

Liepājas Valsts tehnikums  
  
  
  
  
**ProgressKeeper**  
  
Kvalifikācijas eksāmena praktiskās daļas tehniskā dokumentācija

Izglītības programma **33484011 Programmēšana**  
  
Profesionālā kvalifikācija **Programmēšanas tehniķis**

Darba autors: Lukass Gabriels Siliņš

Eksāmena datums 2024. Gada \_\_.jūnijs

Liepāja 2024

Saturs

[Ievads 1](#_Toc193663496)

[1. Uzdevuma noformulējums 1](#_Toc193663497)

[2. Programmatūras prasību specifikācija 2](#_Toc193663498)

[2.1. Produkta perspektīva 2](#_Toc193663499)

[2.2. Sistēmas funkcionālās prasības 2](#_Toc193663500)

[2.2.1. Jauna treniņa izveidošana 2](#_Toc193663501)

[2.2.2. Treniņam ir automātiski jātiek pievienotam tai dienai 2](#_Toc193663502)

[2.2.3. Vingrinājuma izvēle no dotā saraksta 3](#_Toc193663503)

[2.2.4. Personiska vingrinājuma izveide 3](#_Toc193663504)

[2.2.5. Iepriekšēju treniņu nokopēšana uz šobrīdējo dienu 3](#_Toc193663505)

[2.2.6. Treniņu attēlošana kalendārā 4](#_Toc193663506)

[2.2.7. Kalendāra apskatīšana 4](#_Toc193663507)

[2.2.8. Treniņu apskatīšana kalendāra logā 4](#_Toc193663508)

[2.2.9. Treniņa nosaukuma izveide 4](#_Toc193663509)

[2.2.10. Zvaigžņu sistēma kas liecina par progresu 5](#_Toc193663510)

[2.2.11. Pamācības sadaļa 5](#_Toc193663511)

[2.2.12. Iknedēļas zvaigžņu notifikācija 5](#_Toc193663512)

[2.2.13. Neaktivitātes notifikācija 5](#_Toc193663513)

[2.2.14. Senāka treniņa datu rediģēšana 5](#_Toc193663514)

[2.2.15. Piegājiena izdzēšana 6](#_Toc193663515)

[2.2.16. Vingrinājuma izdzēšana 6](#_Toc193663516)

[2.2.17. Treniņa izdzēšana 6](#_Toc193663517)

[2.2.18. Vingrinājumu filtrēšana 6](#_Toc193663518)

[2.2.19. Treniņu atgādinājums 7](#_Toc193663519)

[2.2.20. Piegājiena izlaišana 7](#_Toc193663520)

[2.2.21. Vingrinājuma izlaišana 7](#_Toc193663521)

[2.2.22. Svara pierakstīšana 7](#_Toc193663522)

[2.2.23. Atkārtojuma reižu pierakstīšana 8](#_Toc193663523)

[2.3. Sistēmas nefunkcionālās prasības 8](#_Toc193663524)

[2.4. Gala lietotāja raksturiezīmes 8](#_Toc193663525)

[3. Izstrādes līdzekļu, rīku apraksts un izvēles pamatojums 9](#_Toc193663526)

[3.1. Izvēlēto risinājuma līdzekļu un valodas apraksts 9](#_Toc193663527)

[3.2. Iespējamo (alternatīvo) risinājuma līdzekļu un valodas apraksts 11](#_Toc193663528)

[4. Sistēmas modelēšana un projektēšana 13](#_Toc193663529)

[4.1. Sistēmas struktūras modelis 13](#_Toc193663530)

[4.1.1. Izvietojuma (Deployment) diagramma 13](#_Toc193663531)

[4.1.2. Klašu (Class) diagramma 14](#_Toc193663532)

[4.2. Funkcionālais un dinamiskais sistēmas modelis 15](#_Toc193663533)

[4.2.1. Aktivitāšu (Activity) diagramma 15](#_Toc193663534)

[3. attēls. Aktivitāšu diagramma 15](#_Toc193663535)

[4.2.2. Lietojumgadījumu (Use Case) diagramma 16](#_Toc193663536)

[4. attēls. Lietojumgadījumu diagramma 16](#_Toc193663537)

[4.2.3. Stāvokļu (State) diagramma 17](#_Toc193663538)

[5. attēls. Stāvokļa diagramma 17](#_Toc193663539)

[4.3. Datu struktūru apraksts 18](#_Toc193663540)

[5. Lietotāju ceļvedis 19](#_Toc193663541)

[6. Testēšanas dokumentācija 19](#_Toc193663542)

[6.1. Izvēlētās testēšanas metodes, rīku apraksts un pamatojums 19](#_Toc193663543)

[6.2. Testpiemēru kopa 19](#_Toc193663544)

[6.3. Testēšanas žurnāls 19](#_Toc193663545)

[7. Secinājumi 19](#_Toc193663546)

[8. Lietoto terminu un saīsinājumu skaidrojumi 19](#_Toc193663547)

[9. Literatūras un informācijas avotu saraksts 19](#_Toc193663548)

