**Profesionālās izglītības kompetences centrs „Rīgas Valsts tehnikums”**

Izglītības programma: Programmēšana

**KVALIFIKĀCIJAS DARBS**

**Firmas „WDMarket” uzdevumu menedžēšanas sistēma**

Audzēknis: Edžus Kivilands

Vadītājs: Māris Danne

Normu kontrole: Olga Sabanska

**ANOTĀCIJA**

Kvalifikācijas darbā ir aprakstīta jaunā “WDMarket” uzdevumu menedžēšanas sistēmas izstrādes process. Sistēma dod iespēju projekta vadītājiem viegli un ērti pievienot uzdevumus un projektus, kā arī pārvaldīt tos un redzēt statusus, mainīt prioritātes, veikt komentārus, veidot apakšuzdevumus un pievienot jaunus darbiniekus sistēmai. Kā arī viegli uzskatīt visus datus. Sistēma tika izstrādāta ar Javascript ietvaru/bibliotēku React, Javascript serveri NodeJS un MangoDB objektu orintētu datubāzi.

Kvalifikācijas darbs ietver ievadu, uzdevuma nostādni, prasību specifikāciju, uzdevuma risināšanas līdzekļu izvēles pamatojumu, programmatūras produkta modelēšanu un projektēšanu, datu struktūras aprakstu, lietotāja ceļvedi, nobeigumu, informācijas avotus un pielikumus. Kvalifikācijas darba ievadā ir aprakstīta lielu kompāniju problēma un tās risinājums. Uzdevuma nostādnē ir norādīti uzdevumi, kurus sistēmai būs nepieciešams izpildīt. Prasību specifikācija sastāv no ieejas un izejas informācijas, kā arī no sistēmas funkcionālajām un nefunkcionālajām prasībām. Uzdevumu risināšanas līdzekļu izvēles pamatojumā ir norādīti, kādi līdzekļi tiks izmantoti izstrādāšanai un kādiem nolūkiem tie tiek izmantoti. Programmatūras produkta modelēšanas un projektēšanas apraksts sastāv no sistēmas struktūra modeļa , kas ietver sistēmas arhitektūru un ER modeli, un funkcionālās sistēmas modeļa, kas satur datu plūsmu modeli. Datu struktūras aprakstā tiek parādīta datu bāzes relāciju shēma, kā arī tabulu struktūra ar aprakstu, kas ietver datu tipu un datu garumu norādīšanu. Lietotāja ceļvedī ir norādītas nepieciešamās sistēmas prasības aparatūrai un programmatūrai, sistēmas instalācija un palaišana, kā arī programmas apraksts, kas paskaido kā pareizi lietot sistēmu. Testa piemērā ir dots izveidot jaunu uzdevumu pārvietot tā stāvokli un statusu, kā arī rediģēt noteikto uzdevumu.

Kvalifikācijas darbs sastāv no x lapaspusēm, kurā ietilpst x attēli, x tabulas un x pielikumi. Pielikumi satur sadaļas uzdevumu dizainu, funkcionālās dekompozīcijas diagrammu, ER diagrammu, uzdevuma izveidošanas datu plūsmu diagrammu un datu bāzes ierakstu izgušanas metodes.

**ANNOTATION**

Qualification work describes the process of developing the new WDMarket task management system. The system allows project managers to easily and conveniently add tasks and projects, as well as manage and view statuses, change priorities, make comments, create subtasks and add new employees to the system. As well as easy to view all data. The system was developed with Javascript framework / library React, Javascript server NodeJS and MangoDB object-oriented database.

The qualification work includes an introduction, a task statement, a specification of the requirements, a rationale for choosing the means of solving the task, a modeling and design of the software product, a description of the data structure, a user guide, a conclusion, information sources and attachments. The introduction to the qualification paper describes the problem of large companies and its solution. The task statement specifies the tasks that the system will need to perform. The requirements specification consists of input and output information as well as system functional and non-functional requirements. The rationale for the choice of task-solving tools indicates which tools will be used for development and for what purposes. The software product modeling and design description consists of a system structure model that includes a system architecture and an ER model, and a functional system model that contains a data flow model. The description of the data structure shows the database relational scheme, as well as the table structure with a description that includes the data type and data length. The user guide provides the necessary hardware and software system requirements, system installation and startup, and a program description that explains how to use the system properly. An example of a test is to create a new task, move its position and status, and edit the specified task.

The qualification paper consists of x pages, which includes x iamges, x tables and x attachments. The attachments contain section task designs, functional decomposition diagrams, ER diagrams, task creation data flow diagrams, and database record retrieval methods.

**SATURS**

[**IEVADS**](#_fp4qwj44otwk) **5**

[**Uzdevuma nostādne**](#_ti05t6rk2z2a) **6**

[**Prasību Specifikācija**](#_n10ktyxqqpv) **7**

[Ieejas un izejas informācijas apraksts](#_w8c6h7f0ow1e) 7

[Ieejas informācijas apraksts](#_qj8wrgtsxked) 7

[Izejas informācijas apraksts](#_ilzlglg5c3af) 9

[Funkcionālās prasības](#_rof4scuebko6) 10

[Nefunkcionālās prasības](#_wyprsq3icfag) 12

[**Uzdevuma risināšanas līdzekļu izvēles pamatojums**](#_du1v5lmbidf0) **14**

[**Programmatūras produkta modelēšana un projektēšana**](#_h558u6e8pwu7) **16**

[Sistēmas struktūras modelis](#_pd16c5h3x4ho) 16

[Sistēmas arhitektūra](#_e2uhr41vob0t) 16

[Sistēmas ER modelis](#_6b25juejn5wl) 19

[Funkcionālās sistēmas modelis](#_c9dg3oxi6mtq) 20

[Datu plūsmu modelis](#_7ga06yvxu8p6) 20

[**Datu struktūras apraksts**](#_mpzqupoq1rty) **26**

[**Lietotāja ceļvedis**](#_wu6iyuwr5dn) **29**

[Sistēmas prasības aparatūrai un programmatūrai](#_dtf6irvzzyp8) 29

[Sistēmas instalācija un palaišana](#_bk7pm9s2y2oz) 32

[Programmas apraksts](#_t5up8u3q5bfg) 33

[Testa piemērs](#_cgnhsq6h3vmy) 35

[**Nobeigums**](#_6g4f0kz8bjmg) **37**

[**Izmantotie avoti**](#_rxs4wnyavo10) **38**

# 

# IEVADS

Tehnoloģijām attīstoties paverās daudz jaunu iespēju biznesiem, kā piemēram, biznesa vadītājiem, projekta vadītājiem, mārketinga speciālistiem un darbiniekiem tiek radītas sistēmas, lai atvieglotu ikdienas dzīvi darbā strādājot ar klientiem. Lai atbildīgā persona varētu visi viegli pārskatīt, klientus vai uzdevumu statusus, tiek pielietotas “CRM” Client Relationship managment, jeb klientu attiecību pārvaldības sistēmas, kas atbild par datu uzglabāšanu un precīzu attēlošanu. Šī kvalifikācijas darba mērķis ir izveidot sistēmu, kur projekta vadītāji varēs izveidot projektus ar kuriem ir noslēgts līgums un veidot uzdevumus, darba izpildītājiem, kuri varēs aplūkot šos uzdevumus un pārvietot pa vairākiem statusiem, tādejādi atbildīgā persona varēs redzēt kāds ir darba progress. Sistēmai paredzēts lietotājam draudzīgs un izskatīgs dizains. Kā arī, paredzēts nākotnē šo sistēmu izmantot kā bāzi pārdodot citiem uzņēmumiem un pielāgojot viņu vajadzībām. Protams, šādas sistēmas jau pastāv, bet daudzām ir mīnusi, piemēram, lielāka maksa mēnesī par nepieciešamo papildus funkcionalitāti, vai arī ir pārāk daudz nevajadzīgu funkciju, un daudzām sistēmām ir grūti uztverama informācija, vai arī grūti atrodama.

# Uzdevuma nostādne

Kvalifikācijas darba udzevums ir izveidot uzdevumu menedžēšanas sistēmu uzņēmumam SIA “WDMarket”, pēc jaunā dizaina. Ar sistēmas palīdzību būs viegli izveidot projektus un likt klāt uzdevumus katram projektam, ar prioritātes statusiem, kā arī būs iespēja šos uzdevumus viegli pārlikt ar peles vilcienu, nevis manuāli ar pogu. Pēc jaunā dizaina izstrādes būs viegli pārskatāma informācija visā sistēmā. Sistēmai ir jānodrošina iespēja izveidot projektus, projketiem pievienot uzdevumus, savukārt uzdevumiem jābūt iespējai uzstaisīt ‘checklist’ ar apakš uzdevumiem un komentāriem. Jābūt iekļautām lomām priekš nākotnes sistēmas attīstības, kad tai pievienos papildus funkcionalitāti. Jābūt arī meklētājam jeb filtram kurš izfiltrē uzdevumus pēc projektiem.

SIA “WDMarket” jaunajai sistēmai ir jāizpilda vairākas funkcionalitātes:

* Projektu un uzdevumu izveidošana
* Uzdevumu labošana un komentāru pievienošana
* “Drag and drop” funkcionalitāte, kas nodrošina tasku pārlikšanu ar peles vilkšanu
* Uzdevumu filtrēšana pēc projekta
* Uzdevumu laika skaitīšana
* Autorizācija, lai nepiederošas persona netiek klāt

# 

# Prasību Specifikācija

## Ieejas un izejas informācijas apraksts

### Ieejas informācijas apraksts

Sistēmā tiks nodrošināta šāda ieejas informācijas apstrāde.

1. Informācija par **Lietotājiem** sastāvēs no sekojošiem datiem.

* Vārds - burtu teksts ar izmēru līdz 30 rakstzīmēm
* Uzvārds - burtu teksts ar izmēru līdz 30 rakstzīmēm
* E-pasts - teksta, ciparu un simbolu kombinācija ar izmēru līdz 50 rakstzīmēm un atļauto simbolu “@”
* Telefona numurs - ciparu un simbolu kombinācija ar izmēru līdz 12 rasktzīmēm un atļauto simbolu “+”
* Parole - teksta, ciparu un simbolu kombinācija ar izmēru no 12 līdz 50 rakstzīmēm

1. Informācija par **Uzdevumiem** sastāvēs no sekojošiem datiem.

* Nosaukums - teksta un ciparu kombinācija ar izmēru līdz 50 rakstzīmēm
* Apraksts - teksta un ciparu kombinācija ar visiem simboliem un iespēju ievietot saiti uz kādu saitu
* Statuss - burtu teksts ar izmēru līdz 15 rakstzīmēm
* Prioritāte - burtu teksts ar izmēru līdz 15 rakstzīmēm
* Izpildītājs - burtu teksts ar izmēru līdz 30 rakstzīmēm ( lietotājs kuram jaizpilda uzdevums )

1. Informācija par **Projektiem** sastāvēs no sekojošiem datiem.

* Nosaukums - teksta un ciparu kombinācija ar izmēru līdz 50 rakstzīmēm

1. Informācija par **Apakškategorijām** sastāvēs no sekojošiem datiem.

* Apakšuzdevuma teksts - teksta un ciparu kombinācija ar visiem simboliem un iespēju ievietot saiti uz kādu saitu

1. Informācija par **Komentāriem** sastāvēs no sekojošiem datiem.

* Komentāra teksts - teksta un ciparu kombinācija ar visiem simboliem un iespēju ievietot saiti uz kādu saitu
* Izveidotājs - burtu teksts ar izmēru līdz 30 rakstzīmēm ( lietotājs kurš izveidoja komentāru )

### Izejas informācijas apraksts

1. Uzdevumu vizuālais atspoguļojums. Šeit būs pieejami visi uzdevumi kuri ir izveidoti un izpildītājs ir konkrētais lietotājs. Šos uzdevumus būs iespējams pārvilkt uz citiem statusiem, piemēram, uzdevums stāv zem kolonnas “ToDo” un lietotājs izpilda šo uzdevumu, un pārvelk uz kolonnu “Testing”, tad attiecīgi atbildīgā persona redz ka ir pielikts zem “Testing kolonnas un pārbauda vai uzdevums ir izpildīts pareizi.
2. Projektu vizuālais atspoguļojums. Šeit būs redzami visi projekti, lai sekotu līdzi projekta statusam, kā arī šeit var pievienot klāt funkcionalitāti ko es saucu par “Access”, tas ļauj mums pievienot noderīgu informāciju klāt projektam, kuru vēlāk varam apskatīties pie uzdevumiem ar konkrēto projektu.
3. Uzdevumu filtrācijas atspoguļošana. Šeit būs pieejami uzdevumi lietotāja izvēlētajam projektam, kuru lietotājs ir atfiltrējis. Princips tas pats kas pirmajā punktā, tikai uzdevumiem kas pieder pie specifiskā projekta.

