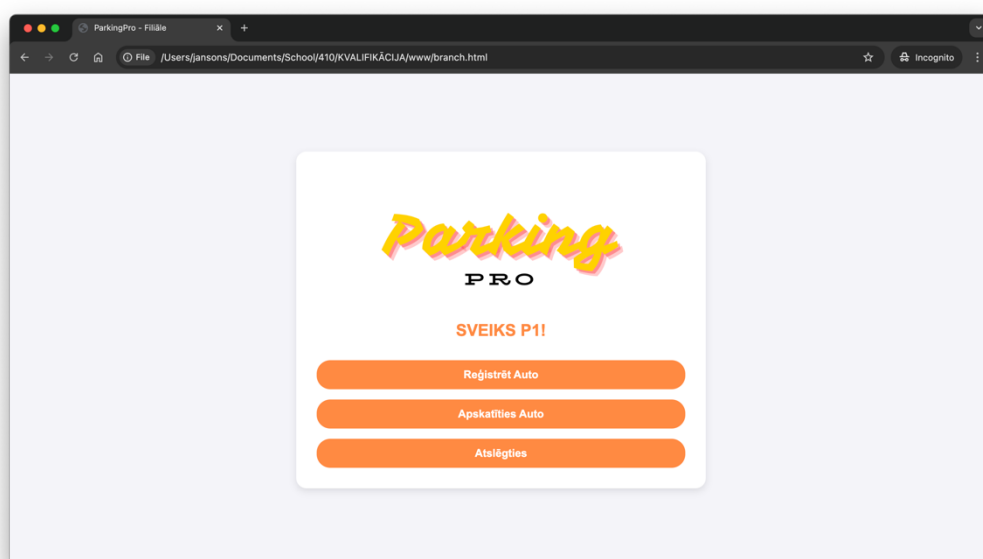


25. att. Pieslēgšanās logs (skat. 1. pielikumā)

- Ja pieteikšanās ir veiksmīga, tiksiet novirzīts vai nu uz administratora sākuma ekrānu (skat. 26.att.), vai filiāles sākuma ekrānu (skat. 27.att.), kur varēsiet redzēt visas pieejamās funkcijas priekš konkrētā lietotāja.



26. att. Administratora sākuma logs (skat. 6. pielikumā)



27. att. Filiāles sākuma logs (skat. 27. pielikumā)

8.2. Transportlīdzekļa reģistrēšana

8.2.1. Jauna transportlīdzekļa reģistrēšana

Lai reģistrētu jaunu transportlīdzekli sistēmā:

- **Pieslēdzieties sistēmai** ar savu administratora vai datu ievades speciālista (filiāles) profilu.
- Pēc pieslēgšanās ekrāna redzēsiet pogu **Reģistrēt transportlīdzekli**.
- Noklikšķiniet uz šīs pogas, lai atvērtu reģistrācijas veidlapu.

8.2.2. Reģistrācijas datu ievade

Aizpildiet nepieciešamos laukus (skat 28. att.):

- **Informācija par klientu:**
 - Klienta vārds
 - Klienta uzvārds
 - Klienta telefona numurs
 - Klienta e-pasta adrese
- **Informācija par transportlīdzekli:**
 - Transportlīdzekļa reģistrācijas numurs
 - Transportlīdzekļa modelis
 - Novietošanas laiks (no un līdz)

Kad visi lauki ir aizpildīti, noklikšķiniet uz **Saglabāt**.

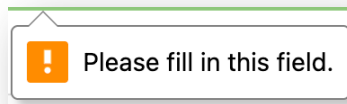
Ja vēlaties darbību atcelt, noklikšķiniet uz **Atcelt**.

28. att. Transportlīdzekļa reģistrācijas logs (skat. 7., 8., 29. un 30. pielikumā)

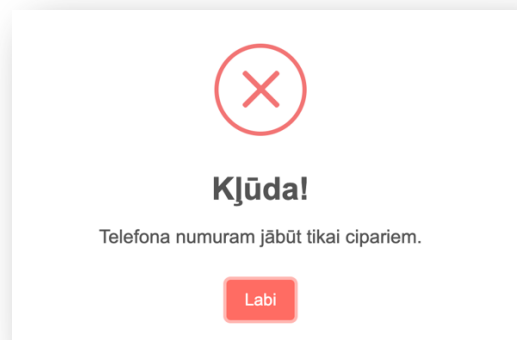
8.2.3. Pārbaudes un kļūdas ziņojumi

Ja kāds no laukiem ir atstāts tukšs, sistēma parādīs paziņojumu, *"Please fill in this field."* (skat. 29. att.).

Ja kādam no laukiem ir sava veida modifikators kā tieši jāaizpilda lauks, bet tas ir aizpildīts nepareizi saglabājot parādās ziņojums (piemēram, ja telefona numura laukā tiek ievadīti simboli (skat. 30. att.))



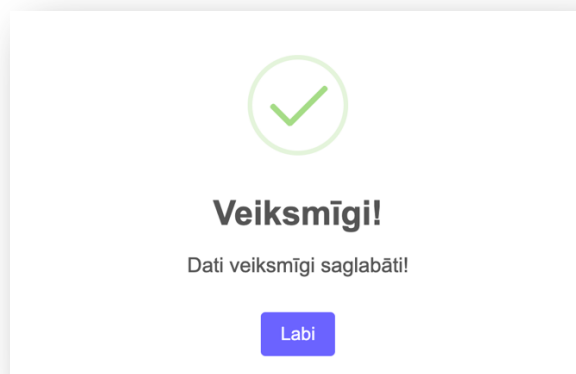
29. att. Kļūdas paziņojums, ja kāds no laukiem nav aizpildīts



30. att. Kļūdas paziņojums, ja kāds no laukiem ir ievadīts nepareizi

8.2.4. Reģistrēšanas pabeigšana

Kad transportlīdzekļa dati ir saglabāti, sistēma parādīs paziņojumu **"Dati veiksmīgi saglabāti!"**, un jaunais transportlīdzeklis tiks pievienots sistēmai (skat. 31. att.).



31. att. Paziņojums, ka dati ir saglabāti datubāzē

8.3. Transportlīdzekļa datu rediģēšana un dzēšana

8.3.1. Kā rediģēt transportlīdzekļa datus

- Pieslēdzieties sistēmai.
- Dodieties uz **Reģistrētie transportlīdzekļi** sadaļu, kur būs pieejama visu reģistrēto transportlīdzekļu saraksts.
- Izvēlieties transportlīdzekli, kuru vēlaties rediģēt.
- Noklikšķiniet uz **Rediģēt** blakus transportlīdzekļa ierakstam (skat. 32.att.).
- Veiciet nepieciešamās izmaiņas un pēc tam noklikšķiniet uz **Saglabāt**, lai saglabātu izmaiņas.

REĢISTRĒTIE TRANSPORTLĪDZEKĻI								
Šobrīd aktīvs				Plānotais		Vāsture		Visa
Vārds	Uzvārds	E-pasts	Telefona numurs	Numura zīme	Auto modelis	Atstāšanas laiks no	Atstāšanas laiks līdz	Darbības
t	t	t@t	3	t	t	2025-04-23 12:05:00	2025-04-30 12:05:00	Rediģēt Dzēst

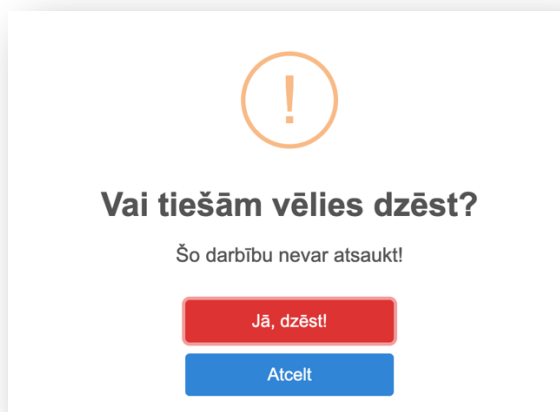
1

Atpakaļ

32. att. Transportlīdzekļa apskatīšanas/meklēšanas/rediģēšanas/dzēšanas logs (skat. 10., 12., 31. un 32. pielikumā)

8.3.2. Kā dzēst transportlīdzekli no sistēmas

- Izvēlieties transportlīdzekli, kuru vēlaties izdzēst.
- Noklikšķiniet uz **Dzēst** pogas (skat. 32.att.).
- Sistēma jautās vai tiešām vēlaties izdzēst (skat. 33. att.).
- Sistēma apstiprinās dzēšanas darbību ar paziņojumu "**Automašīna ir veiksmīgi izdzēsta!**".



33. att. Dzēšanas paziņojums

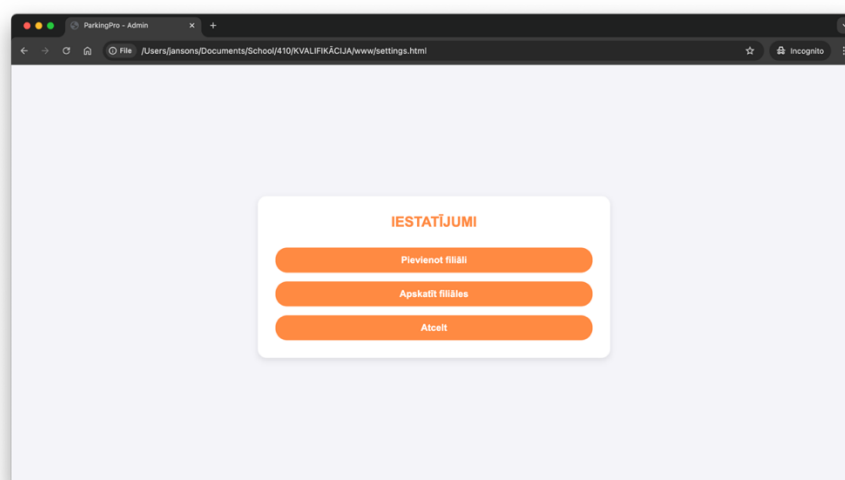
8.4. Transportlīdzekļu Meklēšana

- Lai meklētu transportlīdzekli, izmantojiet Meklēšanas joslu augšējā ekrāna malā (skat. 32.att.).
- Ievadiet reģistrācijas numuru.
- Sistēma atgriezīs visu transportlīdzekļu sarakstu, kas atbilst jūsu meklētajiem kritērijiem.

8.5. Filiāles pārvaldība un iestatījumi

8.5.1. Kā pievienot jaunu filiāli

- Administrators dodas uz **Iestatījumi** (skat. 34.att.).



34. att. Iestatījumu logs (skat. 16. pielikumā)

- Izvēlieties **Pievienot Filiāli** un aizpildiet nepieciešamos laukus (skat. 35.att.):
 - Filiāles nosaukums
 - Filiāles lietotājvārds
 - Filiāles parole

35. att. Filiāles konta pievienošana (skat. 18. pielikumā)

Kad visi lauki ir aizpildīti, noklikšķiniet uz **Saglabāt**, lai pievienotu jaunu filiāli. Ja tomēr darbību vēlas atcelt, tad noklikšķiniet uz **Atcelt**.

8.5.2. Kā rediģēt filiāles datus

- Pieslēdzieties kā **administrators**.
- Dodieties uz **Apskatīt Filiāli** (skat. 36.att.).
- Izvēlieties filiāli, kuru vēlaties rediģēt, un noklikšķiniet uz **Rediģēt**.

FILIĀLE	LIETOTĀJVārds	PAROLE	Rediģēt	Dzēst
P1	p1@parking.lv	parking	Rediģēt	Dzēst
P2	p2@parking.lv	parking	Rediģēt	Dzēst

36. att. Profilu apskates/meklēšanas/rediģēšanas/dzēšanas logs (skat. 22. pielikumā)

8.6. Drošība un datu aizsardzība

ParkingPRO sistēma nodrošina augstu drošības līmeni, lai aizsargātu lietotāju un transportlīdzekļu datus:

- Datu šifrēšana: Visas paroles un sensitīvie dati tiek šifrēti.
- Pieejamība un drošība: Sistēma tiek regulāri pārbaudīta un atjaunināta, lai nodrošinātu tās drošību un stabilitāti.

8.6.1. Sistēmas drošība un datu aizsardzība

Visi dati tiek aizsargāti ar šifrēšanu un uzglabāti drošos serveros. Datu apstrāde tiek veikta saskaņā ar *GDPR* prasībām.

8.7. Sistēmas interfeiss un darbība

8.7.1. Lietotāja interfeiss

Sistēma ir izstrādāta, lai būtu lietotājam draudzīga un intuitīva. Galvenās izvēlnes un pogas ir izvietotas loģiskā secībā, kas ļauj lietotājiem ātri piekļūt nepieciešamajām funkcijām (skat. 26., 27.att.).

8.7.2. Navigācijas ērtības

Navigācija ir vienkārša, izmantojot lielas, viegli sasniedzamas pogas un izkārtotās izvēlnes.

8.7.3. Atsaucīga dizaina iespējas

Sistēma tiek pielāgota visām ierīcēm, nodrošinot ērtu lietošanas pieredzi gan uz datoriem, gan mobilajām ierīcēm.

SECINĀJUMI UN PRIEKŠLIKUMI

1. Projekta rezultātā izstrādāta moderna un lietotājam draudzīga sistēma, kas ļauj uzņēmumam vienkārši un efektīvi reģistrēt, uzglabāt un pārvaldīt klientu un transportlīdzekļu informāciju. Sistēmas intuitīvais interfeiss ļauj lietotājiem ātri un precīzi piekļūt nepieciešamajām funkcijām.
2. Sistēma nodrošina drošu datu apstrādi un uzglabāšanu, ievērojot *GDPR* prasības, kas ir svarīgi, lai garantētu lietotāju datu aizsardzību. Visi sensitīvie dati tiek šifrēti, un sistēma tiek regulāri pārbaudīta, lai nodrošinātu drošību un stabilitāti.
3. Projekta realizācija balstīta uz rūpīgi izstrādātu plānu, kurā ietverts elastīgs grafiks un skaidras prasības. Šī pieeja ļāva veiksmīgi identificēt risinājumus un efektīvi pārvaldīt darba procesus, nodrošinot savlaicīgu projekta pabeigšanu.
4. Sistēmas attīstības laikā iegūtās zināšanas un pieredze, piemēram, darbā ar jaunām tehnoloģijām, ir veicinājušas izstrādātāja profesionālo izaugsmi un kompetences.
5. Lai uzlabotu sistēmas funkcionalitāti un lietojamību, ieteicams nākotnē ieviest papildu funkcijas, piemēram, automatizētu atgādinājumu sistēmu par stāvvietas termiņa beigām, paplašinātu datu meklēšanas iespējas un reāllaika atskaišu ģenerēšanu.
6. Lai nodrošinātu starptautisku izmantojamību, būtu lietderīgi ieviest daudzvalodības atbalstu un pielāgot sistēmu dažādu valstu likumdošanas prasībām.
7. Regulāras drošības pārbaudes un jauninājumi palīdzēs saglabāt sistēmas uzticamību un lietotāju uzticību, novēršot iespējamus riskus un drošības apdraudējumus.