# Ievads

Šajā dokumentā tiks aprakstīts viss par manu projektu “ProgressKeeper”. Dokumentā ietilps vairākas sadaļas, kā uzdevuma noformējums, kur aprakstīšu sasniedzamo rezultātu, kā šis rezultāts tiks sasniegts, kā es konstatēšu ka uzdevuma mērķis ir sasniegts un būs paskaidrota produkta nepieciešamība. Programmatūras prasību specifikācijā tiks aprakstīta produkta perspektīva, funkcionālās prasības, nefunkcionālās prasības un lietotāja raksturiezīmes. Izstrādes līdzekļu aprakstā es aprakstīšu un pamatošu savu rīku izvēli, un iespējamās alternatīvas, ko var izmantot. Sistēmas modelēšana un projektēšanas sadaļā es aprakstīšu sistēmas struktūru ar diagrammām. Šeit ietilps gan sistēmas struktūras modelis, gan funkcionālais un dinamiskais modelis, kā arī datu struktūras apraksts. Šajā dokumentā arī tiks parādīts sistēmas ceļvedis, kas nodrošinās sistēmas izpratni jebkuram. Dokumentā arī ietilpst testēšanas dokumentācija, kur tiek aprakstītas izvēlētās metodes, rīku apraksts un to izvēles pamatojums. Tiks pieminēti alternatīvi rīki un metodes, kā arī testpiemēri un testēšanas žurnāls. Secinājumos tiek aprakstītas sastaptās problēmas un mani izaicinājumi, savi sasniegumi un turpmākie mērķi, kas ir saistīti ar šo projektu. Sadaļā par lietotajiem saīsinājumiem un terminu skaidrojumiem tiek parādīti lietotie saīsinājumi un tiek paskaidroti pilnībā visi lietotie termini. Šie saīsinājumi un termini ir ievietoti tabulā. Avotu sarakstā tiek pieminēti visi izmantotie un apmeklētie avoti projekta izstrādes laikā. Pielikumā ir pievienots ~200 komandrindu garš koda fragments, kas ir komentēts un labi saskatāms.

Mans izvēlētais projekts ir aplikācija cilvēkiem, kas grib saglabāt/aplūkot progresu, ko viņi sasniedz svara zālē. Lietotājs varētu izveidot jaunu, no pieejamiem vingrinājumiem vai pievienot/izveidot citu, un to pievienot tās dienas treniņam treniņu, vai uz to dienu nokopēt iepriekšēju treniņu. Iepriekšējo dienu/nedēļu/mēnešu treniņus varētu redzēt kalendārā. Būtu pamācoša sadaļa, kurā būtu pamācības/video par vingrinājumiem.

# Uzdevuma noformulējums

“ProgressKeeper” ir aplikācija android vidē, kas ir paredzēta priekš cilvēkiem, kas iet uz svara zāli un grib pārskatīt/saglabāt treniņu procesu. Šī aplikācija eksistē, lai cilvēkiem būtu viegli pārskatāma vide par viņu progresu/ iepriekšējiem treniņiem.

Aplikācija sastāv no vairākām daļām. Galvenā no tām ir jauna treniņa pievienošanas sadaļa, kur lietotājs var izvēlēties jebkuru muskuļu grupu, un izvēlēties kādu no piedāvātajiem vingrinājumiem, vai arī izveidot jaunu. Kad ir izvēlēts kāds vingrinājums, tas tiek automātiski pievienots šai dienai. Otra galvenā sadaļa ir nokopēt iepriekšējo treniņu, kur, ar kalendāra palīdzību var redzēt dienas, kad treniņš ir bijis paveikts. Iepriekšējos treniņus var gan apskatīt, gan nokopēt uz šobrīdējo dienu. Aplikācijai ir jautājuma sadaļa, kur tiek pastāstīts gan rakstiskā, gan video formātā par visiem vingrinājumiem, kurus pats lietotājs nav pievienojis. Ja lietotājam ir labāks progress nekā iepriekšējā treniņā (svara vai atkārtojuma daudzumā) parādās zvaigznīte pie specifiskā treniņa, kas liecina par progresu. Lietotājam iknedēļu atnāks notifikācija par to, cik zvaigznes viņš caur nedēļu ir sakrājis, kā arī cik daudz treniņi lietotājam ir bijuši nedēļas laikā.

Iknedēļas notifikācijai par progresu tiks izmantots algoritms, kas sekos līdzi tam, cik zvaigznītes lietotājs ir saņēmis nedēļas laikā. Līdzīgs algoritms tiks izmantots notifikācijai, kas liecina par nedēļas apmeklējuma reizēm. Šis algoritms arī skatīsies, vai 2 nedēļu laikā nav bijis neviens treniņš, ja tas apstiprinās, otrās nedēļas beigās būs atgādinājuma veida notifikācija. Lietotājs arī varēs pievienot atgādinājumu, lai viņam atnāktu paziņojums par to, ka ir jāapmeklē treniņš

Rezultāts tiks sasniegts ar android studio palīdzību, kur tiks izmantota Java un Kotlin programmēšanas valodas. Vizuālais skats tiks izveidots acīm tīkams, kā arī viegli pārskatāms, jo šī aplikācija ir kā tavas piezīmes.

Mans mērķis būs tikai sasniegts, kad aplikācija būs pilnībā funkcionāla, kad lietotāji varēs viegli pārskatīt to un gribēs turpināt izmantot.

"ProgressKeeper" ir nepieciešama, jo tā nodrošina pārskatāmu veidu, kā sekot līdzi progresam, novēršot vajadzību pēc “notes” aplikācijām vai papīra pierakstiem. Tā ietver arī jautājumu sadaļu, palīdzot lietotājiem saprast vingrinājumu veikšanu.

# Programmatūras prasību specifikācija

Šai nodaļā tiks aprakstīts par programmu- kam tā ir paredzēta, tā visas funkcionālās un nefunkcionālās prasības, kā arī tiks aprakstītas programmas lietotāju raksturiezīmes, kas paskaidros kādi lietotāji izmantotu šo programmu.

## Produkta perspektīva

“ProgressKeeper” aplikācija ir paredzēta vieglai svara zāles treniņu apskatei, progresa saglabāšanai un tā līdzās sekošanai:

* Lietotāja treniņu datu pārvaldība (treniņu izveide, vingrinājumu pievienošana, datu rediģēšana, datu dzēšana)
* Lietotāja progresa vizuāla parādīšana (zvaigznītes sistēma)
* Notifikācijas par progresu vai treniņu neapmeklēšanu (iknedēļas / ik 2 nedēļu notifikācijas sistēma)

## Sistēmas funkcionālās prasības

### 2.2.1. Jauna treniņa izveidošana

Mērķis: Funkcija ir nepieciešama, lai lietotājs varētu izveidot jaunu treniņu.