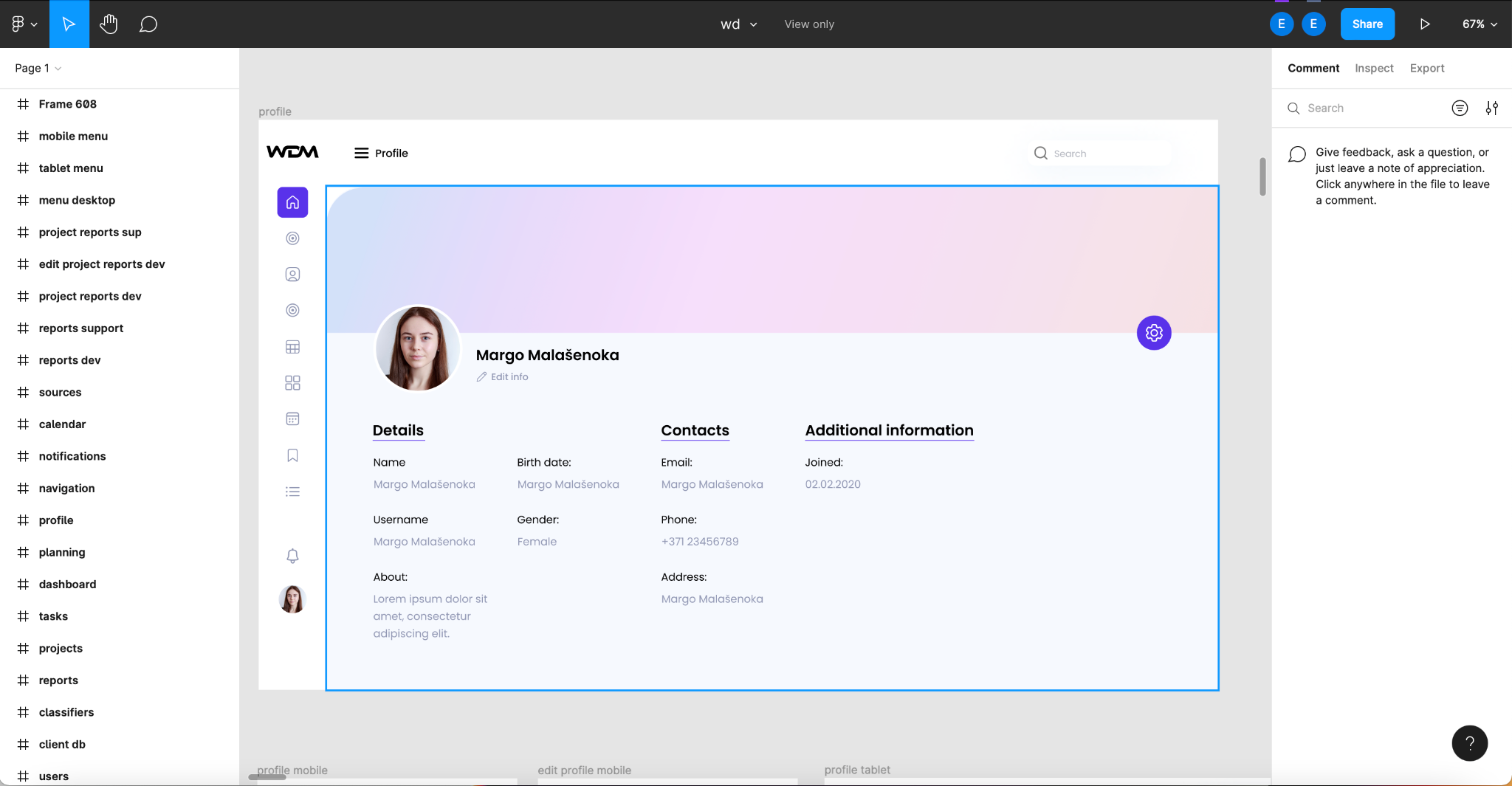
## Funkcionālās prasības

1. Jānodrošina lietotāja autorizācija.
   1. Pārbauda vai lietotāja statuss ir derīgs (JSON web token).
   2. Ja lietotāja statuss ir derīgs, tad to automātiski atcerēties un katru reizi autorizēt lietotāju.
   3. Ja lietotāja statuss ir novecojis, tad automātiski klientu izlikt arā no sistēmas un likt lietotājam autorizēties sistēmā pa jaunam.
   4. Ja kāds no autorizācijas laukiem ir aizpildīts nepareizi parādas paziņojums, ka informācija ievadīta nepareizi.
   5. Jaunu lietotāju varēs izveidot tikai esošs lietotājs, par cik sistēma domāta vienam uzņēmuma vai cilvēkam.
2. Jānodrošina “Drag and drop” funkcionalitāte.
   1. Lietotājam jābut iespējai ar peli pārvilkt uzdevumus no vienas kolonnas uz citu kolonnu.
   2. Pārvelkot uzdevumu vai projektu no vienas kolonnas uz citu ir jāsaglabā jaunais uzdevuma vai projekta statuss.
   3. Uzdevumu vai projektu nebūs iespējams pārlikt tajā pašā kolonnā, ja tas tiek darīts uzdevums vai projekts atlec atpakaļ savā vietā.
3. Jānodrošina uzdevumu un projektu informācijas saglabāšana.
   1. Jābūt iespējai izveidot jaunu projektu.
   2. Jābūt iespējai rediģēt izveidotu projektu.
   3. Jābūt iespējai izdzēst izveidotu projektu.
   4. Jābūt iespējai pievienot “Access” datus klāt projektam.
   5. Jābūt iespējai labot “Access” datus projektam.
   6. Jābūt iespējai izveidot jaunu uzdevumu.
   7. Jābūt iespejai rediģēt esošu uzdevumu.
   8. Jābūt iespējai izdzēst esošu uzdevumu.
   9. Jābūt iespējai atfiltrēt uzdevumu sarakstu pēc noteikta projekta.
   10. Jābūt iespējai pievienot komentārus uzdevumiem.
   11. Jābūt iespējai pievienot apakšuzdevumus klāt uzdevumam.
   12. Jābūt iespējai izdzēst apakšuzdevumus.
   13. Jānodrošina ka noteikto uzdevumu var redzēt tikai izveidotājs un lietotājs kuram noteiktais uzdevums ir uzdots.
4. Lietotāja profilu datu maiņa.
   1. Lietotāja vārdu un uzvārdu rakstzīmēm jābūt vismaz 3 un maksimums 30 rakstzīmes.
   2. Lietotāja e-pastam ir jaatbilst e-pasta izskatam, nevar nomainīt vai izveidot e-pastu kurš ir neatbilstošs.
   3. Lietotāja parolei jābūt vismaz 12 rakstzīmes garai līdz 50 rakstzīmēm.
   4. Nevar izveidot lietotāju ar e-pastu kurš jau eksistē datubāzē.
5. Sistēmas vispārējā funkcionalitāte un ātrums.
   1. Meklējot neeksistējošas lapas caur meklēšanas joslu, sistēma ielādēd 404 lapu, kas piedāvā iespēju doties atpakaļ uz sākumlpau.
   2. Nereģistrētiem lietotājiem nebūs iespēja apskatīt neko, izņemot autorizācijas logu.
   3. Dažas komponentes, tiks ielādētas tikai nepieciešamajā laikā, lai uzlabot aplikācijas ātrumu.
   4. Sistēma veiks informācijas pieprasījumus, tikai kad tas būs nepieciešams, un ja dati nebūs pietiekami ātri ielādējušies, tad uzrādīsies lādēšanās ikona.
   5. Navigāciju ir iespējams paslēpt, ja tā nav nepieciešama, lai labāk varētu uzskatīt datus.
   6. Noteiktu kompontes stāvokļi tiks saglabāti, meklētāja lokālajā datubāzē, lai atcerētos kā lietotājs ir izmantojis sistēmu.

## Nefunkcionālās prasības

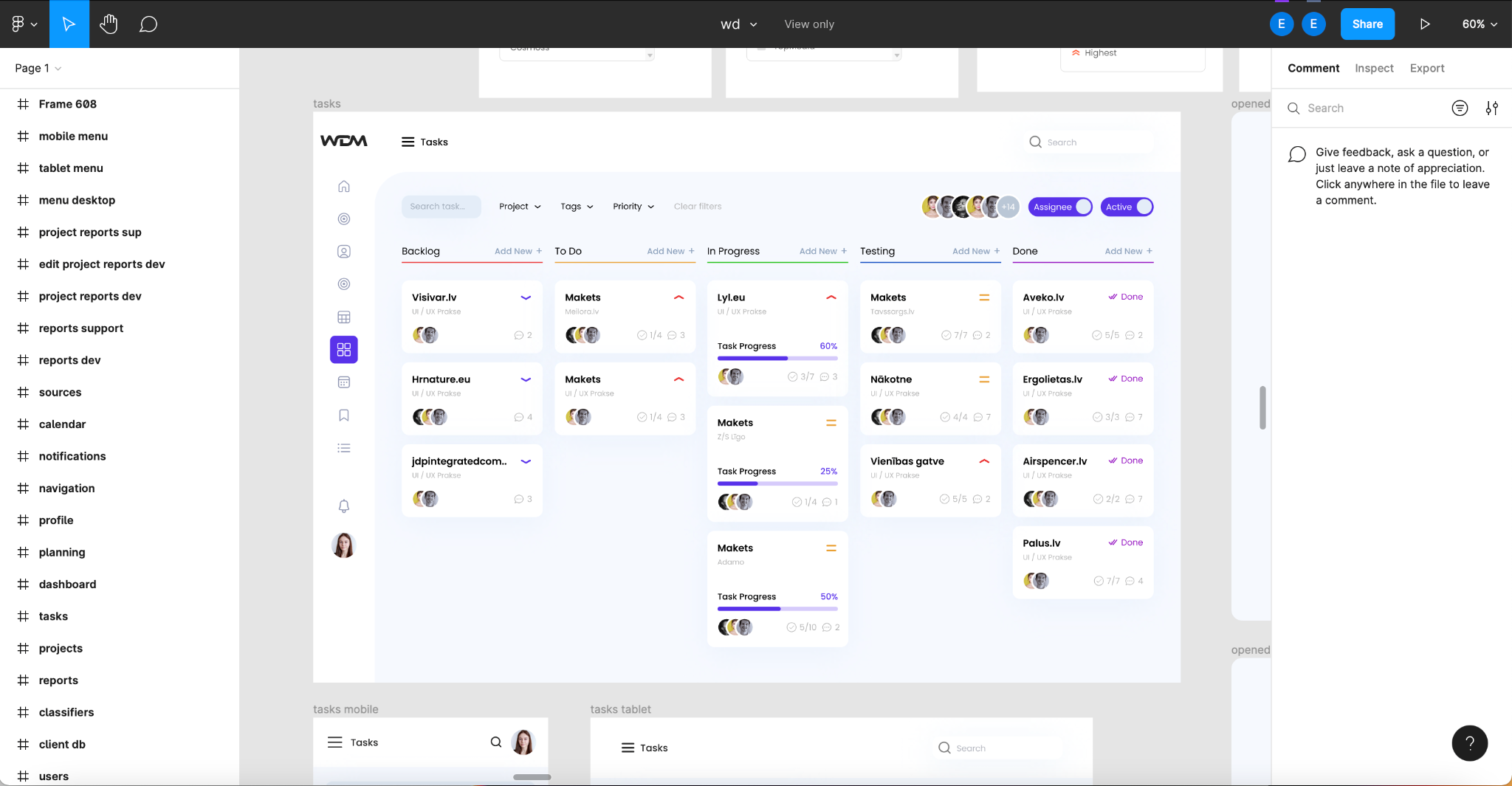
1. Sistēmas saskarnes valodai ir jābūt angļu valodai.
2. Sistēmas pārvietošanās notiek caur lietotāja pusi nevis servera pusi, lai ātrāk redzētu citu lapu, tā vietā lai pārlādētu lapu, nomainas tikai saturs nepārlādējot lapu.
3. Sistēmai jābūt viegli uzskatāmai informācijai, izmantotas maigas krāsas lai viegli uzņemt informāciju un modernisks ar dizains.
4. Navigācija atrodas kreisajā pusē ar iespēju to paslēpt.
5. Uzdevumu statusu un paziņojumu krāsām jābūt lietotājam draudzīgām un saprotamām.
6. Virsrakstiem un nosaukumiem jābūt izceltiem, un atvērtā uzdevuma izkārtojumam jābūt pārskatāmam.
7. Aizvēršanas atvēršanas pogām jābūt novietotām lietotājam ērtās vietās.
8. Apakšuzdevumiem jābūt viegli uzskatāmiem un izpildītajiem jābūt marķētiem ar zaļu krāsu.
9. Sistēmas augšdaļā jāparādas kurā lapā atrodies.

