

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 2025. Gada \_\_.jūnijs

Liepāja 2025

Saturs

[Ievads 1](#_Toc200222441)

[1. Uzdevuma noformulējums 1](#_Toc200222442)

[2. Programmatūras prasību specifikācija 2](#_Toc200222443)

[2.1. Produkta perspektīva 2](#_Toc200222444)

[2.2. Sistēmas funkcionālās prasības 2](#_Toc200222445)

[2.2.1. Jauna treniņa izveidošana ar pogu “Start new workout” 2](#_Toc200222446)

[2.2.2. Jauna treniņa izveidošana ar pogu “+” galvenes daļā 2](#_Toc200222447)

[2.2.3. Jauna vingrinājuma pievienošana ar “+” pogu 3](#_Toc200222448)

[2.2.4. Treniņš tiek automātiski pievienots attiecīgajai dienai 3](#_Toc200222449)

[2.2.5. Vingrinājuma izvēle no dotā saraksta 3](#_Toc200222450)

[2.2.6. Vingrinājumu filtrēšana no saraksta 3](#_Toc200222451)

[2.2.7. Jauna vingrinājuma izveide 4](#_Toc200222452)

[2.2.8. Eksistējoša vingrinājuma pievienošanas mēģinājums 4](#_Toc200222453)

[2.2.9. Svara pierakstīšana 5](#_Toc200222454)

[2.2.10. Atkārtojuma reižu pierakstīšana 5](#_Toc200222455)

[2.2.11. Vecāka treniņa nokopēšana uz pašreizējo dienu 6](#_Toc200222456)

[2.2.12. Treniņu attēlošana kalendārā 6](#_Toc200222457)

[2.2.13. Kalendāra apskatīšana 6](#_Toc200222458)

[2.2.14. Treniņu apskatīšana kalendāra logā 6](#_Toc200222459)

[2.2.15. Pamācības sadaļa 7](#_Toc200222460)

[2.2.16. Piegājiena izdzēšana 7](#_Toc200222461)

[2.2.17. Vingrinājuma izdzēšana 7](#_Toc200222462)

[2.2.18. Treniņa izdzēšana 7](#_Toc200222463)

[2.2.19. Kopējo treniņu skaita attēlošana 8](#_Toc200222464)

[2.2.20. Ik svētdienas notifikācija par nedēļas treniņu skaitu 8](#_Toc200222465)

[2.2.21. Skats par nedēļas sasniegumiem 8](#_Toc200222466)

[2.3. Sistēmas nefunkcionālās prasības 8](#_Toc200222467)

[2.4. Gala lietotāja raksturiezīmes 9](#_Toc200222468)

[3. Izstrādes līdzekļu, rīku apraksts un izvēles pamatojums 10](#_Toc200222469)

[3.1. Izvēlēto risinājuma līdzekļu un valodas apraksts 10](#_Toc200222470)

[3.2. Iespējamo (alternatīvo) risinājuma līdzekļu un valodas apraksts 12](#_Toc200222471)

[4. Sistēmas modelēšana un projektēšana 14](#_Toc200222472)

[4.1. Sistēmas struktūras modelis 14](#_Toc200222473)

[4.1.1. Izvietojuma (Deployment) diagramma 14](#_Toc200222474)

[4.1.2. Klašu (Class) diagramma 15](#_Toc200222475)

[4.2. Funkcionālais un dinamiskais sistēmas modelis 16](#_Toc200222476)

[4.2.1. Aktivitāšu (Activity) diagramma 16](#_Toc200222477)

[4.2.2. Lietojumgadījumu (Use Case) diagramma 17](#_Toc200222478)

[4. attēls. Lietojumgadījumu diagramma 17](#_Toc200222479)

[4.2.3. Stāvokļu (State) diagramma 18](#_Toc200222480)

[5. attēls. Stāvokļu diagramma 18](#_Toc200222481)

[4.3. Datu struktūru apraksts 19](#_Toc200222482)

[5. Lietotāju ceļvedis 20](#_Toc200222483)

[6. Testēšanas dokumentācija 46](#_Toc200222484)

[6.1. Izvēlētās testēšanas metodes, rīku apraksts un pamatojums 46](#_Toc200222485)

[6.2. Alternatīvas testēšanas metodes un rīki 46](#_Toc200222486)

[6.3. Testēšanas kopa 47](#_Toc200222487)

[6.4. Testēšanas žurnāls 53](#_Toc200222488)

[7. Secinājumi 55](#_Toc200222489)

[8. Lietoto saīsinājumu un terminu skaidrojums 56](#_Toc200222490)

[9. Literatūras un informācijas avotu saraksts 57](#_Toc200222491)

[Pielikumi 58](#_Toc200222492)

# 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 vai aplūkot progresu, ko viņi sasniedz svara zālē. Lietotājs varētu izveidot jaunu treniņu, no pieejamiem vingrinājumiem, vai uz to specifisko dienu nokopēt vecāku treniņu. Iepriekšējo dienu, nedēļu, mēnešu treniņus varētu redzēt kalendārā. Būtu pamācoša sadaļa, kurā būtu pamācības vingrinājumiem.

# Uzdevuma noformulējums

“ProgressKeeper” ir aplikācija Android vidē, kas ir paredzēta priekš cilvēkiem, kas iet uz svara zāli un grib pārskatīt vai saglabāt treniņu procesu. Šī aplikācija eksistē, lai cilvēkiem būtu viegli pārskatāma vide par viņu progresu, kā arī 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 pašreizējo dienu. Aplikācijai ir informatīva sadaļa, kur tiek pastāstīts rakstiskā formātā par vingrinājumu izpildes soļiem, kā arī biežāk pieļautajām kļūdām, ko cilvēki pieļauj vingrinājuma izpildes laikā. Katru svētdienu lietotājs saņems paziņojumu ar informāciju par to, cik treniņi tika veikti nedēļas laikā.

Rezultāts tiks sasniegts ar Android Studio palīdzību, kur tiks izmantota Kotlin programmēšanas valoda. Vizuālais skats tiks izveidots acīm tīkams, kā arī viegli pārskatāms, jo šī aplikācija ir kā tavi pieraksti.

Mans mērķis būs tikai sasniegts, kad aplikācija būs pilnībā funkcionāla, un lietotāji varēs viegli pārskatīt, kā arī izmantot.

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

# Programmatūras prasību specifikācija

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

## Produkta perspektīva

“ProgressKeeper” aplikācija ir paredzēta vieglai un pārskatāmai 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

## Sistēmas funkcionālās prasības

### 2.2.1. Jauna treniņa izveidošana ar pogu “Start new workout”

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

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

Izvaddati: 1) Parādās logs ar visām muskuļu grupām.

### 2.2.2. Jauna treniņa izveidošana ar pogu “+” galvenes daļā

Mērķis: Funkcija ir nepieciešama, lai lietotājs varētu izveidot jaunu treniņu, kaut ja netiek uzspiesta “Create new workout” poga.

Apstrādei: Nospiežot pogu “+” lietotājam parādās jauns logs, kur ir pieejamas visas muskuļu grupas.

Izvaddati: 1) Parādās logs ar visām muskuļu grupām.

### 2.2.3. Jauna vingrinājuma pievienošana ar “+” pogu

Mērķis: Funkcija ir nepieciešama, lai lietotājs varētu pievienot vingrinājumu treniņam.

Apstrādei: Nospiežot pogu “+” lietotājam parādās jauns logs, kur ir pieejamas visas muskuļu grupas.

Izvaddati: 1) Parādās logs ar visām muskuļu grupām.

### 2.2.4. Treniņš tiek automātiski pievienots attiecīgajai dienai

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

Apstrādei: Pēc pogas “Create new workout” vai “+” nospiešanas, muskuļu grupas un vingrinājuma izvēles, treniņš tiek automātiski pievienots pašreizējai dienai un parādās kalendārā.

### 2.2.5. Vingrinājuma izvēle no dotā saraksta

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

Apstrādei: Nospiežot uz kādu no vingrinājumiem, lietotāju aizved uz vingrinājuma logu, kur viņš var izveidot piegājienus.

### 2.2.6. Vingrinājumu filtrēšana no saraksta

Mērķis: Funkcija ir nepieciešama, lai filtrēt vingrinājumus pēc to vārdiem.

Apstrādei: Esot specifiskās muskuļu grupas vingrinājumu skatā, iekš meklēšanas joslas lietotājs ieraksta kādu vingrinājuma nosaukumu. Filtrs strādā ar iekļaušanu.

Izvaddati: 1) Vingrinājumi pazūd, ja neviens nesakrīt ar lietotāja tekstu;

2) Lietotāja meklētais vingrinājums/vingrinājumi parādas.

### 2.2.7. Jauna vingrinājuma izveide

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

Ievaddati:

**1. tabula**

**Lietotāja ievadītie dati**

|  |  |  |
| --- | --- | --- |
| Nosaukums | Obligāts | Piezīmes |
| Name | Jā | Vingrinājuma vārds nevar būt vienāds ar jau eksistējošu vingrinājumu |

Apstrādei: Kad lietotājs ieraksta vingrinājuma nosaukumu, un nospiež apstiprināšanas pogu “Save”, šis vingrinājums tiek pievienots starp pārējiem, dotās muskuļu grupas, vingrinājumiem.

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

2) Vingrinājums tiek pievienots pie pārējiem vingrinājumiem.

### 2.2.8. Eksistējoša vingrinājuma pievienošanas mēģinājums

Mērķis: Funkcija ir nepieciešama, lai lietotājs nevarētu izveidot treniņu ar nosaukumu, kāds jau eksistē.

Ievaddati:

**1. tabula**

**Lietotāja ievadītie dati**

|  |  |  |
| --- | --- | --- |
| Nosaukums | Obligāts | Piezīmes |
| Name | Jā | Vingrinājuma vārds nevar būt vienāds ar jau eksistējošu vingrinājumu |

Apstrādei: Kad lietotājs ieraksta vingrinājuma nosaukumu kas ir vienāds ar jau eksistējošu nosaukumu, un nospiež apstiprināšanas pogu “Save”, tiek parādīts logs, kas saka, ka vingrinājums ar šādu nosaukumu jau eksistē.

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

2) Paziņojums par to, ka vingrinājums ar šādu vārdu jau eksistē.

### 2.2.9. 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”.

### 2.2.10. 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”.

### 2.2.11. Vecāka treniņa nokopēšana uz pašreizējo dienu

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

Apstrādei: Kad lietotājs ir nospiež pogu “Copy previous workout”, viņam tiek parādīts kalendārs. Ja tiek izvēlēta diena, kurā iepriekš ir bijis treniņš, parādās logs ar informāciju. Tajā attēloti vingrinājumi, svari, atkārtojumu skaits un piegājienu skaits. Logā ir divas pogas, kas nosaka, vai treniņš tiks nokopēts: “Copy” un “X”. Ja lietotājs nospiež “Copy”, izvēlētās dienas treniņš – ieskaitot svarus, atkārtojumus un piegājienus, tiek nokopēts uz pašreizējo dienu. Ja tiek nospiesta “x” poga, izvēle tiek atcelta. Lietotājam arī ir iespēja nospiest gar logu, lai to aizvērtu.

Izvaddati: 1) Treniņš tiek nokopēts uz pašreizējo dienu;

2) Lietotājam tiek parādīts galvenais skats.

### 2.2.12. 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.

### 2.2.13. Kalendāra apskatīšana

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

Apstrādei: Kad lietotājs nospiež kalendāra pogu, tiek atvērts jauns skats, kurā tiek parādītas dienas un mēneši, kā arī iezīmētās dienas, kurās ir treniņi.

Izvaddati: 1) Tiek atvērts kalendāra skats.

### 2.2.14. Treniņu apskatīšana kalendāra logā

Mērķis: Funkcijas mērķis ir nodrošināt iespēju apskatīt jebkura treniņa datus — vingrinājumus, piegājienus, svaru un atkārtojuma skaitu — kamēr lietotājs atrodas kalendāra skatā.

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) Tiek atvērts neliels logs, kurā tiek attēlots treniņš un tā dati.

