**Profesionālās izglītības kompetences centrs**

**“Rīgas Valsts tehnikums”**

Izglītības programma: Programmēšana

**KVALIFIKĀCIJAS DARBS**

**Interneta veikala sistēma**

Paskaidrojošais raksts: ? lpp

Audzēknis: Deniss Štrombergs

Vadītājs: Vadītājs

Normu kontrole: Normu kontrole

**Rīga**

**2021**

SATURS

[1 ANOTĀCIJA 4](#__RefHeading___Toc2225_2296645101)

[2 IEVADS 5](#__RefHeading___Toc4129_3519069617)

[3 UZDEVUMA NOSTĀDNE 5](#__RefHeading___Toc4617_2378508263)

[4 PRASĪBU SPECIFIKĀCIJA 7](#__RefHeading___Toc2227_2296645101)

[4.1 Ieejas un izejas informācijas apraksts 7](#__RefHeading___Toc2229_2296645101)

[4.1.1 *Ieejas informācijas apraksts* 7](#__RefHeading___Toc2371_1133217771)

[4.1.2 *Izejas informācijas apraksts* 7](#__RefHeading___Toc2373_1133217771)

[4.2 Funkcionālās 7](#__RefHeading___Toc2233_2296645101)

[2.4 Nefunkcionālās 8](#__RefHeading___Toc2235_2296645101)

[3 UZDEVUMA RISINĀŠANAS LĪDZEKĻU IZVĒLES PAMATOJUMS 8](#__RefHeading___Toc2237_2296645101)

[4 PROGRAMMATŪRAS PRODUKTA MODELĒŠANA UN PROJEKTĒŠANA 9](#__RefHeading___Toc2239_2296645101)

[4.1 Sistēmas arhitektūra 9](#__RefHeading___Toc2241_2296645101)

[4.1.1 Administratora daļa. 9](#__RefHeading___Toc2375_1133217771)

[4.1.2 Klienta daļa. 9](#__RefHeading___Toc2377_1133217771)

[4.2 Sistēmas ER modelis 10](#__RefHeading___Toc2243_2296645101)

[4.3 Funkcionālais sistēmas modelis 12](#__RefHeading___Toc2245_2296645101)

[5 DATU STRUKTŪRAS APRAKSTS 13](#__RefHeading___Toc4619_2378508263)

[6 LIETOTĀJA CEĻVEDI 20](#__RefHeading___Toc2247_2296645101)

[6.1 Sistēmas prasības aparatūrai un programmatūrai 20](#__RefHeading___Toc2249_2296645101)

[6.1.1 Windows 20](#__RefHeading___Toc2443_1133217771)

[6.1.2 Linux (Ubuntu) 21](#__RefHeading___Toc2445_1133217771)

[6.2 Sistēmas instalācija un palaišana 21](#__RefHeading___Toc2251_2296645101)

[6.3 Programmas apraksts 22](#__RefHeading___Toc2253_2296645101)

[6.4 Testa piemērs 22](#__RefHeading___Toc2255_2296645101)

[NOBEIGUMS 23](#__RefHeading___Toc4663_2378508263)

[INFORMĀCIJAS AVOTI 24](#__RefHeading___Toc2257_2296645101)

[PIELIKUMI 25](#__RefHeading___Toc4667_2378508263)

[ER diagramma 25](#__RefHeading___Toc2180_4067633674)

[Fiziska struktūra 26](#__RefHeading___Toc4671_2378508263)

[Sistēmas arhitektūra 27](#__RefHeading___Toc4671_23785082631)

# ANOTĀCIJA

Kvalifikācijas darbā ir aprakstīta interneta veikala sistēmas izstrādāšanas process. Sistēma dod iespēju tā lietotājam, apkopot un apstrādāt datus par sava veikala produktiem un klientiem, konkrēti, var veidot produktus, kuri atspoguļo reālus produktus, norādīt produkta atribūtus (cena, krāsa u.c.), kā arī kurus produktus iegādājās pircēji, un kādas atsauksmes tie atstāj. Sistēma tika izstrādāta ar JavaScript programmēšanas valodu, izmantojot React ietvaru, lai realizētu Frontend, un Node ar PostgreSQL un Express, lai realizētu Backend.

Kvalifikācijas darbs ietver ievadu, uzdevumu nostādni, prasību specifikāciju, uzdevuma risināšanas līdzekļu izvēles pamatojumu, programmatūras produkta modelēšanas un projektēšanas aprakstu, datu struktūru aprakstu, lietotāju ceļvedi, nobeigumu un pielikumus.