Profila skats:



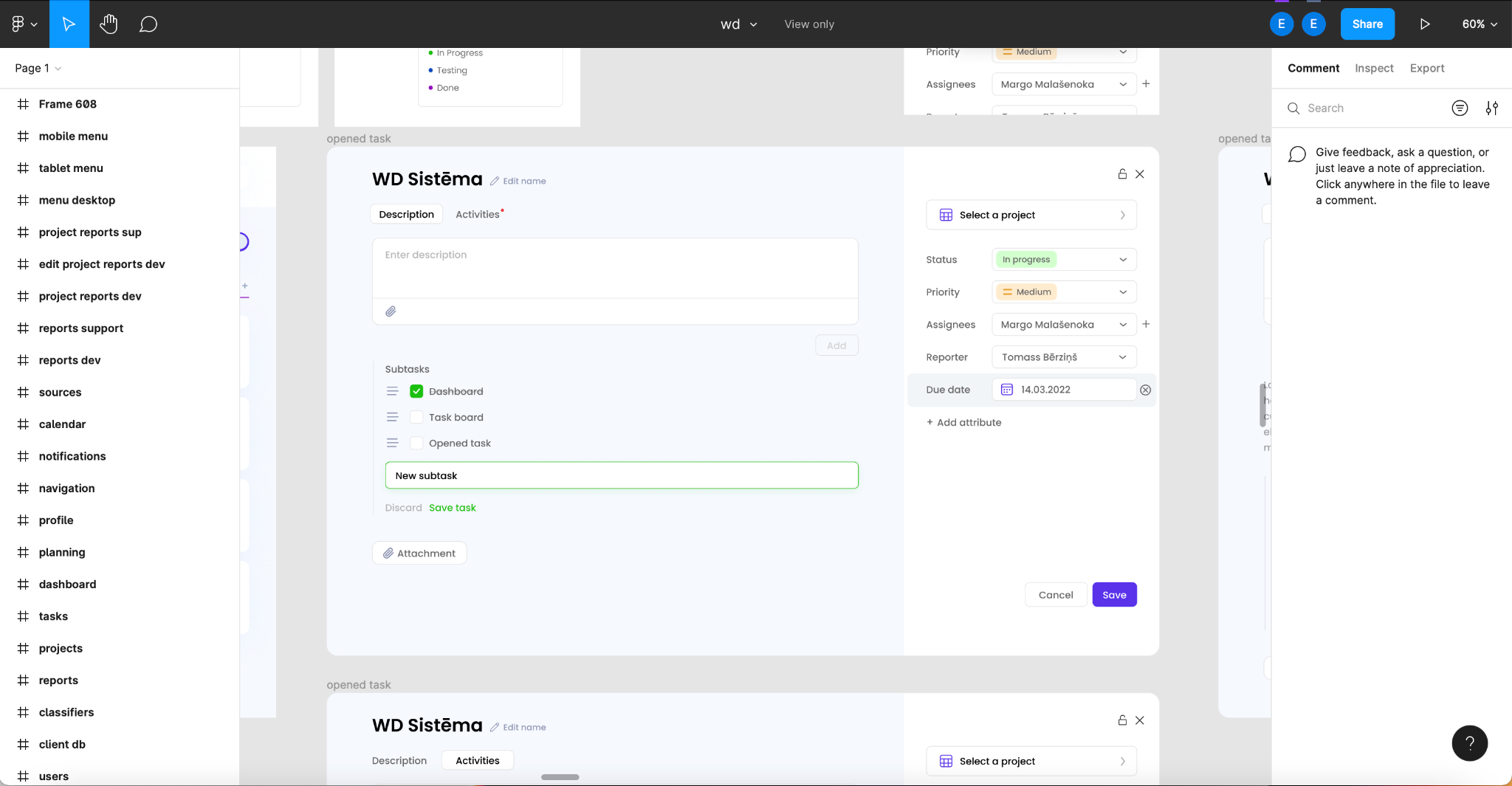
1.att. Profila skats

Uzdevumu saraksta skats:



2.att. Uzdevumu saraksta skats

Atvērta uzdevuma skats:



3.att. Atvērta uzdema skats

# Uzdevuma risināšanas līdzekļu izvēles pamatojums

Izmantotās programmēšanas valodas un tehnoloģijas:

* HTML (versija - 5) - Izmantota sistēmas strktūras izveidei.
* CSS (versija - 3) - Izmantota lai būvētu sistēmas stilu un izkārtojumu.
* Javascript (versija - ECMA-6) - Izmantota priekš sistēmas funkcionalitātes kā arī nodrošina datu struktūru priekš datubāzes un satur visu “backend” funkcionalitāti
* Mangodb (versija 3.3.0) - Izmantota priekš datu glabāšanas un savienošanās ar datubāzi.
* React (versija - 17) - Javascript ietvars ar kuru būvēta visas sistēmas lietotāja puse. React ietvars tiek izmantots, jo viegli darboties ar komponentēm un to stāvokļiem, kā arī React ir ļoti ātrs un lielākā daļa programmētāji vēljoprojām izmanto šo ietvaru, lai veidotu web aplikācijas, tapēc ir daudz resursi un papildinājumi kurus var izmantot kuri nav jātaisa no nulles.
* TailwindCSS (versija - 3) - CSS ietvars, ar kura palīdzību var ātri un vienkārši izveidot stilu priekš komponentēm.
* NodeJs (versija - 16.14.2) - Javascript vide priekš servera puses, ar NodeJs tiek būvēta visa servera puses funkcionalitāte.
* ExpressJs (versija - 4.17) - NodeJs ietvars, lai vieglāk izveidotu servera puses funkcionalitāti.

Izmantotās programmas:

* Visual studio code (versija - 2022 gada) - Izmantots lai rakstītu kodu un visu būtu viegli pārskatīt gan kodu, gan aplikācijas struktūru.
* Github Desktop - Izmantots, lai saglabātu kodu uz github.

Izmantotie moduļi:

* Immer (versija - 9) - izmantota lai labāk strukturizētu komponenšu stāvokļus un izmantotu kopiju no stāvokļa nevis mainītu stāvokli pa taisno.

<https://www.npmjs.com/package/immer>

* React beautiful dnd (versija - 13.1.0) - izmantota priekš “Drag and drop” funkcionalitātes veidošanas. <https://www.npmjs.com/package/react-beautiful-dnd>
* React router dom (versija - 6.3.0) - izmantota priekš lietotāja puses pārvietošanās, lai nav jāparlādē lapa, aizejot uz citu lapu. <https://www.npmjs.com/package/react-router-dom>
* Postcss (versija - 8.4.12) - Izmantota priekš TailwindCSS apstrādes. <https://www.npmjs.com/package/postcss>
* Autoprefixer (versija - 10.4.4) - Izmantota priekš TailwindCSS darbības. <https://www.npmjs.com/package/autoprefixer>
* Babel (versija - 7) - Imanotota priekš Javascript kompilācijas. <https://www.npmjs.com/package/@babel/core> <https://www.npmjs.com/package/@babel/preset-env>
* Axios ( versija - 0.27.2) - Izmantota lai viektu HTTP datu pieprasījumus. <https://www.npmjs.com/package/axios>
* JSON web token (versija - 8.5.1) - Izmanota lai pārbaudītu lietotāja sesijas derīgumu. <https://www.npmjs.com/package/jsonwebtoken>
* MD5 (versija - 2.3.0) - Funkcija lai pārveidotu lietotāja paroli. <https://www.npmjs.com/package/md5>

# Programmatūras produkta modelēšana un projektēšana

## Sistēmas struktūras modelis

### Sistēmas arhitektūra

Datu bāzes projektēšanā datu kopu un saišu starp tām attēlošanai tika lietota realitāšu-saišu diagramma, kas sastāv no divu veidu objektiem – entītēm (loģiskais objekts) un relācijām (attiecības). ER modelis (sk.1.pielikumā) sastāv no 5 entītijam, kas atspoguļo datu aprīti sistēmā.

* **“Lietotājs”** – uzskaita sistēmas lietotājus. Tās atribūtu kopums sevī ietver vārdu, uzvārdu, e-pastu un paroli
* **“Projekts”** – uzskaita sistēmas projektus, kuri ir izstrādes procesā. Tās atribūtu kopums sevī ietver projekta nosaukumu.
* **“Uzdevumi”** – uzskaita projektu uzdevumus. Tās atribūtu kopums sevī ietver uzdevuma nosaukumu, aprakstu, uzdevuma izpildītāju, statusu un prioritāti
* **“Apakšuzdevums”** – uzksiata apakšuzdevumus noteiktajam uzdevumam. Tās atribūtu kopums sevī ietver apakšuzdevuma tekstu.
* **“Komentārs”** – uzskaita uzdevuma komentārus. Tās atribūtu kopums  
  sevī ietver komentāra izveidotāju un komentāra tekstu.

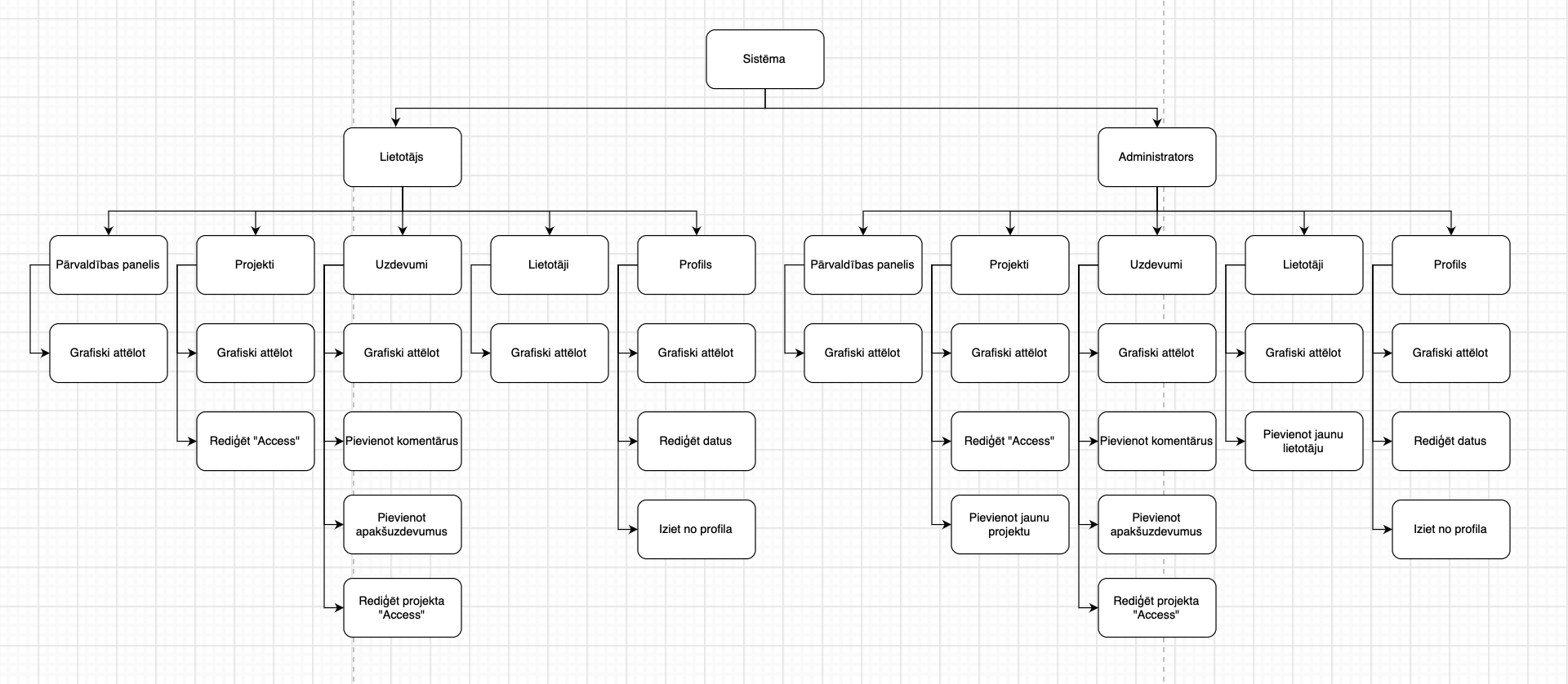
Datu bāzes relācijas uzrāda kā savstarpēji ir savienotas divas vai vairākas entītijas.

* Starp lietotājiem un projektiem attiecība ir viens pret daudziem, jo viens lietotājs tikai var izveidot projektu.
* Starp projektiem un uzdevumiem ir saite daudzi pret daudziem, jo vienam projektam var būt vairāki uzdevumi, un viens uzdevums var būt vienāds vairākiem projektiem.
* Starp uzdevumiem un apakšuzdevumiem attiecība ir viens pret daudziem, jo vienam uzdevumam var pievienot vairākus apakšuzdevumus.
* Starp uzdevumiem un komentāriem attiecība ir viens pret daudziem, jo vienam uzdevumam var pievienot vairākus komentārus.

Sistēmā ir realizēta vispārējām lietotāja darbībām un adminstratora darbībām. Sistēmu paredzēts lietot, tikai piederošām personām, tapēc sistēmā varēs izveidot jaunu lietotāju, vai nu administrators, vai nu ar datubāzes manuālu ieraksta ievietošanu. Sistēmas lietotājam būs iespēja izmantot visu sistēmas funkcionalitāti izņemot pievienot lietotāju un projketus ko attiecīgi varēs darīt administratori. Savukārt nepiederoša persona varēs redzēt tikai “login” logu. Sistēma sastāv no 5 moduļiem.