### 2.2.15. Pamācības sadaļa

Mērķis: Funkcijas mērķis ir nodrošināt sadaļu, kurā tiek aprakstīta katra vingrinājuma pareiza izpilde, izņemot tos vingrinājumus, kurus lietotājs ir pievienojis pats. Zem vingrinājuma apraksta ir norādītas biežāk pieļautās kļūdas, ko cilvēki mēdz pieļaut tā izpildes laikā.

Apstrādei: Kad lietotājs uzspiež uz pamācības sadaļas pogas, viņam tiek parādīts skats ar muskuļu grupām. Pēc kādas muskuļu grupas izvēles lietotājs var izvēlēties konkrētu vingrinājumu. Tad tiek atvērts skats ar vingrinājuma pamācību un biežāk pieļautajām kļūdām.

Izvaddati: 1) Tiek atvērts pamācības sadaļas skats.

### 2.2.16. Piegājiena izdzēšana

Mērķis: Funkcijas mērķis ir nodrošināt lietotājam iespēju izdzēst konkrētu piegājienu pie noteikta vingrinājuma.

Apstrādei: Kad lietotājs atrodas vingrinājuma skatā un izvēlas piegājienu, ko vēlas dzēst, tiek parādīta poga “Delete”. Pēc pogas nospiešanas piegājiens tiek izdzēsts.

Izvaddati: 1) Vingrinājums tiek izdzēsts / tas pazūd

### 2.2.17. Vingrinājuma izdzēšana

Mērķis: Funkcijas mērķis ir nodrošināt lietotājam iespēju izdzēst vingrinājumu no treniņa.

Apstrādei: Kad lietotājs atrodas galvenajā skatā, pie katra vingrinājuma augšējā labajā stūrī ir atkritumu kastes ikona. Nospiežot to, parādās neliels logs, kurā lietotājs var apstiprina dzēšanu. Pēc apstiprināšanas vingrinājums un tā dati tiek dzēsti no dienas treniņa.

Izvaddati: 1) Dzēšanas apstiprināšanas paziņojums;

2) Vingrinājums tiek izdzēsts no treniņa.

### 2.2.18. Treniņa izdzēšana

Mērķis: Funkcijas mērķis ir nodrošināt lietotājam iespēju izdzēst kādu konkrētu treniņu.

Apstrādei: Kad lietotājs atrodas kalendāra skatā, ir izvēlējies dienu, kurā ir treniņš, un redz datus nelielajā logā, tajā ir redzama poga “Delete”. Pēc tās nospiešanas lietotājam parādās apstiprināšanas logs. Pēc apstiprināšanas treniņš tiek dzēsts.

Izvaddati: 1) Dzēšanas apstiprināšanas paziņojums;

2) Treniņš tiek dzēsts un pazūd no kalendāra.

### 2.2.19. Kopējo treniņu skaita attēlošana

Mērķis: Funkcijas mērķis ir nodrošināt lietotājam iespēju redzēt, cik daudz treniņus viņš ir izveidojis.

Apstrādei: Kad lietotājs atrodas kalendāra skatā, galvenes sadaļā tiek attēlots kopējais lietotāja izveidoto treniņu skaits.

Izvaddati: 1) “x” kopējie vingrinājumi tiek parādīti galvenē.

### 2.2.20. Ik svētdienas notifikācija par nedēļas treniņu skaitu

Mērķis: Funkcijas mērķis informēt lietotāju par to, cik treniņus viņš ir izveidojis nedēļas laikā, nosūtot notifikāciju.

Apstrādei: Katru svētdienu lietotājam tiek nosūtīta notifikācija, kurā attēlots viņa izveidoto treniņu skaits nedēļas laikā.

Izvaddati: 1) Lietotājs saņem notifikāciju, kurā ir redzams nedēļas treniņu skaits.

### 2.2.21. Skats par nedēļas sasniegto

Mērķis: Funkcija ir nepieciešama, lai vienā skatā parādītu lietotājam ko viņš nedēļas laikā ir izdarījis. Skats parāda nedēļas izpildītos treniņus, paceltā svara un piegājienu daudzumu.

Apstrādei: Kad lietotājs atrodas galvenajā skatā, galvenes sadaļā viņš nospiež uz trofejas ikonas, kas parādīs skatu ar nedēļas padarīto statistiskā veidā

Izvaddati: 1) “x” Kopēju treniņu skaits nedēļā;

2) “x” Kopējā paceltā svara skaits;

3) “x” Kopējais piegājienu daudzums.

## 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.

## Gala lietotāja raksturiezīmes

Lietotāja vecums būtu no 12 līdz 60 gadiem, kas nodarbojas ar sportu, fiziskām aktivitātēm un ir motivēts šīs aktivitātes turpināt, kā arī 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.

**Microsoft Word**: Ir teksta apstrādes programma, ko es izmantoju priekš dokumentācijas veidošanas. Word ļauj lietotājiem veidot, rediģēt un formatēt tekstus, kā arī tabulas, grafikus un citus elementus. Es izvēlējos Word, jo es to lietoju jau vairākus gadus kā savu primāro rīku ar kuru veidoju dokumentācijas, aprakstus un citus tekstus. Manuprāt, starp pieejamajām teksta apstrādes programmām, Word ir visērtākais un visvienkāršākais.

**Excel**: Ir izklājlapu programma, ko es izmantoju priekš projekta “Black box” un “White box” testu pierakstiem. Excel man ērti atļauj izveidot tabulas, kā arī savadīt iekš tām datus. Līdzīgi kā ar Word, Excel esmu jau labu laiku izmantojis šāda veida uzdevumiem, tādējādi jau ar viņu ir pieredze. Alternatīvas programmas nav nepieciešamas, jo Excel visu man vajadzīgo nodrošina.

## 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.

**LibreOffice Writer**: Šī ir alternatīva teksta apstrādes programma, ko varētu izmantot Microsoft Word vietā. Tā piedāvā plašu funkciju klāstu dokumentu veidošanai, rediģēšanai un formatēšanai. Programma ir ļoti līdzīga Word programmai. Pozitīvi ir tas, ka LibreOffice ir bezmaksas rīks kas piedāvā praktiski to pašu ko Microsoft Word.

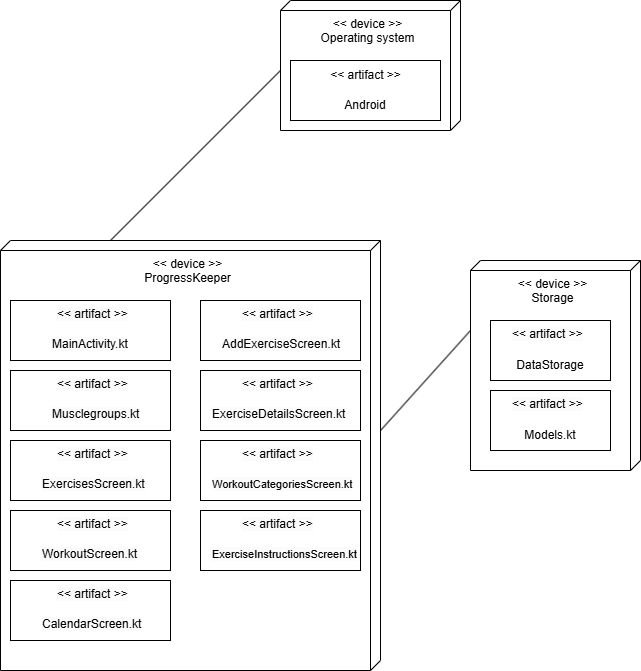
**Google Sheets**: Šī ir alternatīva izklājlapu programma. Tā ļauj izveidot un rediģēt failus ar kuriem pat var dalīties ar citiem lietotājiem. Google Sheets piedāvā nedaudz mazāk funkciju nekā Excel, taču tabulu veidošanai tā ir ļoti piemērota. Līdzīgi kā LibreOffice, Google Sheets ir bezmaksas, turklāt faili un to izmaiņas tiek automātiski saglabātas mākoņkrātuvē.

# Sistēmas modelēšana un projektēšana

## 4.1. Sistēmas struktūras modelis

### 4.1.1. Izvietojuma (Deployment) diagramma

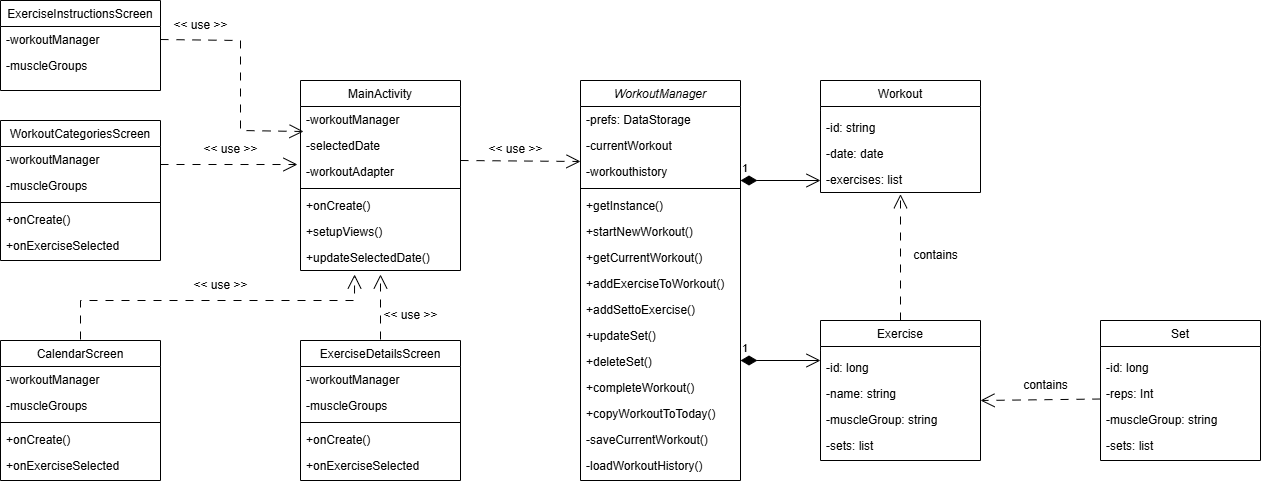
Šeit tiek attēlota izvietojuma diagramma kas parāda sistēmas komponenšu un artifaktu sadarbību. Iekš ProgressKeeper komponentes ir artifakti, kas šijā piemērā ir visi pieejamie lietotāja skati. Ārpus šīs komponentes ir Storage komponente, kurā ir DaraStorage un Models artifakti, kas ir metodes, kas visu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



**1. attēls. Izvietojuma diagramma**

### 4.1.2. Klašu (Class) diagramma

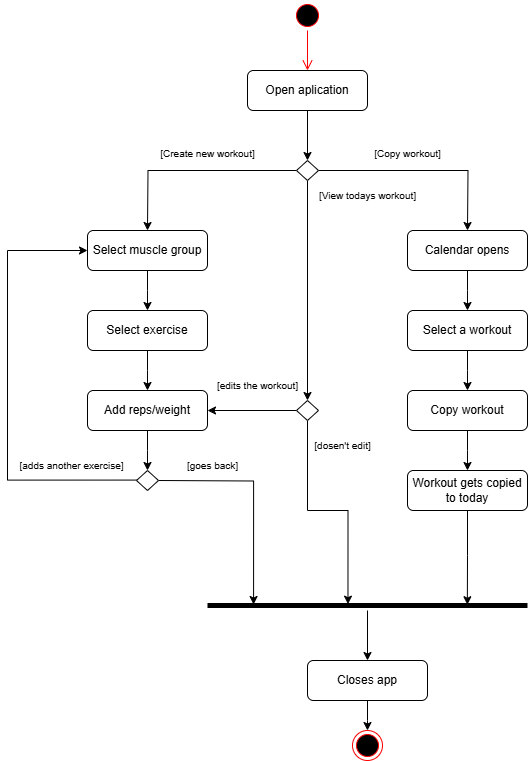
Šeit tiek attēlota sistēmas klašu diagramma, kas parāda visas programmas zemsistēmu. Iekš workout manager ir redzami visi elementi, ko tas manto, kā arī pats izveido sev. Šī būtu galvenā darbības sistēma. Tālāk mums ir Workout, Exercise, Set kas ir mainīgie vingrinājumiem/treniņiem, un tie satur vienu 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 muskuļu grupas. CalendarScreen, kas attēlojas MainActivity screen, kur tas attēlo kalendāru un lietotāja iepriekšējos treniņus. Ir ExerciseInstructions ekrāns, kurā ir pamācības specifiskajiem vingrinājumiem.