Apstrādei: Nospiežot pogu “Create a new workout” lietotājam parādās jauns logs, kur ir pieejamas visas muskuļu grupas.

Izvaddati: 1) Paziņojums par treniņa izveidošanu

### 2.2.2. Treniņam ir automātiski jātiek pievienotam tai dienai

Mērķis: Funkcija ir nepieciešama, lai lietotāja jaunais treniņš automātiski tiktu pievienots tai dienai.

Apstrādei: Nospiežot pogu “Create a new workout” sistēma automātiski pievieno lietotāja treniņu tagadējai dienai, kā arī tas parādās kalendārā.

### 2.2.3. Vingrinājuma izvēle no dotā saraksta

Mērķis: Funkcija ir nepieciešama, lai lietotājs varētu izvēlēties jaunu vingrinājumu, ko pievienot klāt pie tās dienas treniņa.

Apstrādei: Nospiežot uz kādu no vingrinājumiem, tas tiek automātiski pievienots pie tās dienas treniņa.

### 2.2.4. Personiska vingrinājuma izveide

Mērķis: Funkcija ir nepieciešama, lai lietotājs varētu izveidot savu vingrinājumu, kas nav starp jau dotajiem aplikācijas vingrinājumiem.

Ievaddati:

**1. tabula**

**Lietotāja ievadītie dati**

|  |  |  |
| --- | --- | --- |
| Nosaukums | Obligāts | Piezīmes |
| Name | Jā | Līdz 40 rakstzīmēm |
| Category | Jā | Var būt tik 1 kategorija, kur iekļaujas |

Apstrādei: Kad lietotājs ieraksta informāciju, viņš nospiež apstiprināšanas pogu, kas šo vingrinājumu pievieno specifiskajai vingrinājumu kategorijai.

Izvaddati: 1) Paziņojums, ka obligāti ir jāizpilda lauki;

2) Paziņojums par vienādu treniņa nosaukumu;

3) Paziņojums par vingrinājuma veiksmīgu izveidi.

### 2.2.5. Iepriekšēju treniņu nokopēšana uz šobrīdējo dienu

Mērķis: Funkcija ir nepieciešama, lai lietotājs varētu nokopēt iepriekš izveidotu treniņu uz šobrīdējo dienu.

Apstrādei: Kad lietotājs ir uzspiedis “Copy previous workout” pogu, viņam parādās kalendārs. Ja tiek izvēlēta diena, kad iepriekš ir bijis treniņš viņam parādās 2 pogas: “copy” un “cancel”, ja tiek uzspiesta poga “copy”, tad tās dienas treniņš tiek nokopēts uz šobrīdējo dienu, tai skaitā piegājieni un svars, ar ko iepriekšējā treniņā lietotājs darbojās. Ja tiek uzspiesta poga “cancel”, tad izvēle tiek atcelta.

Izvaddati: 1) Paziņojums par veiksmīgi nokopētu treniņu.

### 2.2.6. Treniņu attēlošana kalendārā

Mērķis: Funkcijas mērķis ir attēlot visas dienas, kad kāds treniņš ir darīts. Dienas kad treniņš ir darīts, kalendārā tiek apzīmētas ar zaļu krāsu.

Apstrādei: Kad lietotājs ir izveidojis/nokopējis treniņu, kalendārā, tās dienas datums, tiek automātiski iezīmēts zaļā krāsā.

Izvaddati: 1) Datums tiek iekrāsots zaļā krāsā.

### 2.2.7. Kalendāra apskatīšana

Mērķis: Funkcijas mērķis ir attēlot kalendāru, kad lietotājs nospiež kalendāra pogu.

Apstrādei: Kad lietotājs uzspiež kalendāra pogu, viņam tiek parādīts, jaunā logā, mēneši kuros ir izveidoti treniņi.

### 2.2.8. Treniņu apskatīšana kalendāra logā

Mērķis: Funkcijas mērķis ir apskatīt jebkura treniņa paveikumus, kamēr lietotājs atrodas kalendāra logā.

Apstrādei: Kad lietotājs atrodas kalendāra logā, un uzspiež uz kāda no treniņiem, viņam atveras miniatūras logs, kurā ir redzami visi tā treniņa vingrinājumi, kā arī piegājieni un svara daudzums.t

Izvaddati: 1) Parādās miniatūras logs kur tiek attēlots treniņš.

### 2.2.9. Treniņa nosaukuma izveide

Mērķis: Funkcijas mērķis ir dot iespēju lietotājam nosaukt treniņu savā vēlamajā nosaukumā.

Ievaddati:

**2. tabula**

**Lietotāja nosaukuma ievade**

|  |  |  |
| --- | --- | --- |
| Nosaukums | Obligāts | Piezīmes |
| Workout name | Nē | Līdz 40 rakstzīmēm |

Apstrādei: Kad lietotājs ir ievadījis treniņa nosaukumu ekrāna augšpusē, tas automātiski saglabājas, kad tiek lietotājs uzspiež uz ko citu / kad aiziet uz citu logu.

### 2.2.10. Zvaigžņu sistēma kas liecina par progresu

Mērķis: Funkcijas mērķis ir parādīt, vizuāla veidā, ka lietotājs ir sasniedzis progresu, kopš sava iepriekšējā labākā rezultāta.

Apstrādei: Kad lietotājs pieraksta labāku rezultātu- atkārtojuma vai svara veidā, pie tā piegājiena parādās zvaigznīte, kas lietotājam parāda, ka viņš ir progresējis.

