Modinātājs/taimeris

Funkcionālās prasības

\*\*No klienta puses:\*\*

1. \*\*Modinātāja funkcija:\*\*

- Iespēja iestatīt vairākus modinātājus.

- Iespēja izvēlēties dažādus modinātāja signālus (tostarp lietotāja augšupielādētus skaņu failus).

- Atlikšanas (snooze) funkcija ar pielāgojamu atlikšanas laiku.

- Iespēja iestatīt atkārtotus modinātājus (ikdienas, nedēļas utt.).

2. \*\*Taimera funkcija:\*\*

- Iespēja iestatīt vairākus taimerus vienlaicīgi.

- Skaņas signāls, kad taimeris beidzas.

- Atpakaļskaitīšanas vizualizācija.

3. \*\*Papildu funkcionalitāte:\*\*

- Integrācija ar kalendāru un notikumiem.

- Iespēja iestatīt modinātāju ar balsi (balss atpazīšanas funkcija).

- Intuitīvs un viegli lietojams interfeiss.

\*\*No programmētāja puses:\*\*

1. \*\*Modinātāja funkcija:\*\*

- Modinātāju pārvaldības sistēma ar CRUD (Create, Read, Update, Delete) operācijām.

- Laika apstrādes modulis, kas atbalsta vairākas laika zonas.

2. \*\*Taimera funkcija:\*\*

- Neatkarīgu taimeru pārvaldība, kas spēj darboties vienlaicīgi.

- Notikumu dzinējs, kas izsauc noteiktas darbības, kad taimeris beidzas.

3. \*\*Papildu funkcionalitāte:\*\*

- API integrācija ar kalendāra pakalpojumiem.

- Balss atpazīšanas modulis.

- UX/UI dizaina vadlīnijas ievērošana, balstoties uz klienta sniegtajiem prototipiem vai stila vadlīnijām.

Nefunkcionālās prasības

\*\*No klienta puses:\*\*

1. \*\*Veiktspēja:\*\*

- Modinātājam jādarbojas bez kavēšanās un precīzi iestatītajā laikā.

- Taimera atpakaļskaitīšana jābūt reāllaikā, bez aizkaves.

2. \*\*Lietojamība:\*\*

- Lietotājam draudzīgs interfeiss.

- Intuitīva navigācija un skaidri saprotami paziņojumi.

3. \*\*Drošība:\*\*

- Lietotāja datu aizsardzība, īpaši balss komandu un personīgo skaņu failu gadījumā.

\*\*No programmētāja puses:\*\*

1. \*\*Veiktspēja:\*\*

- Lietojumprogrammai jābūt optimizētai, lai tā darbotiesu efektīvi arī uz vecākiem vai mazāk jaudīgiem ierīcēm.

- Minimāls resursu patēriņš, lai saglabātu akumulatora darbības laiku.

2. \*\*Mērogojamība:\*\*

- Kods jāstrukturē tā, lai to varētu viegli paplašināt ar jaunām funkcijām.

- API jābūt modulāram, lai atvieglotu integrāciju ar citām sistēmām.

3. \*\*Uzturēšana:\*\*

- Kods jābūt labi dokumentētam, lai atvieglotu uzturēšanu un turpmākus uzlabojumus.

- Jāizstrādā automatizēti testi, lai nodrošinātu lietojumprogrammas stabilitāti un ātru kļūdu novēršanu.