**2. attēls. Klašu diagramma**

## 4.2. Funkcionālais un dinamiskais sistēmas modelis

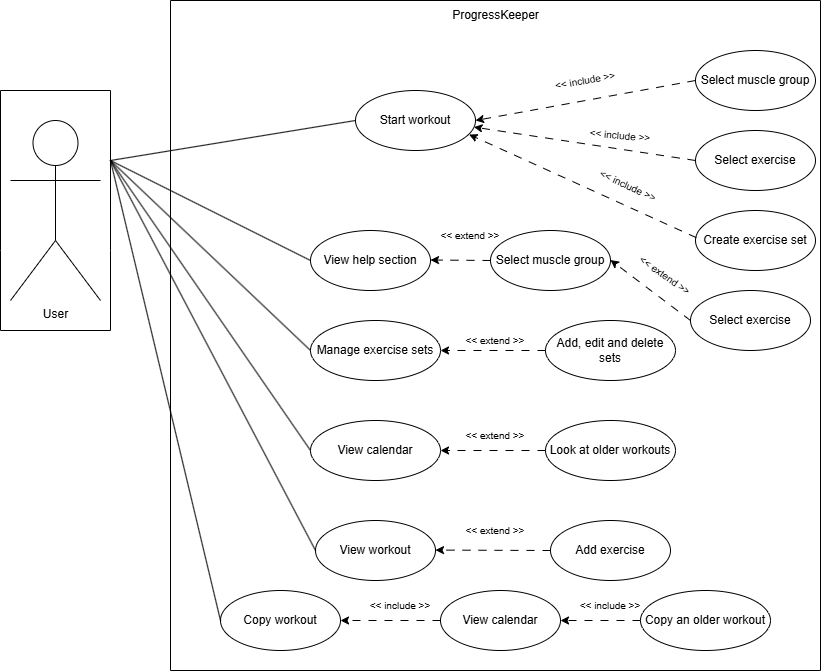
### 4.2.1. Aktivitāšu (Activity) diagramma

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.

**3. attēls. Aktivitāšu diagramma**

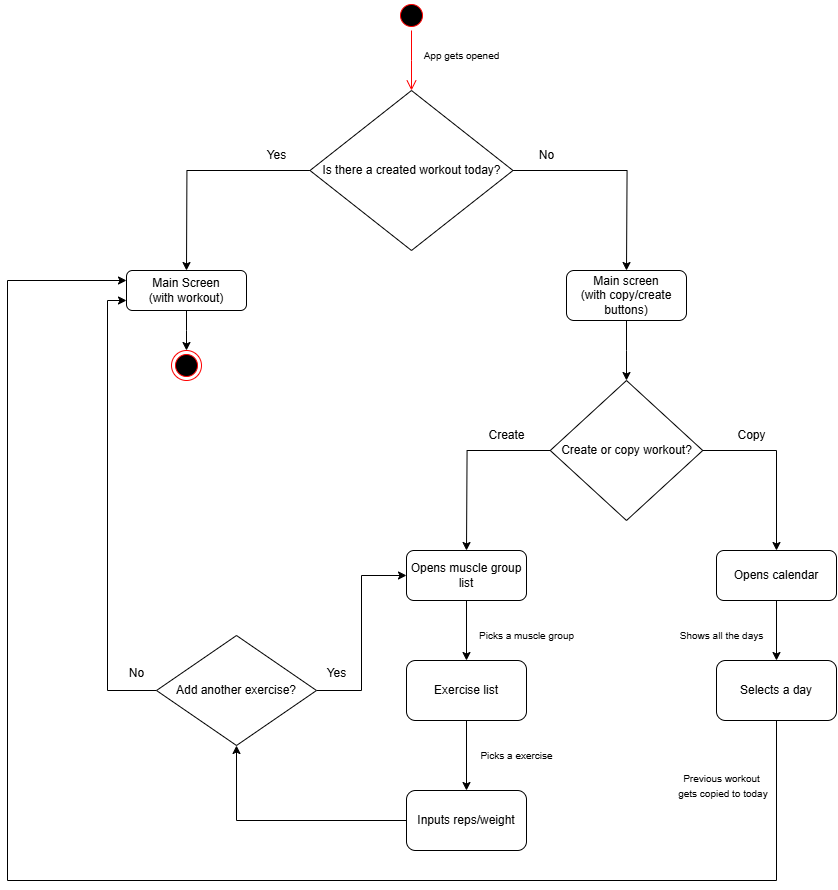
### 4.2.2. Lietojumgadījumu (Use Case) diagramma

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. attēls. Lietojumgadījumu diagramma

### 4.2.3. Stāvokļu (State) diagramma

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.

### 5. attēls. Stāvokļu diagramma

## 4.3. 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.

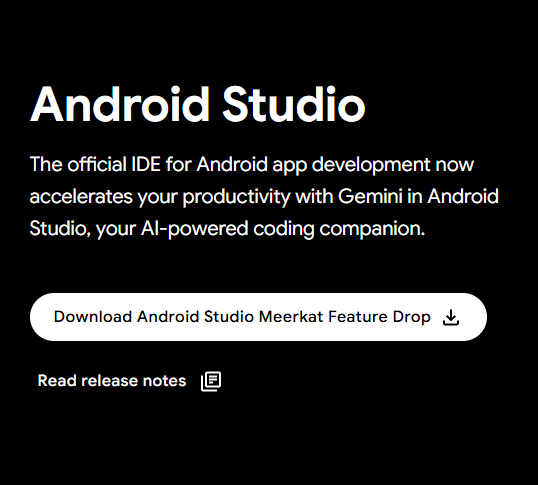
Lietojumgadījumu diagramma tika izvēlēta, jo tā attēlo lietotāja pieejamos scenārijus, piemēram apskata kalendāru, un nokopē vecāku treniņu uz šodienu. Iekš šīs diagrammas ir attēlotas visas svarīgākās darbības.

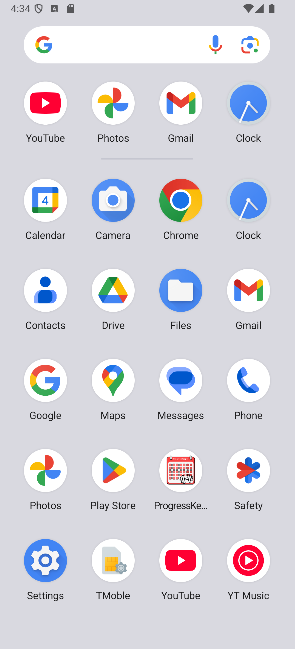
Stāvokļu diagramma tika izmantota, lai attēlotu lietotnes stāvokļus, piemēram kā galvenā ekrāna stāvokli, kas ir atkarīgs no tā, vai šodienā jau ir izveidots/nokopēts treniņš, vai vēl nē. Šī diagramma ļauj izprast, kā lietotnes stāvokļi mainās, atkarībā no lietotāja izvēlēm.

# Lietotāju ceļvedis

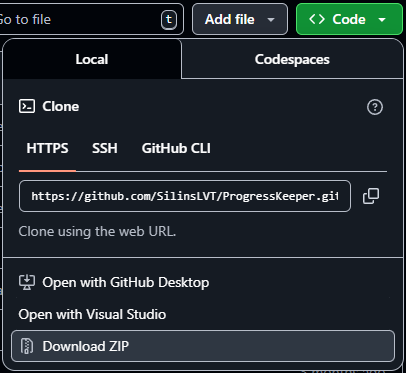
Šajā nodaļā tiks parādīts lietotāja ceļš un galvenās darbības, ko iespējams veikt aplikācijā “ProgressKepper”. Aplikācija ir izstrādāta, lai palīdzētu lietotājiem plānot un sekot līdzi saviem treniņiem, nodrošinot ērtu veidu, kā izveidot treniņus, izvēlēties vingrinājumus, pierakstīt atkārtojuma reizes un lietoto svaru, rediģēt šo informāciju, kā arī dzēst to vai pašus treniņus. Zemāk redzamie ekrānuzņēmumi palīdzēs saprast, kā aplikāciju lejupielādēt un kā to izmantot.

Sakarā ar to, ka aplikācija nav pieejama ārpus GitHub saites, pirmais solis ir lejupielādēt Android Studio aplikāciju. Saite kur to lejupielādēt (<https://developer.android.com/studio>)



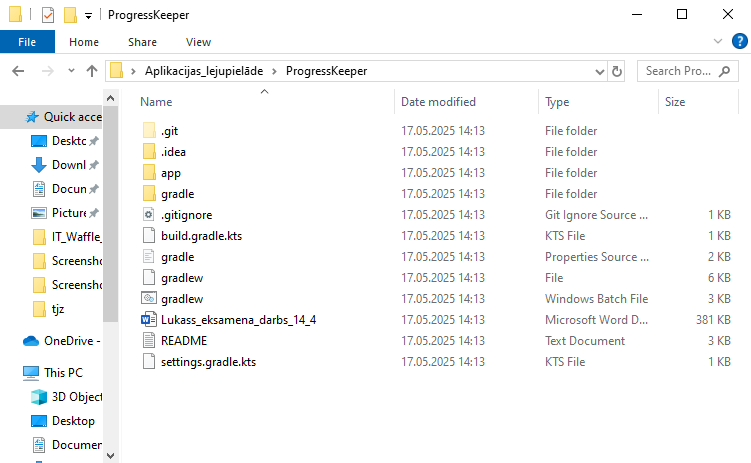
**6. attēls. Android Studio lejupielāde**

Tālāk ir jāapmeklē mana GitHub repozitorija, no kuras lejupielādēt “ProgessKeeper” projektu kā arhivētu mapi. Saite kur to atrast un lejupielādēt (<https://github.com/SilinsLVT/ProgressKeeper>)



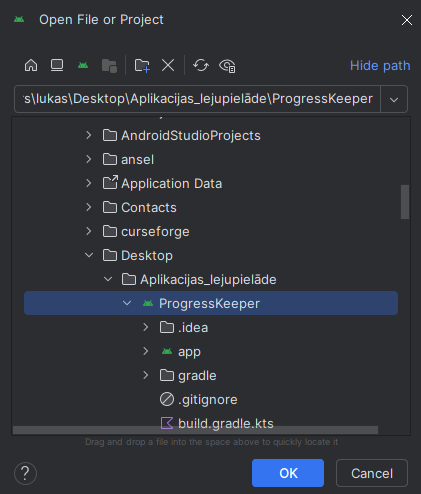
**7. attēls. GitHub repozitorijas mapes lejupielāde**

Pēc mapes lejupielādēšanas, to ir jāatarhivē, un jāsaglabā vietā, kur to varēs atrast, kā iekš lejupielādes mapes vai uz datora darbvirsmas.

****

**8. attēls. Lejupielādētā mapes atarhivēšana un saglabāšana**

Lietotājam ir jāatver Android Studio, kur viņš augšējā labajā stūrī uzspiež un pogas “Main Menu”. Pēc tam ir jānospiež uz pogas “File” tad “Open” un jāatrod kur ir lejupielādētā mape no GitHub virsmas. Lietotājam ir jāizvēlas atvērt aplikāciju “ProgressKeeper”.



**9. attēls. “ProgressKeeper” projekta atvēršana**

 Lietotājam ir jānospiež ar sarkano apli iezīmētā poga, lai sinhronizētu projektu un visas tā bibliotēkas.