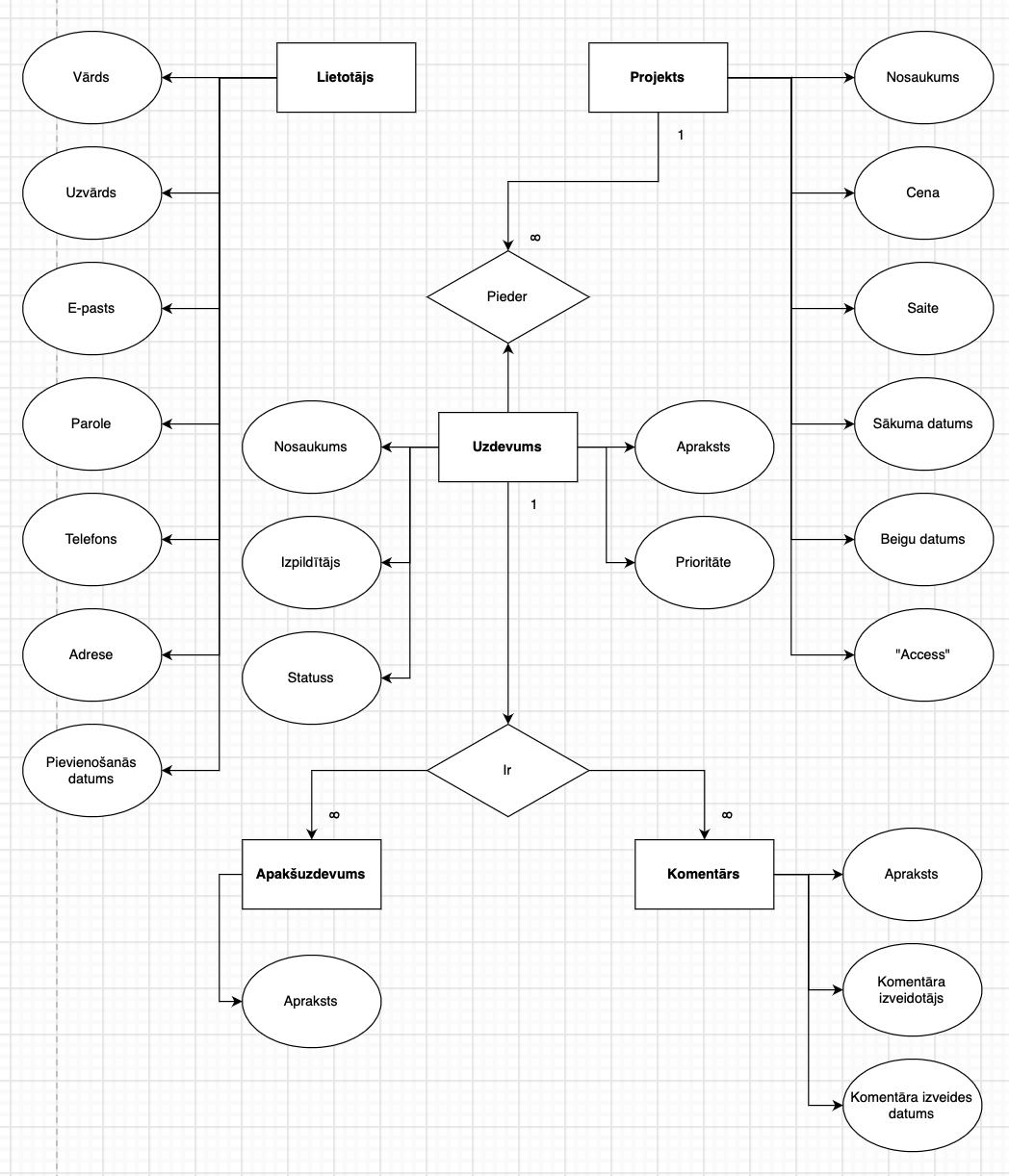
Sistēma visā vietnē, tiek izmantota, lai grafiski reprezentētu datus lietotājam un apstrādāt tos. Iekļauj sevī galvenās ārējās funkcionalitātes.

* Modulis “Uzdevumi” – lietotājs var apskatīt visus uzdevumus, kas ir saistīti ar viņu kā arī atfiltrēt uzdevumus pēc izvēlētā projekta. Parvelkot uzdevumu uz kollonu var nomainīt statusu, un uzspiežot virsū lietotājs var redzēt uzdevuma informāciju sīkāk
  + Grafiski Attēlot – grafiski tiek attēloti uzdevumi, to nosaukums, saistītais projekts, statuss un prioritāte
  + Pievienot komentāru - atverot uzdevuma logu, lietotājs var ievadīt komentārus
  + Pievienot apakšuzdevumus - atverot uzdevuma logu, lietotājs var atķeksēt izpildītos apakšuzdevumus, nodzēst tos un pievienot jaunus.
  + Rediģēt Access - atverot uzdevuma logu, lietotājs var rediģēt projekta “Access”, jeb piekļuves.
* Modulis “Projekti” - lietotājs var apskatīt visus projektus, kas ir izveidoti. Parvelkot projektu uz kollonu var nomainīt statusu, un uzspiežot virsū lietotājs var redzēt projekta informāciju sīkāk
  + Grafiski Attēlot – grafiski tiek attēloti projketi, to nosaukums un statuss, atverot projekta logu vaļā, var apskatīt projekta “Access”.
  + Rediģēt Access - atverot projketa logu, lietotājs var rediģēt projekta “Access”, jeb piekļuves.
  + Pievienot jaunu projektu - atverot jauno lietotāja izveides logu **administratori** varēs pievienot jaunu lietotāju.
* Modulis “Dashboard” - lietotājs var apskatīt aktuālāko informāciju par uzdevumiem un projektiem.
  + Grafiski attēlot - grafiski attēloti aktuālo uzdevumu statusi un aktuālie projketi.
* Modulis “Users” - lietotājs var apskatīt visus lietotājus, kā arī pievienot jaunu lietotāju.
  + Grafiski attēlot - grafiski attēlot visus lietotājus kuri ir autorizēti sistēmai.
  + Pievienot jaunu lietotāju - izveidot jaunu lietotāju pēc nepieciešmības. ( tikai **administratori** )
* Modelis “Profils” - lietotājs var apskatīt visus savus datus kā arī izmainīt tos, un iziet arā no sava profila.
  + Grafiski attēlot - grafiski attēlot visus lietotāja datus.
  + Nomainīt lietotāja datus - lietotājs var izmainīt savus datus pēc vajadzības.
  + Iziet no profila - Iziet arā no sava profila (logout)



5.1.att. Sistēmas arhitektūras modelis

### Sistēmas ER modelis



5.2.att. ER modelis

## Funkcionālās sistēmas modelis

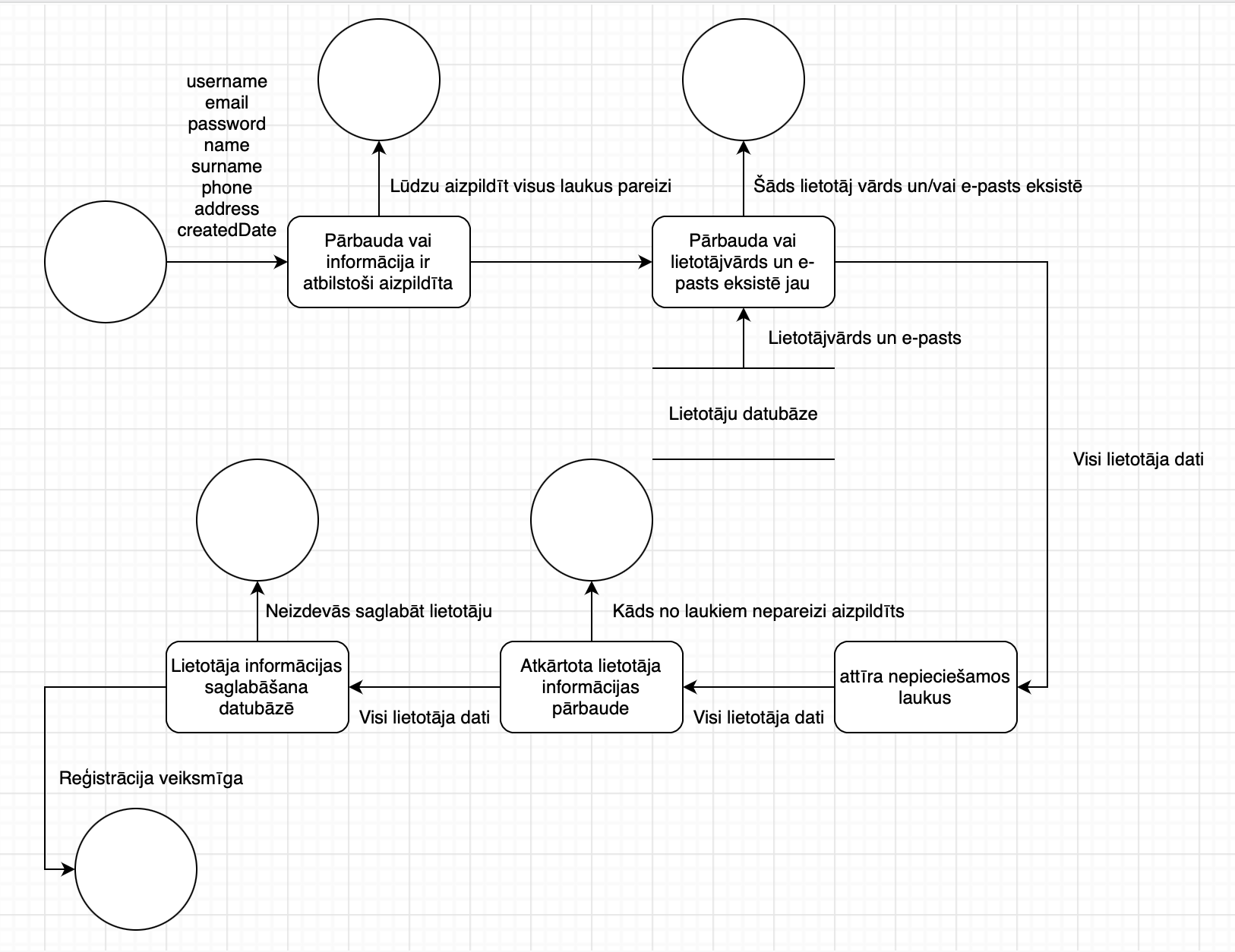
### Datu plūsmu modelis

1. **Lietoāja reģistrācija**

Jauna lietotāja izveidošana ir svarīga, jo savādāk paredzētais lietotājs nespēs piekļut klāt sistēmai. Tākā šī sistēma ir paredzēta tikai vienas kompānijas ietvaros, tad izveidot jaunu lietotāju spēs tikai sistēmas administrators, lai izvairītos no liekas informācijas noplūdes. Lietotāja izveides process noteik sekojoši. (skat. 5.3. att.)

Vispirms administrators navigē uz logu “Users” uzspiež pogu “Add new user”, tālak uznirstošajā laukā ievada visus nepieciešamos datus. Ja kāds no laukiem nav aizpildīts vai aizpildīts nepareizi parādas reāllaika paziņojums.

Kad esam pārliecinājušies ka visi dati ir ievadīti pareizi, tad tiek sūtīts pieprasījums uz serveri ar atbilstošajiem datiem un papildus lauku “createdDate” kurš satur informāciju kad lietotājs ir izveidots. Servera lietotāju kontrolieris izsauc moduļa funkciju, kura savukārt sastāv no solījuma un vēl 2 funkcijām, kuras ir sekojošas - attīrīšanas funkcija, ši funkcija veic teksta attīrīšanu nepieciešamajos laukos, piemēram, ja lietotāja vārds ievadīts ar tukšām atstarpēm, tad šīs atstarpes tiek noņemtas nost, otrā funkcija vēlreiz validē ievadītos datus, vai visos laukos ir ievadīti dati un vai tie ir ievadīti pareizi. Kad validācija un attīrīšana ir pabeigta, tālāk parole tiek samiksēta ar md5 hash algoritmu, tad serveris ievieto jaunu dokumentu datubāzē. Ja viss ir izdevies veiksmīgi, tiek nosutīts izdevies solījums atpakaļ uz kontrolieri, un šī informācija tiek padota atpakļ uz “frontend” sistēmu kur tiek atspoguļota lietotāja pievienošana kā arī parādas paziņojums par lietotāja veiksmīgu pievienošanu.



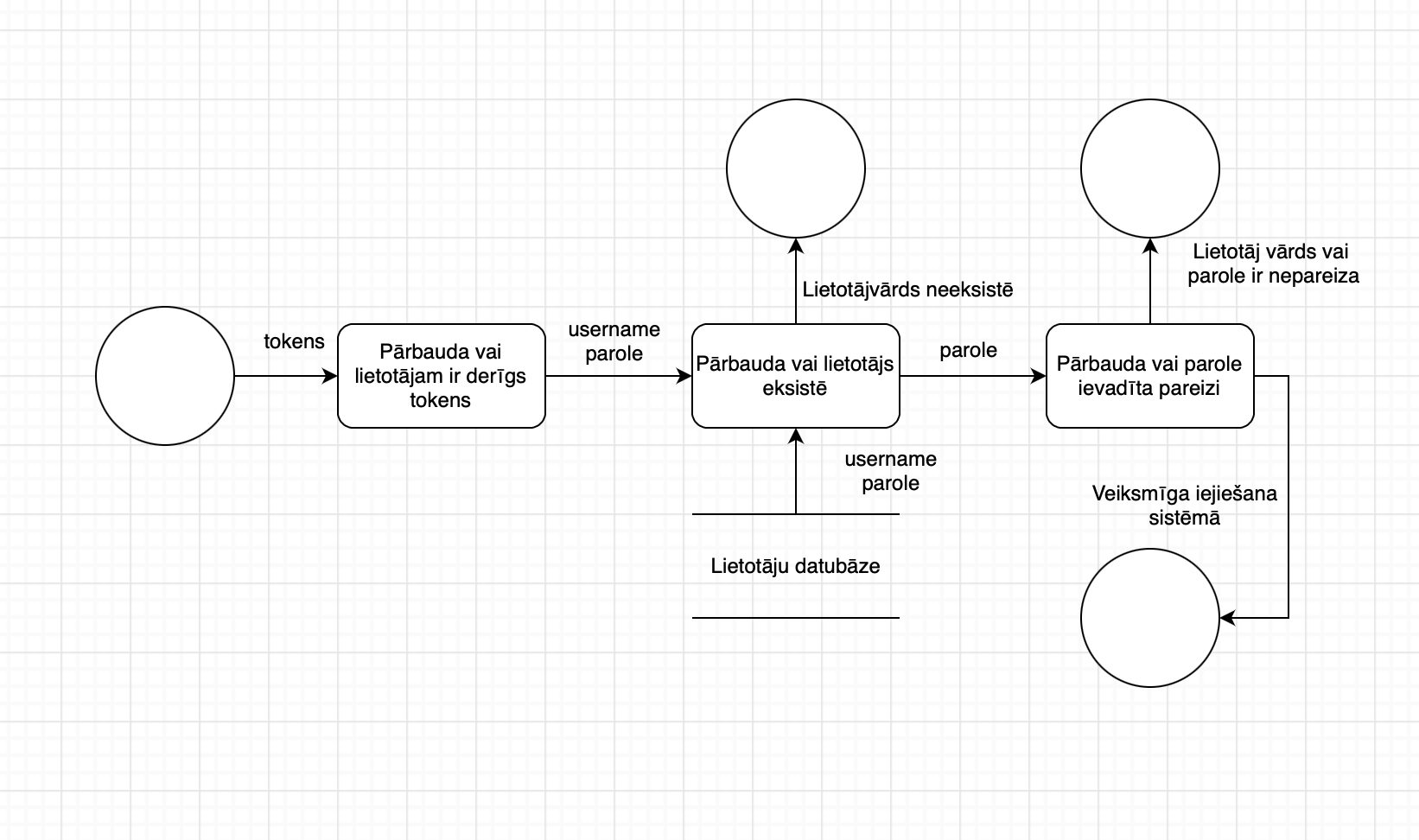
5.3.att. Lietotāja reģistrācijas datu plūsma

1. **Lietotāja iejiešana sistēmā**

Lietotāja autorizācija ir svarīga, lai nepiederošas personas netiek klāt sistēmai, kā arī autorizācijai nevajadzētu notikt katru reizi kad pārlādē tīmekli, tapēc tiek izmantoti WEB tokeni. Lietotāja autorizācijas process noteik sekojoši. (skat. 5.4. att.)