IZMANTOTO AVOTU SARAKSTS

- [1] *CSS Introduction* [Tiešsaite] Pieejams: https://www.w3schools.com/css/css_intro.asp [Skatīts 20-Nov-2024]
- [2] *Database management made easy* [Tiešsaite] Pieejams: <https://tableplus.com/> [Skatīts 20-Nov-2024]
- [3] *dbdiagram.io* 2024 [Tiešsaite] Pieejams: <https://dbdiagram.io/home> [Skatīts 10-Dec-2024]
- [4] *europark.lv* 2024 [Tiešsaite] Pieejams: <https://www.europark.lv/> [Skatīts 9-Okt-2024]
- [5] *HTML Introduction* [Tiešsaite] Pieejams: https://www.w3schools.com/html/html_intro.asp [Skatīts 20-Nov-2024]
- [6] *lucid.co* 2024 [Tiešsaite] Pieejams: <https://lucid.co> [Skatīts 19-Nov-2024]
- [7] *mobilly.lv* 2024 [Tiešsaite] Pieejams: <https://mobilly.lv/> [Skatīts 9-Okt-2024]
- [8] *parksmart.lv* 2024 [Tiešsaite] Pieejams: <https://www.parksmart.lv/lv> [Skatīts 6-Nov-2024]
- [9] *parksmart.lv* 2024 [Tiešsaite] Pieejams: https://www.parksmart.lv/lv/how_it_works [Skatīts 6-Nov-2024]
- [10] *PHP Introduction* [Tiešsaite] Pieejams: https://www.w3schools.com/php/php_intro.asp [Skatīts 20-Nov-2024]
- [11] *scanva.com* [Tiešsaite] Pieejams: <https://canva.com> [Skatīts 24-Sep-2024]
- [12] *What is JavaScript?* [Tiešsaite] Pieejams: https://www.w3schools.com/whatis/whatis_js.asp [Skatīts 20-Nov-2024]
- [13] *Why did we build Visual Studio Code?* [Tiešsaite] <https://code.visualstudio.com/docs/editor/whyvscode> [Skatīts 20-Nov-2024]
- [14] *XAMPP* [Tiešsaite] Pieejams: <https://en.wikipedia.org/wiki/XAMPP> [Skatīts 20-Nov-2024]

PIELIKUMI

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8 -->
  <meta name="viewport" content="width=device-width, initial-
scale=1.0"> <!-- Pielāgo lapas izskatu mobilajām ierīcēm -->
  <title>ParkingPRO - Pieteikšanās</title> <!-- Lapas nosaukums, kas
parādās pārlūkprogrammas cilnē -->
  <!-- Ikona, kas tiek attēlota pārlūkprogrammas cilnē -->
  <link rel="icon" type="image/png" href="img/pic.png">
  <!-- Stila lapa, kas nosaka lapas dizainu -->
  <link rel="stylesheet" href="css/login.css">
  <!-- Pievieno SweetAlert2 bibliotēku, lai izmantotu uzlabotus
paziņojumus -->
  <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
</head>
<body>
  <!-- Kontainers, kas satur pieteikšanās formu -->
  <div class="login-container">
    <!-- Uzņēmuma logotips -->
    
    <!-- Virsraksts pieteikšanās sadaļai -->
    <div class="header">
      <h1>Pieteikšanās</h1>
    </div>
    <!-- Pieteikšanās forma, kas nosūta ievadītos datus uz
"php/login.php" -->
    <form id="login-form" action="php/login.php" method="POST">
      <!-- Lietotājvārda ievades lauks -->
      <input type="text" id="username" name="username"
placeholder="Lietotājvārds" required autocomplete="off">
      <!-- Paroles ievades lauks -->
      <input type="password" id="password" name="password"
placeholder="Parole" required>
      <!-- Poga, lai iesniegtu formu -->
      <button type="submit">Pieteikties</button>
    </form>
  </div>
  <!-- JavaScript fails, kas pārvalda interaktivitāti -->
  <script src="js/index.js"></script>
</body>
</html>
```

2. pielikums **index.js**

```
// Izveido URL parametru objektu, lai varētu pārbaudīt un iegūt vērtības no
pašreizējā URL
const urlParams = new URLSearchParams(window.location.search);
// Pārbauda, vai URL satur parametru 'error' un tā vērtība ir '1'
if (urlParams.has('error') && urlParams.get('error') === '1') {
    // Izsauc SweetAlert2 paziņojumu, lai informētu lietotāju par
    pieteikšanās kļūdu
    Swal.fire({
        icon: 'error', // Parāda sarkanu kļūdas ikonu
        title: 'Pieteikšanās kļūda!', // Paziņojuma virsraksts
        text: 'Nepareizs lietotājvārds un/vai parole! Problēmu gadījumā
sazināties ar Administratoru!', // Paziņojuma teksts
        confirmButtonText: 'Labi', // Apstiprināšanas pogas teksts
        // Pievieno pielāgotu CSS klasi apstiprināšanas pogai
        customClass: {
            confirmButton: 'custom-swal-button' // Šī klase var būt
definēta ārējā CSS failā
        },
        // Izslēdz SweetAlert2 noklusējuma stilu pogām (ja pielāgota klase
        tiek izmantota)
        buttonsStyling: false
    });
}
```

3. pielikums **login.css**

```
/* Atiestata noklusējuma margin un padding visiem elementiem un nodrošina
vienotu kastes izmēru aprēķinu */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box; /* Nodrošina, ka padding un border nepalielina
elementa kopējo izmēru */
}
/* Stilizē body (galveno lapas sadaļu) */
body {
    display: flex; /* Izmanto flexbox, lai centrētu saturu */
    justify-content: center; /* Centrē saturu horizontāli */
    align-items: center; /* Centrē saturu vertikāli */
    min-height: 100vh; /* Nodrošina, ka lapas augstums ir vismaz 100% no
skatloga augstuma */
    background-color: #f4f4f9; /* Gaiši pelēka fona krāsa */
    font-family: Arial, sans-serif; /* Norāda fontu, ko izmantot visā lapā
*/
}
/* Kontainers pieteikšanās formai */
.login-container {
    width: 320px; /* Formas platums */
    padding: 30px; /* Iekšējā atstarpe */
    background-color: #fff; /* Baltas krāsas fons */
    border-radius: 15px; /* Noapaļoti stūri */
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); /* Viegls ēnojums, kas
piešķir 3D efektu */
    text-align: center; /* Centrē tekstu konteinerī */
}
/* Attēla stilizēšana formā */
.login-container img {
    max-width: 100%; /* Nodrošina, ka attēls neiziet ārpus konteinerā
platuma */
    margin-bottom: 20px; /* Apakšējā atstarpe no nākamā elementa */
}
/* Galvenes sekcija (piemēram, virsraksts) */
.header {
    color: #ff8a42; /* Oranža krāsa */
    margin-bottom: 20px; /* Atstarpe zem galvenes */
}
/* Galvenes virsraksts */
.header h1 {
    font-size: 32px; /* Liels fonta izmērs */
    font-weight: bold; /* Padara tekstu treknrakstā */
}
/* Teksta ievades lauki (lietotājvārds un parole) */
input[type="text"],
input[type="password"] {
    width: 100%; /* Ievades lauks aizņem visu pieejamo platumu */
    padding: 10px; /* Iekšējā atstarpe */
    margin: 10px 0; /* Atstarpes starp laukiem */
    border: 1px solid #ddd; /* Pelēka apmale */
    border-radius: 5px; /* Noapaļoti stūri */
    font-size: 16px; /* Vidēja izmēra fonts */
}
/* Stilizē pogu */
button {
    width: 100%; /* Poga aizņem visu pieejamo platumu */
    padding: 12px; /* Iekšējā atstarpe */
    margin-top: 15px; /* Atstarpe virs pogas */
}
```

```

background-color: #ff8a42; /* Oranža fona krāsa */
color: #fff; /* Balta teksta krāsa */
font-size: 16px; /* Vidēja izmēra fonts */
font-weight: bold; /* Padara tekstu treknrakstā */
border: none; /* Noņem apmali */
border-radius: 20px; /* Noapaļoti stūri */
cursor: pointer; /* Maina kursoru uz pointeri, lai norādītu, ka poga ir
klikšķināma */
transition: background-color 0.3s ease; /* Nodrošina gludu pāreju,
mainot pogas krāsu */
}
/* Kad lietotājs uzbrauc ar peli uz pogas, tā maina krāsu */
button:hover {
background-color: #ff7a30; /* Tumšāka oranža krāsa */
}
/* Pielāgota SweetAlert2 poga */
.custom-swal-button {
background-color: #ff8a42; /* Oranža poga */
color: #fff; /* Baltas krāsas teksts */
font-size: 16px; /* Vidēja izmēra fonts */
font-weight: bold; /* Padara tekstu treknrakstā */
border: none; /* Noņem apmali */
border-radius: 20px; /* Noapaļoti stūri */
padding: 10px 20px; /* Iekšējā atstarpe */
cursor: pointer; /* Maina kursoru uz pointeri */
transition: background-color 0.3s ease; /* Gluda pāreja, mainot fonu */
}
/* Kad lietotājs uzbrauc ar peli uz SweetAlert2 pogas */
.custom-swal-button:hover {
background-color: #ff7a30; /* Tumšāka oranža krāsa */
}

```

```

<?php
// Sāk sesiju, lai pārvaldītu lietotāja pieteikšanās datus
session_start();
// Iekļauj datubāzes savienojuma failu
include('../php/db.php');
// Iegūst un attīra lietotājvārdu un paroli no POST pieprasījuma
$user = trim($_POST['username']); // Noņem liekās atstarpes no
lietotājvārda
$pass = trim($_POST['password']); // Noņem liekās atstarpes no paroles
// Pārbauda, vai lietotājvārds un parole nav tukši
if (empty($user) || empty($pass)) {
    // Ja kāds no laukiem ir tukšs, pāradresē uz pieteikšanās lapu ar
    kļūdas parametru
    header("Location: ../login.html?error=1");
    exit();
}
// Sagatavo SQL vaicājumu, lai pārbaudītu, vai lietotājvārds un parole ir
datubāzē
$sql = "SELECT id, branch_name, username FROM branches WHERE username = ?
AND password = ?";
$stmt = $conn->prepare($sql);
// Piesaista lietotājvārdu un paroli SQL vaicājumam, izmantojot sagatavoto
vaicājumu (drošības nolūkos)
$stmt->bind_param("ss", $user, $pass);
// Izpilda vaicājumu
$stmt->execute();
// Iegūst rezultātus
$result = $stmt->get_result();
// Pārbauda, vai lietotājs tika atrasts datubāzē
if ($result->num_rows > 0) {
    // Iegūst lietotāja informāciju no datubāzes
    $row = $result->fetch_assoc();
    // Saglabā lietotāja datus sesijā
    $_SESSION['branch_name'] = $row['branch_name']; // Saglabā filiāles
nosaukumu
    $_SESSION['username'] = $row['username']; // Saglabā lietotājvārdu
    $_SESSION['user_id'] = $row['id']; // Saglabā lietotāja ID
    $_SESSION['branch_id'] = $row['id']; // Saglabā filiāles ID (ja
nepieciešams)
    // Pārbauda, vai lietotājs ir administrators (id = 1)
    if ($row['id'] == 1) {
        // Ja lietotājs ir administrators, pāradresē uz administrācijas
paneli
        header("Location: ../admin.html");
    } else {
        // Ja lietotājs ir parasts filiāles lietotājs, pāradresē uz
filiāles paneli
        header("Location: ../branch.php");
    }
    exit(); // Pārtrauc skripta izpildi pēc pāradresācijas
} else {
    // Ja lietotājs netika atrasts vai parole ir nepareiza, pāradresē
atpakaļ uz pieteikšanās lapu ar kļūdas parametru
    header("Location: ../login.html?error=1");
    exit();
}
// Aizver sagatavoto vaicājumu un aizver datubāzes savienojumu
$stmt->close();
$conn->close();
?>

```

5. pielikums **admin.css**

```

/* Atiestata noklusējuma margin un padding visiem elementiem un nodrošina
vienotu kastes izmēru aprēķinu */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box; /* Nodrošina, ka padding un border nepalielina
elementa kopējo izmēru */
}
/* Stilizē `body` (galveno lapas sadaļu) */
body {
    display: flex; /* Izmanto flexbox, lai centrētu saturu */
    justify-content: center; /* Centrē saturu horizontāli */
    align-items: center; /* Centrē saturu vertikāli */
    min-height: 100vh; /* Nodrošina, ka lapas augstums ir vismaz 100% no
skatloga augstuma */
    background-color: #f4f4f9; /* Pelēkzila fona krāsa */
    font-family: Arial, sans-serif; /* Norāda, kādu fontu izmantot visā
lapā */
}
/* Konteiners, kas ietver pieteikšanās vai reģistrācijas formu */
.form-container {
    width: 350px; /* Konteinera platums */
    padding: 30px; /* Iekšējā atstarpe */
    background-color: #fff; /* Baltas krāsas fons */
    border-radius: 10px; /* Noapaļoti stūri */
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); /* Viegls ēnojums, kas
piešķir 3D efektu */
    text-align: center; /* Centrē tekstu konteinerī */
}
/* Attēla stilizēšana formā */
.form-container img {
    max-width: 100%; /* Nodrošina, ka attēls neiziet ārpus konteinera
platuma */
    margin-bottom: 20px; /* Atstarpe zem attēla */
}
/* Formas virsraksts */
.form-container h2 {
    color: #333; /* Tumši pelēka krāsa */
    font-size: 24px; /* Vidēja izmēra fonts */
    font-weight: bold; /* Padara tekstu treknrakstā */
    margin-bottom: 20px; /* Apakšējā atstarpe no nākamā elementa */
}
/* Navigācijas pogu konteiners */
.navigation button {
    width: 100%; /* Poga aizņem visu pieejamo platumu */
    padding: 12px; /* Iekšējā atstarpe */
    margin-bottom: 15px; /* Apakšējā atstarpe starp pogām */
    background-color: #ff8a42; /* Oranža fona krāsa */
    color: #fff; /* Balta teksta krāsa */
    font-size: 16px; /* Vidēja izmēra fonts */
    font-weight: bold; /* Padara tekstu treknrakstā */
    border: none; /* Noņem apmali */
    border-radius: 5px; /* Noapaļoti stūri */
    cursor: pointer; /* Maina kursoru uz pointeri, lai norādītu, ka poga ir
klikšķināma */
    transition: background-color 0.3s ease; /* Nodrošina gludu pāreju,
mainot pogas krāsu */
}
/* Kad lietotājs uzbrauc ar peli uz pogas, tā maina krāsu */
.navigation button:hover {

```

```
background-color: #ff7a30; /* Tumšāka oranža krāsa */  
}
```

6. pielikums **admin.html**

```

<!DOCTYPE html>
<html lang="lv"> <!-- Dokumenta valoda ir latviešu -->
<head>
    <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8, lai
atbalstītu latviešu valodu -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
<!-- Nodrošina responsīvu dizainu mobilajām ierīcēm -->
    <title>PPRO - admin</title> <!-- Lapas nosaukums, kas parādās
pārlūkprogrammas cilnē -->
    <!-- Favicon ikona, kas tiek attēlota pārlūkprogrammas cilnē -->
    <link rel="icon" type="image/png" href="img/pic.png">
    <!-- Pievieno ārējo CSS failu, kas nosaka lapas dizainu -->
    <link rel="stylesheet" href="css/styles.css">
</head>
<body>
    <!-- Kontainers, kas satur administrācijas paneli -->
    <div class="form-container">
        <!-- Uzņēmuma logotips -->
        
        <!-- Lapas virsraksts -->
        <h2>ADMIN PANELIS</h2>
        <!-- Navigācijas sadaļa ar pogām dažādām administrācijas funkcijām
-->
        <div class="navigation">
            <!-- Poga, lai reģistrētu jaunu transportlīdzekli -->
            <button
onclick="window.location.href='registeradmin.php'">Reģistrēt
transportlīdzekli</button>
            <!-- Poga, lai apskatītu visus reģistrētos transportlīdzekļus -
->
            <button
onclick="window.location.href='viewadmin.php'">Reģistrētie
transportlīdzekļi</button>
            <!-- Poga, lai piekļūtu iestatījumiem -->
            <button
onclick="window.location.href='settings.html'">Iestatījumi</button>
            <!-- Poga, lai izrakstītos no administrācijas paneļa -->
            <button
onclick="window.location.href='php/logout.php'">Atslēgties</button>
            </div>
        </div>
    </body>
</html>

```

7. pielikums **registeradmin.php**

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Iegūst ziņojumu un tā tipu no sesijas, ja tie ir iestatīti
$message = $_SESSION['message'] ?? null;
$messageType = $_SESSION['message_type'] ?? null;
// Iegūst saglabātos ievades laukus no sesijas gadījumā, ja forma tiek
atkārtoti ielādēta pēc kļūdas
$name = $_SESSION['form_data']['name'] ?? '';
$surname = $_SESSION['form_data']['surname'] ?? '';
$email = $_SESSION['form_data']['email'] ?? '';
$phone = $_SESSION['form_data']['phone'] ?? '';
$car_number = $_SESSION['form_data']['car_number'] ?? '';
$car_model = $_SESSION['form_data']['car_model'] ?? '';
$start_time = $_SESSION['form_data']['start_time'] ?? '';
$end_time = $_SESSION['form_data']['end_time'] ?? '';
// Ja ziņojuma tips ir "success", noņemam saglabātos ievades datus, jo tie
vairs nav vajadzīgi
if ($messageType === 'success') {
    unset($_SESSION['form_data']);
}
// Attīra sesijas ziņojumu un ziņojuma tipu, lai tie netiktu parādīti
atkārtoti
unset($_SESSION['message'], $_SESSION['message_type']);
?>
<!DOCTYPE html>
<html lang="lv">
<head>
    <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8 -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- Pielāgo lapas izmēru mobilajām ierīcēm -->
    <title>PPRO - admin</title> <!-- Lapas nosaukums, kas parādīsies
pārlūkprogrammas cilnē -->
    <!-- Ikona, kas tiek attēlota pārlūkprogrammas cilnē -->
    <link rel="icon" type="image/png" href="img/pic.png">
    <!-- SweetAlert2 bibliotēka vizuāliem paziņojumiem -->
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/sweetalert2@11/dist/sweetalert2.min.css"
>
    <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
    <!-- Stila lapa, kas nosaka lapas vizuālo izskatu -->
    <link rel="stylesheet" href="css/reg.css">
</head>
<body>
    <?php if ($message && $messageType === 'success'): ?>
        <!-- Ja ir veiksmīgs ziņojums, parādīt SweetAlert paziņojumu -->
        <script>
            Swal.fire({
                icon: 'success',
                title: 'Veiksmīgi!',
                text: '<?php echo htmlspecialchars($message); ?>',
                confirmButtonText: 'Labi',
                confirmButtonColor: '#6c63ff',
            }).then((result) => {
                if (result.isConfirmed) {
                    window.location.href = 'viewadmin.php'; // Pāradresē uz
administrācijas skata lapu pēc apstiprinājuma
                }
            });
        </script>
    </body>