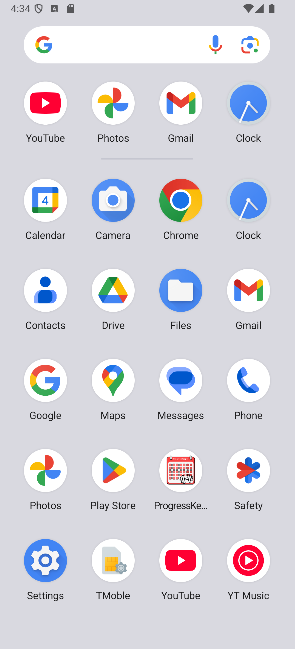
**10. attēls. “ProgressKeeper” projekta atvēršana**

Lietotājam ir jānospiež ar sarkano apli iezīmētā poga, lai startētu aplikāciju savā virtuālajā vidē.

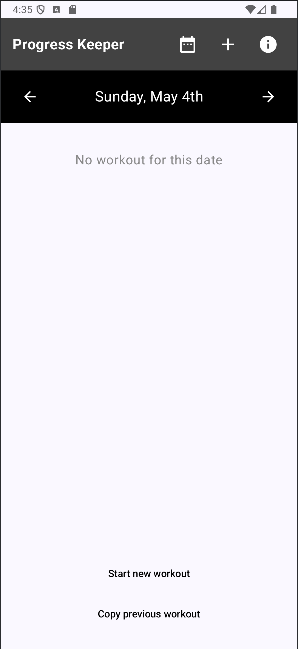


**11. attēls. “ProgressKeeper” projekta startēšana**

Android vidē ir redzama aplikācija ar kalendāra ikonu vārdā “ProgressKeeper”. Lietotājs uzspiež uz tās.

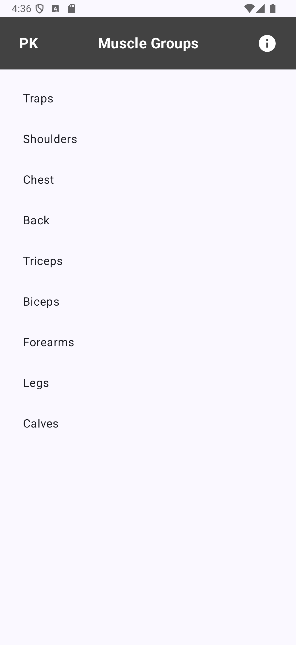
**12. attēls. “ProgressKeeper” aplikācijas atvēršana**

Lietotājs ir pirmo reizi atvēris aplikāciju. Viņš vēl nav izveidojis nevienu treniņu. Viņam ir vairākas opcijas, ko viņš var izvēlēties, kā sākt jaunu treniņu, apskatīt kalendāru un aplūkot pamācības sadaļu.

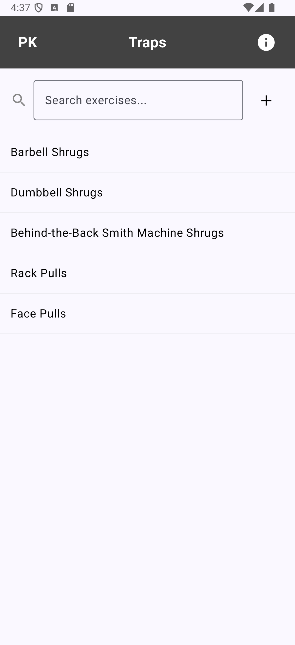


**13. attēls. Tukšs galvenais skats**

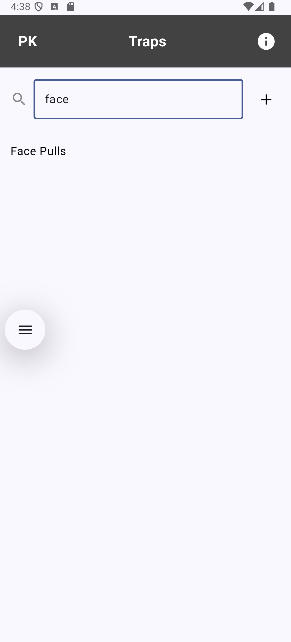
Lietotājs izvēlas uzsākt jaunu treniņu. Viņam parādās cits skats, kur var izvēlēties starp muskuļu grupām.

**14. attēls. Muskuļu grupu izvēle**

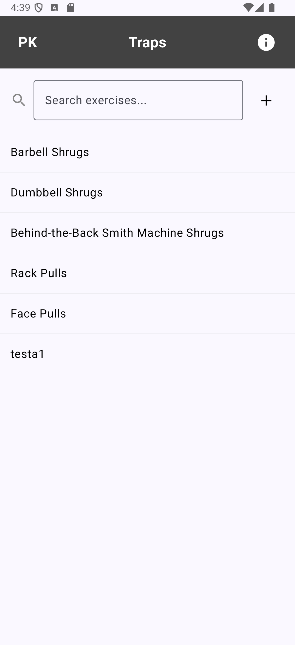
Pēc izvēlētās muskuļu grupas izvēles lietotājam atveras cits skats, kurā ir redzami visi šīs muskuļa grupas vingrinājumi.

**15. attēls. Vingrinājumu izvēle**

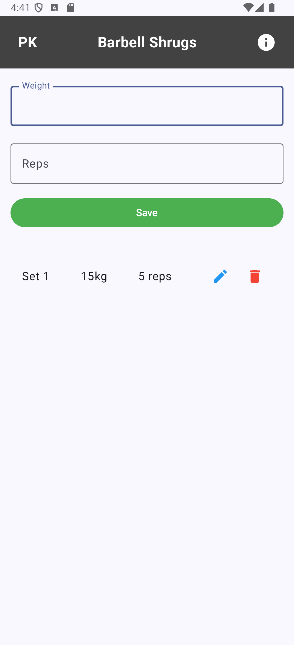
Lietotājam ir iespēja meklēt kādu specifisku vingrinājumu, izmantojot meklēšanas joslu.

**16. attēls. Vingrinājuma meklēšana**

Lietotājs var pievienot pats savu vingrinājumu izvēlētajai muskuļu grupai esošo vingrinājumu sarakstā. Pievienotais vingrinājums tiek ievietots starp pārējiem vingrinājumiem.

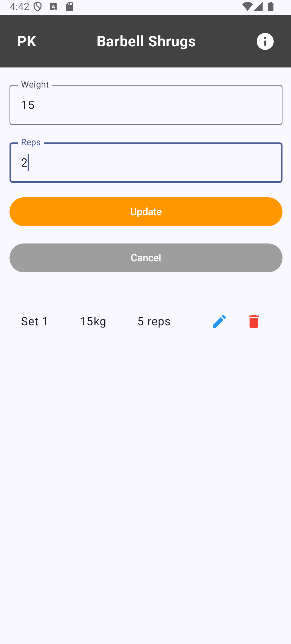
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
**17. un 18. attēls. Vingrinājuma pievienošana**

Lietotājam, pēc vingrinājuma izvēles, tiek parādīts jauns logs, kurā viņš var pievienot vingrinājuma datus (piegājienus, svaru, atkārtojuma reizes). Viņš izveido 1. piegājienu.



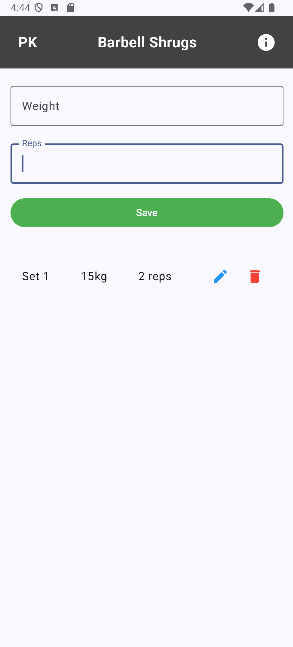
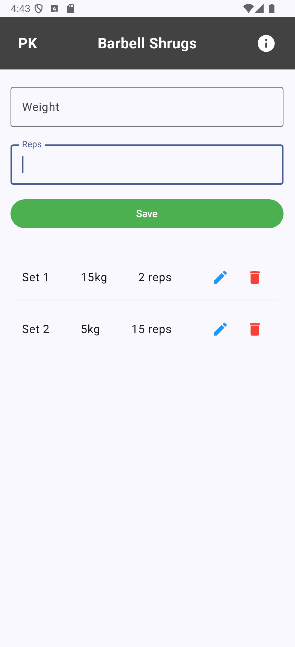
**19. attēls. Pirmā piegājiena izveide**

Lietotājam ir iespēja rediģēt savus piegājienus.



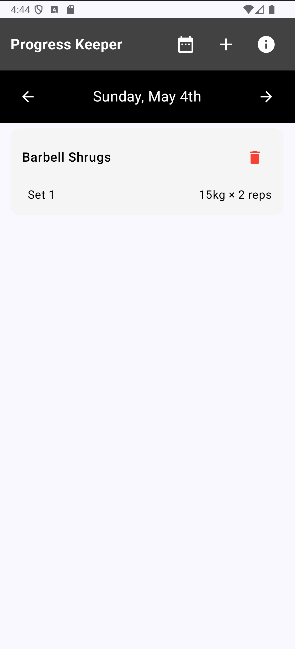
**20. attēls. Piegājiena rediģēšana**

Lietotājs arī var dzēst kādu no izveidotajiem piegājieniem.

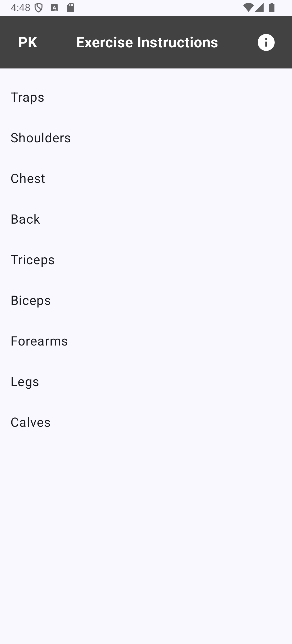


**21. un 22. attēls. Piegājiena dzēšana**

Lietotājam atgriežoties galvenajā skatā ir redzams, ka uz doto dienu ir izveidots treniņš, kā arī tā dati (piegājieni, svars, atkārtojuma reizes) ir redzami.

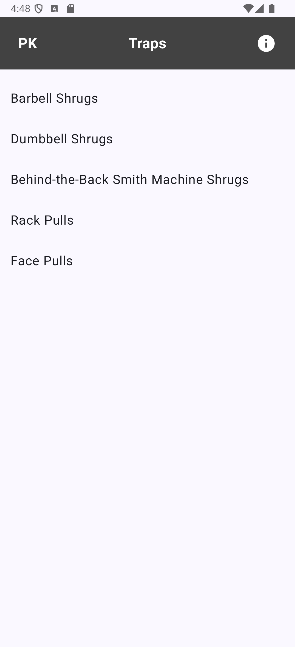
**23. attēls. Treniņš galvenajā skatā**

Lietotājs var atvērt pamācības sadaļu (“i” ikonu) galvenes sadaļā, kur būs muskuļu grupu izvēle.



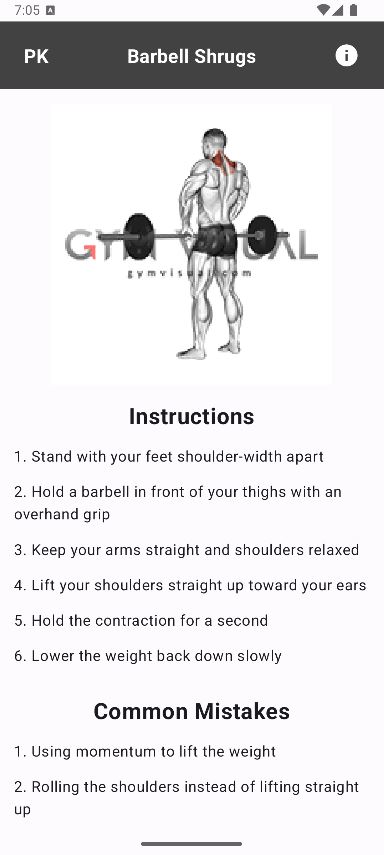
**24. attēls. Pamācības sadaļa**

Pēc muskuļu grupas izvēles, lietotājam tiek atvērts skats, kurā redzami visi muskuļu grupas vingrinājumi, kas bija aplikācijas lejupielādes laikā.