### 2.2.11. Pamācības sadaļa

Mērķis: Funkcijas mērķis ir sadaļa, kur tiek detalizēti aprakstīts, kā arī video formātā parādīts, kā darīt katru no vingrinājumiem, kurus lietotājs pats nav pievienojis.

Apstrādei: Kad lietotājs uzspiež uz pamācības sadaļas, viņam parādās līdzīgs logs vingrinājuma izvēles logam, tik kad lietotājs uzspiež uz jebkādu vingrinājumu, viņam parādās logs, kur ir gan teksta formātā, gan video formātā parādīts, kā paveikt vēlamo vingrinājumu.

### 2.2.12. Iknedēļas zvaigžņu notifikācija

Mērķis: Funkcijas mērķis ir dot lietotājam paziņojumu par to, cik zvaigznes viņš nedēļas laikā ir sasniedzis.

Apstrādei: Aplikācija skatās uz to, cik daudzas reizes nedēļā ir iegūta zvaigzne pie kāda treniņa. Aplikācija šo informāciju apkopo un nedēļas beigās parāda to lietotājam.

Izvaddati: 1) Paziņojums/notifikācija parāda cik zvaigznes lietotājs nedēļā ir dabūjis.

### 2.2.13. Neaktivitātes notifikācija

Mērķis: Funkcijas mērķis ir atgādināt lietotāju par to, ka 2 nedēļu laikā nav pievienots neviens jauns treniņš.

Apstrādei: Aplikācija skatās uz to, vai 2 nedēļu periodā tiek pievienots jauns treniņš kalendāram. Ja nav pat 1 pievienots, lietotājam paradās atgādinājums par neaktivitāti.

Izvaddati: 1) Paziņojums/notifikācija, kas atgādina lietotājam, ka 2 nedēļas nav trenējies.

### 2.2.14. Senāka treniņa datu rediģēšana

Mērķis: Funkcijas mērķis ir ļaut lietotājam rediģēt informāciju par senāku treniņu.

Apstrādei: Kad lietotājs izvēlas vecāku treniņu, viņš var izmainīt tā informāciju, kā atkārtojuma reizes vai svara daudzumu.

### 2.2.15. Piegājiena izdzēšana

Mērķis: Atļaut lietotājam izdzēst kādu piegājienu pie specifiskā vingrinājuma.

Apstrādei: Kad lietotājs izvēlas kādu piegājienu no kāda no saviem vingrinājumiem. Uzspiežot uz šī piegājiena, parādās poga “delete”, uzspiežot šo pogu, piegājiens tiek izdzēsts.

Izvaddati: 1) Dzēšanas apstiprināšanas paziņojums;

2) Veiksmīgas dzēšanas paziņojums.

### 2.2.16. Vingrinājuma izdzēšana

Mērķis: Atļaut lietotājam izdzēst kādu vingrinājuma pie specifiska treniņa.

Apstrādei: Kad lietotājs izvēlas kādu vingrinājumu no kāda no saviem treniņiem ko grib dzēst, viņam ir jāiziet atpakaļ uz pilno dienas skatu, un jāpatur pirksts uz vingrinājumu, ko viņš grib dzēst. Kad tas ir izdarīts, ekrāna augšā parādās dzēšanas poga, ja tā tiek nospiesta, vingrinājums tiek izdzēsts.

Izvaddati: 1) Dzēšanas apstiprināšanas paziņojums;

2) Veiksmīgas dzēšanas paziņojums.

### 2.2.17. Treniņa izdzēšana

Mērķis: Atļaut lietotājam izdzēst kādu specifisku treniņu.

Apstrādei: Kad lietotājs izvēlas izdzēst kādu treniņu, viņam ir jāiziet uz kalendāra skatu, un jāpatur pirksts uz treniņa, ko viņš vēlas dzēst. Kad tas ir izdarīts, ekrāna augšā parādās dzēšanas poga, ja tā tiek nospiesta, treniņš tiek izdzēsts.

Izvaddati: 1) Dzēšanas apstiprināšanas paziņojums;

2) Veiksmīgas dzēšanas paziņojums.

### 2.2.18. Vingrinājumu filtrēšana

Mērķis: Funkcija ir nepieciešama, lai filtrēt vingrinājumus pēc to vārdiem, kaut ja muskuļu grupas atšķiras.

Apstrādei: Kad lietotājs pievieno jaunu vingrinājumu, virs visām muskuļu grupām ir meklēšanas josla, kurā ierakstot kādu vārdu, izmetīsies visi vingrinājumi, kuros ir iekļauts šis vārds.

### 2.2.19. Treniņu atgādinājums

Mērķis: Funkcija ir nepieciešama, lai lietotājs varētu sev iepriekš atgādināt, ka ir jāapmeklē/jāizveido treniņš kādā specifiskā dienā, stundā, minūtē.

Apstrādei: Kad lietotājs nospiež atgādinājuma pogu, atveras jauns logs, kurā ir iespēja iestatīt atgādinājumu pēc specifiska laika, vai arī uz specifiku dienu un laiku.

Izvaddati: 1) Paziņojums, kas parādās specifiski noteiktajā laikā.

### 2.2.20. Piegājiena izlaišana

Mērķis: Funkcija ir nepieciešama, lai lietotājs varētu atzīmēt piegājienu, ko uz todienu bija izlaidis, bet zina, ka nākotnē neizlaidīs. Kā arī ar šo funkciju nav nepieciešams lieki dzēst piegājienu.

Apstrādei: Kad lietotājs uzspiež “skip” pogu, kas ir labajā pusē piegājienam, piegājiens tiek atzīmēts kā būtu izlaists.

### 2.2.21. Vingrinājuma izlaišana

Mērķis: Funkcija ir nepieciešama, lai lietotājs varētu atzīmēt vingrinājumu, ko uz todienu bija izlaidis, bet zina, ka nākotnē neizlaidīs. Kā arī ar šo funkciju nav nepieciešams lieki dzēst vingrinājumu.