```

```

<?php elseif ($message && $messageType === 'error'): ?>
    <!-- Ja ir kļūdas ziņojums, parādīt SweetAlert paziņojumu -->
    <script>
        Swal.fire({
            icon: 'error',
            title: 'Kļūda!',
            text: '<?php echo htmlspecialchars($message); ?>',
            confirmButtonText: 'Labi',
            confirmButtonColor: '#ff6c63',
        });
    </script>
<?php endif; ?>
<!-- Kontainers, kas satur transportlīdzekļa reģistrācijas formu -->
<div class="form-container">
    <h2>REĢISTRĒT TRANSPORTLĪDZEKLI</h2>
    <!-- Forma, kas nosūta datus uz savecar.php -->
    <form action="php/savecar.php" method="POST">
        <div class="form-grid">
            <!-- Īpašnieka informācijas sadaļa -->
            <div class="owner-info">
                <h3>Informācija par īpašnieku</h3>
                <label for="name">Vārds:</label>
                <input type="text" id="name" name="name"
placeholder="Ievadiet vārdu" value="<?php echo htmlspecialchars($name); ?>"
autocomplete="off" required>
                <label for="surname">Uzvārds:</label>
                <input type="text" id="surname" name="surname"
placeholder="Ievadiet uzvārdu" value="<?php echo
htmlspecialchars($surname); ?>" autocomplete="off" required>
                <label for="email">E-pasts:</label>
                <input type="email" id="email" name="email"
placeholder="Ievadiet e-pasta adresi" value="<?php echo
htmlspecialchars($email); ?>" autocomplete="off" required>

                <label for="phone">Telefona numurs:</label>
                <input type="tel" id="phone" name="phone"
placeholder="Ievadiet telefona numuru" value="<?php echo
htmlspecialchars($phone); ?>" autocomplete="off" required>
            </div>
            <!-- Auto informācijas sadaļa -->
            <div class="car-info">
                <h3>Informācija par auto</h3>
                <label for="car_number">Numura zīme:</label>
                <input type="text" id="car_number" name="car_number"
placeholder="Ievadiet numura zīmi" value="<?php echo
htmlspecialchars($car_number); ?>" autocomplete="off" required>
                <label for="car_model">Marka:</label>
                <input type="text" id="car_model" name="car_model"
placeholder="Ievadiet auto marku" value="<?php echo
htmlspecialchars($car_model); ?>" autocomplete="off" required>
                <label for="start_time">Atstāšanas laiks no:</label>
                <input type="datetime-local" id="start_time"
name="start_time" value="<?php echo htmlspecialchars($start_time); ?>"
autocomplete="off" required>
                <label for="end_time">Atstāšanas laiks līdz:</label>
                <input type="datetime-local" id="end_time"
name="end_time" value="<?php echo htmlspecialchars($end_time); ?>"
autocomplete="off" required>
            </div>
        </div>
    <!-- Pogu sadaļa -->

```

```
        <button type="submit">Saglabāt</button> <!-- Nosūta datus uz
serveri -->
        <button type="button"
onclick="window.location.href='admin.html'">Atpakaļ</button> <!-- Atgriežas
atpakaļ uz administrācijas paneli -->
    </form>
</div>
</body>
</html>
```

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Iekļauj datubāzes savienojuma failu
include('db.php');
// Pārbauda, vai lietotājs ir pieteicies
if (!isset($_SESSION['username'])) {
    // Ja lietotāja sesijas dati nav pieejami, saglabā kļūdas ziņojumu un
    // pāradresē uz reģistrācijas lapu
    $_SESSION['message'] = 'Lietotāja identifikators nav atrasts. Lūdzu,
    piesakieties vēlreiz.';
    $_SESSION['message_type'] = 'error';
    header('Location: ../registeradmin.php');
    exit();
}
// Saglabā pašreizējā lietotāja lietotājevārdu
$username = $_SESSION['username'];
// Iegūst un attīra ievades datus no POST pieprasījuma, izmantojot `trim()`
// nevajadzīgu atstarpju noņemšanai
$name = trim($_POST['name'] ?? '');
$surname = trim($_POST['surname'] ?? '');
$email = trim($_POST['email'] ?? '');
$phone = trim($_POST['phone'] ?? '');
$car_number = trim($_POST['car_number'] ?? '');
$car_model = trim($_POST['car_model'] ?? '');
$start_time = $_POST['start_time'] ?? '';
$end_time = $_POST['end_time'] ?? '';
// Masīvs, kas saglabās validācijas kļūdas
$errors = [];
// Pārbauda, vai visi lauki ir aizpildīti
if (empty($name) || empty($surname) || empty($email) || empty($phone) ||
    empty($car_number) || empty($car_model) || empty($start_time) ||
    empty($end_time)) {
    $errors[] = 'Visi lauki ir obligāti jāaizpilda.';
}
// Pārbauda, vai telefona numurs satur tikai ciparus
if (!preg_match('/^\d+$/', $phone)) {
    $errors[] = 'Telefona numuram jābūt tikai cipariem.';
}
// Pārbauda, vai atstāšanas beigu laiks ir pēc sākuma laika
if (strtotime($start_time) >= strtotime($end_time)) {
    $errors[] = 'Atstāšanas laikam līdz jābūt vēlākam par atstāšanas laiku
    no.';
}
// Ja ir kļūdas, saglabā tās sesijā un pāradresē atpakaļ uz reģistrācijas
// lapu
if (!empty($errors)) {
    $_SESSION['message'] = implode(' ', $errors); // Apvieno kļūdas vienā
    // ziņojumā
    $_SESSION['message_type'] = 'error';
    $_SESSION['form_data'] = $_POST; // Saglabā lietotāja ievadītos datus,
    // lai tie netiktu zaudēti
    header('Location: ../registeradmin.php');
    exit();
}
// Mēģina saglabāt datus datubāzē
try {
    // Sagatavo SQL vaicājumu, lai ievietotu datus `cars` tabulā

```

```

        $stmt = $conn->prepare("INSERT INTO cars (name, surname, email, phone,
car_number, car_model, start_time, end_time, account_name) VALUES (?, ?, ?,
?, ?, ?, ?, ?, ?)");
        // Sasien parametru vērtības ar SQL vaicājumu, lai izvairītos no SQL
injekcijām
        $stmt->bind_param("sssssssss", $name, $surname, $email, $phone,
$car_number, $car_model, $start_time, $end_time, $username);
        // Izpilda vaicājumu un pārbauda, vai tas ir veiksmīgi izpildīts
        if ($stmt->execute()) {
            $_SESSION['message'] = 'Dati veiksmīgi saglabāti!';
            $_SESSION['message_type'] = 'success';
            header('Location: ../registeradmin.php'); // Pāradresē uz
reģistrācijas lapu
        } else {
            // Ja izpilde neizdodas, izmet kļūdu
            throw new Exception('Neizdevās saglabāt datus. Mēģiniet vēlreiz.');
```

```

        }
    } catch (Exception $e) {
        // Ja rodas kļūda, saglabā kļūdas ziņojumu sesijā un pāradresē atpakaļ
        uz reģistrācijas lapu
        $_SESSION['message'] = 'Kļūda: ' . $e->getMessage();
        $_SESSION['message_type'] = 'error';
        $_SESSION['form_data'] = $_POST; // Saglabā lietotāja ievadītos datus
        header('Location: ../registeradmin.php');
```

```

    }
    ?>
```

```

/* Atiestata noklusējuma margin un padding visiem elementiem un nodrošina
vienotu kastes izmēru aprēķinu */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box; /* Nodrošina, ka padding un border nepalielina
elementa kopējo izmēru */
}
/* Stilizē body (galveno lapas sadaļu) */
body {
    display: flex; /* Izmanto flexbox, lai centrētu saturu */
    justify-content: center; /* Centrē saturu horizontāli */
    align-items: center; /* Centrē saturu vertikāli */
    min-height: 100vh; /* Nodrošina, ka lapas augstums ir vismaz 100% no
skatloga augstuma */
    background-color: #f4f4f9; /* Gaiši pelēka fona krāsa */
    font-family: Arial, sans-serif; /* Norāda fontu, ko izmantot visā lapā
*/
}
/* Konteiners, kas ietver formu */
.form-container {
    width: 90%; /* Nodrošina, ka konteiners ir responsīvs */
    max-width: 800px; /* Maksimālais platums, lai saglabātu labu izskatu
arī uz lielākiem ekrāniem */
    padding: 30px; /* Iekšējā atstarpe */
    background-color: #fff; /* Baltas krāsas fons */
    border-radius: 15px; /* Noapaļoti stūri */
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); /* Viegls ēnojums, kas
piešķir 3D efektu */
    text-align: center; /* Centrē tekstu konteinerī */
}
/* Formas virsraksts */
.form-container h2 {
    margin-bottom: 20px; /* Atstarpe starp virsrakstu un saturu */
    color: #ff8a42; /* Oranža krāsa */
}
/* Izmanto grid izkārtojumu, lai sakārtotu formu divās kolonnās */
.form-grid {
    display: grid;
    grid-template-columns: 1fr 1fr; /* Izveido divas vienāda izmēra
kolonnas */
    gap: 20px; /* Atstarpe starp laukiem */
}
/* Sadaļu virsraksti (piemēram, "Informācija par īpašnieku") */
h3 {
    margin-top: 20px; /* Augšējā atstarpe, lai atdalītu sadaļas */
    margin-bottom: 10px; /* Apakšējā atstarpe, lai nodalītu no nākamā
elementa */
    color: #333; /* Tumši pelēka krāsa */
    text-align: left; /* Labo teksta izlīdzinājumu */
}
/* Etiķetes (label) formā */
label {
    display: block; /* Padara elementu bloka tipa */
    margin-top: 10px; /* Atstarpe virs etiķetes */
    font-size: 14px; /* Mazs fonta izmērs */
    color: #333; /* Tumši pelēka krāsa */
}
/* Teksta ievades lauki formā */
input[type="text"],

```

```

input[type="email"],
input[type="tel"],
input[type="datetime-local"] {
    width: 100%; /* Ievades lauks aizņem visu pieejamo platumu */
    padding: 10px; /* Iekšējā atstarpe */
    margin: 10px 0; /* Atstarpes starp laukiem */
    border: 1px solid #ddd; /* Pelēka apmale */
    border-radius: 5px; /* Noapaļoti stūri */
    font-size: 16px; /* Vidēja izmēra fonts */
    background-color: #f9f9f9; /* Gaiši pelēks fons */
}
/* Stilizē pogu */
button {
    width: 100%; /* Poga aizņem visu pieejamo platumu */
    padding: 15px; /* Iekšējā atstarpe */
    margin-top: 20px; /* Atstarpe virs pogas */
    background-color: #ff8a42; /* Oranža fona krāsa */
    color: #fff; /* Balta teksta krāsa */
    font-size: 16px; /* Vidēja izmēra fonts */
    font-weight: bold; /* Padara tekstu treknrakstā */
    border: none; /* Noņem apmali */
    border-radius: 30px; /* Noapaļoti stūri */
    cursor: pointer; /* Maina kursoru uz pointeri, lai norādītu, ka poga ir
klikšķināma */
    transition: background-color 0.3s ease; /* Nodrošina gludu pāreju,
mainot pogas krāsu */
}
/* Kad lietotājs uzbrauc ar peli uz pogas, tā maina krāsu */
button:hover {
    background-color: #ff7a30; /* Tumšāka oranža krāsa */
}

```

```

<?php
// Iekļauj datubāzes pieslēguma failu
include('php/db.php');
// Iegūst pašreizējo lapas numuru no URL parametra "page", ja nav iestatīts
- izmanto noklusējumu 1
$page = isset($_GET['page']) ? (int)$_GET['page'] : 1;
// Ierakstu skaits, kas tiek rādīts vienā lapā
$recordsPerPage = 5;
// Aprēķina no kura ieraksta sākt vaicājumu
$startIndex = ($page - 1) * $recordsPerPage;
// Iegūst filtrēšanas parametru no URL, ja nav iestatīts - izmanto
noklusējumu "all"
$filter = isset($_GET['filter']) ? $_GET['filter'] : 'all';
// Iegūst meklēšanas kritēriju no URL
$search = isset($_GET['car_number']) ? $_GET['car_number'] : '';
// SQL vaicājums, lai atlasītu transportlīdzekļus
$query = "SELECT * FROM cars WHERE 1"; // "WHERE 1" nodrošina, ka varam
pievienot papildu nosacījumus bez sintakses kļūdām
// Ja ir ievadīts meklēšanas kritērijs, pievieno nosacījumu
if (!empty($search)) {
    $query .= " AND car_number LIKE ?";
}
// Pievieno filtrēšanas kritērijus atkarībā no izvēlēta filtra
if ($filter === 'active') {
    $query .= " AND start_time <= NOW() AND end_time >= NOW()";
} elseif ($filter === 'planned') {
    $query .= " AND start_time > NOW()";
} elseif ($filter === 'history') {
    $query .= " AND end_time < NOW()";
}
// Pievieno lapošanas limitu vaicājumam
$query .= " LIMIT $startIndex, $recordsPerPage";
// Sagatavo SQL vaicājumu
$stmt = $conn->prepare($query);
// Ja ir meklēšanas kritērijs, sasien attiecīgos parametrus vaicājumam
if (!empty($search)) {
    $searchParam = "%$search%";
    $stmt->bind_param("s", $searchParam);
}

// Izpilda vaicājumu
$stmt->execute();
// Iegūst rezultātus no izpildītā vaicājuma
$result = $stmt->get_result();
// Vaicājums, lai iegūtu kopējo transportlīdzekļu skaitu (lapošanai)
$totalQuery = "SELECT COUNT(*) as count FROM cars WHERE 1";
// Pievieno meklēšanas un filtra kritērijus kopējam ierakstu skaitam
if (!empty($search)) {
    $totalQuery .= " AND car_number LIKE ?";
}
if ($filter === 'active') {
    $totalQuery .= " AND start_time <= NOW() AND end_time >= NOW()";
} elseif ($filter === 'planned') {
    $totalQuery .= " AND start_time > NOW()";
} elseif ($filter === 'history') {
    $totalQuery .= " AND end_time < NOW()";
}
// Sagatavo un izpilda kopējo ierakstu skaita vaicājumu
$totalStmt = $conn->prepare($totalQuery);
if (!empty($search)) {

```

```

        $totalStmt->bind_param("s", $searchParam);
    }
    $totalStmt->execute();
    $totalResult = $totalStmt->get_result();
    $totalRows = $totalResult->fetch_assoc()['count'];
    // Aprēķina kopējo lapu skaitu lāpošanai
    $totalPages = ceil($totalRows / $recordsPerPage);
    // Aizver datubāzes savienojumu
    $conn->close();
    ?>
<!DOCTYPE html>
<html lang="lv">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>PPRO - admin</title>
    <link rel="icon" type="image/png" href="img/pic.png">
    <link rel="stylesheet" href="css/viewcars.css">
</head>
<body>
    <div class="form-container">
        <h2>REGISTRĒTIE TRANSPORTLĪDZEKĻI</h2>
        <!-- Top bar: Filter buttons (left) + Search form (right) -->
    <div class="top-bar">
        <!-- Filter Buttons -->
        <div class="filter-buttons">
            <a href="?filter=active&page=1" class="filter-btn <?php echo
$filter === 'active' ? 'active' : ''; ?>">Šobrīd aktīvs</a>
            <a href="?filter=planned&page=1" class="filter-btn <?php echo
$filter === 'planned' ? 'active' : ''; ?>">Plānotais</a>
            <a href="?filter=history&page=1" class="filter-btn <?php echo
$filter === 'history' ? 'active' : ''; ?>">Vēsture</a>
            <a href="?filter=all&page=1" class="filter-btn <?php echo $filter
=== 'all' ? 'active' : ''; ?>">Viss</a>
        </div>
        <!-- Search Form -->
        <form method="GET" action="viewadmin.php" class="search-form-inline">
            <input type="text" name="car_number" placeholder="Meklēt pēc
numura"
                value="<?php echo isset($_GET['car_number']) ?
htmlspecialchars($_GET['car_number']) : ''; ?>">
            <button type="submit">Meklēt</button>
        </form>
    </div>
    <!-- Tabula, kurā tiek attēloti transportlīdzekļi -->
    <table>
        <thead>
            <tr>
                <th>Vārds</th>
                <th>Uzvārds</th>
                <th>E-pasts</th>
                <th>Telefona numurs</th>
                <th>Numura zīme</th>
                <th>Marka</th>
                <th>Atstāšanas laiks no</th>
                <th>Atstāšanas laiks līdz</th>
                <th>Filāle</th>
                <th>Darbības</th>
            </tr>
        </thead>
        <tbody>

```

```

        <?php while ($row = $result->fetch_assoc()): ?>
            <tr>
                <td><?php echo htmlspecialchars($row['name']);
?></td>
                <td><?php echo htmlspecialchars($row['surname']);
?></td>
                <td><?php echo htmlspecialchars($row['email']);
?></td>
                <td><?php echo htmlspecialchars($row['phone']);
?></td>
                <td><?php echo
htmlspecialchars($row['car_number']); ?></td>
                <td><?php echo htmlspecialchars($row['car_model']);
?></td>
                <td><?php echo
htmlspecialchars($row['start_time']); ?></td>
                <td><?php echo htmlspecialchars($row['end_time']);
?></td>
                <td><?php echo
htmlspecialchars($row['account_name']); ?></td>
                <td>
                    <!-- Pogu sadaļa transportlīdzekļu rediģēšanai
un dzēšanai -->
                    <button class="edit-btn"
onclick="window.location.href='editcar.php?id=<?= $row['id'];
?>'>Rediģēt</button>
                    <button class="delete-btn"
onclick="deletecar(<?= $row['id']; ?>)">Dzēst</button>
                    </td>
            </tr>
        <?php endwhile; ?>
    </tbody>
</table>
<!-- Lapu navigācija (lapošana) -->
<div class="pagination">
    <?php for ($i = 1; $i <= $totalPages; $i++): ?>
        <a href="?filter=<?php echo htmlspecialchars($filter);
?>&search=<?php echo htmlspecialchars($search); ?>&page=<?php echo $i; ?>"
class="<?php echo $i === $page ? 'active' : ''; ?>">
            <?php echo $i; ?>
        </a>
    <?php endfor; ?>
</div>
<!-- Poga, lai atgrieztos atpakaļ uz admin paneli -->
<button class="back-button"
onclick="window.location.href='admin.html'">Atpakaļ</button>
</div>
<!-- Iekļauj SweetAlert2 bibliotēku uznirstošiem paziņojumiem -->
<script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
<!-- Iekļauj ārējo JavaScript failu, kas pārvalda lapas interaktivitāti
-->
<script src="js/viewadmin.js"></script>
</body>
</html>

```