**25. attēls. Pamācības sadaļas vingrinājumi**

Pēc vingrinājuma izvēles, lietotājam tiek atvērts skats, kurā ir redzama instrukcija par to, kā darīt specifisko vingrinājumu. Ir norādīta bilde, kur parāda kā vingrinājums tiek izpildīts, kā arī instrukcijas soļi. Pēc instrukcijas ir aprakstītas dažas biežāk pieļautās kļūdas.

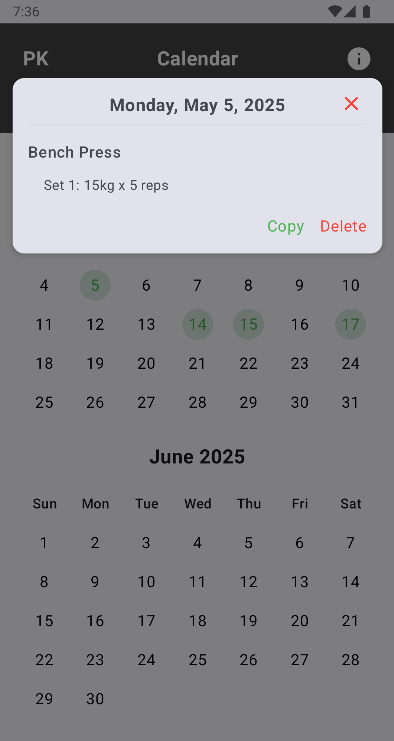


**26. attēls. Vingrinājuma instrukcija**

Kad lietotājs ir galvenajā skatā, viņš var nospiest kalendāra pogu. Tā atvērs kalendāra skatu kurā var redzēt visus treniņus, un kopēju treniņu skaitu.

**27. attēls. Kalendāra skats ar treniņiem**

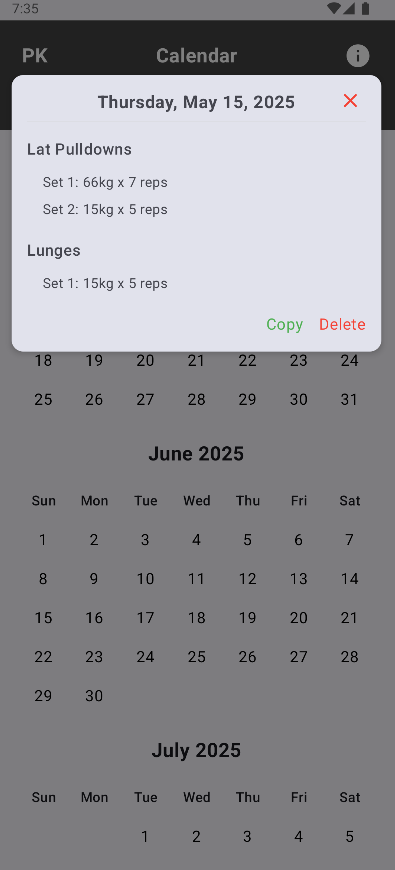
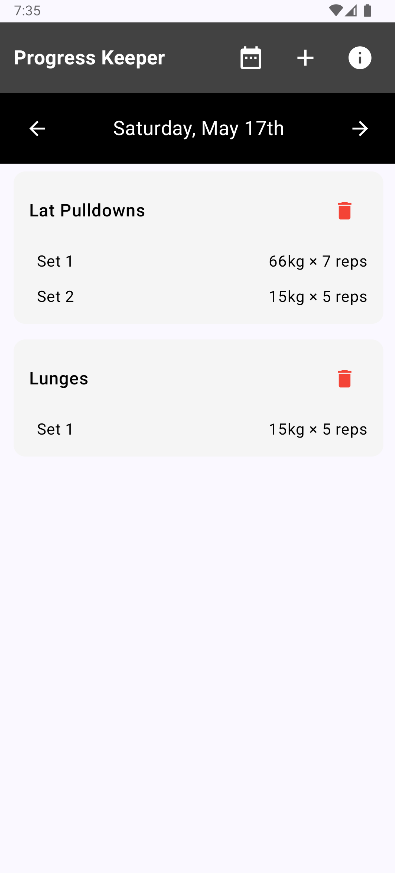
Lietotājs var apskatīt senāka treniņa informāciju.

**28. attēls. Aplūko treniņa datus kalendāra skatā**

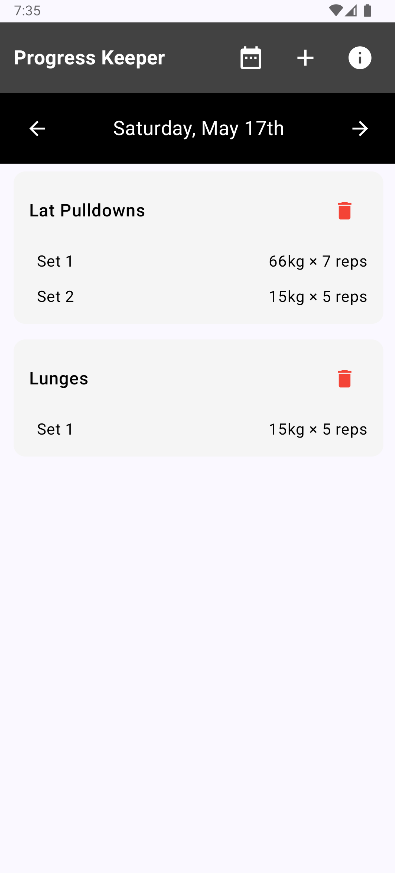
Lietotājs galvenajā skatā var arī nokopēt vecāku treniņu, ja noteiktajā dienā vēl nav izveidots treniņš.

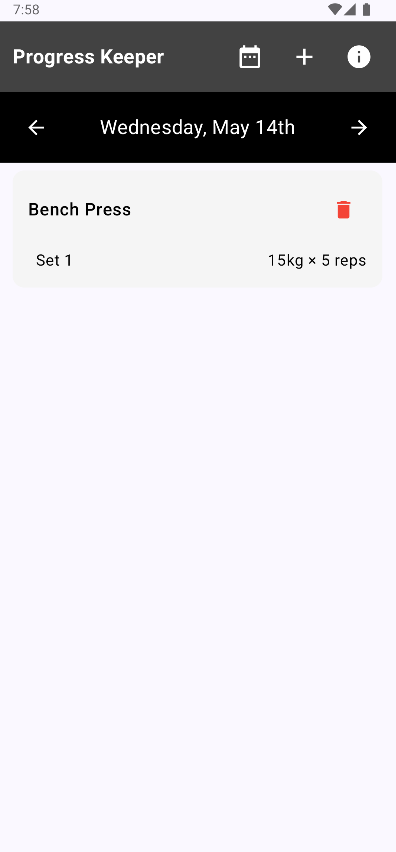
**29. attēls. Lietotājs nokopēs treniņu**

Pēc “Copy previous workout” pogas nospiešanas, lietotājam parādās kalendāra skats, kurā viņš var nokopēt vecāku treniņu. Pēc treniņa nokopēšanas, lietotājs tiek pārnests uz galveno skatu, kur ir redzams nokopētais treniņš.

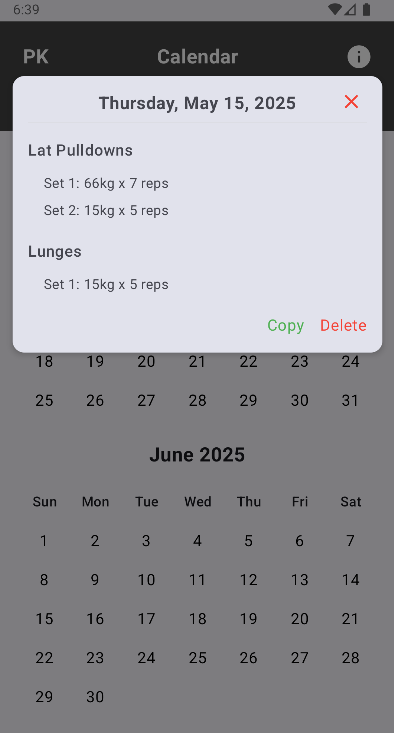


**31. un 32. attēls. Lietotājs kopē vecāku treniņu**

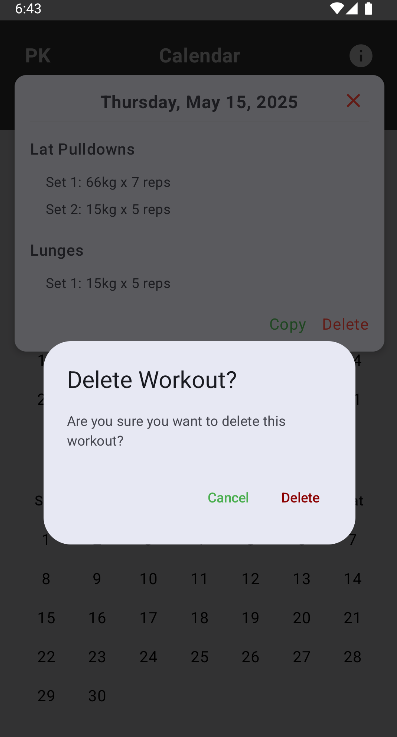
Lietotājs, atrodoties galvenajā skatā, arī var aplūkot vecākus treniņu, nospiežot “🡨” pogu.

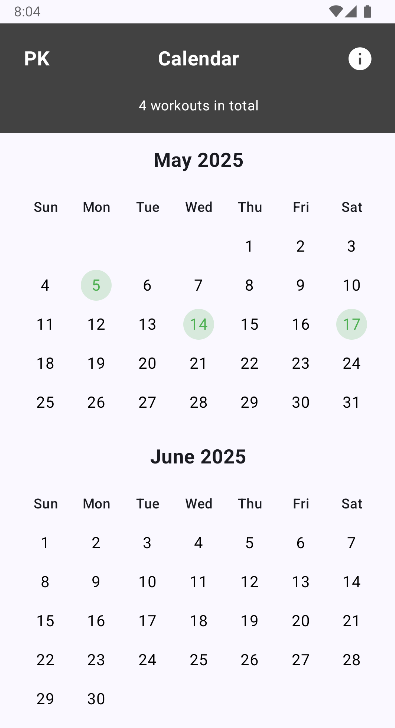
****  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
**33. un 34. attēls. Vecāka treniņa aplūkošana no galvenā skata**

Lietotājs, atrodoties kalendāra skatā, var arī dzēst jebkuru treniņu. Uz iezīmētās dienas ir jānospiež, un kad atveras mazais logs ar treniņa informāciju, ir jānospiež “Delete” poga.



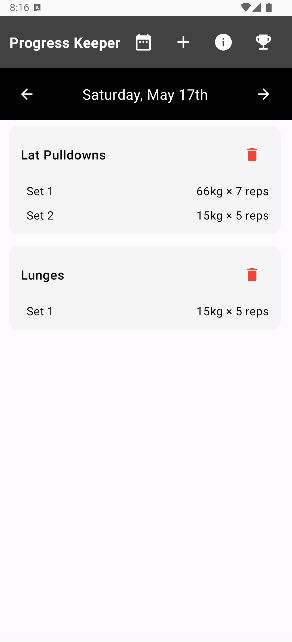
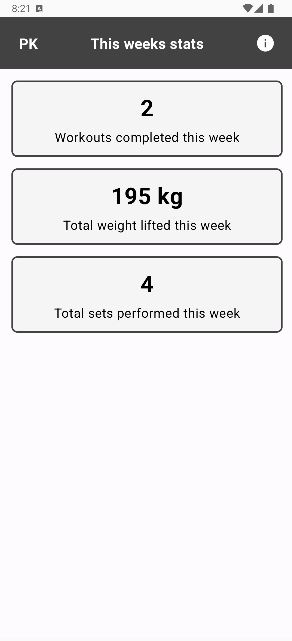
**35. attēls. Treniņa dzēšana**

Kad lietotājs mēģina dzēst vecāku treniņu, viņam tas ir jāapstiprina. Pēc dzēšanas, treniņš pazūd no kalendāra, un kopējo treniņu skaits mainās.