Kad lietotājs ielādē sistēmu, no sākuma sistēma pārbauda vai lokālajā atmiņā ir saglabāts lietotāja tokens, tālāk tā pārbauda vai šis tokens ir piederošs un kam tas pieder.

Ja sistēma atpazīst lietotāju, viņš tiek automātiski autorizēts sistēmai. Ja lietotājam nav saglabāts tokens vai tas ir neatbilstošs, tad jāievada lietotājvārds un parole. Kad dati ir ievadīti, pēc pogas “Login” nospiešanas tiek veikts peiprasījums uz serveri izsaucot “login” funkciju lietotājam. Tiek izveidots solījums, tālāk tiek veikta lietotāja meklēšana datubāzē pēc lietotājvārda, ja lietotājvārds tiek atrasts, tālāk notiek pārbaude vai lietotāja parole sakrīt ar ievadīto paroli. Ja viss norit pareizi un nav problēmas ar datubāzes ieraksta meklēšanu, tad tiks atgriezts veiksmīgs solījums ar lietotāja datiem un lietotājs tiks autorizēts sistēmai un uzrādīsies paziņojums ka lietotājs ir autorizējies veiksmīgi.

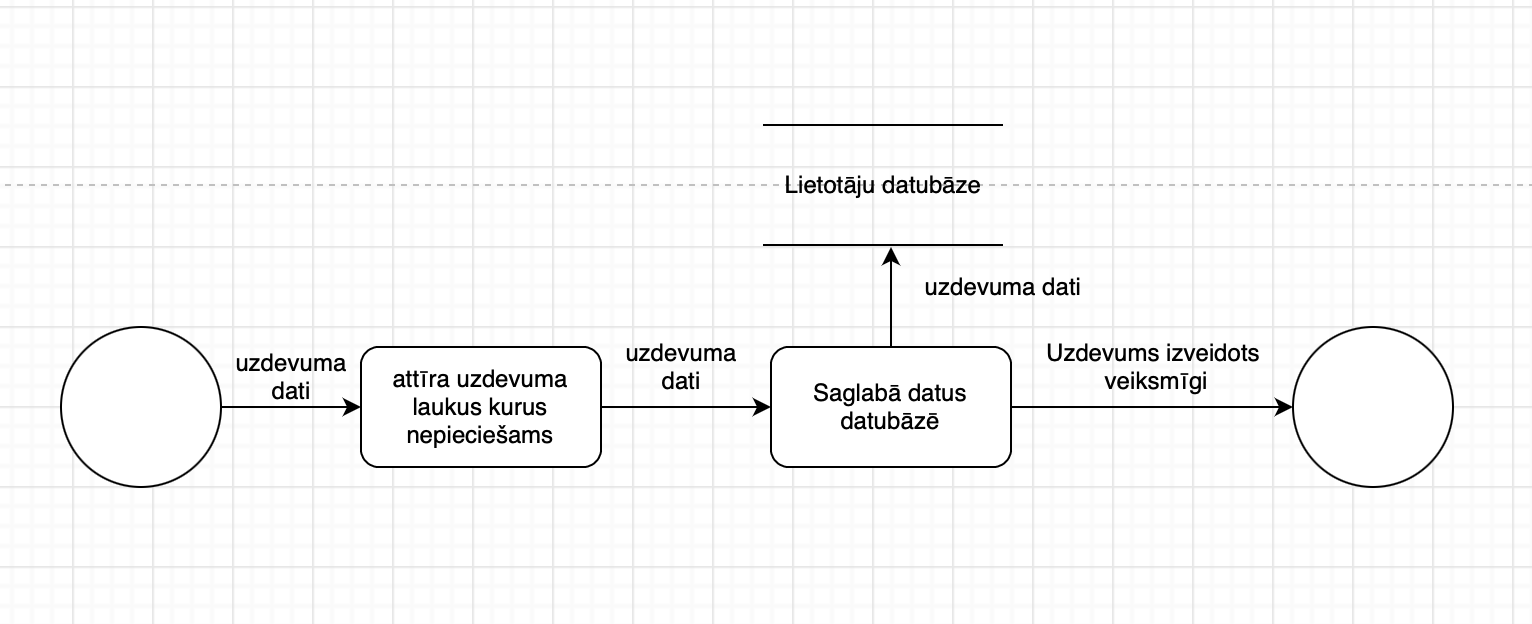


5.4.att. Lietotāja iejiešana sistēmā datu plūsma

1. **Jauna uzdevuma izveidošana**

Uzdevumos nepieciešamā informācija ir ļoti maza, jo dažreiz udevums var sastāvēt nosaukuma un apraksta, bet citas reizes varbūt gan apakšuzdevumi,gan komentāri. Tāpēc uzdevumu izveidošana nav tik strikta kā jauna lietotāja. Uzdevuma izveidošana notiek sekojoši. (skat. 5.5. att.)

Uzdevumā ievada nepieciešamos laukus, kā nosaukumu, aprakstu un prioritāti, tālāk dati tiek nosūtīti uz serveri. Serveris izsauc taska izveidošanas funkciju, attīra laukus - nosaukumu un aprakstu no nevajadzīgām atstarpēm, tālāk datus saglabā datubāze un atgriež veiksmīgu statusu par uzdevuma izveidošanu, ja viss noritēja pareizi.

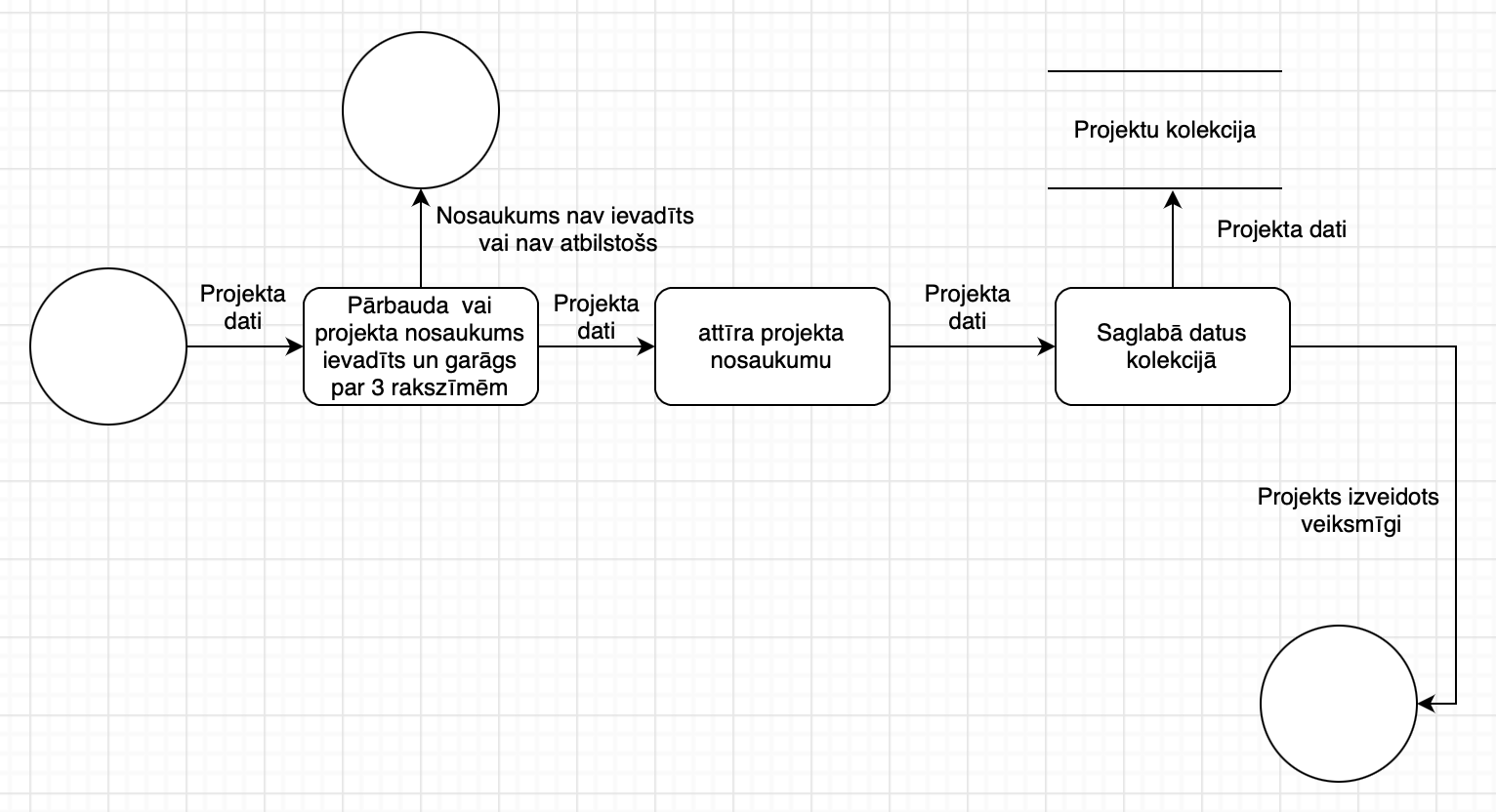


5.5.att. Uzdevuma izveidošanas datu plūsma

1. **Jauna projekta izveidošana**

Ļoti līdzīgi uzdevuma izveidošanai, projektos nepieciešams tikai nosaukums lai izveidotu, protams, var pievienot vēl laukus kā piemēram no kura datuma un līdz kuram datumam jāizveido projekts, vai projekta cena, bet projekts var mierīgi pastāvēt bez visiem laukiem. Galvenais ka ir nosaukums kurš būs piesaistīts uzdevumiem, jo no šī projekta nāks “access” lauks ar projekta pieejam kuras nākotnē noteikti vajadzēs, lai nebūtu pa datoru jāmeklē katra projekta piekļuves. Projekta izveidošana notiek sekojoši. (skat. 5.6. att.)

Ievadam visus projekta datus. No sākuma pārbauda vai nosaukuma laukā kautkas ir ievadīts un vai nosaukums ir garāks par 3 rakstzīmēm, tad projekta dati tiek nosūtīti uz servera kontrolieri kurš piemeklē nepieciešamo funkciju. Funkcija attīra no nevajadzīgām atstarpēm, un tad saglabā datubāzē. Ja viss notiek korekti tiek sūtīts atpakaļ paziņojums ka projekts ir izveidots un tas parādas uzdevumu logā.