```

/* Noklusētie iestatījumi visiem elementiem */
* {
    margin: 0; /* Noņem ārējo malu */
    padding: 0; /* Noņem iekšējo malu */
    box-sizing: border-box; /* Iekšējais izmērs ietver padding un border */
}
/* Pamatstilojums body elementam */
body {
    display: flex; /* Flexbox izkārtojums */
    justify-content: center; /* Centrē saturu horizontāli */
    align-items: center; /* Centrē saturu vertikāli */
    min-height: 100vh; /* Minimālais augstums - visa skatloga augstums */
    background-color: #f4f4f9; /* Gaišs fona tonis */
    font-family: Arial, sans-serif; /* Fonta tips */
}
/* Forma kontainers */
.form-container {
    width: 90%; /* 90% no pieejamā platuma */
    max-width: 1200px; /* Maksimālais platums */
    padding: 30px; /* Iekšējā mala */
    background-color: #fff; /* Balts fons */
    border-radius: 15px; /* Noapaļoti stūri */
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); /* Ēna ap formu */
    text-align: center; /* Centrēts teksts */
}
/* Virsraksts formā */
.form-container h2 {
    margin-bottom: 15px; /* Attālums apakšā */
    color: #ff8a42; /* Oranža krāsa */
}
/* Etiķetes formā */
label {
    display: block;
    margin-top: 10px;
    font-size: 14px;
    color: #333;
}
/* Ievades lauki */
input[type="text"],
input[type="email"],
input[type="tel"],
input[type="datetime-local"] {
    width: 100%;
    padding: 10px;
    margin: 10px 0;
    border: 1px solid #ddd;
    border-radius: 5px;
    font-size: 16px;
}
/* Poga formā */
button {
    width: 100%;
    padding: 12px;
    margin-top: 15px;
    background-color: #ff8a42;
    color: #fff;
    font-size: 16px;
    font-weight: bold;
    border: none;
    border-radius: 20px;
}

```

```

        cursor: pointer;
        transition: background-color 0.3s ease;
    }
    /* Poga - hover efekts */
    button:hover {
        background-color: #ff7a30;
    }
    /* Tabula - pamata stils */
    table {
        width: 100%;
        margin-top: 20px;
        border-collapse: collapse;
        table-layout: fixed;
    }
    /* Tabulas šūnas */
    table th, table td {
        padding: 12px;
        border-bottom: 1px solid #ddd;
        text-align: left;
        font-size: 16px;
        color: #333;
        word-wrap: break-word;
        overflow: hidden;
        text-overflow: ellipsis;
        white-space: normal;
    }
    /* Tabulas virsraksta šūnas */
    table th {
        background-color: #ff8a42;
        color: #fff;
        font-weight: bold;
        text-transform: uppercase;
    }
    /* Katrs otrais rindīņas fons */
    table tr:nth-child(even) {
        background-color: #f9f9f9;
    }
    /* Pogas tabulā */
    table button {
        padding: 8px 12px;
        margin: 0 5px;
        font-size: 14px;
        border: none;
        border-radius: 5px;
        cursor: pointer;
        transition: background-color 0.3s ease;
    }
    /* Rediģēšanas poga */
    table button.edit-btn {
        background-color: #4CAF50;
        color: #fff;
    }
    /* Dzēšanas poga */
    table button.delete-btn {
        background-color: #f44336;
        color: #fff;
    }
    /* Hover efekti tabulas pogām */
    table button.edit-btn:hover {
        background-color: #45a049;
    }

```

```

table button.delete-btn:hover {
    background-color: #d32f2f;
}
/* Atpakaļ poga */
.back-button {
    width: 100%;
    padding: 12px;
    margin-top: 20px;
    background-color: #ff8a42;
    color: #fff;
    font-size: 16px;
    font-weight: bold;
    border: none;
    border-radius: 20px;
    cursor: pointer;
    transition: background-color 0.3s ease;
    text-decoration: none;
    text-align: center;
    display: inline-block;
}
.back-button:hover {
    background-color: #ff7a30;
}
/* Meklēšanas konteiners */
.search-container {
    display: flex;
    align-items: center;
    justify-content: flex-start;
    gap: 10px;
    margin-bottom: 20px;
}
/* Meklēšanas ievades lauks */
.search-container input {
    padding: 8px;
    font-size: 14px;
    width: 250px;
    border: 1px solid #ddd;
    border-radius: 5px;
}
/* Meklēšanas poga */
.search-btn {
    padding: 8px 15px;
    font-size: 14px;
    background-color: #ddd;
    color: #333;
    border: 1px solid #ccc;
    border-radius: 5px;
    cursor: pointer;
    transition: background-color 0.3s ease;
}
.search-btn:hover {
    background-color: #bbb;
}
/* Filtru pogu konteiners */
.filter-buttons {
    display: flex;
    justify-content: center;
    gap: 15px;
    margin-bottom: 20px;
}
/* Filtru poga */

```

```

.filter-btn {
    background-color: #ff8a42;
    color: #fff;
    padding: 10px 20px;
    font-size: 16px;
    font-weight: bold;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: background-color 0.3s ease;
}
.filter-btn:hover {
    background-color: #ff7a30;
}
/* Tabula vēlreiz (atkārtojums, bet ar nedaudz citām vērtībām) */
table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 10px;
}
table th, table td {
    padding: 12px;
    border-bottom: 1px solid #ddd;
    text-align: left;
    font-size: 14px;
}
table th {
    background-color: #ff8a42;
    color: #fff;
}
table tr:nth-child(even) {
    background-color: #f9f9f9;
}
/* Atpakaļ poga atkārtota */
.back-button {
    display: inline-block;
    margin-top: 20px;
    padding: 12px;
    background-color: #ff8a42;
    color: white;
    font-weight: bold;
    border-radius: 20px;
    text-decoration: none;
    transition: background-color 0.3s ease;
}
.back-button:hover {
    background-color: #ff7a30;
}
/* Lapu navigācija (paginācija) */
.pagination {
    display: flex;
    justify-content: center;
    margin-top: 20px;
    gap: 5px;
}
.pagination a {
    padding: 8px 12px;
    text-decoration: none;
    background-color: #ff8a42;
    color: white;
    border-radius: 5px;
}

```

```

        transition: background-color 0.3s ease;
    }
    .pagination a:hover {
        background-color: #ff7a30;
    }
    .pagination a.active {
        background-color: #e06722;
        font-weight: bold;
    }
    /* Papildu stils paginācijai */
    .pagination {
        margin: 20px 0;
        text-align: center;
    }
    .pagination-btn {
        display: inline-block;
        padding: 10px 15px;
        margin: 0 5px;
        background-color: #ddd;
        color: #333;
        border-radius: 5px;
        text-decoration: none;
    }
    .pagination-btn.active {
        background-color: #ff8a42;
        color: #fff;
    }
    /* Lejupielādes poga */
    .download-btn {
        padding: 8px 15px;
        font-size: 14px;
        background-color: #007bff;
        color: #fff;
        border: 1px solid #0056b3;
        border-radius: 5px;
        cursor: pointer;
        transition: background-color 0.3s ease;
        height: 100%;
    }
    .download-btn:hover {
        background-color: #0056b3;
    }
    /* Rediģēšanas un dzēšanas pogas */
    .btn-edit {
        background-color: #4CAF50;
        color: white;
        padding: 5px 10px;
        border: none;
        border-radius: 3px;
        cursor: pointer;
    }
    .btn-delete {
        background-color: #f44336;
        color: white;
        padding: 5px 10px;
        border: none;
        border-radius: 3px;
        cursor: pointer;
    }
    .btn-edit:hover {
        background-color: #45a049;
    }

```

```
}
.btn-delete:hover {
    background-color: #d32f2f;
}
/* Augšējais rīkjoslas kontainers */
.top-bar {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 20px;
    flex-wrap: wrap;
}
/* Iekšējais meklēšanas forms */
.search-form-inline {
    display: flex;
    gap: 10px;
}
/* Teksta ievade rīkjoslā */
.search-form-inline input[type="text"] {
    width: 250px !important;
    padding: 8px !important;
    margin: 0 !important;
    border: 1px solid #ccc;
    border-radius: 5px;
}
/* Poga rīkjoslā */
.search-form-inline button {
    padding: 8px 12px !important;
    margin: 0 !important;
    width: auto !important;
    background-color: #ff8a42;
    color: white;
    border: none;
    border-radius: 5px;
    font-weight: bold;
    cursor: pointer;
}
```

```

// Funkcija dzēšanai, kas pieņem automašīnas ID kā argumentu
function deletecar(carId) {
  // Izsauc SweetAlert2 paziņojumu ar apstiprinājuma dialogu
  Swal.fire({
    title: "Vai tiešām vēlies dzēst?", // Dialoga virsraksts
    text: "Šo darbību nevar atsaukt!", // Brīdinājuma teksts
    icon: "warning", // Brīdinājuma ikona (dzeltenais brīdinājuma
simbols)
    showCancelButton: true, // Parāda "Atcelt" pogu
    confirmButtonColor: "#d33", // Sarkanā krāsa apstiprināšanas pogai
    cancelButtonColor: "#3085d6", // Zilā krāsa atcelšanas pogai
    confirmButtonText: "Jā, dzēst!", // Apstiprinājuma pogas teksts
    cancelButtonText: "Atcelt" // Atcelšanas pogas teksts
  }).then((result) => {
    // Pārbauda, vai lietotājs apstiprināja dzēšanu
    if (result.isConfirmed) {
      // Izmanto Fetch API, lai nosūtītu GET pieprasījumu uz
      `delete.php`, nododot automašīnas ID
      fetch(`php/delete.php?id=${carId}`)
        .then(response => response.json()) // Konvertē servera
atbildi JSON formātā
        .then(data => {
          // Pārbauda, vai servera atbilde norāda uz veiksmīgu
dzēšanu
          if (data.success) {
            // Ja dzēšana veiksmīga, parāda SweetAlert2
paziņojumu
            Swal.fire({
              title: "Dati veiksmīgi izdzēsti!", //
Paziņojuma virsraksts
              icon: "success", // Zaļā veiksmīga darbības
ikona
              timer: 2000, // Paziņojums automātiski pazūd
pēc 2 sekundēm
              showConfirmButton: false // Nerāda
apstiprinājuma pogu
            }).then(() => {
              // Pēc paziņojuma aizvēršanas pāradresē
lietotāju uz `viewadmin.php`
              window.location.href = "viewadmin.php";
            });
          } else {
            // Ja serveris atgrieza kļūdu, parāda kļūdas
paziņojumu
            Swal.fire({
              title: "Kļūda!", // Paziņojuma virsraksts
              text: data.error || "Dzēšana neizdevās!", // Ja
serveris atgrieza kļūdas ziņu, tā tiek parādīta
              icon: "error" // Sarkanā kļūdas ikona
            });
          }
        })
        .catch(error => {
          // Ja radās tīkla vai servera kļūda, parāda vispārīgu
kļūdas paziņojumu
          Swal.fire({
            title: "Kļūda!", // Paziņojuma virsraksts
            text: "Kaut kas nogāja greizi!", // Vispārīgs
kļūdas paziņojums
            icon: "error" // Sarkanā kļūdas ikona

```

[illegible]

```

<?php
// Iekļauj datubāzes savienojuma failu
include 'db.php';
// Norāda, ka atbilde būs JSON formātā
header('Content-Type: application/json');
// Pārbauda, vai pieprasījumā ir norādīts 'id' parametrs
if (isset($_GET['id'])) {
    // Konvertē saņemto ID uz veselu skaitli, lai novērstu potenciālus
    drošības draudus
    $id = intval($_GET['id']);
    // SQL vaicājums, lai dzēstu transportlīdzekļa ierakstu pēc ID
    $sql = "DELETE FROM cars WHERE id = ?";
    // Sagatavo SQL vaicājumu, lai izvairītos no SQL injekcijas
    $stmt = $conn->prepare($sql)
    // Piesaista ID kā skaitli (`integer`)
    $stmt->bind_param("i", $id);
    // Izpilda vaicājumu un pārbauda, vai tas ir veiksmīgi izpildīts
    if ($stmt->execute()) {
        // Ja dzēšana veiksmīga, atgriež JSON atbildi ar `success: true`
        echo json_encode(['success' => true]);
    } else {
        // Ja dzēšana neizdevās, atgriež kļūdas ziņojumu JSON formātā
        echo json_encode(['success' => false, 'error' => $conn->error]);
    }
    // Aizver sagatavoto vaicājumu
    $stmt->close();
    // Aizver datubāzes savienojumu
    $conn->close();
} else {
    // Ja 'id' nav norādīts pieprasījumā, atgriež kļūdas paziņojumu JSON
    formātā
    echo json_encode(['success' => false, 'error' => 'Invalid request']);
}
?>

```

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Iekļauj datubāzes savienojuma failu
include 'db.php';
// Pārbauda, vai pieprasījuma metode ir POST (tikai POST metode ir atļauta)
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Iegūst un validē ievades laukus no POST pieprasījuma, nodrošinot, ka
    tie nav nenoteikti
    $id = $_POST['id'] ?? null; // Transportlīdzekļa ID
    $name = $_POST['name'] ?? ''; // Īpašnieka vārds
    $surname = $_POST['surname'] ?? ''; // Īpašnieka uzvārds
    $email = $_POST['email'] ?? ''; // Īpašnieka e-pasts
    $phone = $_POST['phone'] ?? ''; // Īpašnieka telefona numurs
    $car_number = $_POST['car_number'] ?? ''; // Transportlīdzekļa numura
    zīme
    $car_model = $_POST['car_model'] ?? ''; // Transportlīdzekļa
    marka/modelis
    $start_time = $_POST['start_time'] ?? ''; // Autostāvvietas sākuma
    laiks
    $end_time = $_POST['end_time'] ?? ''; // Autostāvvietas beigu laiks
    // Pārbauda, vai visi lauki ir aizpildīti un vai ID nav tukšs
    if ($id && $name && $surname && $email && $phone && $car_number &&
    $car_model && $start_time && $end_time) {
        // SQL vaicājums, lai atjauninātu transportlīdzekļa datus datubāzē
        $query = "UPDATE cars SET name = ?, surname = ?, email = ?, phone =
        ?, car_number = ?, car_model = ?, start_time = ?, end_time = ? WHERE id =
        ?";

        // Sagatavo SQL vaicājumu, lai izvairītos no SQL injekcijām
        $stmt = $conn->prepare($query);
        // Pārbauda, vai vaicājums tika sagatavots veiksmīgi
        if ($stmt) {
            // Sasien ievadītos datus ar vaicājumu
            $stmt->bind_param('ssssssssi', $name, $surname, $email, $phone,
            $car_number, $car_model, $start_time, $end_time, $id);
            // Izpilda vaicājumu un pārbauda, vai tas tika veiksmīgi
            izpildīts
            if ($stmt->execute()) {
                $_SESSION['message'] = "Dati veiksmīgi saglabāti!";
                $_SESSION['message_type'] = "success"; // Iestata veiksmes
                paziņojumu sesijā
            } else {
                $_SESSION['message'] = "Kļūda, saglabājot datus.";
                $_SESSION['message_type'] = "error"; // Iestata kļūdas
                paziņojumu sesijā
            }
            // Aizver sagatavoto vaicājumu
            $stmt->close();
        } else {
            // Saglabā kļūdas ziņojumu, ja SQL vaicājums nevarēja tikt
            sagatavots
            $_SESSION['message'] = "Kļūda sagatavojot vaicājumu.";
            $_SESSION['message_type'] = "error";
        }
    } else {
        // Saglabā kļūdas ziņojumu, ja kāds no laukiem nav aizpildīts
        $_SESSION['message'] = "Visi lauki ir obligāti jāaizpilda!";
        $_SESSION['message_type'] = "error";
    }
} else {

```

```
// Ja pieprasījums nav veikts ar POST metodi, iestata kļūdas ziņojumu
$_SESSION['message'] = "Nederīgs pieprasījums.";
$_SESSION['message_type'] = "error";
}
// Aizver datubāzes savienojumu
$conn->close();
// Pāradresē atpakaļ uz transportlīdzekļa rediģēšanas lapu ar attiecīgo ID
header("Location: ../editcar.php?id=$id");
exit;
?>
```

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Ieslēdz kļūdu attēlošanu izstrādes nolūkiem
ini_set('display_errors', 1);
error_reporting(E_ALL);
// Iekļauj datubāzes savienojuma failu
include 'php/db.php';
// Pārbauda, vai datubāzes savienojums ir izveidots
if (!isset($conn)) {
    die('Datubāzes savienojums neizdevās. Pārbaudiet db.php ceļu un
konfigurāciju.');
```

```

}
// Iegūst transportlīdzekļa ID no URL parametra
$id = $_GET['id'] ?? null;
// Pārbauda, vai ID ir norādīts
if (!$id) {
    die('Nav norādīts derīgs ID.');
```

```

}
// Sagatavo SQL vaicājumu, lai atlasītu konkrēto transportlīdzekli
$query = "SELECT * FROM cars WHERE id = ?";
$stmt = $conn->prepare($query);
// Pārbauda, vai vaicājums tika sagatavots pareizi
if (!$stmt) {
    die('SQL vaicājuma kļūda: ' . $conn->error);
}
// Sasien ID ar SQL vaicājumu un izpilda to
$stmt->bind_param('i', $id);
$stmt->execute();
$result = $stmt->get_result();
// Pārbauda, vai ieraksts tika atrasts
if ($result->num_rows === 0) {
    die('Ieraksts netika atrasts.');
```