**36. un 37. attēls. Pēc treniņa dzēšanas**

Lietotājam ir iespēja aplūkot nedēļas apkopotos rezultātus. To var palūkot no galvenās sadaļas, uzspiežot uz trofejas ikonas. Šajā sadaļā lietotājam ir pieejami dati par to, cik treniņi ir bijuši nedēļas laikā, kopējais pacelto kg skaits, kā arī cik piegājieni ir paveikti.



**38. un 39. attēls. Nedēļas apkopotais rezultāts**

# Testēšanas dokumentācija

Šī dokumentācijas nodaļa ir izstrādāta sistēmas testēšanai. Šeit varēs iepazīties ar manu izmantoto testēšanas veidu un metodi, kā arī ar alternatīviem risinājumiem. Nodaļā būs aprakstīta testēšanas gaita un parādīti sasniegtie rezultāti.

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

Šī projekta testēšanai es izvēlējos veikt manuālo testēšanu, jo automatizētu testu veidošana šajā gadījumā šķita nevajadzīga. Skriptu rakstīšana ir laikietilpīga, un visas galvenās programmas funkcijas ir vizuāli pārbaudāmas, jo lietotājs var redzēt izmaiņas ekrānā, ko izraisa viņa darbības.

Šo iemeslu dēļ esmu izvēlējies sistēmas testēšanai izmantot manuālo melnās kastes testēšanu. Šim testēšanas veidam, manuprāt, ir divas priekšrocības: pirmkārt, man jau ir pieredze ar melnās kastes testēšanu no iepriekšējiem projektiem, kā arī šī metode neprasa piekļuvi programmas kodam, lai varētu pilnvērtīgi notestēt visu sistēmas darbību.

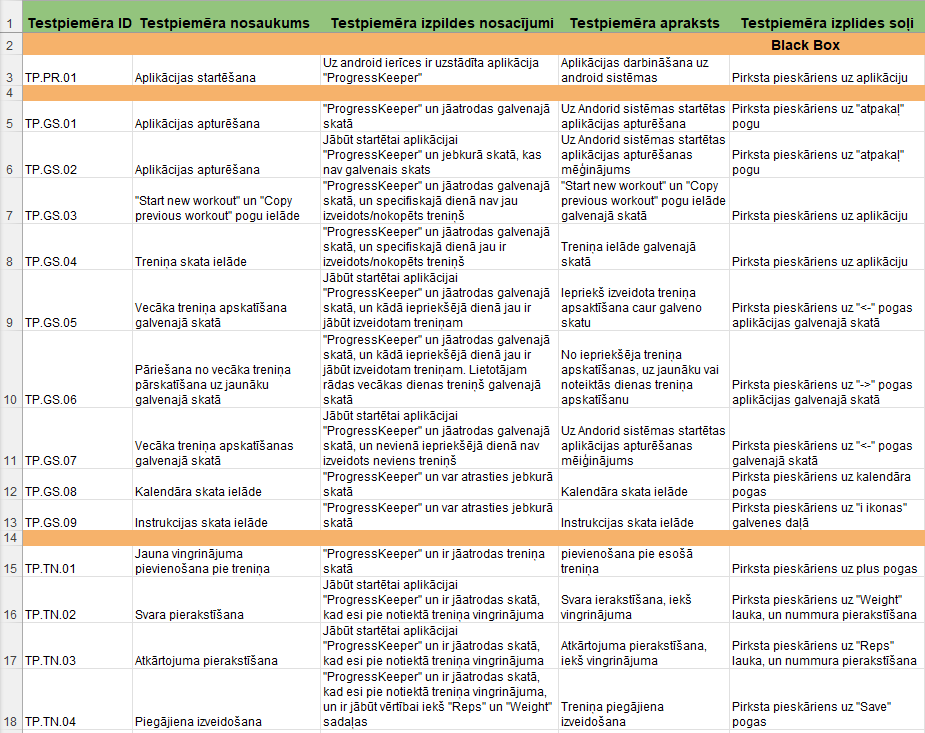
Tāpat vēlos pieminēt, ka visa programmas izstrādes procesa laikā fonā darbojās Lint, kas ir iebūvēta sistēma, kas automātiski veica statisku koda pārbaudi, palīdzot man savlaicīgi pamanīt iespējamas kļūdas un uzlabot koda kvalitāti.

## 6.2. Alternatīvas testēšanas metodes un rīki

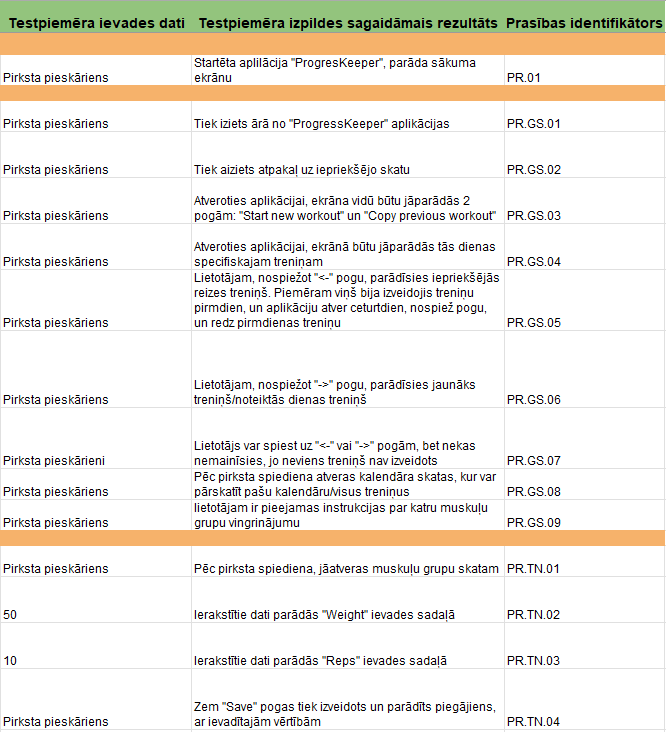
Labs alternatīvs manis izmantotajai metodei – manuālajai melnās kastes (Black Box) testēšanai – ir automatizētā testēšana, kuras laikā testētājs programmē testus programmai. Šāda pieeja testēšanai dažiem liekas ērtāka, jo nav nepieciešams manuāli pārskatīt visu programmu. Testētājam ir tikai jāuzraksta testi un aplūkot to rezultātus.

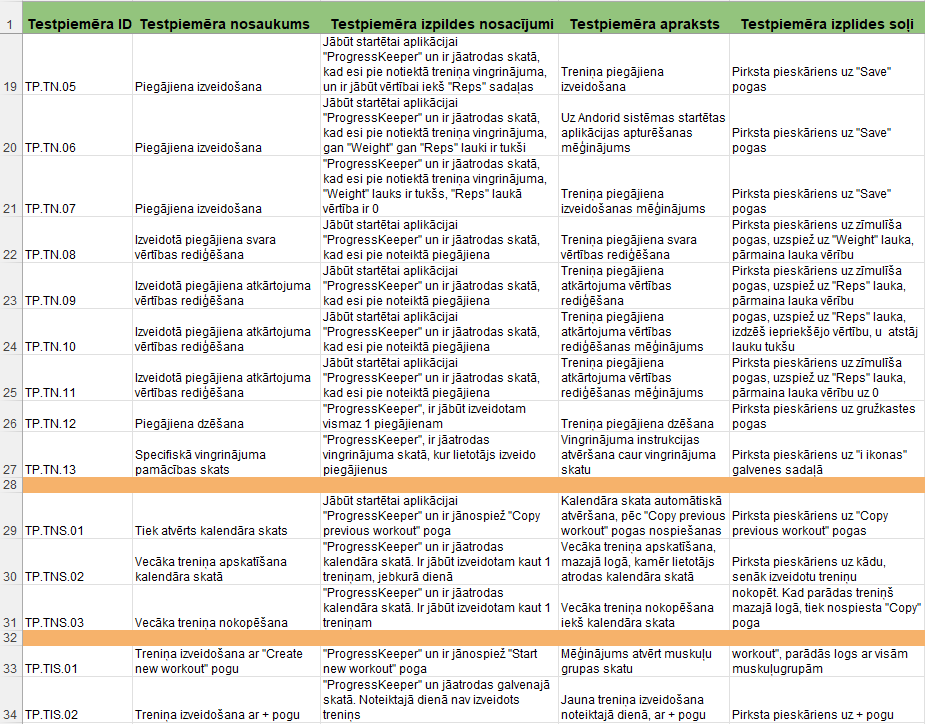
Lai šai aplikācijai izmantotu automatizēto testēšanu, var izmantot rīkus, piemēram JUnit, kas ir paredzēts vienību testēšanai. Cits rīks ir Espresso, ar ko testē UI elementus, tāpat var izmantot Mockito, ar ko testē objektus un programmas izolāciju.

## Testēšanas kopa

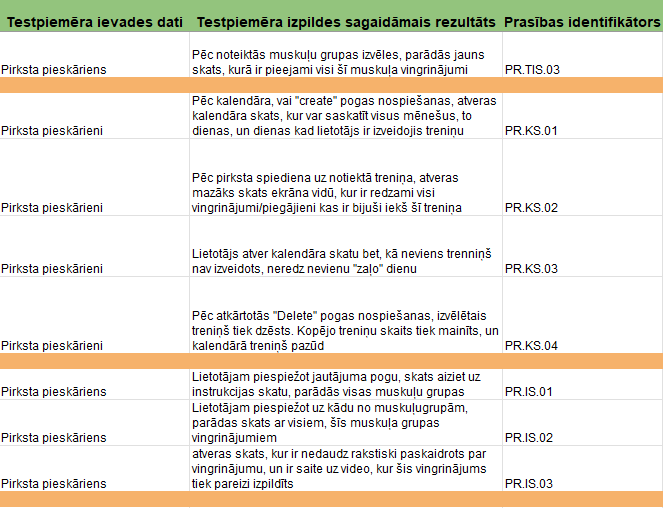
Zemāk ir redzami visi mani izstrādātie testpiemēri priekš “ProgressKeeper” aplikācijas.

**40. attēls. Testpiemēri (1-18) 1. daļa**

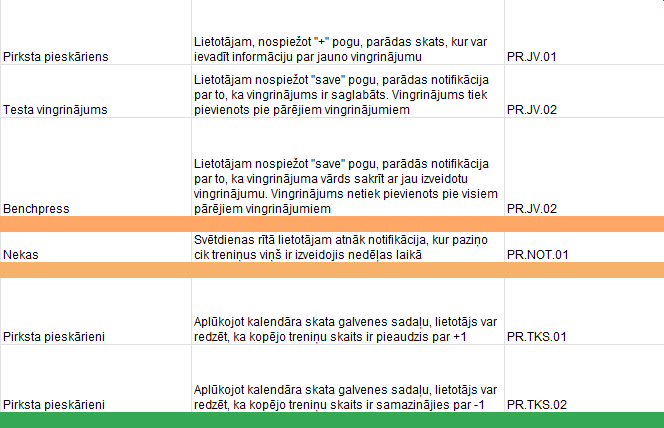
**41. attēls. Testpiemēri (1-18) 2. daļa**