Apstrādei: Kad lietotājs uzspiež “skip” pogu, kas ir labajā augšējā pusē pie vingrinājuma, tas tiek atzīmēts kā būtu izlaists.

### 2.2.22. Svara pierakstīšana

Mērķis: Funkcija ir nepieciešama, lai lietotājs, kad ir izveidojis jaunu treniņu un vingrinājumu, varētu pierakstīt svara daudzumu pie vingrinājuma pēc sava piegājiena.

Ievaddati:

**3. tabula**

**Lietotāja ieraksti**

|  |  |
| --- | --- |
| Nosaukums | Obligāts |
| Weight | Jā |

Apstrādei: Kad lietotājs ir pierakstījis gan svaru, gan piegājienu skaitu, ir iespējams saglabāt piegājienu, kad nospiež pogu “save”.

### 2.2.23. Atkārtojuma reižu pierakstīšana

Mērķis: Funkcija ir nepieciešama, lai lietotājs, kad ir izveidojis jaunu treniņu un vingrinājumu, varētu pierakstīt atkārtojuma reižu daudzumu pie vingrinājuma pēc sava piegājiena.

Ievaddati:

**3. tabula**

**Lietotāja ieraksti**

|  |  |
| --- | --- |
| Nosaukums | Obligāts |
| Reps | Jā |

Apstrādei: Kad lietotājs ir pierakstījis gan svaru, gan piegājienu skaitu, ir iespējams saglabāt piegājienu, kad nospiež pogu “save”.

## Sistēmas nefunkcionālās prasības

1. Aplikācijai jāstrādā arī bez interneta savienojuma;
2. Aplikācijai ir jābūt saderīgai ar vismaz Android 8.0 versiju, kā arī ar jaunākām;
3. Visu funkciju responsivitātei ir jābūt zem 1 sekundes garai;
4. Aplikācijai jāizmanto pēc iespējas mazāk operatīvās atmiņas, ne vairāk kā 300MB
5. Aplikācijas dizainam ir jābūt viegli pārskatāmam, ar neuzkrītošam krāsām/dizainu;
6. Jānodrošina iespēju nomainīt aplikācijas valodu(Angļu, Krievu un Latviešu)

## Gala lietotāja raksturiezīmes

Lietotāja vecums būtu starp 14 līdz 60 gadiem, kas nodarbojas ar sportu / fiziskām aktivitātēm un ir motivēts šīs aktivitātes turpināt/sekot līdzi savam progresam. Lietotājs var būt sporta iesācējs kurš tikko ir sācis sportot, vai kāds kas vēlas sevi uzturēt savā šobrīdējā formā, iespējams ka šāds lietotājs sporto tik reizi vai divas nedēļā, vai pat kāds pieredzējis sportists, kurš nedēļā sporto praktiski katru dienu. Lietotājs var būt gan sieviete, gan vīrietis. Protams, primāri, aplikācija ir domāta cilvēkiem, kas iet uz svara zāli, bet pašam lietotājam nav jāiet uz svara zāli, lai izmantotu aplikāciju, jo ir iekļauti vingrinājumi ko viņi var mājās, vai citviet paši darīt- skriešana, piepumpēšanās, pievilkšanās, un lietotājs pats var izveidot savu vingrinājumu un pielikt pie sava treniņa. Lietotāja galvenais aplikācijas izmantošanas mērķis ir sekot līdzi, kā arī viegli pārskatīt savus treniņus un savu progresu vienviet. Lietotājs izmantotu šo aplikāciju uz android viedtālruņa, vai android planšetdatora, jo vispasaulē ap 70% cilvēku viedierīces izmanto android operētājsistēmu.

# Izstrādes līdzekļu, rīku apraksts un izvēles pamatojums

Šajā nodaļā es aprakstīšu savus risinājumu līdzekļus, manis izvēlētās programmēšanas valodas, kā arī alternatīvus risinājuma līdzekļus un valodas.

## 3.1. Izvēlēto risinājuma līdzekļu un valodas apraksts

**Android Studio**: Ir šī projekta izvēlētā programmēšanas vide. Android studio ir populārākā programmēšanas vide priekš android aplikāciju izveides, kā arī to oficiāli atbalsta Google. Tā ir bezmaksas aplikācija. Tiek atbalstītas 4 programmēšanas valodas- Kotlin, Java, XML un C++. Vidē ir iebūvēts android emulators, kas var palaist aplikācijas uz virtuāla telefona. Vidē ir iebūvēts IntelliJ koda palīgs, kas atvieglo darbu/koda rakstīšanu, jo var pabeigt kodu, kā arī piedāvāt risinājumus un koda pārbaudi.

Izvēles pamatojums: Šo vidi izvēlējos, jo tā nodrošina visu nepieciešamo, lai izveidotu android aplikāciju. Android studio vidē var veidot, testēt un pat publicēt aplikāciju, kā arī ir pieejamas daudzas bibliotēkas. Vēl viens iemesls bija tas, ka iepriekš ir bijusi pieredze ar Java programmēšanas valodu, un Kotlin, kas ir primārā android studio programmēšanas valoda, un ko es izmantošu, sintakse ir līdzīga Javai, kas manu darbu atvieglos. Par Android studio internetā ir daudzi resursi, kā arī video par aplikāciju projektiem un risinājumiem, kas kādu problēmu vai aplikācijas izstrādi var paātrināt.

**Git**: Ir šī projekta versionēšanas programma. Git ir pasaules populārākā versionēšanas programma, kuru izmanto miljoniem lietotāju un uzņēmumu. Git piedāvā uzticamu un ātru versionēšanu, kā arī ir iespēja izveidot “atzarus” ar kuriem viegli var apvienot vairākas projekta versijas, atgriezties uz vecāku projekta versiju vai arī vienlaicīgi sadarboties ar vairākiem cilvēkiem pie 1 projekta. Git ir bezmaksas aplikācija, kura ir pieejama jebkuram, kā arī to var integrēt ar populārām izstrādes vidēm, kā Android Studio, Visual Studio Code u.c.