```

}
// Iegūst transportlīdzekļa datus
$car = $result->fetch_assoc();
?>
<!DOCTYPE html>
<html lang="lv">
<head>
    <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8 -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- Pielāgo lapas izskatu mobilajām ierīcēm -->
    <title>PPRO - admin</title> <!-- Lapas nosaukums -->
    <!-- Ikona, kas tiek attēlota pārlūkprogrammas cilnē -->
    <link rel="icon" type="image/png" href="img/pic.png">
    <!-- SweetAlert2 bibliotēka paziņojumu parādīšanai -->
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/sweetalert2@11/dist/sweetalert2.min.css"
>
    <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
    <!-- Stila lapa, kas nosaka lapas vizuālo izskatu -->
    <link rel="stylesheet" href="css/reg.css">
</head>
<body>
    <!-- Konteiners, kas satur rediģēšanas formu -->
    <div class="form-container">
        <h2>REDIĢĒT AUTOMAŠĪNU</h2>
        <!-- Forma, kas nosūta datus uz updatecar.php -->
        <form action="php/updatecar.php" method="POST">
```

```

        <!-- Slēptais lauks, kas satur transportlīdzekļa ID -->
        <input type="hidden" name="id" value="<?php echo $car['id'];
?>">

        <div class="form-grid">
            <!-- Transportlīdzekļa īpašnieka informācijas sadaļa -->
            <div class="owner-info">
                <h3>Informācija par īpašnieku</h3>
                <label for="name">Vārds:</label>
                <input type="text" id="name" name="name" value="<?php
echo htmlspecialchars($car['name']); ?>" required>
                <label for="surname">Uzvārds:</label>
                <input type="text" id="surname" name="surname"
value="<?php echo htmlspecialchars($car['surname']); ?>" required>
                <label for="email">E-pasts:</label>
                <input type="email" id="email" name="email"
value="<?php echo htmlspecialchars($car['email']); ?>" required>
                <label for="phone">Telefona numurs:</label>
                <input type="tel" id="phone" name="phone" value="<?php
echo htmlspecialchars($car['phone']); ?>" required>
            </div>
            <!-- Transportlīdzekļa informācijas sadaļa -->
            <div class="car-info">
                <h3>Informācija par auto</h3>
                <label for="car_number">Numura zīme:</label>
                <input type="text" id="car_number" name="car_number"
value="<?php echo htmlspecialchars($car['car_number']); ?>" required>
                <label for="car_model">Marka:</label>
                <input type="text" id="car_model" name="car_model"
value="<?php echo htmlspecialchars($car['car_model']); ?>" required>
                <label for="start_time">Atstāšanas laiks no:</label>
                <input type="datetime-local" id="start_time"
name="start_time" value="<?php echo htmlspecialchars(date('Y-m-d\TH:i',
strtotime($car['start_time']))); ?>" required>
                <label for="end_time">Atstāšanas laiks līdz:</label>
                <input type="datetime-local" id="end_time"
name="end_time" value="<?php echo htmlspecialchars(date('Y-m-d\TH:i',
strtotime($car['end_time']))); ?>" required>
            </div>
        </div>
        <!-- Pogu sadaļa -->
        <button type="submit">Saglabāt</button> <!-- Iesniedz datus
rediģēšanai -->
        <button type="button"
onclick="window.location.href='viewadmin.php'">Atpakaļ</button> <!--
Atgriežas uz administrācijas lapu -->
    </form>
</div>
<!-- Paziņojumu attēlošana, ja sesijā ir saglabāti ziņojumi -->
<?php
if (isset($_SESSION['message']) && isset($_SESSION['message_type'])) {
    echo "<script>
        Swal.fire({
            title: ' " . ($_SESSION['message_type'] === 'success' ?
'Veiksmīgi!' : 'Kļūda!') . " ',
            text: ' " . $_SESSION['message'] . " ',
            icon: ' " . $_SESSION['message_type'] . " ',
            confirmButtonText: 'Labi'
        }).then((result) => {
            if (result.isConfirmed) {
                window.location.href = 'viewadmin.php';
            }
        });
    </script>";
}

```

```
        });  
    </script>";  
    // Pēc paziņojuma attēlošanas dzēš ziņojumus no sesijas  
    unset($_SESSION['message']);  
    unset($_SESSION['message_type']);  
    }  
    ?>  
</body>  
</html>  
<?php  
// Aizver datubāzes savienojumu  
$conn->close();  
?>
```

```

<!DOCTYPE html>
<html lang="en"> <!-- Dokumenta valoda angļu -->
<head>
  <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8 -->
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <!-- Pielāgo lapas izmēru mobilajām ierīcēm -->
  <title>PPRO - admin</title> <!-- Lapas nosaukums, kas parādīsies
pārlūkprogrammas cilnē -->
  <!-- Ikona, kas tiek attēlota pārlūkprogrammas cilnē -->
  <link rel="icon" type="image/png" href="img/pic.png">
  <!-- Stila lapa, kas nosaka lapas vizuālo izskatu -->
  <link rel="stylesheet" href="css/styles.css">
</head>
<body>
  <!-- Kontainers, kas satur iestatījumu pogas -->
  <div class="form-container">
    <h2>Iestatījumi</h2> <!-- Virsraksts "Iestatījumi" -->
    <!-- Poga, lai pārietu uz filiāles pievienošanas lapu -->
    <button onclick="window.location.href='addbranch.html'">Pievienot
filiāli</button>
    <!-- Poga, lai apskatītu visas filiāles -->
    <button onclick="window.location.href='lookbranch.php'">Apskatīt
filiāles</button>
    <!-- Poga, lai atceltu un atgrieztos atpakaļ uz admin lapu -->
    <button onclick="window.location.href='admin.html'">Atcelt</button>
  </div>
</body>
</html>

```

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Iegūst ziņojumu un ziņojuma tipu no sesijas, ja tie ir iestatīti
$message = $_SESSION['message'] ?? null;
$messageType = $_SESSION['message_type'] ?? null;
// Iegūst ievadītos datus no sesijas, lai saglabātu ievadītos laukus
gadījumā, ja forma tiek atkārtoti ielādēta
$name = $_SESSION['form_data']['name'] ?? '';
$surname = $_SESSION['form_data']['surname'] ?? '';
$email = $_SESSION['form_data']['email'] ?? '';
$phone = $_SESSION['form_data']['phone'] ?? '';
$car_number = $_SESSION['form_data']['car_number'] ?? '';
$car_model = $_SESSION['form_data']['car_model'] ?? '';
$start_time = $_SESSION['form_data']['start_time'] ?? '';
$end_time = $_SESSION['form_data']['end_time'] ?? '';
// Ja ziņojuma tips ir "success", noņemam saglabātos ievades datus
if ($messageType === 'success') {
    unset($_SESSION['form_data']);
}
// Attīra sesijas ziņojumu un ziņojuma tipu, lai tie netiktu parādīti
atkārtoti
unset($_SESSION['message'], $_SESSION['message_type']);
?>
<!DOCTYPE html>
<html lang="lv">
<head>
    <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8 -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- Pielāgo lapas izmēru mobilajām ierīcēm -->
    <title>ParkingPro - <?php echo
htmlspecialchars($_SESSION['branch_name']); ?> - reģistrēt
transportlīdzekli</title> <!-- Dinamisks lapas nosaukums ar filiāles
nosaukumu -->
    <!-- Ikona, kas tiek attēlota pārlūkprogrammas cilnē -->
    <link rel="icon" type="image/png" href="img/pic.png">
    <!-- SweetAlert2 bibliotēka vizuāliem paziņojumiem -->
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/sweetalert2@11/dist/sweetalert2.min.css"
>
    <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
    <!-- Stila lapa, kas nosaka lapas vizuālo izskatu -->
    <link rel="stylesheet" href="css/reg.css">
</head>
<body>
    <?php if ($message && $messageType === 'success'): ?>
        <!-- Ja ir veiksmīgs ziņojums, parādīt SweetAlert paziņojumu -->
        <script>
            Swal.fire({
                icon: 'success',
                title: 'Veiksmīgi!',
                text: '<?php echo htmlspecialchars($message); ?>',
                confirmButtonText: 'Labi',
                confirmButtonColor: '#6c63ff',
            }).then((result) => {
                if (result.isConfirmed) {
                    window.location.href = 'viewbranch.php'; // Pāradresē
uz filiāles skata lapu pēc apstiprinājuma
                }
            });
        </script>
    </?php>

```

```

</script>
<?php elseif ($message && $messageType === 'error'): ?>
    <!-- Ja ir kļūdas ziņojums, parādīt SweetAlert paziņojumu -->
    <script>
        Swal.fire({
            icon: 'error',
            title: 'Kļūda!',
            text: '<?php echo htmlspecialchars($message); ?>',
            confirmButtonText: 'Labi',
            confirmButtonColor: '#ff6c63',
        });
    </script>
<?php endif; ?>
<!-- Kontainers, kas satur transportlīdzekļa reģistrācijas formu -->
<div class="form-container">
    <h2>REĢISTRĒT TRANSPORTLĪDZEKLI</h2>
    <!-- Forma, kas nosūta datus uz savecarbranch.php -->
    <form action="php/savecarbranch.php" method="POST">
        <div class="form-grid">
            <!-- Īpašnieka informācijas sadaļa -->
            <div class="owner-info">
                <h3>Informācija par īpašnieku</h3>
                <label for="name">Vārds:</label>
                <input type="text" id="name" name="name"
placeholder="Ievadiet vārdu" value="<?php echo htmlspecialchars($name); ?>"
autocomplete="off" required>
                <label for="surname">Uzvārds:</label>
                <input type="text" id="surname" name="surname"
placeholder="Ievadiet uzvārdu" value="<?php echo
htmlspecialchars($surname); ?>" autocomplete="off" required>
                <label for="email">E-pasts:</label>
                <input type="email" id="email" name="email"
placeholder="Ievadiet e-pasta adresi" value="<?php echo
htmlspecialchars($email); ?>" autocomplete="off" required>

                <label for="phone">Telefona numurs:</label>
                <input type="tel" id="phone" name="phone"
placeholder="Ievadiet telefona numuru" value="<?php echo
htmlspecialchars($phone); ?>" autocomplete="off" required>
            </div>
            <!-- Auto informācijas sadaļa -->
            <div class="car-info">
                <h3>Informācija par auto</h3>
                <label for="car_number">Numura zīme:</label>
                <input type="text" id="car_number" name="car_number"
placeholder="Ievadiet numura zīmi" value="<?php echo
htmlspecialchars($car_number); ?>" autocomplete="off" required>
                <label for="car_model">Marka:</label>
                <input type="text" id="car_model" name="car_model"
placeholder="Ievadiet auto marku" value="<?php echo
htmlspecialchars($car_model); ?>" autocomplete="off" required>
                <label for="start_time">Atstāšanas laiks no:</label>
                <input type="datetime-local" id="start_time"
name="start_time" value="<?php echo htmlspecialchars($start_time); ?>"
autocomplete="off" required>
                <label for="end_time">Atstāšanas laiks līdz:</label>
                <input type="datetime-local" id="end_time"
name="end_time" value="<?php echo htmlspecialchars($end_time); ?>"
autocomplete="off" required>
            </div>
        </div>
    </form>
</div>

```

```
        <!-- Pogu sadaļa -->
        <button type="submit">Saglabāt</button> <!-- Nosūta datus uz
serveri -->
        <button type="button"
onclick="window.location.href='branch.php'">Atpakaļ</button> <!-- Atgriežas
atpakaļ uz filiāles lapu -->
    </form>
</div>
</body>
</html>
```

```

<?php
// Pārbauda, vai pieprasījuma metode ir POST (tas nozīmē, ka dati tiek
sūtīti no formas)
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Iegūst un apstrādā lietotāja ievadītos datus
    $branch_name = strtoupper(trim($_POST['branch_name'])); // Noņem
atstarpes un pārveido par lielajiem burtiem
    $username = trim($_POST['username']); // Noņem atstarpes no
lietotājevārda
    $password = trim($_POST['password']); // Noņem atstarpes no paroles
    // Izveido savienojumu ar datubāzi
    $conn = new mysqli("localhost", "root", "", "parkingpro");
    // Pārbauda, vai savienojums ar datubāzi ir veiksmīgs
    if ($conn->connect_error) {
        die("Savienojuma kļūda: " . $conn->connect_error); // Ja
savienojums neizdevās, izvada kļūdu un pārtrauc skriptu
    }
    // Pārbauda, vai lietotājevārds jau eksistē datubāzē
    $check_user_sql = "SELECT * FROM branches WHERE username = ?";
    $stmt = $conn->prepare($check_user_sql); // Sagatavo SQL vaicājumu, lai
novērstu SQL injekcijas
    $stmt->bind_param("s", $username); // Piesaista lietotājevārdu kā
stringu
    $stmt->execute(); // Izpilda vaicājumu
    $result = $stmt->get_result(); // Iegūst rezultātu
    // Ja lietotājevārds jau pastāv, pāradresē atpakaļ uz filiāles
pievienošanas lapu ar kļūdas ziņojumu
    if ($result->num_rows > 0) {
        header("Location: ../addbranch.html?error=userexists");
        exit();
    }
    // Sagatavo SQL vaicājumu jaunas filiāles pievienošanai
    $sql = "INSERT INTO branches (branch_name, username, password) VALUES
(?, ?, ?)";
    $stmt = $conn->prepare($sql); // Sagatavo vaicājumu
    $stmt->bind_param("sss", $branch_name, $username, $password); //
Piesaista datus SQL vaicājumam
    // Izpilda vaicājumu un pārbauda, vai ieraksts veiksmīgi pievienots
    if ($stmt->execute()) {
        header("Location: ../settings.html?success=1"); // Ja pievienošana
veiksmīga, pāradresē uz iestatījumu lapu
        exit();
    } else {
        header("Location: ../addbranch.html?error=1"); // Ja kļūda,
pāradresē atpakaļ ar kļūdas ziņojumu
        exit();
    }
    // Aizver SQL vaicājumu un datubāzes savienojumu
    $stmt->close();
    $conn->close();
} else {
    // Ja pieprasījuma metode nav POST, pāradresē atpakaļ uz filiāles
pievienošanas lapu
    header("Location: ../addbranch.html");
    exit();
}
?>

```

```

// Izveido URL parametru objektu, lai varētu pārbaudīt un iegūt vērtības no
pašreizējā URL
const urlParams = new URLSearchParams(window.location.search);
// Pārbauda, vai URL satur parametru 'success', kas norāda, ka filiāle tika
veiksmīgi pievienota
if (urlParams.has('success')) {
    // Parāda SweetAlert2 paziņojumu par veiksmīgu filiāles pievienošanu
    Swal.fire({
        icon: 'success', // Zaļa veiksmes ikona
        title: 'Veiksmīgi!', // Paziņojuma virsraksts
        text: 'Filiāle veiksmīgi pievienota!', // Paziņojuma teksts
        confirmButtonText: 'Labi' // Poga, lai aizvērtu paziņojumu
    });
}
// Ja URL satur parametru 'error', apstrādā kļūdas gadījumus
else if (urlParams.has('error')) {
    // Iegūst 'error' parametra vērtību
    const error = urlParams.get('error');
    // Pārbauda, vai kļūdas veids ir 'userexists' (lietotājevārds jau
eksistē)
    if (error === 'userexists') {
        Swal.fire({
            icon: 'error', // Sarkanā kļūdas ikona
            title: 'Kļūda!', // Paziņojuma virsraksts
            text: 'Lietotājevārds jau eksistē!', // Paziņojuma teksts
            confirmButtonText: 'Labi' // Poga, lai aizvērtu paziņojumu
        });
    }
    // Ja kļūdas veids nav norādīts, parāda vispārīgu kļūdas paziņojumu
    else {
        Swal.fire({
            icon: 'error', // Sarkanā kļūdas ikona
            title: 'Kļūda!', // Paziņojuma virsraksts
            text: 'Radās problēma pievienojot filiāli!', // Paziņojuma
teksts
            confirmButtonText: 'Labi' // Poga, lai aizvērtu paziņojumu
        });
    }
}
}

```

```

/* Nodzēš noklusējuma margin un padding visiem elementiem un iestata box-
sizing uz border-box */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
/* Stilizē body (galveno lapas sadaļu) */
body {
    display: flex; /* Izmanto flexbox, lai centrētu saturu */
    justify-content: center; /* Centrē saturu horizontāli */
    align-items: center; /* Centrē saturu vertikāli */
    min-height: 100vh; /* Nodrošina, ka lapas augstums ir vismaz 100% no
skatloga augstuma */
    background-color: #f4f4f9; /* Iestata pelēkzilu fona krāsu */
    font-family: Arial, sans-serif; /* Norāda fontu, ko izmantot visā lapā
*/
}
/* Kontainers pieteikšanās formai */
.login-container {
    width: 400px; /* Konteinera platums */
    padding: 30px; /* Iekšējā atstarpe */
    background-color: #fff; /* Baltas krāsas fons */
    border-radius: 15px; /* Noapaļoti stūri */
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); /* Viegls ēnojums, kas
piešķir 3D efektu */
    text-align: center; /* Centrē tekstu konteinerī */
}
/* Pieteikšanās formas virsraksts */
.login-container h1 {
    color: #ff8a42; /* Oranža krāsa */
    font-size: 28px; /* Liela fonta izmērs */
    margin-bottom: 20px; /* Apakšējā atstarpe no nākamā elementa */
}
/* Stilizē formu tekstu nosaukumus (label) */
form label {
    display: block; /* Padara elementu bloka tipa */
    text-align: left; /* Labo teksta izlīdzinājumu */
    margin: 10px 0 5px; /* Iestata augšējo un apakšējo atstarpi */
    font-size: 14px; /* Mazs fonta izmērs */
    font-weight: bold; /* Padara tekstu treknrakstā */
    color: #333; /* Tumši pelēka krāsa */
}
/* Stilizē ievades laukus lietotājvārdam un parolei */
form input[type="text"],
form input[type="password"] {
    width: 100%; /* Ievades lauks aizņem visu pieejamo platumu */
    padding: 10px; /* Iekšējā atstarpe */
    margin-bottom: 20px; /* Apakšējā atstarpe starp laukiem */
    border: 1px solid #ddd; /* Pelēka apmale */
    border-radius: 5px; /* Nedaudz noapaļoti stūri */
    font-size: 16px; /* Vidēja izmēra fonts */
}
/* Stilizē pogu vispārīgi */
button {
    width: 100%; /* Aizņem visu pieejamo platumu */
    padding: 12px; /* Iekšējā atstarpe */
    margin-top: 10px; /* Augšējā atstarpe no citiem elementiem */
    font-size: 16px; /* Vidēja izmēra fonts */
    font-weight: bold; /* Padara tekstu treknrakstā */
}