**42. attēls. Testpiemēri (19-34) 1.daļa**

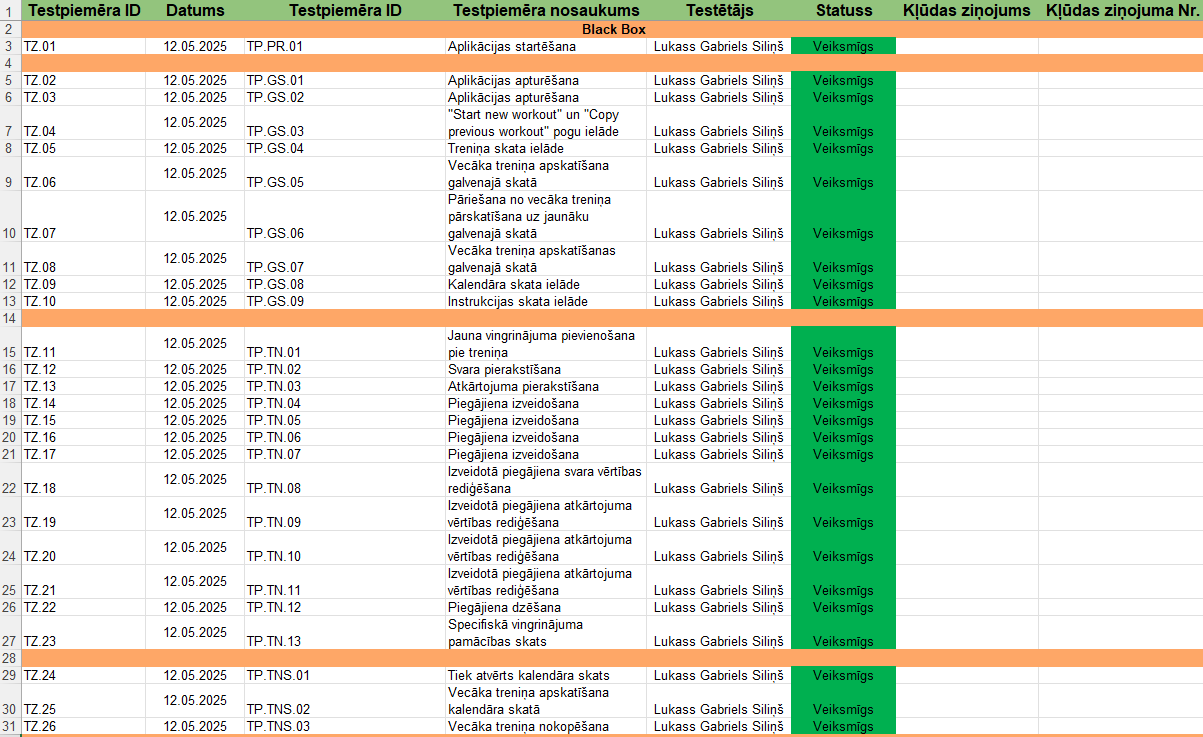
**  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
43. attēls. Testpiemēri (19-34) 2.daļa**

****

**44. un 45. attēls. Testpiemēri (35-45)**

**46. un 47. attēls. Testpiemēri (46-54)**

## 6.4. Testēšanas žurnāls

  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
**48. attēls. Testēšanas žurnāls (1-31)**

**  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
49. attēls. Testēšanas žurnāls (32-53)**

# 7. Secinājumi

Šo projektu sāku aptuveni februāra sākumā, tātad kopumā izstrādei veltīju apmēram 3,5 mēnešus. No visiem iepriekšējiem projektiem, šis ir lielākais, ko esmu izstrādājis, gan salīdzinot ar tehnikuma uzdevumiem un eksāmeniem, kā arī uzdevumiem prakses laikā. Projekta sākums gāja samērā lēni, jo uzsākot projektu man nebija visas prasmes, kā arī projekta laikā apguvu jaunas zināšanas. Saņēmu ieteikumus no skolotājiem, kā arī no prakses kolēģiem, kā uzlabot projektu, vai ko pamainīt, izdarīt labāk.

Projekta izstrādes laikā esmu paplašinājis savas zināšanas par Kotlin programmēšanas valodu, kā arī tās metodēm, kā SharedPreferences, kas saglabā treniņu informāciju, lokālās notifikācijas par treniņu daudzumu. Dokumentācijas izstrādes laikā es atkārtoju jau iepriekš zinātās prasmes, ko apguvu tehnikumā, kā diagrammu veidošanu, Excel izmantošanu un strukturētu dokumentācijas veidošanu.

Projekta izstrāde kopumā man patika, protams, bija brīži kad kaut kas negāja vai ievilkās uz ilgu laiku, vai vienkārši negribējās neko veidot. Aplikācija man pašam liekas lietderīga, kā arī vizuāli neuzkrītoša.

Nākotnē, ja turpināšu izstrādāt projektu, gribu ieviest funkcionalitātes, kā zvaigžņu sistēma, kas pievienotu zvaigzni pie kāda piegājiena, kur lietotājs ir sasniedzis lielāku svaru, vai arī vairāk atkārtojumu, kā iepriekš. Vēl gribētu ieviest datu importu un eksportu, jo ir iespēja, ka lietotājs mainīs ierīci un gribēs saglabāt treniņu datus, bet lai šo ieviestu, man būtu jāpārveido datu saglabāšanas funkcionalitāte, jo pašreizējais API ko izmantoju (SharedPreferences) neļauj tiešu datu importu un eksportu. Arī gribētu ieviest kādu statistiku, kā grafiku, kur parādās, kā lietotājs progresē ar specifisku vingrinājumu svara un atkārtojuma reižu skaitā.

# 8. Lietoto saīsinājumu un terminu skaidrojums

Vispārīgi:

* API (no angļu valodas *Application programming interface*) – Lietojumprogrammas saskarne ar iepriekš definētu klašu, procedūru, funkciju, struktūru un konstanšu kopumu, kas tiek piedāvāts kā interfeisa tips, kuru iespējams izmantot ārējiem programmatūras projektiem.
* Artifakts – Rezultējošs fails vai komponents (piemēram, fails, programmas u.c.), kas tiek attēloti programmatūras diagrammās.
* Klase – Programmēšanas struktūra, kas definē objektus ar laukiem un metodēm.
* Komponentes – Atsevišķas lietotnes daļas, kā servisi, kas tiek attēloti diagrammās.
* SharedPreferences – Android API, kas nodrošina datu saglabāšanu lokāli.

Testēšana:

* GS – Galvenais aplikācijas skats.
* IS – Instrukcijas skats.
* JV – Jauns vingrinājums.
* KS – Kalendāra skats.
* NOT – Notifikācija.
* PR – Sistēmas funkcionālā prasība.
* TIS – Treniņa izveidošanas skats.
* TKS – Treniņu kopējais skaits.
* TN – Treniņa skats.
* TNS – Treniņa nokopēšanas skats.
* TP – Testpiemērs.
* TZ – Testēšanas žurnāls.

# 9. Literatūras un informācijas avotu saraksts

<https://androidcodelabs.dev/> (27.03.2025)

<https://app.diagrams.net/> (12.04.2025)

<https://chat.deepseek.com/> (09.05.2025)

<https://chatgpt.com/> (18.05.2025)

<https://developer.android.com/courses> (01.04.2025)

<https://geeksforgeeks.org/> (02.05.2025)

<https://lucidchart.com/pages?usecase=uml> (10.04.2025)

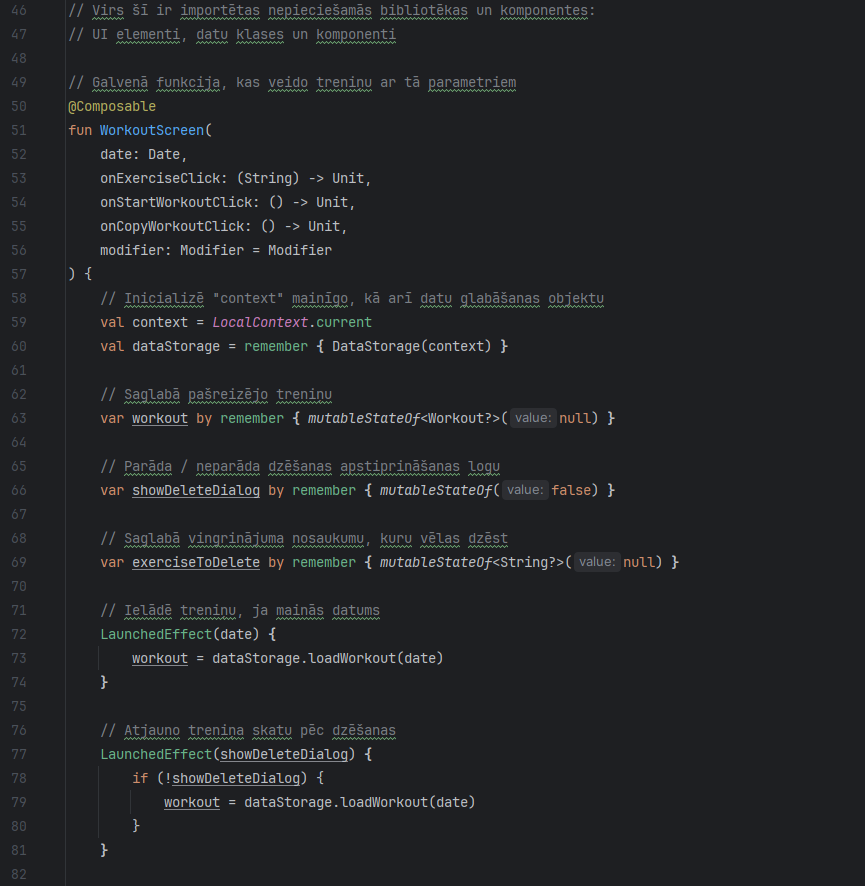
<https://stackoverflow.com/questions> (07.05.2025)

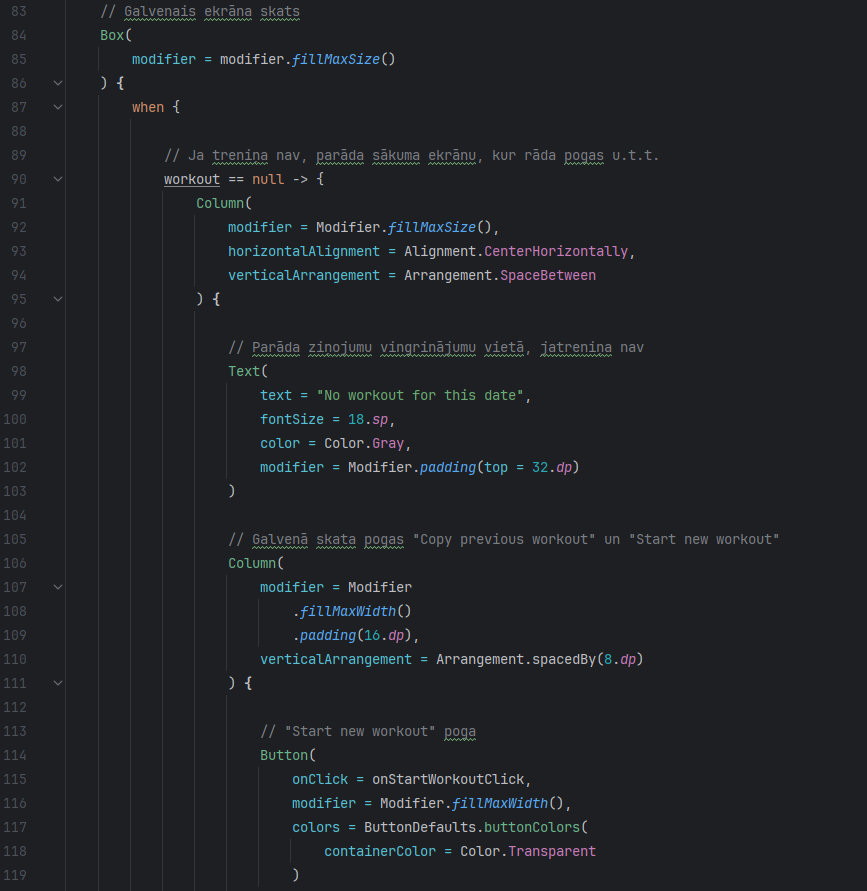
<https://visual-paradigm.com/> (10.04.2025)