Izvēles pamatojums: Es izvēlējos Git, jo ir bijusi iepriekšēja pieredze ar viņu, kā arī tā izmantošana, manuprāt, ir saprotama un visi pakalpojumi ir bezmaksas, kas studentam nenāk par sliktu. Vēl viens iemesls, kāpēc izvēlējos Git ir, jo izmantoju GitHub, kas ir saite kur Git repozitorijas tiek mitinātas, kā arī var redzēt projekta darbību/papildināšanas vēsturi.

**Github**: Ir šī projekta mitināšanas saite. Github ir pasaulē lielākā projektu mitināšanas saite, ar vairāk kā 100 miljonu lietotāju, un ar vairāk kā 350 miljonu repozitoriju.

Github glabā un pārvalda projektus, tādējādi tiem var piekļūt no jebkuras ierīces, kā arī citi projekta dalībnieki, var piekļūt klāt šim projektam no šīs mājaslapas. Ar GitHub var sekot līdzi projekta izmaiņām, un var pat testēt kodu. GitHub ļauj veidot gan privātas repozitroijas, gan publiskas. GitHub atbalsta samērā daudzus DevOps rīkus, kā Docker, Jenkins, Jira, AWS, u.c.

Izvēles pamatojums: Es izvēlējos izmantot GitHub, jo man ar viņu ir bijusi iepriekšēja pieredze, kā arī izmantoju Git versionēšanas programmu, kas viegli sadarbojas ar Github. Kā ar Git, arī GitHub var izmantot bezmaksas, kā arī GitHub saitei ir viegla saskarsne, kā arī tā ir viegli pārskatāma.

**Kotlin**: Ir programmēšanas valoda, ko es izmantošu šai projekta izstrādei. Kotlin ir samērā jauna programmēšanas valoda, ko izstrādāja JetBrains. Kotlin ir pilnībā savienojama ar Java programmēšanas valodu, tādēļ tā ir piemērota Android aplikāciju izstrādei un citiem programmēšanas projektiem. Kotlin sintakse ir viegli lasāma un ļoti efektīva. Tā ieviesa null-safety drošību, kas novērš biežas kļūdas, ar kurām varēja sastapties, kad izmantoja Java. Kotlin ir ātra un veiktspējīga valoda, kas ir ieviesta kā oficiālā Android izstrādes valoda, kas viņu, android izstrādes vidē, ir padarījis par vispopulārāko.

Izvēles pamatojums: Es izvēlējos Kotlin valodu kā 1 no valodām ko pielietošu projekta izstrādei, jo tās sintakse ir līdzīga Java valodai, ar kuru jau man ir iepriekšēja pieredze, kā arī Kotlin ir oficiālā Android aplikāciju programmēšanas valoda. Tā ir efektīva, moderna ar daudziem interneta resursiem par tās izmantošanu un pat tās izmantošanas apmācību.

**Java**: Ir programmēšanas valoda, ko es izmantošu projekta izstrādei. Java ir pasaulē viena no populārākajām programmēšanas valodām. To izmanto daudzos pielietojumos, kā mājaslapu backend daļās, sistēmās, mākslīgajos intelektos, spēlēs, mākoņdatņošanā, mobilajās aplikācijās u.c. Tai ir plašs bibliotēku un rīku atbalsts, kas atvieglo projektu izstrādi. Java strādā pēc interesanta principa “raksti vienreiz, palaid jebkur”, kas nozīmē ka programmētājs var veidot programmu uz 1 datora, un turpināt veidot to uz cita, bez jebkādām komplikācijām. Java valodai ir plaša kopiena, kā arī dokumentācija un video resursi. Java var vienlaikus strādāt ar Kotlin, kas programmētājiem ir liels bonuss.

Izvēles pamatojums: Java valodu es izmantošu ar kādu bibliotēku, kas vēl nav pārgājusi uz Kotlin valodu. Primāri izmantošu Kotlin, bet Java citviet arī tiks izmantota.

## 3.2. Iespējamo (alternatīvo) risinājuma līdzekļu un valodas apraksts

**Visual Studio Code**: Šī ir alternatīva programmēšanas vide. Visual Studio Code ir pasaulē viena no populārākajām programmēšanas vidēm, ar efektīvu programmēšanas vidi. Tā atbalsta daudzas programmēšanas valodas, rīkus un bibliotēkās. Visual Studio Code ir daudzas funkcijas, kā IntelliSense, kas automātiski pabeidz kodu/ piedāvā risinājumus un parāda kļūdas. Ir iebūvēts terminālis, kur var palaist komandrindas. Iebūvēta Git versionēšana, kas atbalsta Git, GitLab, GitHub u.c. Visual Studio Code ir milzīgs paplašinājumu atbalsts, kur publicētāji var valodām, vai pašai videi izstrādāt kādu paplašinājumu. Ir iebūvēts Live Share rīks, ar ko var vairāki cilvēki, vienlaicīgi, strādāt pie viena projekta. Visual Studio Code arī ir bezmaksas, parasti patērē maz sistēmas resursus kā arī darbojas uz Windows, macOS un Linux operētājsistēmām. Ar Visual Studio Code arī var izstrādāt Android aplikācijas, bet salīdzinot ar Android Studio, kas ir specifiski izstrādāts priekš Android aplikāciju izveides, tas nav vienādā līmenī. Visual Studio Code vajag papildināt ar r;īkiem/bibliotēkām, lai tas strādātu līdzvērtīgi Android Studio.