```

```
    color: #fff; /* Teksta krāsa balta */
    background-color: #ff8a42; /* Oranža fona krāsa */
    border: none; /* Noņem apmali */
    border-radius: 20px; /* Noapaļoti stūri */
    cursor: pointer; /* Maina kursoru uz pointeri, lai norādītu, ka poga ir
klikšķināma */
    transition: background-color 0.3s ease; /* Nodrošina gludu pāreju,
mainot pogas krāsu */
}
/* Kad lietotājs uzbrauc ar peli uz pogas, tā maina krāsu */
button:hover {
    background-color: #ff7a30; /* Tumšāka oranža krāsa */
}
/* Stilizē speciālas pogas ar `type="button"` */
button[type="button"] {
    background-color: #ff8a42; /* Oranža fona krāsa */
    color: #fff; /* Baltas krāsas teksts */
}
/* Kad lietotājs uzbrauc ar peli uz speciālās pogas, tā maina krāsu */
button[type="button"]:hover {
    background-color: #ff7a30; /* Tumšāka oranža krāsa */
}
```

```

<?php
// Izveido savienojumu ar datubāzi
$conn = new mysqli("localhost", "root", "", "parkingpro");
// Pārbauda, vai savienojums ar datubāzi ir izdevies
if ($conn->connect_error) {
    // Ja savienojums neizdodas, izvada kļūdas ziņojumu un pārtrauc skriptu
    izpildi
        die("Savienojuma kļūda: " . $conn->connect_error);
}
// Pārbauda, vai pieprasījums tika veikts ar POST metodi
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Iegūst un attīra ievadītos datus no POST pieprasījuma
    $id = $_POST['id']; // Filiāles unikālais ID
    $branch_name = strtoupper(trim($_POST['branch_name'])); // Filiāles
    nosaukums (pārvērsts par lielajiem burtiem un attīrīts no liekajām
    atstarpēm)
    $username = trim($_POST['username']); // Lietotājvārds, attīrīts no
    liekajām atstarpēm
    $password = trim($_POST['password']); // Parole, attīrīta no liekajām
    atstarpēm
    // Sagatavo SQL vaicājumu, lai atjauninātu filiāles informāciju
    $sql = "UPDATE branches SET branch_name = ?, username = ?, password = ?
    WHERE id = ?";
    $stmt = $conn->prepare($sql); // Sagatavo vaicājumu, lai izvairītos no
    SQL injekcijām
    // Piesaista ievadītos datus SQL vaicājumam
    $stmt->bind_param("sssi", $branch_name, $username, $password, $id);
    // Izpilda SQL vaicājumu un pārbauda, vai tas tika veiksmīgi izpildīts
    if ($stmt->execute()) {
        // Ja atjaunināšana bija veiksmīga, parādesē uz "lookbranch.php"
        ar parametru edited=1
        header("Location: ../lookbranch.php?edited=1");
        exit();
    } else {
        // Ja atjaunināšana neizdevās, parādesē uz "lookbranch.php" ar
        parametru edited=0
        header("Location: ../lookbranch.php?edited=0");
        exit();
    }
}
?>

```

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8 -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
<!-- Pielāgo lapas izskatu mobilajām ierīcēm -->
<title>PPRO - admin</title> <!-- Lapas nosaukums -->
<!-- Ikona, kas tiek attēlota pārlūkprogrammas cilnē -->
<link rel="icon" type="image/png" href="img/pic.png">
<!-- Pievieno CSS failu, kas nosaka lapas dizainu -->
<link rel="stylesheet" href="css/styles.css">
<!-- Pievieno SweetAlert2 bibliotēku paziņojumiem un apstiprinājumiem -
-->
    <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
</head>
<body>
    <!-- Konteiners, kas satur tabulu ar pievienotajām filiālēm -->
    <div class="form-container">
        <h2>Pievienotās filiāles</h2>
        <!-- Tabula, kas attēlo filiāļu datus -->
        <table>
            <thead>
                <tr>
                    <th>FILIĀLE</th>
                    <th>LIETOTĀJVĀRDS</th>
                    <th>PAROLE</th>
                    <th>DARBĪBAS</th>
                </tr>
            </thead>
            <tbody>
                <?php
                    // Izveido savienojumu ar datubāzi
                    $conn = new mysqli("localhost", "root", "", "parkingpro");
                    // Pārbauda, vai savienojums izdevies, ja nē - pārtrauc
                    izpildi un izvada kļūdu
                    if ($conn->connect_error) {
                        die("Savienojuma kļūda: " . $conn->connect_error);
                    }
                    // SQL vaicājums, lai atlasītu visas filiāles, izņemot
                    galveno (id = 1)
                    $sql = "SELECT * FROM branches WHERE id != 1";
                    $result = $conn->query($sql);
                    // Cikls, kas izvada katru filiāli tabulā
                    while ($row = $result->fetch_assoc()):
                        ?>
                        <tr>
                            <!-- Izvada filiāles nosaukumu, lietotājvārdu un paroli
                            -->
                            <td><?= htmlspecialchars($row['branch_name']); ?></td>
                            <td><?= htmlspecialchars($row['username']); ?></td>
                            <td><?= htmlspecialchars($row['password']); ?></td>
                            <td>
                                <!-- Poga, lai rediģētu filiāles datus -->
                                <button class="edit-btn"
                                    onclick="window.location.href='php/editbranch.php?id=<?= $row['id'];
                                    ?>'>Rediģēt</button>
                                <!-- Poga, lai dzēstu filiāli ar apstiprinājumu -->
                                <button class="delete-btn"
                                    onclick="deleteBranch(<?= $row['id']; ?>)">Dzēst</button>
                            </td>
                        </tr>
                    endwhile;
                </tbody>
            </table>
        </div>
    </body>
</html>

```

```
        </tr>
        <?php endwhile; ?>
    </tbody>
</table>
<!-- Poga, lai atgrieztos atpakaļ uz iestatījumu lapu -->
<button
onclick="window.location.href='settings.html'">Atcelt</button>
</div>
<!-- Pievieno JavaScript failu, kas pārvalda filiāļu dzēšanu -->
<script src="js/lookbranch.js"></script>
</body>
</html>
```

23. pielikums **lookbranch.js**

```

// Izveido URL parametru objektu, lai varētu pārbaudīt un iegūt vērtības no
pašreizējā URL
const urlParams = new URLSearchParams(window.location.search);
// Pārbauda, vai URL ir parametrs 'edited'
if (urlParams.has('edited')) {
  // Iegūst 'edited' parametra vērtību
  const edited = urlParams.get('edited');
  // Ja edited = '1', tas nozīmē, ka dati veiksmīgi laboti
  if (edited === '1') {
    Swal.fire({
      icon: 'success', // Zaļa veiksmes ikona
      title: 'Veiksmīgi!', // Paziņojuma virsraksts
      text: 'Dati veiksmīgi laboti!', // Paziņojuma teksts
      confirmButtonText: 'Labi' // Poga, lai aizvērtu paziņojumu
    });
  }
  // Ja edited = '0', tas nozīmē, ka radās kļūda, labojot datus
  else if (edited === '0') {
    Swal.fire({
      icon: 'error', // Sarkanā kļūdas ikona
      title: 'Kļūda!', // Paziņojuma virsraksts
      text: 'Radās problēma labojot datus!', // Paziņojuma teksts
      confirmButtonText: 'Labi' // Poga, lai aizvērtu paziņojumu
    });
  }
}
// Funkcija filiāles dzēšanai, kas pieņem filiāles ID kā argumentu
function deleteBranch(id) {
  // Izsauc SweetAlert2 paziņojumu ar apstiprinājuma dialogu
  Swal.fire({
    title: 'Vai tiešām vēlaties dzēst šo filiāli?', // Brīdinājuma
    virsraksts
    icon: 'warning', // Dzeltenā brīdinājuma ikona
    showCancelButton: true, // Parāda "Atcelt" pogu
    confirmButtonText: 'Dzēst', // Apstiprinājuma pogas teksts
    cancelButtonText: 'Atcelt' // Atcelšanas pogas teksts
  }).then((result) => {
    // Pārbauda, vai lietotājs apstiprināja dzēšanu
    if (result.isConfirmed) {
      // Pāradresē uz `deletebranch.php`, nododot filiāles ID
      dzēšanai
      window.location.href = `php/deletebranch.php?id=${id}`;
    }
  });
}

```

```

<?php
// Izveido savienojumu ar datubāzi
$conn = new mysqli("localhost", "root", "", "parkingpro");
// Pārbauda, vai savienojums ar datubāzi ir veiksmīgi izveidots
if ($conn->connect_error) {
    die("Savienojuma kļūda: " . $conn->connect_error); // Ja savienojums
neizdevās, izvada kļūdas ziņojumu un aptur skriptu
}
// Pārbauda, vai URL pieprasījumā ir norādīts 'id' parametrs (filiāles ID)
if (isset($_GET['id'])) {
    $id = $_GET['id']; // Iegūst filiāles ID no URL
    // Sagatavo SQL vaicājumu, lai dzēstu konkrēto filiāli no datubāzes
    $sql = "DELETE FROM branches WHERE id = ?";
    // Sagatavo SQL vaicājumu, lai novērstu SQL injekcijas
    $stmt = $conn->prepare($sql);
    // Piesaista ID kā integer tipu
    $stmt->bind_param("i", $id);
    // Izpilda vaicājumu un pārbauda, vai tas tika veiksmīgi izpildīts
    if ($stmt->execute()) {
        // Ja filiāle veiksmīgi izdzēsta, parādesē lietotāju uz
'lookbranch.php' ar parametru deleted=1
        header("Location: ../lookbranch.php?deleted=1");
        exit(); // Pārtrauc skripta izpildi pēc parādesācījas
    } else {
        // Ja dzēšana neizdevās, parādesē uz 'lookbranch.php' ar parametru
deleted=0
        header("Location: ../lookbranch.php?deleted=0");
        exit();
    }
} else {
    // Ja 'id' nav norādīts, arī parādesē uz 'lookbranch.php' ar parametru
deleted=0
    header("Location: ../lookbranch.php?deleted=0");
    exit();
}
?>

```

```

<?php
// Izveido savienojumu ar datubāzi
$conn = new mysqli("localhost", "root", "", "parkingpro");
// Pārbauda, vai savienojums ar datubāzi ir veiksmīgs
if ($conn->connect_error) {
    die("Savienojuma kļūda: " . $conn->connect_error); // Ja savienojums
neizdevās, izvada kļūdas ziņojumu un pārtrauc skripta izpildi
}
// Pārbauda, vai URL pieprasījumā ir norādīts 'id' parametrs
if (isset($_GET['id'])) {
    $id = $_GET['id']; // Iegūst filiāles ID no URL
    // Sagatavo SQL vaicājumu, lai atlasītu konkrēto filiāli pēc ID
    $sql = "SELECT * FROM branches WHERE id = ?";
    $stmt = $conn->prepare($sql); // Sagatavo SQL vaicājumu, lai novērstu
SQL injekcijas
    $stmt->bind_param("i", $id); // Piesaista ID kā integer tipu
    $stmt->execute(); // Izpilda vaicājumu
    $result = $stmt->get_result(); // Iegūst rezultātu no vaicājuma
    $branch = $result->fetch_assoc(); // Iegūst rezultātu kā asociatīvo
masīvu
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8 -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
<!-- Nodrošina responsīvu dizainu -->
    <title>Rediģēt filiāli</title> <!-- Lapas nosaukums -->
    <link rel="stylesheet" href="../css/styles.css"> <!-- Pievieno ārējo
CSS failu, kas nosaka lapas dizainu -->
</head>
<body>
    <div class="form-container"> <!-- Kontainers, kas satur formu -->
        <h2>Rediģēt filiāli</h2> <!-- Lapas virsraksts -->
        <!-- Forma filiāles datu rediģēšanai -->
        <form action="savebranch.php" method="POST">
            <!-- Slēptais lauks, kas saglabā filiāles ID, lai zinātu, kuru
ierakstu atjaunināt -->
            <input type="hidden" name="id" value="<?=
htmlspecialchars($branch['id']); ?>">
            <!-- Ievades lauks filiāles nosaukuma rediģēšanai -->
            <label for="branch_name">Filiāles nosaukums:</label>
            <input type="text" id="branch_name" name="branch_name"
value="<?= htmlspecialchars($branch['branch_name']); ?>" required>
            <!-- Ievades lauks lietotājevārdam -->
            <label for="username">Lietotājevārds:</label>
            <input type="text" id="username" name="username" value="<?=
htmlspecialchars($branch['username']); ?>" required>
            <!-- Ievades lauks parolei -->
            <label for="password">Parole:</label>
            <input type="text" id="password" name="password" value="<?=
htmlspecialchars($branch['password']); ?>" required>
            <!-- Poga formas iesniegšanai, lai saglabātu izmaiņas -->
            <button type="submit">Saglabāt</button>
        </form>
    </div>
</body>
</html>

```

```

<?php
// Sāk sesiju, lai varētu izmantot sesijas mainīgos
session_start();
// Datubāzes savienojuma iestatījumi
$host = "127.0.0.1"; // Datubāzes servera adrese (lokālais serveris)
$username = "root"; // Datubāzes lietotājvārds
$password = ""; // Datubāzes parole (tukša pēc noklusējuma)
$dbname = "parkingpro"; // Datubāzes nosaukums
// Izveido savienojumu ar datubāzi
$conn = new mysqli($host, $username, $password, $dbname);
// Pārbauda, vai savienojums izveidots veiksmīgi
if ($conn->connect_error) {
    die("Savienojuma kļūda: " . $conn->connect_error); // Ja kļūda, izvada
    ziņojumu un aptur skriptu
}
// Iegūst datus no POST pieprasījuma un attīra tos
$id = $_POST['id']; // Filiāles ID
$branch_name = strtoupper(trim($_POST['branch_name'])); // Noņem liekās
    atstarpes un pārvērš lielajiem burtiem
$username = trim($_POST['username']); // Noņem liekās atstarpes no
    lietotājvārda
$password = trim($_POST['password']); // Noņem liekās atstarpes no paroles
// Sagatavo SQL vaicājumu, lai atjauninātu filiāles informāciju
$sql = "UPDATE branches SET branch_name = ?, username = ?, password = ?
    WHERE id = ?";
// Sagatavo un aizsargā vaicājumu pret SQL injekcijām
$stmt = $conn->prepare($sql);
$stmt->bind_param("sssi", $branch_name, $username, $password, $id);
// Izpilda vaicājumu un pārbauda, vai tas tika veiksmīgi izpildīts
if ($stmt->execute()) {
    // Ja atjaunināšana veiksmīga, pāradresē lietotāju uz "lookbranch.php"
    ar veiksmes ziņojumu
    header("Location: ../lookbranch.php?success=branch_updated");
} else {
    // Ja atjaunināšana neizdevās, pāradresē ar kļūdas paziņojumu
    header("Location: ../lookbranch.php?error=update_failed");
}
// Aizver sagatavoto vaicājumu un datubāzes savienojumu
$stmt->close();
$conn->close();
?>

```

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Pārbauda, vai sesijas mainīgais 'branch_name' ir iestatīts un vai
lietotājs nav administrators (user_id == 1)
if (!isset($_SESSION['branch_name']) || $_SESSION['user_id'] == 1) {
    // Ja lietotājs nav pieteicies vai ir administrators, pāradresē uz
    pieteikšanās lapu
    header("Location: login.html");
    exit(); // Pārtrauc turpmāko koda izpildi
}
?>
<!DOCTYPE html>
<html lang="lv">
<head>
    <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8 -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- Pielāgo lapas izskatu mobilajām ierīcēm -->
    <title>PPRO - <?php echo htmlspecialchars($_SESSION['branch_name']);
?></title> <!-- Dinamisks lapas nosaukums ar filiāles nosaukumu -->
    <!-- Ikona, kas tiek attēlota pārlūkprogrammas cilnē -->
    <link rel="icon" type="image/png" href="img/pic.png">
    <!-- Pievieno ārējo stila lapu, kas nosaka lapas dizainu -->
    <link rel="stylesheet" href="css/styles.css">
</head>
<body>
    <!-- Kontainers, kas satur lietotāja interfeisu -->
    <div class="form-container">
        <!-- Uzņēmuma logotips -->
        
        <!-- Dinamisks sveiciens teksts ar filiāles nosaukumu -->
        <h2>SVEIKS <?php echo htmlspecialchars($_SESSION['branch_name']);
?>!</h2>
        <!-- Navigācijas pogas filiāles funkcionalitātei -->
        <div class="navigation">
            <!-- Poga, lai reģistrētu jaunu transportlīdzekli -->
            <button
onclick="window.location.href='registerbranch.php'">Reģistrēt
transportlīdzekli</button>
            <!-- Poga, lai apskatītu jau reģistrētos transportlīdzekļus -->
            <button
onclick="window.location.href='viewbranch.php'">Reģistrētie
transportlīdzekļi</button>
            <!-- Poga, lai izrakstītos no sistēmas -->
            <button
onclick="window.location.href='php/logout.php'">Atslēgties</button>
        </div>
    </div>
</body>
</html>