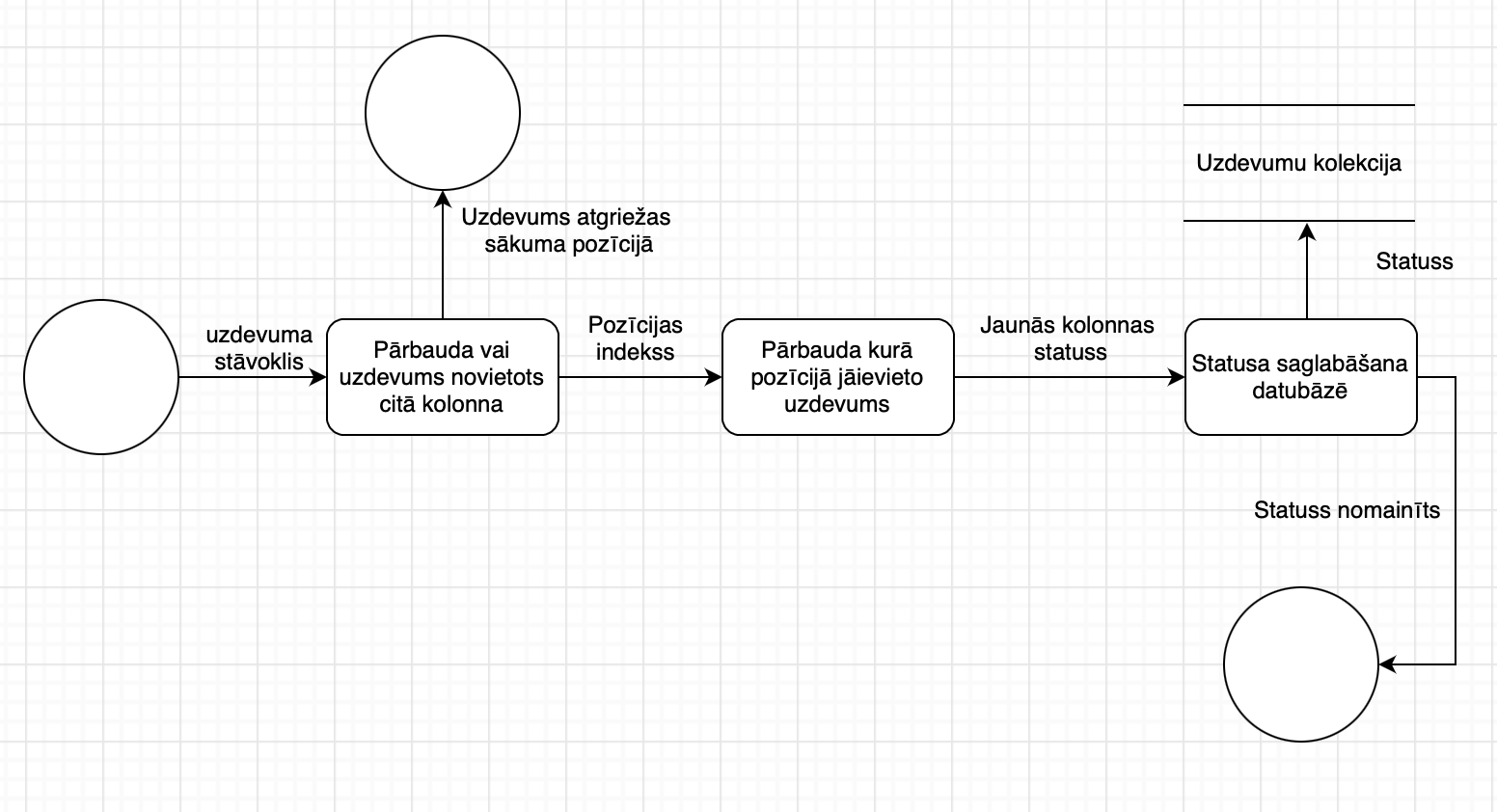


5.6.att. Projekta izveidošanas datu plūsma

1. **Uzdevuma statusa maiņa**

Uzdevuma statusa maiņa ir vienkārša, bet svarīga priekš “drag and drop” funkcionalitātes. Kad mēs pārvelkam uzdevumu uz citu kolonnu mums vajag ne tikai nomainīt lauka kolonnu kurā tas atrodas, bet arī nomainīt statusu gan, lai tas tiktu attēlots, gan lai ielādējot lapu pa jaunam mūsu pārvietotais uzdevums paliek tajā kolonna uz kuru tika pārvietots. Uzdevuma statusa maiņa notiek sekojoši. (skat. 5.7. att.)

Kad mēs uzdevumu pārvelkam uz citu kolonnu, mēs pārbaudam vai tā nav tā paša kolonna, kā arī pārbaudam pozīcijas indeksu, lai zinātu kurā vietā mums jāievieto uzdevums. Tad mēs nomainam statusu un atjaunojam uzdevuma statusu datubāzē.

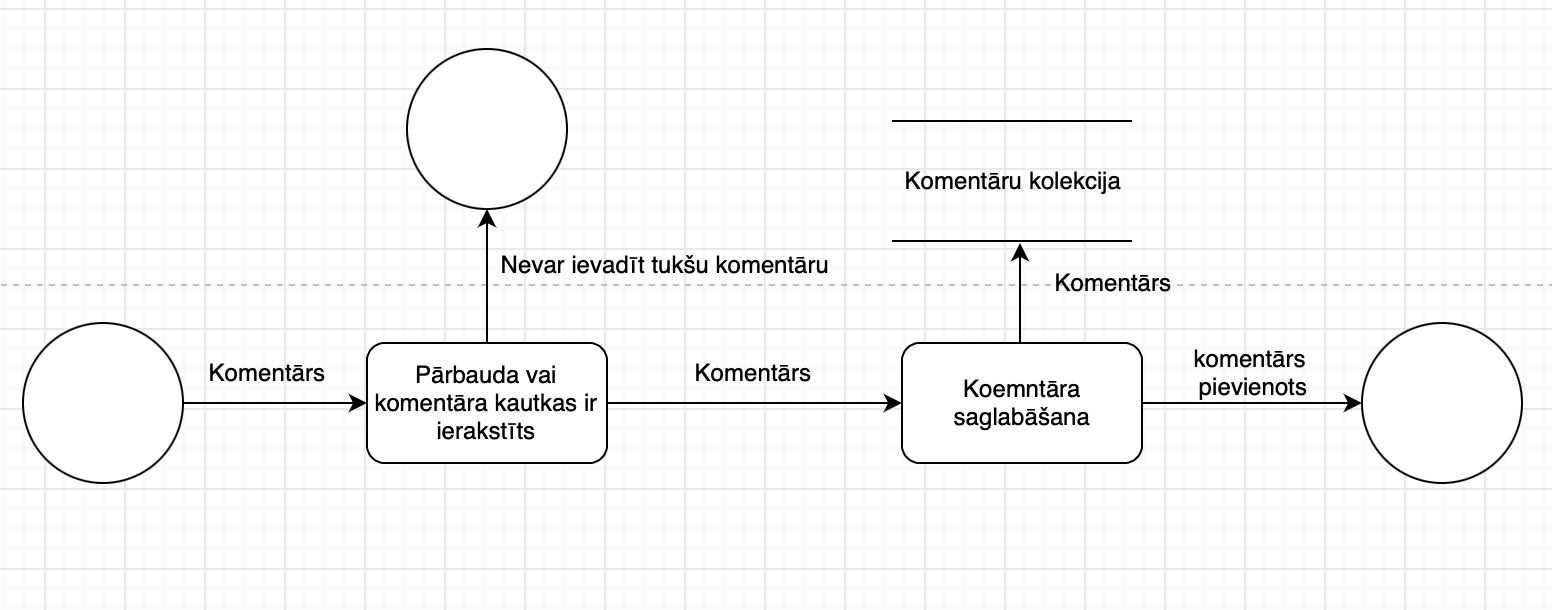


5.7.att. Uzdevuma statusa maiņas datu plūsma

1. **Komentāra pievienošana**

Komentāri glabājas uzdevumos un sastāv tikai no teksta kas tiek ievadīts un datuma kad veikts komentārs, lai varam sakārtot secīgi komentārus, ja nu gadījuma komentāru secība datubāzē izmainas, kam nevajadzētu notikt. Komentāru izveide notiek sekojoši. (skat. 5.8. att.)

Mēs ievadam komentāra tekstu un pārbaudam vai kautkas ir ievadīts šajā laukā, ja laukā ir ievadīts kaut kāds teksts tad tas ir deīgs komentārs. Tālāk mēs saglabājam komentāru datubāzē.

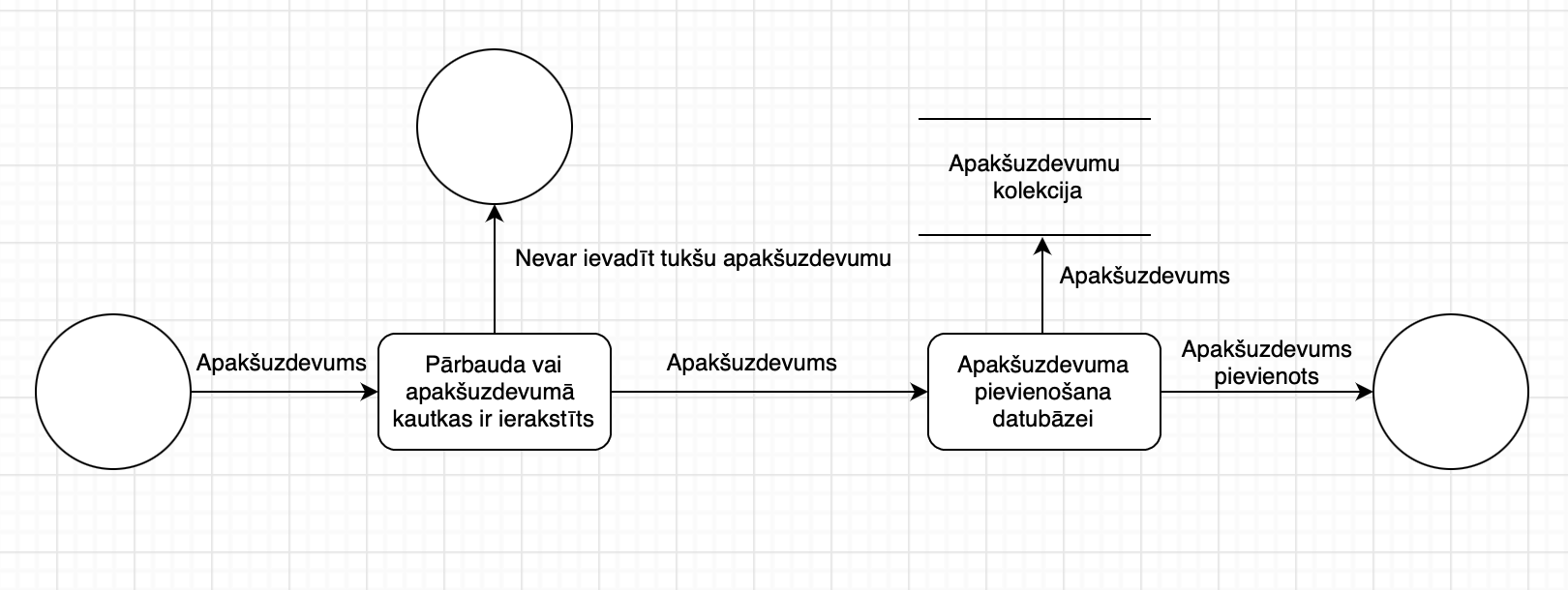


5.8.att. Komentāra pievienošanas datu plūsma

1. **Apakšuzdevuma pievienošana**

Apakšuzdevumu datu plūsma ir gandrīz tādi pata kā komentāriem, tikai apakšuzdevumiem nav nepieciešamības pēc datuma, jo vienalga kā tie ir sakārtoti. Apakšuzdevumu pievienošana notiek sekojoši. (skat. 5.9. att.)

Mēs ievadam apakšuzdevuma tekstu un pārbaudam vai kautkas ir ievadīts šajā laukā, ja laukā ir ievadīts kaut kāds teksts tad tas ir deīgs apakšuzdevums. Tālāk mēs saglabājam apakšuzdevumu datubāzē.



5.9.att. Apakšuzdevuma pievienošanas datu plūsma

# 

# Datu struktūras apraksts

Datu bāzes projektēšanas rezultātā tika veidotas vairākas tabulas un starp tām tika definēta tabulu relācija jeb saistība, kas norāda, ka abās tabulās saistītajos laukos ir vienādi dati. Saites var būt trejādas: viens-pret-vienu, viens-pret-daudziem, daudzi-pret-daudziem. Datu bāzē visas tabulas ir saistītas ar saiti viens-pret-daudziem. Datu bāze veidota ar nosql datubāzi MongoDB. Visi saglabātie dati būs “String” tips.

Datu bāze sastāv no 5 tabulām, kuras satur informāciju par lietotājiem, projektiem, uzdevumiem, apakšuzdevumiem un komentāriem.

Tabula „users” satur reģistrētu lietotāju datus.

3.1. tabula

Tabulas „**users**” struktūra

| **NR** | **Nosaukums** | **Tips** | **Izmērs** | **Apraksts** |
| --- | --- | --- | --- | --- |
| 1 | \_id | String | 25 | Objekta id |
| 2 | username | String | 3-50 | lietotājvārds lai iejietu sistēmā |
| 3 | email | String | 50 | e-pasts |
| 4 | password | String | 50 | parole (tā tiks glabāta saujaukta ar md5 hash algoritmu) |
| 5 | name | String | 50 | lietoāja vārds |
| 6 | surname | String | 50 | lietotāja uzvārds |
| 7 | phone | String | 12 | lietotāja telefona numurs |
| 8 | address | String | 50 | lietotāja adrese |
| 9 | admin | Boolean | true/false | vai lietotājs ir admins vai nē |

3.2. tabula

Tabulas „**tasks**” struktūra

| **NR** | **Nosaukums** | **Tips** | **Izmērs** | **Apraksts** |
| --- | --- | --- | --- | --- |
| 1 | \_id | String | 25 | Objekta id |
| 2 | title | String | 50 | Uzdevuma nosaukums |
| 3 | content | String | 0 - neierobežots | Uzdevuma apraksts |
| 4 | status | String | 4-20 | Uzdevuma statuss |
| 5 | priority | String | 4-20 | Uzdevuma prioritāte |
| 6 | project | String | 50 | Uzdevuma projkets, pie kura piesaistīts uzdevums |
| 7 | assignee | String | 50 | Lietoājs kuram jāpilda uzdevums |

3.3. tabula

Tabulas „**projects**” struktūra

| **NR** | **Nosaukums** | **Tips** | **Izmērs** | **Apraksts** |
| --- | --- | --- | --- | --- |
| 1 | \_id | String | 25 | Objekta id |
| 2 | name | String | 50 | Projekta nosaukums |
| 3 | price | String | 50 | Projekta cena |
| 4 | link | String | 100 | Projekta saite |
| 5 | startDate | String | 20 | Projekta sākuma datums |
| 6 | endDate | String | 20 | Projekta beigu datums |
| 7 | access | String | 0-neierobežots | Projekta atrā piekļuve |

Datu bāze sastāv no kolekcijām, un katrai kolekcija sastāv no dokumentiem.