**JetBrains Intellij IDEA**: Šī ir alternatīva programmēšanas vide. Šī vide ir populārā Java izmantotāju vidē, kā arī šī vide atbalsta citas valodas, kā JavaScript, TypeScript, Python, PHP u.c. Šai videi ir spējīgs atkļūdošanas rīks, kā arī tas piedāvā intuitīvu darba vidi. Šai videi ir pieejamas daudzas bibliotēkas, kas ļauj izstrādāt dažādus projektus, kā aplikācijas, spēles un mājaslapas. Iebūvēta Git versionēšana, kas atbalsta Git. JetBrains ir bezmaksas, bet arī ir pieejama maksas versija, kas piedāvā papildu funkcijas un web izstrādes atbalstu. Kopumā, JetBrains Intellij IDEA ir daudzpusēja vide, ar daudzām izstrādes opcijām.

**Mercurial**: Šī ir alternatīva versionēšanas programma. Šī ir populāra versionēšanas programma, kas strādā līdzīgi kā Git. Katrs projekta izstrādātājs var strādāt viens pats pie savas daļas, un vēlāk apvienot savu daļu ar galveno projektu. Saka, ka šī programma ir pārskatāmāka nekā git, un ir vieglāk saprast kā to izmantot.

**BitBucket**: Šī ir alternatīva mitināšanas saite. BitBucket ir iekšā iebūvēts rīks, kas testē kodu, un to kompilē, tādējādi automātiski veicot testus. Sākotnēji Bitbucket atbalstīja gan Mercurial, gan Git, bet pirms 4 gadiem BitBucket beidza atbalstīt Mercurial, un šobrīdēji, Git ir vienīgā versionēšanas programma, ko tas atbalsta. BitBucket ir vērsts uz komerciāliem projektiem un komandu sadarbību. Ir gan maksas versija, gan bezmaksas versija, kur pie 1 projekta var strādāt līdz 5 cilvēkiem.

**SourceForge**: Šī ir alternatīva mitināšanas saite. SourceForge ir viena no pasaulē populārākajām mitināšanas saitēm. Tā sadarbojas ar daudzām versionēšanas programmām, kā git, mercurial, subversion u.c. SourceForge izmanto kad privātpersonas, gan kompānijas. SourceForge ļauj publicētājiem publicēt projekta vairākas versijas, kā 1.0, 1.1 u.t.t. SourceForge var izmantot bezmaksas.

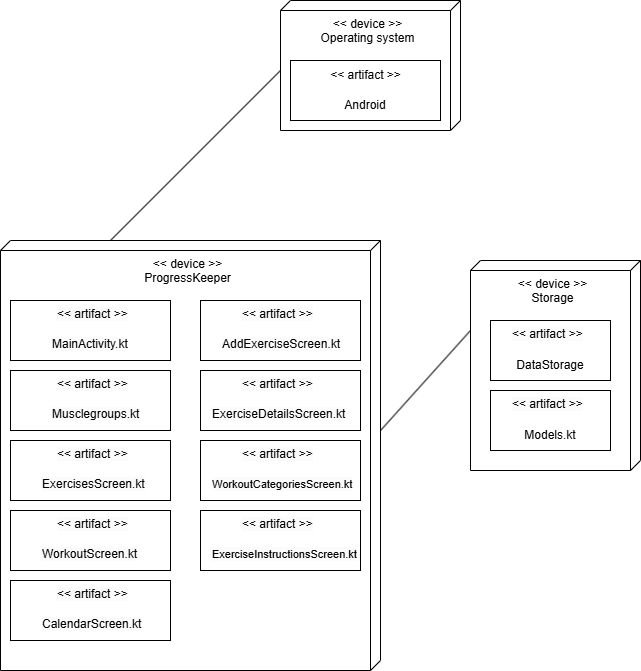
**Dart (ar Flutter):** Šī ir alternatīva programmēšanas valoda. Dart ir moderna programmēšanas valoda, lai veidotu cross-platform aplikācijas. Dart izstrādāja Google. Dart ir īpašības, kā statiskā tipizēšana, kas ļauj atrast kļūdas jau programmas izstrādes laikā, kā arī atbalsta asinhronu programmēšanu. Dart galvenā īpašība ir tā, ka var vienlaikus izveidot apklikāciju uz android un ios vides, bet tad ir jāiekļauj Flutter UI. Flutter var viegli iekļaut starp Dart, un flutter strādā kā programmas vizuālā daļa.

**Javascript (ar React Native):** Šī ir alternatīva programmēšnas valoda. Primāri, JavaScript izmanto tīmekļa vietnēm, bet ar React Native var izmantot JavaScript, lai izveidotu mobilās aplikācijas, kas strādā gan ar Andorid, gan ar iOS ierīcēm. React Native bibliotēku izveidoja Gacebook, lai veidotu mobilās aplikācijas ar React (JavaScript bibliotēku). Javascript tiek izmantots kā loģika (funkcijas, viss kas notiek fonā) kamēr React Native tiek izmantots vizuālajai daļai. Gan JavaScript, gan react Native populāras valodas, kurām ir daudzi resursi, kā arī kopienas. Izmantot šo kombināciju ir parocīgi, jo vajag zināt Javascript, jo React Native ir balstīts uz to, kā arī izstrādātā aplikācija strādās gan uz Android ierīcēm, gan uz iOS ierīcēm.