```

```

/* Vispārīgi stili visiem HTML elementiem */
* {
    margin: 0; /* Noņem ārējo malu */
    padding: 0; /* Noņem iekšējo malu */
    box-sizing: border-box; /* Padara elementa izmēru vieglāk prognozējamu
*/
}
/* Body - galvenais kontainers */
body {
    display: flex; /* Aktivē Flexbox izkārtojumu */
    justify-content: center; /* Centrē saturu horizontāli */
    align-items: center; /* Centrē saturu vertikāli */
    min-height: 100vh; /* Minimālais augstums ir 100% no redzamā laukuma */
    background-color: #f4f4f9; /* Gaiši pelēks fona tonis */
    font-family: Arial, sans-serif; /* Fonta tips visam saturam */
}
/* Kontainers formai */
.form-container {
    width: 600px; /* Fiksēts platums */
    padding: 30px; /* Iekšējā mala */
    background-color: #fff; /* Balts fons */
    border-radius: 15px; /* Noapaļoti stūri */
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1); /* Viegla ēna */
    text-align: center; /* Centrēts teksts */
}
/* Attēls formā */
.form-container img {
    max-width: 70%; /* Maksimālais platums - 70% no konteinera */
    margin-bottom: 1px; /* Neliels attālums no nākamā elementa */
}
/* Formas virsraksts */
.form-container h2 {
    margin-bottom: 15px;
    color: #ff8a42; /* Oranžs tonis */
}
/* Izvietojums divās kolonnās */
.form-grid {
    display: grid;
    grid-template-columns: 1fr 1fr; /* Divas vienādas kolonnas */
    gap: 20px; /* Atstarpe starp laukiem */
}
/* Mazāks virsraksts, piemēram, sadaļām */
h3 {
    margin-top: 20px;
    margin-bottom: 10px;
    color: #333;
}
/* Etiķetes ievades laukiem */
label {
    display: block;
    margin-top: 10px;
    font-size: 14px;
    color: #333;
}
/* Ievades lauki */
input[type="text"],
input[type="email"],
input[type="tel"],
input[type="datetime-local"] {
    width: 100%;

```

```

padding: 10px;
margin: 10px 0;
border: 1px solid #ddd;
border-radius: 5px;
font-size: 16px;
}
/* Standarta poga */
button {
width: 100%;
padding: 12px;
margin-top: 15px;
background-color: #ff8a42;
color: #fff;
font-size: 16px;
font-weight: bold;
border: none;
border-radius: 20px;
cursor: pointer;
transition: background-color 0.3s ease;
}
/* Hover efekts pogai */
button:hover {
background-color: #ff7a30;
}
/* Tabulas stils */
table {
width: 100%;
margin-top: 20px;
border-collapse: collapse; /* Noņem dubultās līnijas */
table-layout: fixed; /* Vienmērīgs kolonnu platums */
}

/* Tabulas šūnas (gan virsraksta, gan datu) */
table th, table td {
padding: 12px;
border-bottom: 1px solid #ddd;
text-align: left;
font-size: 16px;
color: #333;
word-wrap: break-word; /* Atdala garus vārdus */
white-space: normal;
}
/* Tabulas virsraksti */
table th {
background-color: #ff8a42;
color: #fff;
font-weight: bold;
text-transform: uppercase; /* Pārvērš burtus lielajos */
}
/* Katrs otrais rindas fons */
table tr:nth-child(even) {
background-color: #f9f9f9;
}
/* Pogas tabulas šūnās */
table button {
padding: 8px 12px;
margin: 0 5px;
font-size: 14px;
font-weight: bold;
border: none;
border-radius: 5px;

```

```

        cursor: pointer;
        transition: background-color 0.3s ease;
    }
    /* Rediģēšanas poga tabulā */
    table button.edit-btn {
        background-color: #ff8a42;
        color: #fff;
    }
    table button.edit-btn:hover {
        background-color: #ff7a30;
    }
    /* Dzēšanas poga tabulā */
    table button.delete-btn {
        background-color: #ff4c4c;
        color: #fff;
    }
    table button.delete-btn:hover {
        background-color: #e60000;
    }
    /* Dublikāts body stilam (atkārtojas, bet ar nelielām atšķirībām) */
    body {
        font-family: Arial, sans-serif;
        background-color: #f9f9f9;
        margin: 0;
        padding: 0;
        display: flex;
        justify-content: center;
        align-items: center;
        height: 100vh;
    }
    /* Vēl viens stils form-container klasei */
    .form-container {
        background-color: white;
        border-radius: 10px;
        box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
        padding: 20px 30px;
        width: 500px;
    }
    /* Formas virsraksts */
    .form-title {
        text-align: center;
        color: #ff6f00;
        margin-bottom: 20px;
    }
    /* Formas struktūra - kolonnas */
    .form {
        display: flex;
        flex-direction: column;
        gap: 15px;
    }
    /* Formas sadaļa */
    .form-section {
        margin-bottom: 20px;
    }
    /* Formas sadaļas virsraksts */
    .form-section-title {
        font-size: 16px;
        font-weight: bold;
        margin-bottom: 10px;
        color: #333;
    }
}

```

```

/* Etikete ievades laukiem (atkārtojums ar niansēm) */
label {
    display: block;
    font-size: 14px;
    margin-bottom: 5px;
    color: #555;
}
/* Universālais ievades lauks */
input {
    width: 100%;
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 5px;
    font-size: 14px;
}
/* Fokusēts ievades lauks */
input:focus {
    outline: none;
    border-color: #ff6f00;
    box-shadow: 0 0 3px rgba(255, 111, 0, 0.5);
}
/* Pogu izkārtojums - divas kolonnas */
.form-buttons {
    display: flex;
    justify-content: space-between;
}
/* Universāla poga */
.btn {
    padding: 10px 20px;
    border: none;
    border-radius: 5px;
    font-size: 14px;
    cursor: pointer;
}
/* Saglabāšanas poga */
.btn-save {
    background-color: #ff6f00;
    color: white;
}
.btn-save:hover {
    background-color: #e65c00;
}
/* Atcelšanas poga */
.btn-cancel {
    background-color: #ddd;
    color: black;
}
.btn-cancel:hover {
    background-color: #bbb;
}
/* Ziņojuma bloks */
.message {
    margin-top: 20px;
    text-align: center;
    font-weight: bold;
}

```

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Iekļauj datubāzes savienojuma failu
include('db.php');
// Pārbauda, vai lietotājs ir pieteicies
if (!isset($_SESSION['username'])) {
    // Ja lietotāja sesijas dati nav pieejami, saglabā kļūdas ziņojumu un
    // pāradresē uz pieteikšanās lapu
    $_SESSION['message'] = 'Lietotāja identifikators nav atrasts. Lūdzu,
    piesakieties vēlreiz.';
    $_SESSION['message_type'] = 'error';
    header('Location: ../login.html');
    exit();
}
// Saglabā pašreizējā lietotāja lietotājvārdu
$username = $_SESSION['username'];
// Iegūst un attīra ievades datus no POST pieprasījuma, izmantojot `trim()`
// nevajadzīgu atstarpju noņemšanai
$name = trim($_POST['name'] ?? '');
$surname = trim($_POST['surname'] ?? '');
$email = trim($_POST['email'] ?? '');
$phone = trim($_POST['phone'] ?? '');
$car_number = trim($_POST['car_number'] ?? '');
$car_model = trim($_POST['car_model'] ?? '');
$start_time = $_POST['start_time'] ?? '';
$end_time = $_POST['end_time'] ?? '';
// Masīvs, kas saglabās validācijas kļūdas
$errors = [];
// Pārbauda, vai visi lauki ir aizpildīti
if (empty($name) || empty($surname) || empty($email) || empty($phone) ||
    empty($car_number) || empty($car_model) || empty($start_time) ||
    empty($end_time)) {
    $errors[] = 'Visi lauki ir obligāti jāaizpilda.';
}
// Pārbauda, vai telefona numurs satur tikai ciparus
if (!preg_match('/^\d+$/', $phone)) {
    $errors[] = 'Telefona numuram jābūt tikai cipariem.';
}
// Pārbauda, vai atstāšanas beigu laiks ir pēc sākuma laika
if (strtotime($start_time) >= strtotime($end_time)) {
    $errors[] = 'Atstāšanas laikam līdz jābūt vēlākam par atstāšanas laiku
    no.';
}
// Ja ir kļūdas, saglabā tās sesijā un pāradresē atpakaļ uz reģistrācijas
// lapu
if (!empty($errors)) {
    $_SESSION['message'] = implode(' ', $errors); // Apvieno kļūdas vienā
    // ziņojumā
    $_SESSION['message_type'] = 'error';
    $_SESSION['form_data'] = $_POST; // Saglabā lietotāja ievadītos datus,
    // lai tie netiktu zaudēti
    header('Location: ../registerbranch.php');
    exit();
}
// Mēģina saglabāt datus datubāzē
try {
    // Sagatavo SQL vaicājumu, lai ievietotu datus `cars` tabulā

```

```

        $stmt = $conn->prepare("INSERT INTO cars (name, surname, email, phone,
car_number, car_model, start_time, end_time, account_name) VALUES (?, ?, ?,
?, ?, ?, ?, ?, ?)");
        // Sasien parametru vērtības ar SQL vaicājumu, lai izvairītos no SQL
injekcijām
        $stmt->bind_param("sssssssss", $name, $surname, $email, $phone,
$car_number, $car_model, $start_time, $end_time, $username);
        // Izpilda vaicājumu un pārbauda, vai tas ir veiksmīgi izpildīts
        if ($stmt->execute()) {
            $_SESSION['message'] = 'Dati veiksmīgi saglabāti!';
            $_SESSION['message_type'] = 'success';
            header('Location: ../registerbranch.php'); // Pāradresē uz
reģistrācijas lapu
        } else {
            // Ja izpilde neizdodas, izmet kļūdu
            throw new Exception('Neizdevās saglabāt datus. Mēģiniet vēlreiz.');
```

```

        }
    } catch (Exception $e) {
        // Ja rodas kļūda, saglabā kļūdas ziņojumu sesijā un pāradresē atpakaļ
        uz reģistrācijas lapu
        $_SESSION['message'] = 'Kļūda: ' . $e->getMessage();
        $_SESSION['message_type'] = 'error';
        $_SESSION['form_data'] = $_POST; // Saglabā lietotāja ievadītos datus
        header('Location: ../registerbranch.php');
```

```

    }
    ?>

```

```

<?php
// Pārbauda, vai pieprasījuma metode ir POST (tas nozīmē, ka dati tiek
sūtīti no formas)
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Iegūst un apstrādā lietotāja ievadītos datus
    $branch_name = strtoupper(trim($_POST['branch_name'])); // Noņem
atstarpes un pārveido par lielajiem burtiem
    $username = trim($_POST['username']); // Noņem atstarpes no
lietotājbārda
    $password = trim($_POST['password']); // Noņem atstarpes no paroles
    // Izveido savienojumu ar datubāzi
    $conn = new mysqli("localhost", "root", "", "parkingpro");
    // Pārbauda, vai savienojums ar datubāzi ir veiksmīgs
    if ($conn->connect_error) {
        die("Savienojuma kļūda: " . $conn->connect_error); // Ja
savienojums neizdevās, izvada kļūdu un pārtrauc skriptu
    }
    // Pārbauda, vai lietotājbārds jau eksistē datubāzē
    $check_user_sql = "SELECT * FROM branches WHERE username = ?";
    $stmt = $conn->prepare($check_user_sql); // Sagatavo SQL vaicājumu, lai
novērstu SQL injekcijas
    $stmt->bind_param("s", $username); // Piesaista lietotājbārdu kā
stringu
    $stmt->execute(); // Izpilda vaicājumu
    $result = $stmt->get_result(); // Iegūst rezultātu
    // Ja lietotājbārds jau pastāv, pāradresē atpakaļ uz filiāles
pievienošanas lapu ar kļūdas ziņojumu
    if ($result->num_rows > 0) {
        header("Location: ../addbranch.html?error=userexists");
        exit();
    }
    // Sagatavo SQL vaicājumu jaunas filiāles pievienošanai
    $sql = "INSERT INTO branches (branch_name, username, password) VALUES
(?, ?, ?)";
    $stmt = $conn->prepare($sql); // Sagatavo vaicājumu
    $stmt->bind_param("sss", $branch_name, $username, $password); //
Piesaista datus SQL vaicājumam
    // Izpilda vaicājumu un pārbauda, vai ieraksts veiksmīgi pievienots
    if ($stmt->execute()) {
        header("Location: ../settings.html?success=1"); // Ja pievienošana
veiksmīga, pāradresē uz iestatījumu lapu
        exit();
    } else {
        header("Location: ../addbranch.html?error=1"); // Ja kļūda,
pāradresē atpakaļ ar kļūdas ziņojumu
        exit();
    }
    // Aizver SQL vaicājumu un datubāzes savienojumu
    $stmt->close();
    $conn->close();
} else {
    // Ja pieprasījuma metode nav POST, pāradresē atpakaļ uz filiāles
pievienošanas lapu
    header("Location: ../addbranch.html");
    exit();
}
?>

```

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Iekļauj datubāzes pieslēguma failu
include('php/db.php');
// Pārbauda, vai lietotājs ir pierakstījies, ja nav - parāda kļūdas
ziņojumu un beidz izpildi
if (!isset($_SESSION['username'])) {
    die("Nav piekļuves tiesību! Lūdzu, piesakieties no jauna.");
}
// Saglabā lietotāja lietotājvārdu no sesijas
$account_name = $_SESSION['username'];
// Iegūst filtrēšanas parametru no URL, ja nav iestatīts - izmanto
noklusējumu "all"
$filter = isset($_GET['filter']) ? $_GET['filter'] : 'all';
// Iegūst pašreizējo lapas numuru no URL, ja nav iestatīts - izmanto
noklusējumu 1
$page = isset($_GET['page']) ? (int)$_GET['page'] : 1;
// Ierakstu skaits, kas tiek rādīts vienā lapā
$recordsPerPage = 5;
// Aprēķina no kura ieraksta sākt vaicājumu
$startIndex = ($page - 1) * $recordsPerPage;
// Iegūst meklēšanas kritēriju no URL
$search = isset($_GET['car_number']) ? $_GET['car_number'] : '';
// SQL vaicājums, lai atlasītu transportlīdzekļus, kas saistīti ar
pierakstīto lietotāju
$query = "SELECT id, name, surname, email, phone, car_number, car_model,
start_time, end_time
FROM cars
WHERE account_name = ?";
// Ja ir ievadīts meklēšanas kritērijs, pievieno meklēšanas nosacījumu
vaicājumam
if (!empty($search)) {
    $query .= " AND car_number LIKE ?";
}
// Pievieno filtrēšanas kritērijus atkarībā no izvēlēta filtra
if ($filter === 'active') {
    $query .= " AND start_time <= NOW() AND end_time >= NOW()";
} elseif ($filter === 'planned') {
    $query .= " AND start_time > NOW()";
} elseif ($filter === 'history') {
    $query .= " AND end_time < NOW()";
}
// Pievieno lapošanas limitu vaicājumam
$query .= " LIMIT ?, ?";
// Sagatavo SQL vaicājumu
$stmt = $conn->prepare($query);
// Ja ir meklēšanas kritērijs, sasien attiecīgos parametrus vaicājumam
if (!empty($search)) {
    $searchParam = "%$search%";
    $stmt->bind_param("ssii", $account_name, $searchParam, $startIndex,
$recordsPerPage);
} else {
    $stmt->bind_param("sii", $account_name, $startIndex, $recordsPerPage);
}
// Izpilda vaicājumu
$stmt->execute();
// Iegūst rezultātus no izpildītā vaicājuma
$result = $stmt->get_result();

```

```

// Vaicājums, lai iegūtu kopējo transportlīdzekļu skaitu lietotāja kontā
(lapošanai)
$totalQuery = "SELECT COUNT(*) as count FROM cars WHERE account_name = ?";
// Pievieno meklēšanas un filtra kritērijus kopējam ierakstu skaitam
if (!empty($search)) {
    $totalQuery .= " AND car_number LIKE ?";
}
if ($filter === 'active') {
    $totalQuery .= " AND start_time <= NOW() AND end_time >= NOW()";
} elseif ($filter === 'planned') {
    $totalQuery .= " AND start_time > NOW()";
} elseif ($filter === 'history') {
    $totalQuery .= " AND end_time < NOW()";
}
// Sagatavo un izpilda kopējo ierakstu skaita vaicājumu
$totalStmt = $conn->prepare($totalQuery);
if (!empty($search)) {
    $totalStmt->bind_param("ss", $account_name, $searchParam);
} else {
    $totalStmt->bind_param("s", $account_name);
}
$totalStmt->execute();
$totalResult = $totalStmt->get_result();
$totalRecords = $totalResult->fetch_assoc()['count'];

// Aprēķina kopējo lapu skaitu, lai izmantotu lapošanai
$totalPages = ceil($totalRecords / $recordsPerPage);
?>
<!DOCTYPE html>
<html lang="lv">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>PPRO - <?php echo htmlspecialchars($_SESSION['branch_name']); ?>
- reģistrētie transportlīdzekļi</title>
    <link rel="icon" type="image/png" href="img/pic.png">
    <link rel="stylesheet" href="css/viewcars.css">
</head>
<body>
    <div class="form-container">
        <h2>REĢISTRĒTIE TRANSPORTLĪDZEKĻI</h2>
        <!-- Filtra pogas, kas ļauj izvēlēties skatāmos transportlīdzekļus
-->

        <!-- Container for action buttons and search -->
<div class="top-controls">
<!-- Top bar: Filter buttons (left) + Search form (right) -->
<div class="top-bar">
    <!-- Filter Buttons -->
    <div class="filter-buttons">
        <a href="?filter=active&page=1" class="filter-btn <?php echo
$filter === 'active' ? 'active' : ''; ?>">Šobrīd aktīvs</a>
        <a href="?filter=planned&page=1" class="filter-btn <?php echo
$filter === 'planned' ? 'active' : ''; ?>">Plānotais</a>
        <a href="?filter=history&page=1" class="filter-btn <?php echo
$filter === 'history' ? 'active' : ''; ?>">Vēsture</a>
        <a href="?filter=all&page=1" class="filter-btn <?php echo $filter
=== 'all' ? 'active' : ''; ?>">Viss</a>
    </div>
    <!-- Search Form -->
    <form method="GET" action="viewbranch.php" class="search-form-inline">