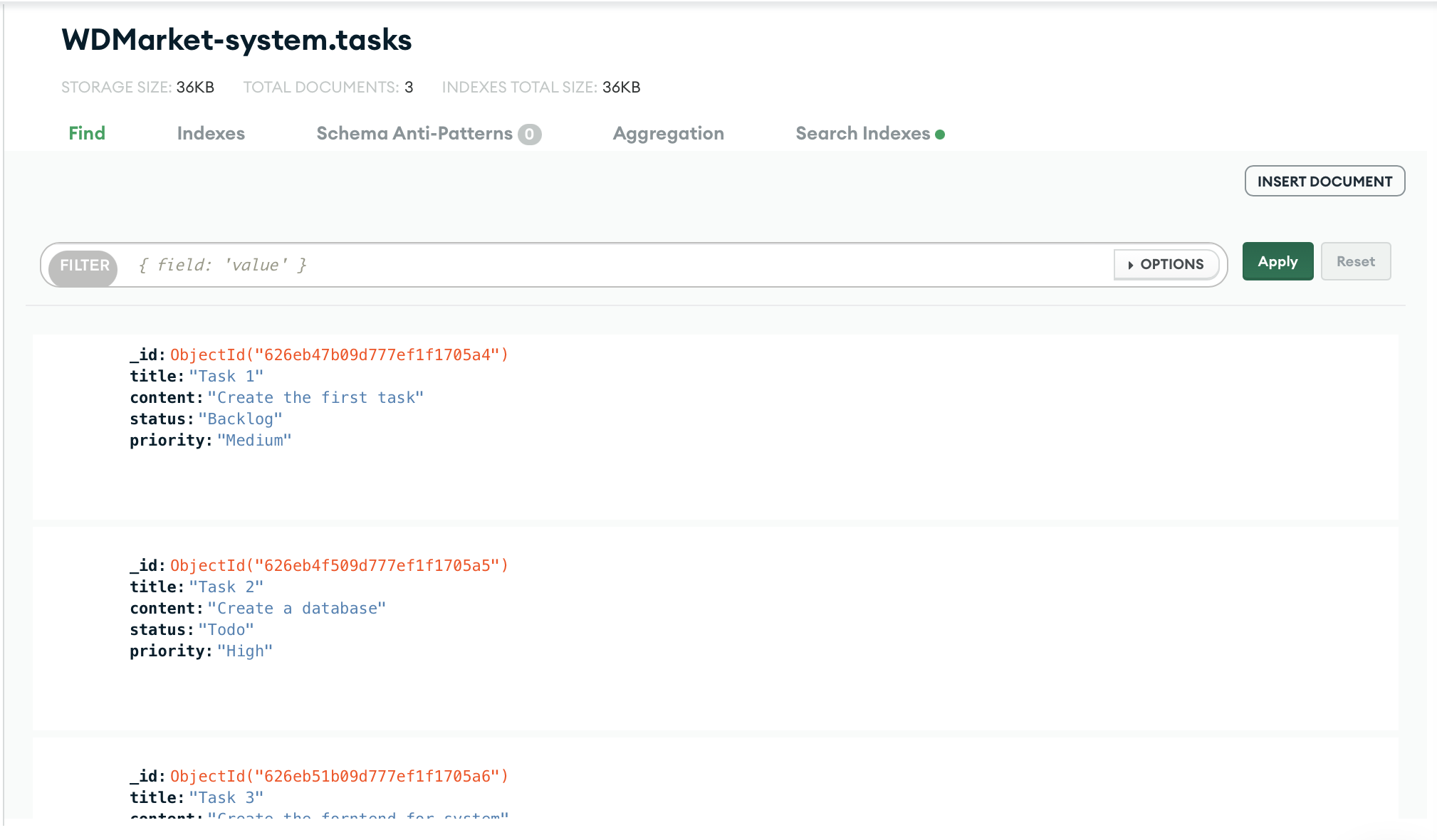
**Sistēmas kolekcijas**

5.pielikums



**Sistēmas kolekcija “users”**

6.pielikums



# Lietotāja ceļvedis

## Sistēmas prasības aparatūrai un programmatūrai

Lai sāktu veiksmīgu darbu pie sistēmas, ir nepieciešamas divas lietas:

* lietotāja programmatūra;
* serveris.

Lietotāja programmatūra ir programmatūra, kas ļauj lietotājam aplūkot uz tīmekļa serveriem esošus dokumentus. Šajā gadījumā sistēma ir uz tīmekļa servera atrodošais dokumentu kopums. Populārākās pārlūkprogrammas ir Google Chrome, Mozilla Firefox, Safari un Microsoft Internet Explorer. Par cik programmu paredzēts lietot kompānijām kuras dara daudz uzdevumus un ir nepieciešama kārtība, tad lielāko ties nevajadzētu būt nekādām problēmām to palaist un nevajadzētu būt nekādām vizuālām problēmām. Zemāk ir aplūkojama uzskatāma informācija par lapas atbalstu uz dažādām pārlūkprogrammām (skat. 7.1. tabulu).

7.1. Tabula

Lapas atbalstītās pārlūkprogrammas

| **Pārlūkprogrammas** | **Versija** |
| --- | --- |
| Firefox | 2.0 + |
| Chrome | 1.0 + |
| Safari | 4.0 + |
| Opera | 9.0 + |
| Edge | 12.0 + |
| Internet Explorer | 9.0 + |
| Android Browser | 3.0 + |

Lai veiksmīgi un stabili pārlūkotu lapu, katrai pārlūkprogrammai ir savas noteiktās minimālās prasības attiecībā uz programmatūru un lietotāja datora specifikācijām. Zemāk ir aplūkojuma tabula, kas detalizēti parāda, kādas minimālās prasības tiek izvirzītas optimālai sistēmas darbībai un tās lietošanai no lietotāja puses. (skat. 7.2. tabulu).

7.2. Tabula

Minimālās prasības lietotāja programmatūrai un viņa datora specifikācijām

| **Specifikācija** | **Internet explorer 9** | **Firefox** | **Chrome** | **Opera** | **Safari** |
| --- | --- | --- | --- | --- | --- |
| Windows processors | 1-GHz | Pentium 4 | Pentium 4 | Pentium II | 500-MHz Pentium-class |
| Mac processors | intel | intel / m1 | intel / m1 | intel / m1 | intel / m1 |
| Min RAM | - | - | 128MB | 128MB | 256MB |
| Ieteicamais RAM | 512MB | 512MB | - | 256MB | - |
| Minimālais diska daudzums | - | - | 100MB | 20MB | - |
| Ieteicamais diska daudzums | 120MB | 200MB | - | 100MB | - |
| Windows | Windows Vista | Windows XP SP2 (FF 13) | Windows XP SP2 | Windows 2000 | Windows XP SP2 |
| OS X | - | OS X 10.5 | OS X 10.5.6 | OS X 10.5 | OS X 10.5.8 |
| Linux | - | Jebkurš jaunākais | Ubuntu 10.04,  Debian 6,  OpenSuse 11.3,  Fedora Linux 14 | Jebkurš jaunākais | - |

Neskaitot 6.1.2. tabulā aplūkotās prasības, lietotājam ir nepieciešama tastatūra un pele vai skārienjūtīgais paliktnis, lai spētu ievadīt un apstrādāt informāciju sistēmā.

Lai sistēma strādātu, lai to varētu varētu palaist un uzturēt, ir nepieciešams serveris ar uzstādītu programmatūru. Mums ir divas iespējas palaist programmas kodu uz hostēta servera vai pašiem uzlikt uz sava servera. Servera ieteicamās prasības (skat. 7.2. tabulu).

7.3. Tabula

Ieteicamās prasības uzsturošā servera resursiem

| Procesors | 3Ghz procesors |
| --- | --- |
| Operatīvā atmiņa | 16GB RAM |
| Ārējā atmiņa | 500GB |

## Sistēmas instalācija un palaišana

Sistēmai ir divu veidu palaišana. Pirmā no tām ir lietotāja puses palaišana. Lietotāja puses palaišana nav nekas vairāk par vienkāršu ieiešanu lapā ar pārlūkprogrammas starpniecību. Viss, kas lietotājam ir jāizdara – jāievada pārlūkprogrammā lapas saite un ar to arī sistēmas palaišana beidzas – lietotājs var brīvi aplūkot lapu.

Otrais palaišanas veids ir administratora pusē. Šis palaišanas veids aktivizē pašu sistēmas darbību un sagatavo lapu publikai, līdz ar to atļaujot lietotājam to izmantot un brīvi aplūkot. Sistēmas sagatavošana publikai, tās konfigurēšana un uzstādīšana noris sekojoši.

* Sistēmas serveris jeb “backend” jāuzliek uz hostojošā servera.
* “Frontend” pusē jānomaina Axios base\_url uz backend linku.

## Programmas apraksts

Sistēmā būs divas lomas “user”, jeb sistēmas lietotājs un “admin”, jeb projekta vadītājs vai atbildīgas. Sistēmai nepiederošas personas nevarēs tikt klāt sistēmai bez piekļuvēm. Sistēmas lietotājs varēs pārlikt uzdevumus un izveidot uzdevumus, likt komentārus uzdevumiem, izveidot apakšuzdevumus, pievienot projektiem “access” lauku, mainīt profila informāciju, savukārt admins varēs izdarīt visas lietotāja atļautās komandas kā arī papildus izveidot projektus un pievienot jaunus lietotājus. Ieejot sistēma, galvenais logs “Dashboard” rādīs noteiktajam lietotājam relatīvo informāciju par uzdevumiem un projektiem. Logā “Tasks” būs uzdevumu saraksts, un lietotājs varēs pārvilkt uzdevumus uz citiem statusiem atspolguļojot uzdevuma statusu. Logā “Projects” būs redzami visi projekti un to statusi. Logā “Users” būs redzami visi lietotāji un to informācija, kā arī varēs pievienot jaunu lietotāju. Logā “Profile” varēs redzēt savu profila informāciju un iziet no sistēmas. Visi logi ir ietverti konteinerī ar galveni un navigāciju kreisajā pusē, sistēmas galvenē uzrāda kādā logā atrodamies un varam paslēpt vai parādīt navigāciju.

Lietotāja “user” ceļvedis:

* Pievienot uzdevumu:
  + Navigēt uz logu “Tasks”
  + Nopsiest pogu “Add new”
  + Ievadīt visu nepieciešamo informāciju
  + nospiest pogu “Create task”
* Nomainīt “Access” projektam:
  + Navigēt uz logu “Tasks”
  + Nospiest uz uzdevuma ar attiecīgo projektu
  + Nospiest uz pogas “Access”
  + “Access” logā nospiest pogu “Edit”
  + Nomainīt informāciju
  + Nospiest pogu “Save”
* Pievieno komentāru uzdevumam:
  + Navigēt uz logu “Tasks”
  + Atvērt attiecīgo uzdevumu
  + “Comments” laukā ievadīt komentāru
  + Nospiest pogu “Comment”
* Noaminīt profila informāciju:
  + Navigēt uz logu “Profile”
  + Nospiest pogu “Edit”
  + Noaminīt informāciju dotajos laukos
  + Nospiest pogu “Save”
* Nomainīt uzdevuma statusu:
  + Navigēt uz logu “Tasks”
  + Pārvilkt noteikto uzdevumu uz kolonnu ar attiecīgo statusu
* Iziet arā no profila:
  + Navigēt uz logu “Profile”
  + Nospiest pogu “Logout”

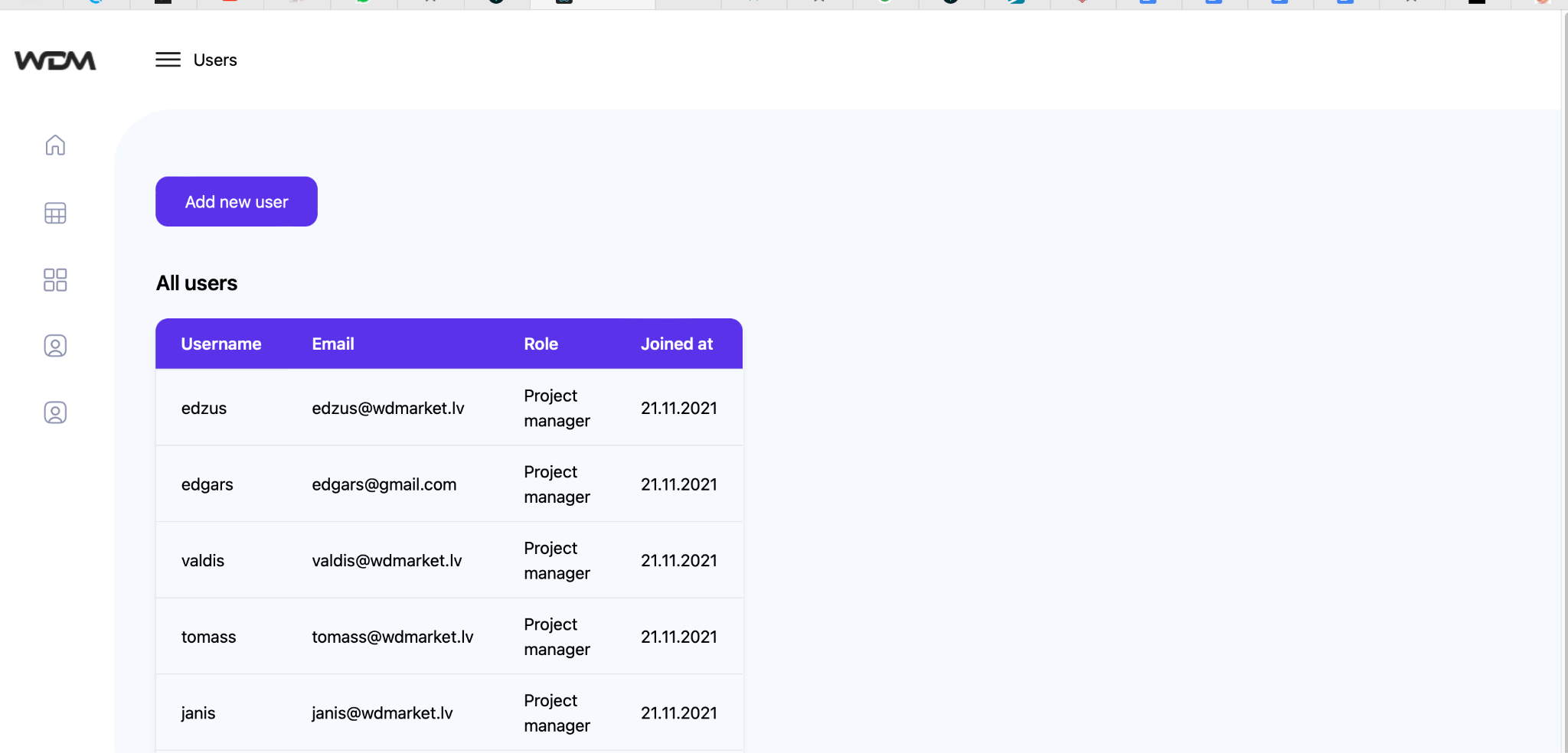
Lietotāja “admin” ceļvedis:

* Izveidot jaunu projektu:
  + Navigēt uz logu “Projects”
  + Nospiest pogu “Add new”
  + Ievadīt nepieciešamo informāciju
  + Nospiest pogu “Create”
* Izveidot jaunu lietotāju sistēmai:
  + Navigēt uz logu “Users”
  + Nospiest pogu “Add new”
  + Ievadīt nepieciešamo informāciju
  + Nospiest pogu “Create user”

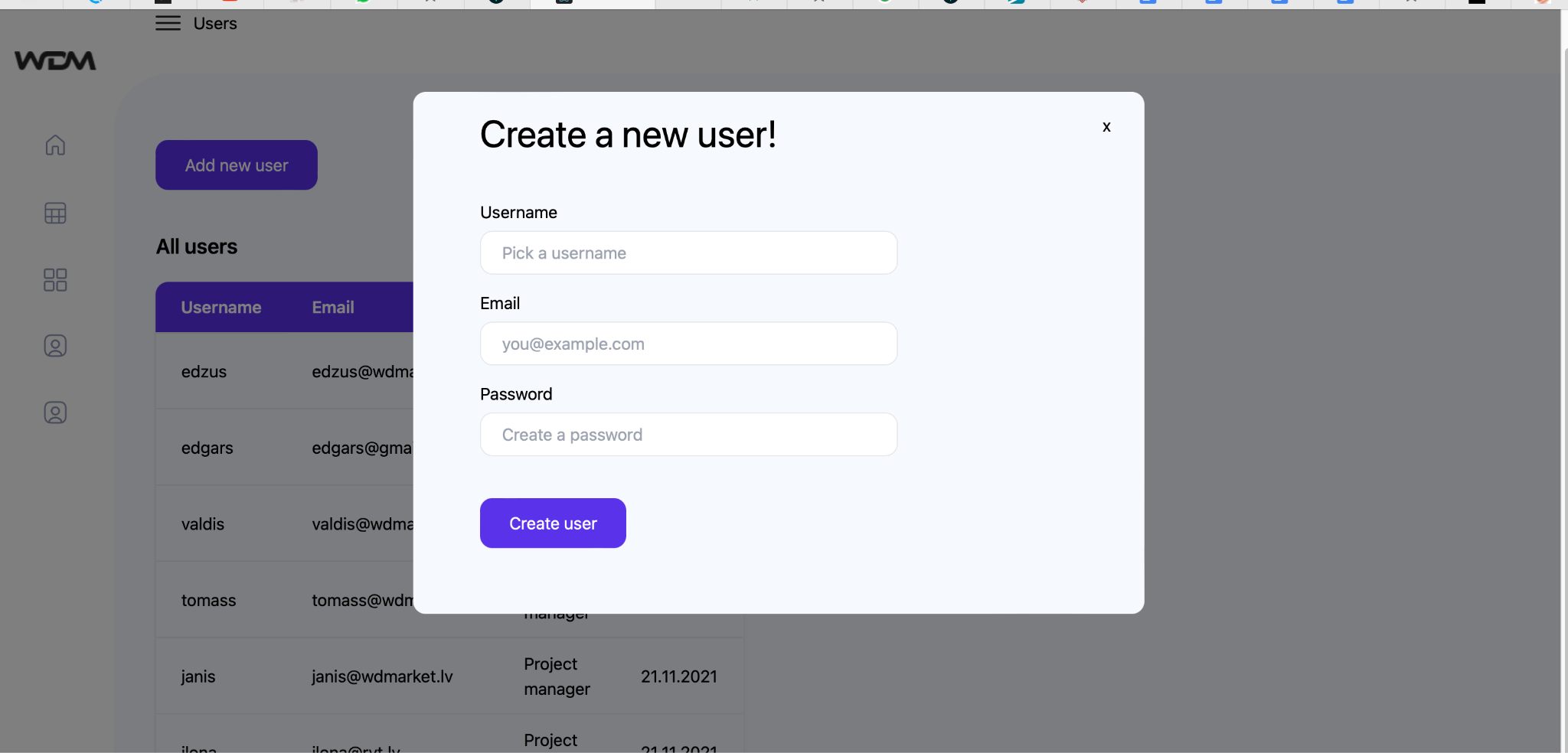
## Testa piemērs

Testa piemēra nolūks ir parādīt sistēmas jauna lietotāja pievienošanu.

Pirmais solis uz vēlamā rezultāta sasniegšanu ir autorizēties sistēmā (lai pievienotu jaunu lietotāju – jāpieslēdzas kā administratoram, jeb projekta vadītājam). Kad tas izdarīts, jādodas uz logu “Users” – saite atrodama galvenajā izvēlnē, kas atrodas kreisajā pusē – un jāspiež uz pogas „Add new user” kura atrodas loga kreisajā augšējā stūrī (skat. 7.1. att.). Pēc pogas nospiešanas atveras uznirstoša loga izveides forma, kur ievadam testa datus. Spiežam uz pogas „Create user”, lai izveidotu jaunu lietotāju (skat. 7.2.. att.).

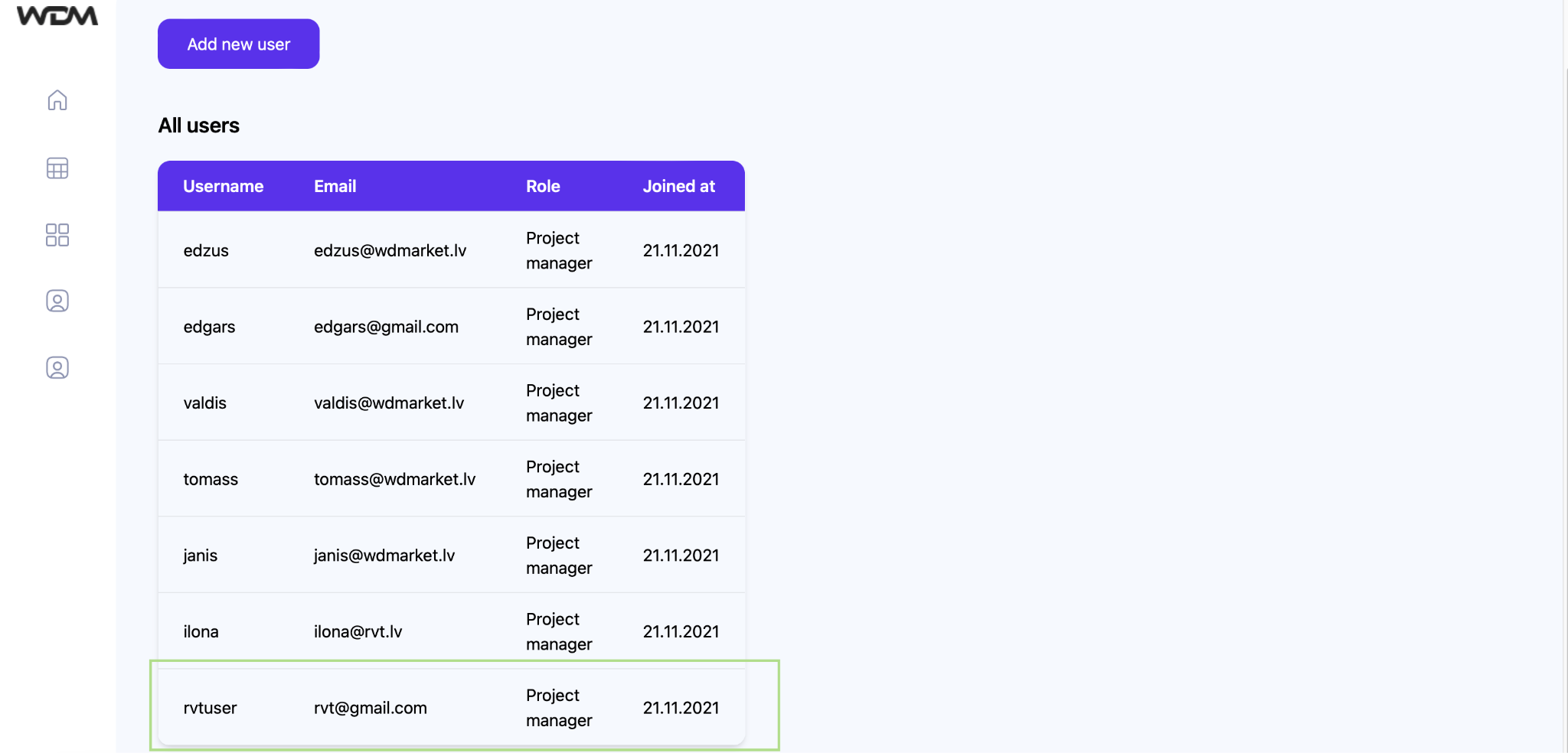


7.1. att. loga “Users” skats



7.2. att. Jauna lietotāja izveidošanas uznirstosā loga forma

Ja lietotājs veiksmiīgi izveidot parādās paziņojums un varam apskatīties izveidoto lietotāju saraksta apakšā (skatīt 7.3. att.).



7.3. att. Lietotāju saraksts ar jauno lietotāju

## 

# Nobeigums

Kopumā sistēma dara to kam tā ir paredzēta, viegla tasku menedžēšana, viegla kontrole pār lietotājiem, pietiekami droša, protams, ir vietas kur varētu uzlabot drošību. Sistēmai ir minimālistisks stils, lai varētu uzņemt viegli informāciju, un viss ir uzskatāms un saredzams, liels mīnuss ir tas ka nav paredzēta mobīlajām ierīcēm.

Sistēmai ir savi mīnusi, kā jau minēts par pielāgošanos mobīlajām ierīcēm, bet uzskatu ka tās ir vienkārši iespējas kā padarīt sistēmu labāku nākotnē, pievienot komplicētāku filtrāciju visu caur sistēmai, pievienot čatu, pievienot privātās ziņas, laikā uzņemšana uzdevumiem, iztērētais laiks uz projektiem spriežot pēc uzdevumu izpildes laikiem, kalendārs, uzdevumu likšana nākotnei kalendārā, kā arī pievienot klientu attiecību menedžēšanas sadaļu. Izvēlētās tehnoloģijas manuprāt, ir piemērotas React ir viens no populārākajiem ietvariem / biebliotēkām priekš Javascript tāpēc būvēt sistēmu bija nedaudz vieglāk, jo ir pieejami vairāk resursi un vairāki veidi kā izpildīt nepieciešamo funkcionalitāti. Datubāzes NOSQL izvēle šajā gadījumā ir labāka, jo daudzos gadījumos lauki var tikt neievadīti un sistēmas izaugsmei, manuprāt, MongoDB būs labāka tehnoloģija nekā SQL.

# 

# Izmantotie avoti

React dokumentācija - <https://reactjs.org/docs/getting-started.html> - (resurs skatīts sākot no 24.11.2021)

React router dokumentācija - <https://reactrouter.com/docs/en/v6/getting-started/overview> - (resurs skatīts sākot no 20.04.2021)

React beautiful dnd izmantošanas pamācība - <https://egghead.io/courses/beautiful-and-accessible-drag-and-drop-with-react-beautiful-dnd> - (resurs skatīts sākot no 25.04.2022)

TailwindCSS uzstādīšana - <https://www.youtube.com/watch?v=Pe1Vo2N3Z2c&t=733s> (resurs skatīts 27.04.2022)

TailwindCSS klašu špikeris - <https://nerdcave.com/tailwind-cheat-sheet> (resurs skatīts sākot no 27.04.2022)

Brad Schiff (React kurss) - <https://www.udemy.com/course/react-for-the-rest-of-us/> - (resurs skatīts sākot no 24.11.2021)

Brad Schiff (Javascript kurss) - <https://www.udemy.com/course/web-design-for-beginners-real-world-coding-in-html-css/> - (resurs skatīts sākot no 1.05.2022)

React immer dokumentācija - <https://immerjs.github.io/immer/> (resurs skatīts sākot no 21.04.2022)

ExpressJs dokumnetācija - <https://expressjs.com/en/guide/routing.html> (resurs skatīts sākot no 01.05.2022)

Mongodb nodejs dokumentācija - <https://www.mongodb.com/docs/drivers/node/current/> - (resurs skatīts sākot no 01.05.2022)