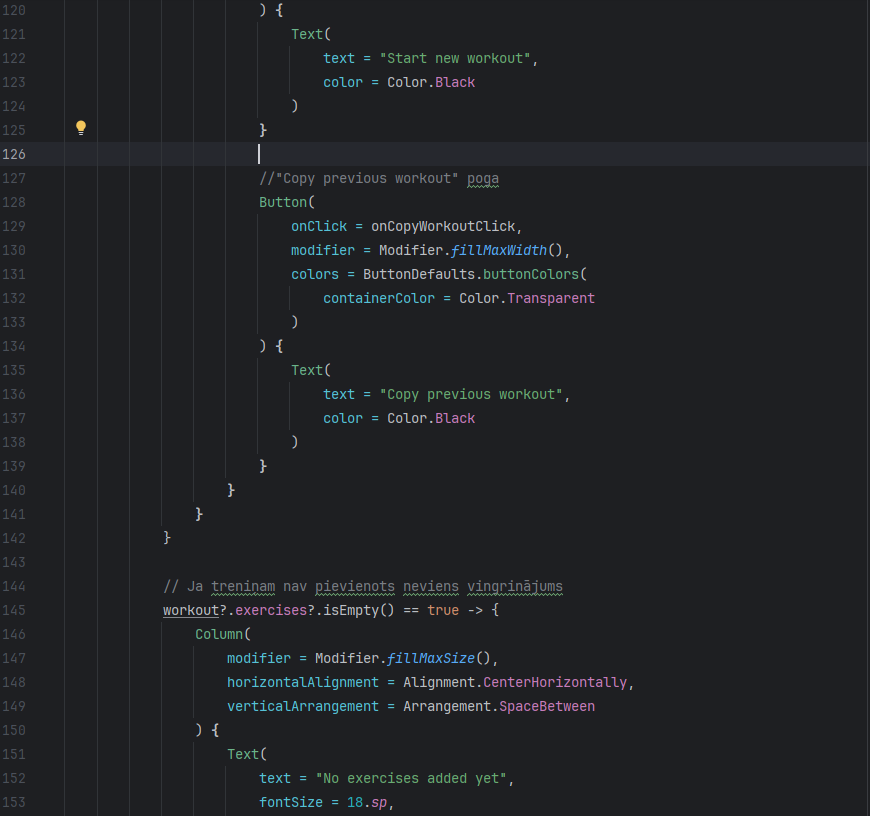
<https://youtube.com/> (10.05.2025)

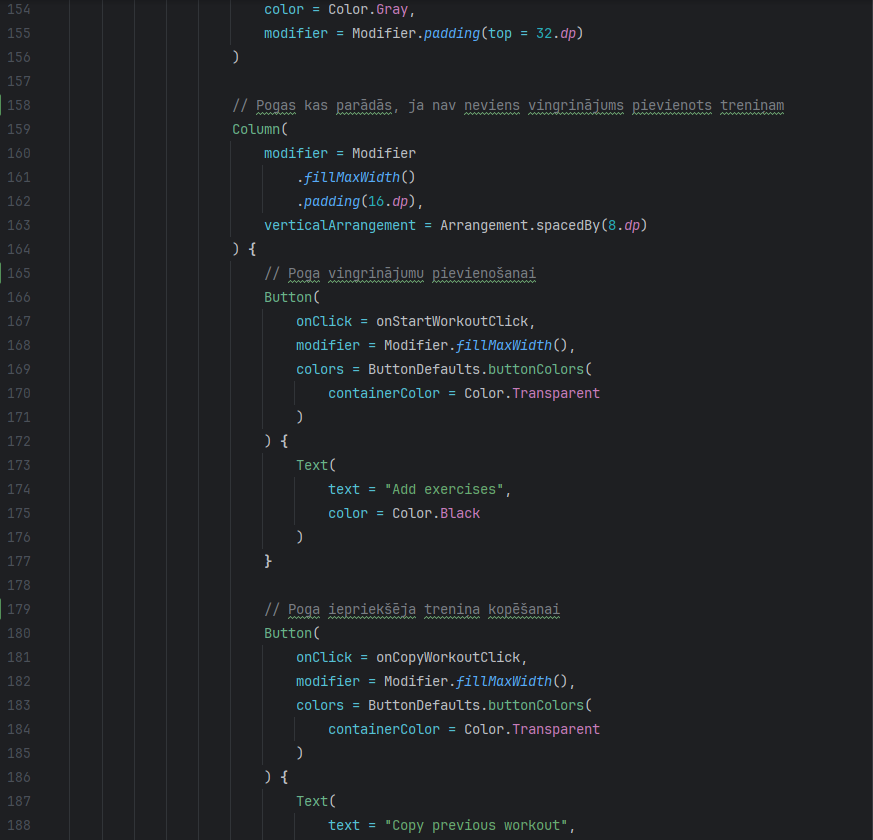
# Pielikumi

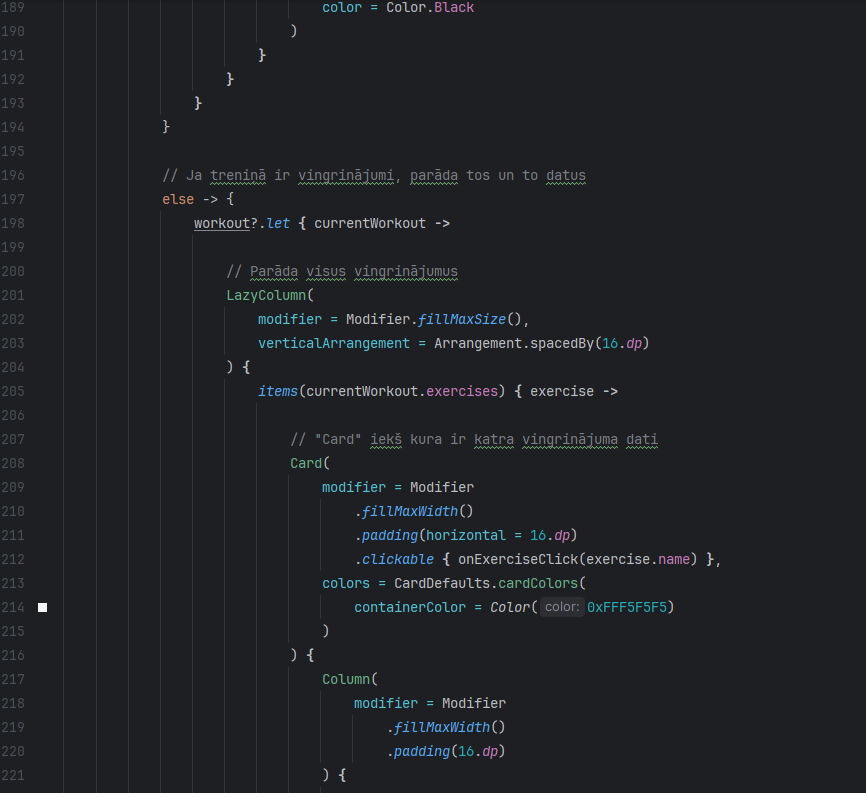
Fails “WorkoutScreen.kt” kas parāda noteiktās dienas treniņu, parādot visus vingrinājumus, izmantoto svaru, atkārtojuma reizes un piegājienus. Fails atļauj lietotājiem sākt jaunus treniņus, kopēt vecākus, kā arī dzēst vingrinājumus iekš treniņa.

1. pielikums

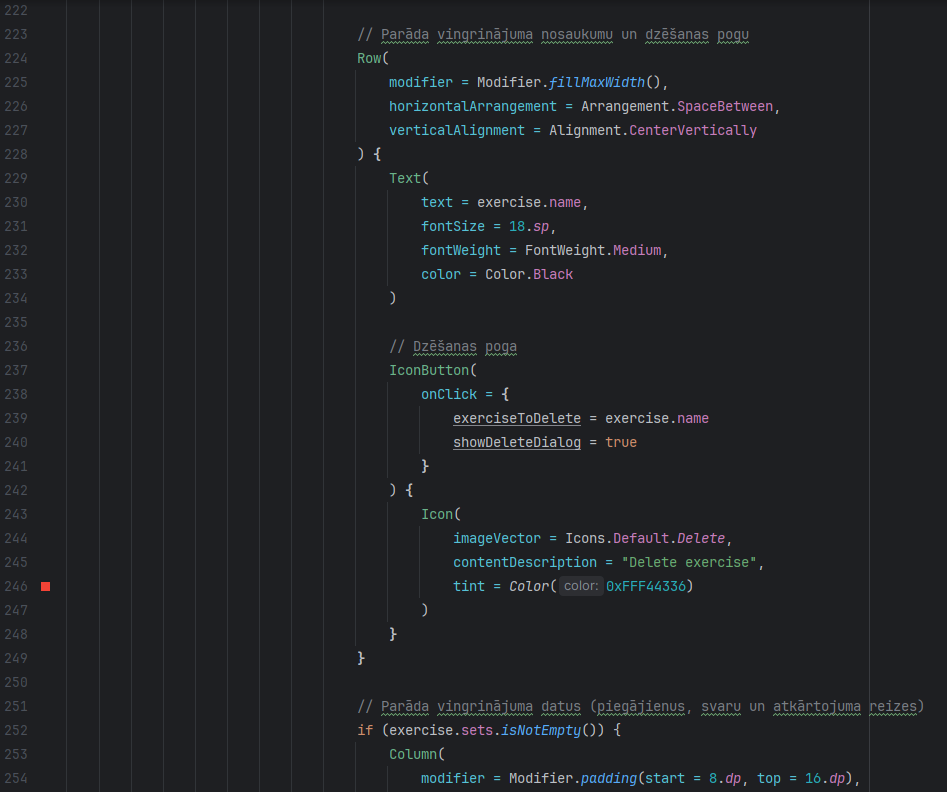
2. pielikums

3. pielikums

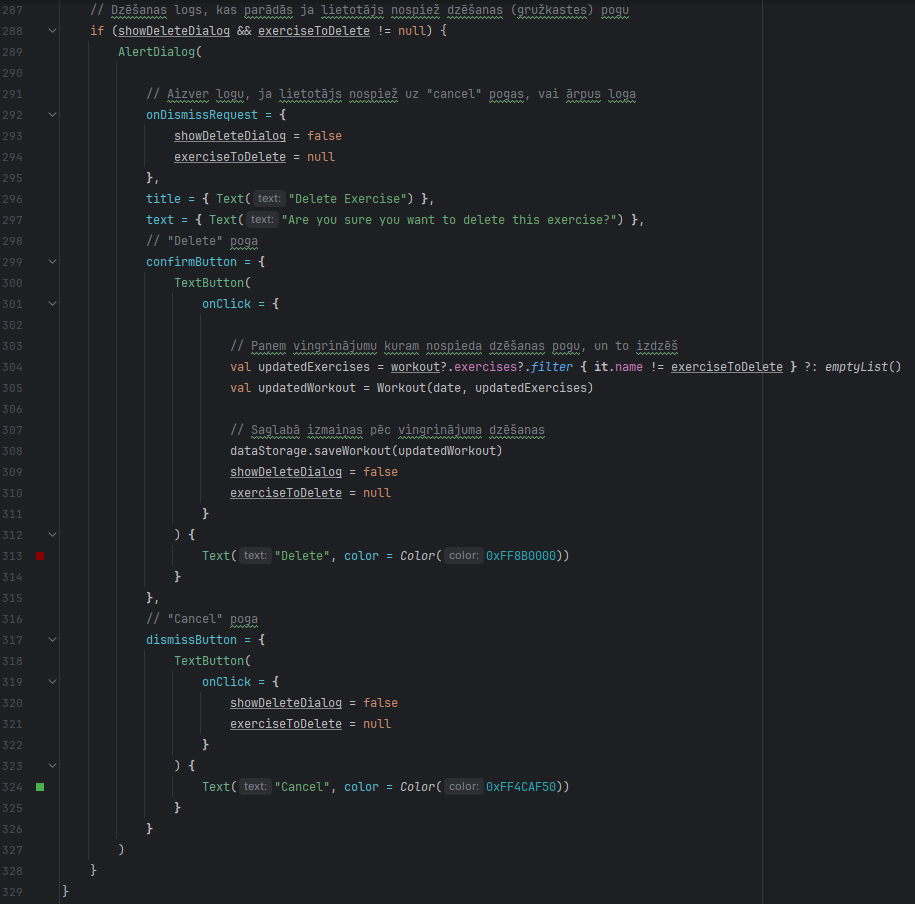
4. pielikums

5. pielikums

6. pielikums



7. pielikums

8. pielikums