```

```

        <input type="text" name="car_number" placeholder="Meklēt pēc
numura"
        value="<?php echo isset($_GET['car_number']) ?
htmlspecialchars($_GET['car_number']) : ''; ?>"
        <button type="submit">Meklēt</button>
    </form>
</div>

<!-- Tabula, kurā tiek attēloti transportlīdzekļi -->
<table>
    <thead>
        <tr>
            <th>Vārds</th>
            <th>Uzvārds</th>
            <th>E-pasts</th>
            <th>Telefona numurs</th>
            <th>Numura zīme</th>
            <th>Auto modelis</th>
            <th>Atstāšanas laiks no</th>
            <th>Atstāšanas laiks līdz</th>
            <th>Darbības</th>
        </tr>
    </thead>
    <tbody>
        <?php while ($row = $result->fetch_assoc()): ?>
            <tr>
                <td><?php echo htmlspecialchars($row['name']); ?></td>
                <td><?php echo htmlspecialchars($row['surname']);
?></td>

                <td><?php echo htmlspecialchars($row['email']); ?></td>
                <td><?php echo htmlspecialchars($row['phone']); ?></td>
                <td><?php echo htmlspecialchars($row['car_number']);
?></td>

                <td><?php echo htmlspecialchars($row['car_model']);
?></td>

                <td><?php echo htmlspecialchars($row['start_time']);
?></td>

                <td><?php echo htmlspecialchars($row['end_time']);
?></td>

                <td>
                    <!-- Pogu sadaļa transportlīdzekļu rediģēšanai un
dzēšanai -->
                    <button class="edit-btn"
onclick="window.location.href='editcarbr.php?id=<?= $row['id'];
?>'>Rediģēt</button>
                    <button class="delete-btn" onclick="deletecar(<?=
$row['id']; ?>)">Dzēst</button>
                </td>
            </tr>
        <?php endwhile; ?>
    </tbody>
</table>

<!-- Lapu navigācija (lapošana) -->
<div class="pagination">
    <?php for ($i = 1; $i <= $totalPages; $i++): ?>
        <a href="?page=<?php echo $i; ?>" <?php if ($i == $page)
echo 'class="active"'; ?>>
            <?php echo $i; ?>
        </a>
    <?php endfor; ?>
</div>

```

```
        <button  
onclick="window.location.href='branch.php'">Atpakał</button>  
    </div>  
    <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>  
    <script src="js/viewbranch.js"></script>  
</body>  
</html>
```

```

// Funkcija dzēšanai, kas pieņem automašīnas ID kā argumentu
function deletecar(carId) {
  // Izsauc SweetAlert2 paziņojumu ar apstiprinājuma dialogu
  Swal.fire({
    title: "Vai tiešām vēlies dzēst?", // Dialoga virsraksts
    text: "Šo darbību nevar atsaukt!", // Brīdinājuma teksts
    icon: "warning", // Brīdinājuma ikona
    showCancelButton: true, // Parāda "Atcelt" pogu
    confirmButtonColor: "#d33", // Sarkanā krāsa apstiprināšanas pogai
    cancelButtonColor: "#3085d6", // Zilā krāsa atcelšanas pogai
    confirmButtonText: "Jā, dzēst!", // Apstiprinājuma pogas teksts
    cancelButtonText: "Atcelt" // Atcelšanas pogas teksts
  }).then((result) => {
    // Ja lietotājs apstiprina dzēšanu
    if (result.isConfirmed) {
      // Izmanto `fetch()` funkciju, lai nosūtītu dzēšanas
      // pieprasījumu uz `delete.php`
      fetch(`php/delete.php?id=${carId}`)
        .then(response => response.json()) // Pārveido atbildi par
        // JSON
        .then(data => {
          // Ja dzēšana veiksmīga (`data.success` ir true)
          if (data.success) {
            // Parāda paziņojumu par veiksmīgu dzēšanu
            Swal.fire({
              title: "Dati veiksmīgi izdzēsti!", //
              // Paziņojuma virsraksts
              icon: "success", // Veiksmes ikona
              timer: 2000, // Paziņojums pazūd pēc 2 sekundēm
              showConfirmButton: false // Nerāda
              // apstiprināšanas pogu
            }).then(() => {
              // Pēc paziņojuma aizvēršanas pāradresē uz
              // `viewbranch.php`
              window.location.href = "viewbranch.php";
            });
          } else {
            // Ja dzēšana neizdevās, parāda kļūdas paziņojumu
            Swal.fire({
              title: "Kļūda!", // Paziņojuma virsraksts
              text: data.error || "Dzēšana neizdevās!", // Ja
              // serveris atgrieza kļūdu, tā tiek parādīta
              icon: "error" // Kļūdas ikona
            });
          }
        })
        .catch(error => {
          // Ja radās tīkla kļūda, parāda vispārīgu kļūdas
          // paziņojumu
          Swal.fire({
            title: "Kļūda!", // Paziņojuma virsraksts
            text: "Kaut kas nogāja greizi!", // Vispārīgs
            // kļūdas paziņojums
            icon: "error" // Kļūdas ikona
          });
        })
    }
  });
}

```

```

<?php
// Iekļauj datubāzes savienojuma failu
include 'db.php';
// Norāda, ka atbilde būs JSON formātā
header('Content-Type: application/json');
// Pārbauda, vai pieprasījumā ir norādīts 'id' parametrs
if (isset($_GET['id'])) {
    // Konvertē saņemto ID uz veselu skaitli, lai novērstu potenciālus
    drošības draudus
    $id = intval($_GET['id']);
    // SQL vaicājums, lai dzēstu transportlīdzekļa ierakstu pēc ID
    $sql = "DELETE FROM cars WHERE id = ?";
    // Sagatavo SQL vaicājumu, lai izvairītos no SQL injekcijas
    $stmt = $conn->prepare($sql);
    // Piesaista ID kā skaitli (`integer`)
    $stmt->bind_param("i", $id);
    // Izpilda vaicājumu un pārbauda, vai tas ir veiksmīgi izpildīts
    if ($stmt->execute()) {
        // Ja dzēšana veiksmīga, atgriež JSON atbildi ar `success: true`
        echo json_encode(['success' => true]);
    } else {
        // Ja dzēšana neizdevās, atgriež kļūdas ziņojumu JSON formātā
        echo json_encode(['success' => false, 'error' => $conn->error]);
    }
    // Aizver sagatavoto vaicājumu
    $stmt->close();
    // Aizver datubāzes savienojumu
    $conn->close();
} else {
    // Ja 'id' nav norādīts pieprasījumā, atgriež kļūdas paziņojumu JSON
    formātā
    echo json_encode(['success' => false, 'error' => 'Invalid request']);
}
?>

```

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Ieslēdz kļūdu attēlošanu izstrādes nolūkiem
ini_set('display_errors', 1);
error_reporting(E_ALL);
// Iekļauj datubāzes savienojuma failu
include 'php/db.php';
// Pārbauda, vai datubāzes savienojums ir izveidots
if (!isset($conn)) {
    die('Datubāzes savienojums neizdevās. Pārbaudiet db.php ceļu un
konfigurāciju.');
```

```

}
// Iegūst transportlīdzekļa ID no URL parametra
$id = $_GET['id'] ?? null;
// Pārbauda, vai ID ir norādīts
if (!$id) {
    die('Nav norādīts derīgs ID.');
```

```

}
// Sagatavo SQL vaicājumu, lai atlasītu konkrēto transportlīdzekli
$query = "SELECT * FROM cars WHERE id = ?";
$stmt = $conn->prepare($query);
// Pārbauda, vai vaicājums tika sagatavots pareizi
if (!$stmt) {
    die('SQL vaicājuma kļūda: ' . $conn->error);
}
// Sasien ID ar SQL vaicājumu un izpilda to
$stmt->bind_param('i', $id);
$stmt->execute();
$result = $stmt->get_result();
// Pārbauda, vai ieraksts tika atrasts
if ($result->num_rows === 0) {
    die('Ieraksts netika atrasts.');
```

```

}
// Iegūst transportlīdzekļa datus
$car = $result->fetch_assoc();
?>
<!DOCTYPE html>
<html lang="lv">
<head>
    <meta charset="UTF-8"> <!-- Norāda rakstzīmju kodējumu UTF-8 -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- Pielāgo lapas izskatu mobilajām ierīcēm -->
    <title>Rediģēt automašīnu</title> <!-- Lapas nosaukums -->
    <!-- Ikona, kas tiek attēlota pārlūkprogrammas cilnē -->
    <link rel="icon" type="image/png" href="img/pic.png">
    <!-- SweetAlert2 bibliotēka paziņojumu parādīšanai -->
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/sweetalert2@11/dist/sweetalert2.min.css"
>
    <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
    <!-- Stila lapa, kas nosaka lapas vizuālo izskatu -->
    <link rel="stylesheet" href="css/reg.css">
</head>
<body>
    <!-- Konteiners, kas satur rediģēšanas formu -->
    <div class="form-container">
        <h2>REDIĢĒT AUTOMAŠĪNU</h2>
        <!-- Forma, kas nosūta datus uz updatecarbr.php -->
        <form action="php/updatecarbr.php" method="POST">
```

```

        <!-- Slēptais lauks, kas satur transportlīdzekļa ID -->
        <input type="hidden" name="id" value="<?php echo $car['id'];
?>">

        <div class="form-grid">
            <!-- Transportlīdzekļa īpašnieka informācijas sadaļa -->
            <div class="owner-info">
                <h3>Informācija par īpašnieku</h3>
                <label for="name">Vārds:</label>
                <input type="text" id="name" name="name" value="<?php
echo htmlspecialchars($car['name']); ?>" required>
                <label for="surname">Uzvārds:</label>
                <input type="text" id="surname" name="surname"
value="<?php echo htmlspecialchars($car['surname']); ?>" required>
                <label for="email">E-pasts:</label>
                <input type="email" id="email" name="email"
value="<?php echo htmlspecialchars($car['email']); ?>" required>
                <label for="phone">Telefona numurs:</label>
                <input type="tel" id="phone" name="phone" value="<?php
echo htmlspecialchars($car['phone']); ?>" required>
            </div>
            <!-- Transportlīdzekļa informācijas sadaļa -->
            <div class="car-info">
                <h3>Informācija par auto</h3>
                <label for="car_number">Numura zīme:</label>
                <input type="text" id="car_number" name="car_number"
value="<?php echo htmlspecialchars($car['car_number']); ?>" required>
                <label for="car_model">Marka:</label>
                <input type="text" id="car_model" name="car_model"
value="<?php echo htmlspecialchars($car['car_model']); ?>" required>
                <label for="start_time">Atstāšanas laiks no:</label>
                <input type="datetime-local" id="start_time"
name="start_time" value="<?php echo htmlspecialchars(date('Y-m-d\TH:i',
strtotime($car['start_time']))); ?>" required>
                <label for="end_time">Atstāšanas laiks līdz:</label>
                <input type="datetime-local" id="end_time"
name="end_time" value="<?php echo htmlspecialchars(date('Y-m-d\TH:i',
strtotime($car['end_time']))); ?>" required>
            </div>
        </div>
        <!-- Pogu sadaļa -->
        <button type="submit">Saglabāt</button> <!-- Iesniedz datus
rediģēšanai -->
        <button type="button"
onclick="window.location.href='viewbranch.php'">Atpakaļ</button> <!--
Atgriežas uz sarakstu -->
    </form>
</div>
<!-- Paziņojumu attēlošana, ja sesijā ir saglabāti ziņojumi -->
<?php
if (isset($_SESSION['message']) && isset($_SESSION['message_type'])) {
    echo "<script>
        Swal.fire({
            title: '" . ($_SESSION['message_type'] === 'success' ?
'Veiksmīgi!' : 'Kļūda!') . '",
            text: '" . $_SESSION['message'] . '" ,
            icon: '" . $_SESSION['message_type'] . '" ,
            confirmButtonText: 'Labi'
        }).then((result) => {
            if (result.isConfirmed) {
                window.location.href = 'viewbranch.php';
            }
        });
    </script>";
}

```

```
        });  
</script>";  
// Pēc paziņojuma attēlošanas dzēš ziņojumus no sesijas  
unset($_SESSION['message']);  
unset($_SESSION['message_type']);  
}  
?>  
  
</body>  
</html>  
<?php  
// Aizver datubāzes savienojumu  
$conn->close();  
?>
```

```

<?php
// Sāk sesiju, lai piekļūtu sesijas mainīgajiem
session_start();
// Iekļauj datubāzes savienojuma failu
include 'db.php';
// Pārbauda, vai pieprasījuma metode ir POST
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Iegūst datus no POST pieprasījuma, izmantojot null-coalescing
    // operatoru, lai izvairītos no nenoteiktiem mainīgajiem
    $id = $_POST['id'] ?? null;
    $name = $_POST['name'] ?? '';
    $surname = $_POST['surname'] ?? '';
    $email = $_POST['email'] ?? '';
    $phone = $_POST['phone'] ?? '';
    $car_number = $_POST['car_number'] ?? '';
    $car_model = $_POST['car_model'] ?? '';
    $start_time = $_POST['start_time'] ?? '';
    $end_time = $_POST['end_time'] ?? '';
    // Pārbauda, vai visi nepieciešamie lauki ir aizpildīti
    if ($id && $name && $surname && $email && $phone && $car_number &&
    $car_model && $start_time && $end_time) {
        // SQL vaicājums, lai atjauninātu transportlīdzekļa datus datubāzē
        $query = "UPDATE cars SET name = ?, surname = ?, email = ?, phone =
        ?, car_number = ?, car_model = ?, start_time = ?, end_time = ? WHERE id =
        ?";

        // Sagatavo SQL vaicājumu, lai izvairītos no SQL injekcijām
        $stmt = $conn->prepare($query);
        // Pārbauda, vai vaicājums tika sagatavots veiksmīgi
        if ($stmt) {
            // Sasien ievadītos datus ar vaicājumu
            $stmt->bind_param('ssssssssi', $name, $surname, $email, $phone,
            $car_number, $car_model, $start_time, $end_time, $id);
            // Izpilda vaicājumu un pārbauda, vai tas tika veiksmīgi
            izpildīts
            if ($stmt->execute()) {
                $_SESSION['message'] = "Dati veiksmīgi saglabāti!";
                $_SESSION['message_type'] = "success";
            } else {
                $_SESSION['message'] = "Kļūda, saglabājot datus.";
                $_SESSION['message_type'] = "error";
            }
            // Aizver sagatavoto vaicājumu
            $stmt->close();
        } else {
            // Saglabā kļūdas ziņojumu, ja SQL vaicājums nevarēja tikt
            sagatavots
            $_SESSION['message'] = "Kļūda sagatavojot vaicājumu.";
            $_SESSION['message_type'] = "error";
        }
    } else {
        // Saglabā kļūdas ziņojumu, ja kāds no laukiem nav aizpildīts
        $_SESSION['message'] = "Visi lauki ir obligāti jāaizpilda!";
        $_SESSION['message_type'] = "error";
    }
} else {
    // Ja pieprasījums nav veikts ar POST metodi, iestata kļūdas ziņojumu
    $_SESSION['message'] = "Nederīgs pieprasījums.";
    $_SESSION['message_type'] = "error";
}
// Aizver datubāzes savienojumu

```

```
$conn->close();  
// Pāradresē atpakaļ uz transportlīdzekļa rediģēšanas lapu ar attiecīgo ID  
header("Location: ../editcarbr.php?id=$id");  
exit;  
?>
```

```
<?php
// Sāk sesiju, lai piekļūtu un pārvaldītu sesijas mainīgos
session_start();
// Iznīcina pašreizējo sesiju un dzēš visus sesijas mainīgos
session_destroy();
// Pāradresē lietotāju uz sākumlapu (index.php) pēc izrakstīšanās
header("Location: ../index.php");
// Pārtrauc skripta izpildi, lai pārliecinātos, ka netiek izpildīts nekas
cits
exit();
?>
```

```
<?php
// Norāda datubāzes servera adresi (localhost / vietējais serveris)
$servername = "127.0.0.1";
// Datubāzes lietotājvārds (noklusējuma iestatījums XAMPP un MAMP vidē ir
"root")
$username = "root";
// Datubāzes nosaukums, kurā tiks veikti pieprasījumi
$dbname = "parkingpro";
// Datubāzes parole (noklusējuma iestatījumos XAMPP/MAMP ir tukša)
$password = "";
// Izveido savienojumu ar datubāzi, izmantojot `mysqli` klasi
$conn = new mysqli($servername, $username, $password, $dbname);
// Pārbauda, vai savienojums ar datubāzi ir izdevies
if ($conn->connect_error) {
    // Ja savienojums neizdodas, izvada kļūdas ziņojumu un aptur skriptu
    die("Savienojuma kļūda: " . $conn->connect_error);
}
// Ja savienojums ir veiksmīgi izveidots, kods turpina darboties bez kļūdas
// paziņojuma
?>
```