# Sistēmas modelēšana un projektēšana

## 4.1. Sistēmas struktūras modelis

### 4.1.1. Izvietojuma (Deployment) diagramma



**1. attēls. Šeit tiek attēlota izvietojuma diagramma kas parāda sistēmas komponenšu un artifaktu sadarbību. Iekš ProgressKeeper komponentes ir artifakti, kas šijā piemērā ir visi pieejamie lietotāja skati. Ārpus šīs komponentes ir Storage komponente, kurā ir DaraStorage un Models artifakti, kas ir metodes, kas visu lietotāja sarakstīto informāciju saglabā lietotāja telefona atmiņā. Iekš Operating System komponentes ir android artifakts, uz kā ir visa šī sistēma, lai varētu darbināt aplikāciju.**

### 4.1.2. Klašu (Class) diagramma

**2. attēls. Šeit tiek attēlota sistēmas klašu diagramma, kas parāda visas programmas zemsistēmu. Iekš workout manager ir redzami visi elementi, ko tas manto, kā arī pats izveido sev. Šī būtu galvenā darbības sistēma. Tālāk mums ir Workout, Exercise, Set kas ir mainīgie vingrinājumiem/treniņiem, un tie satur vienu otru un tos satur WorkoutExercise. Tālāk ir attēlots MainActivityScreen, kurā tiek attēloti treniņi. Tad ir ExerciseDetailsScreen kas izmanto WorkoutManager un MainActivity lai attēlotu treniņu. Muscle groups ari izmanto MainActivity, kur attēlo visas muskuļugrupas. CalendarScreen, kas attēlojas MainActivity screen, kur tas attēlo kalendāru un lietotāja iepriekšējos treniņus. Ir ExerciseInstructions ekrāns, kurā ir pamācības specifiskajiem vingrinājumiem.**

## 4.2. Funkcionālais un dinamiskais sistēmas modelis

### 4.2.1. Aktivitāšu (Activity) diagramma 3. attēls. Šī ir aktivitāšu diagramma kur tiek attēlota sistēmas plūsma no vienas aktivitātes ar citu sistēmu vai procesu / parāda visu darbības plūsmu. Tiek parādītas lietotāja pieejamās opcijas (ko viņš var darīt/paveikt aplikācijā. Kā lietotājam ir dažas iespējas, ko var paveikt, kad atver aplikāciju, kā izveidot jaunu treniņu, nokopēt vecāku treniņu vai apskatīties šīs dienas treniņu. Katrai izvēlei ir savs ceļš un citi ceļi “saduras” viens ar otru.

### 4.2.2. Lietojumgadījumu (Use Case) diagramma

### 4. attēls. Šī ir lietojumgadījumu diagramma, kas attēlo lietotāja mijiedarbību ar aplikāciju un kādas darbības lietotājs var tajā veikt. Diagramma palīdz saprast sistēmas darbību / funkcionalitāti no lietotāja puses. Lietotājam ir pieejamas galvenās darbības, kā treniņa / -u apskatīšana, vingrinājumu pievienošana, rediģēšana dzēšana, piegājienu pievienošana, rediģēšana dzēšana un kalendāra apskatīšana.

### 4.2.3. Stāvokļu (State) diagramma 5. attēls. Šī ir stāvokļa diagramma, kas parāda kā aplikācijas objekti maina savu stāvokli atkarībā no notikumiem (sistēmas un/vai lietotāja). Diagramma arī parāda / apraksta kas notiek sistēmā, šo pāreju laikā. Aplikācijas galvenais stāvoklis, ko var redzēt, ir galvenais ekrāns (MainActivity) kas mainās pēc lietotāja darbībām. Piemēram pēc jauna treniņa izveidošanas.

## Datu struktūru apraksts

Lai jebkuram, kas pārskata “ProgressKeeper” aplikāciju, būtu skaidrs tās darbības process un iespējamās darbības, tika izveidotas piecas sistēmas modeļa diagrammas: Izvietojuma (Deployment), Klašu (Class), Aktivitāšu (Activity), Lietojumgadījumu (Use Case) un Stāvokļa (State). Katra no šīm diagrammām parāda skatu par lietotnes uzbūvi, datu plūsmu, funkcionalitāti un darbības iespējām. Kopā, izvēlētās diagrammas atspoguļo pilnu sistēmas modeli, un sistēmas piedāvātās/iespējamās funkcijas.

Izvietojuma diagramma tika izvēlēta, lai varētu attēlot, kā aplikācija darbojas, kā arī ko tā izmanto/dara. Ar šīs diagrammas palīdzību var saskatīt programmas daļu sadalījumu un saistību vienai ar otru.

Klašu diagramma tika izvēlēta, lai parādītu aplikācijas datu struktūru un tās loģiku/darbības procesu. Tā parāda, kā objekti, iekš aplikācijas, sadarbojas, kādas vērtības un metodes tiem ir nepieciešami.

Aktivitāšu diagramma tika izvēlēta, lai attēlotu lietotāja darbību plūsmu, piemēram, kā lietotājs izveido treniņu, izvēlas muskuļa grupas un vingrinājumus tām, savada svara/piegājiena vērtības, vai arī kā viņš nokopē vecāku treniņu. Diagramma palīdz saskatīt iespējamo darba plūsmu un tā rezultātus.

Lietojumgadījumu diagramma tika izvēlēta, jo tā attēlo lietotāja pieejamos scenārijus, piemēram apskata kalendāru, un nokopē vecāku treniņu uz šodienu. Iekš šīs diagrammas ir attēlotas visas svarīgākās darbības.

Stāvokļu diagramma tika izmantota, lai attēlotu lietotnes stāvokļus, piemēram kā galvenā ekrāna stāvokli, kas ir atkarīgs no tā, vai šodienā jau ir izveidots/nokopēts treniņš, vai vēl nē. Šī diagramma ļauj izprast, kā lietotnes stāvokļi mainās, atkarībā no lietotāja izvēlēm.

# 5. Lietotāju ceļvedis

# 6. Testēšanas dokumentācija

## 6.1. Izvēlētās testēšanas metodes, rīku apraksts un pamatojums

## 6.2. Testpiemēru kopa

## 6.3. Testēšanas žurnāls

# 7. Secinājumi

# 8. Lietoto terminu un saīsinājumu skaidrojumi

# 9. Literatūras un informācijas avotu saraksts