Kvalifikācijas darba ievadā ir aprakstīta vairumtirdzniecības aktuālā problēma un tās risinājumus. Uzdevumu nostādnē ir norādīti uzdevumi, kurus sistēmai būs nepieciešams veikt. Prasību specifikācija sastāv no ieejas un izejas informācijas, kā arī no sistēmas funkcionālajām un nefunkcionālajām prasībām. Uzdevuma 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 struktūru 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 paskaidro, kā pareizi lietot sistēmu. Testa piemērā ir dots detalizēt rezervācijas pievienošanas apraksts ar vizuāliem

Kvalifikācijas darbs sastāv no ? lappusēm, kurā ietilpst ? attēli, ? tabulas un pielikumi. Pielikumi satur sadaļas: ER diagrammu, datu plūsmu diagrammu un datu bāzes relāciju shēmu.

# IEVADS

Kvalifikācijas darba mērķis ir izveidot interneta veikala sistēmu, līdzīgi jau eksistējošiem variantiem, kā Magento 2, Shopify, BigCommerce un citiem, lai varētu nodrošināt varētu apkopot sava veikala datus vienā sistēma un, lai paplašinātu pircēju loku un pieejamību balstoties uz to, kā veikals ir arī pieejams internetā, kas joprojām ir ļoti aktuāli. Protams nav jēgās būvēt sistēmu, kura spēs kļūt par konkurent ar iepriekš minētām, jau eksistējošām sistēmām. Bet tas ar ko šī interneta veikala sistēma atšķirsies ir ar to, ka tā vienkāršāka. Šīs sistēmas nodrošina visas nepieciešamas bāzes funkcionalitātes, kuras varētu būt nepieciešamas, taču dažādas integrācijas ir neobligātas un nav pēc nosacījuma implementētas, piemēram, Payment Gateway, vai Shipping API, nebūs pieejami šajā programmā, vismaz pagaidām, tām vajag būt implementētam, bāzējoties uz konkrēta gadījuma nepieciešamībām. Īsumā, šī sistēma nodrošina pamata funkcionalitāti, taču specifiskas lietas, jau jābūt implementētam balstoties uz konkrētu gadījumu.

# UZDEVUMA NOSTĀDNE

Kvalifikācijas darba uzdevums ir izveidot interneta veikala sistēmu. Sistēma sastāvēs no 2 galvenām daļām administrator un klients. Administrators, ir kā interneta veikala īpašnieks, tam ir pieeja visiem nepieciešamiem datiem un tas var to manipulēt. Klient, ir kā pircējs, tam ir pieeja veikala priekšpusei, tas var likt pirkumus grozā, un veikt pasūtījumus.

Interneta veikala sistēmas sistēmai ir jānodrošina sekojošās funkcionalitātes:

Administrators:

* Dažādu datu (produkti, kategorijas, lapas, atribūti, u.c.), apkopošana un manipulācija
* Kopējās veikala konfigurācijas uzstādīšana
* ...

Klients:

* Produktu meklēšana
* Produkta lapas apskatīšana un iespēja pievienot grozā
* Atsauksmju sistēma
* Pircēja akaunta veidošana un rediģēšana
* ...

# **PRASĪBU SPECIFIKĀCIJA**

## Ieejas un izejas informācijas apraksts

### ***Ieejas informācijas apraksts***

Sistēmā tiks nodrošināta šāda ieejas informācijas apstrāde.

1. Informācijas par lietotājiem sastāvēs no sekojošiem datiem.

* E-pasts -
* Parole -
* Vārds -
* Uzvārds -
* Telefona numurs -
* Valsts -
* Pilsēta -
* Novads -
* Pasta kods -

### ***Izejas informācijas apraksts***

Meklēšanas rezultātu atspoguļojums

Rēķinu izveide

## Funkcionālās

1. Jānodrošina lietotāja autorizācija.
   1. Ja ... tad ...
   2. adsdsa
2. Jānodrošina pircēju informācijas apstrāde.
   1. Adsads
   2. adsads
   3. Jānodrošina sdfiog

## Nefunkcionālās

Dizaina apraksts

# **UZDEVUMA RISINĀŠANAS LĪDZEKĻU IZVĒLES PAMATOJUMS**

1. Frontend:
   1. JavaScript – ir praktiski vienīga programmēšanas valoda, kas tiek izmantota šajā projektā, un visas darbības ar HTML, SCSS, PostgreSQL tiek veiktas arī ar JavaScript. Izvēles pamatojums ir tāds, kā šī programmēšanas valoda ļoti labi der mājaslapu veidošanai, tagad, ne tikai, lai nodrošinātu lietotāja interfeisu, bet arī, lai nodrošinātu datu apstrādi un datu bāzes manipulācijas. JavasScript ir ļoti populāra un attīstīta programmēšanas valoda, ar lielu daudzumu palīg resursu, tāpēc, izvēle ir ļoti izdevīga, jo atvieglo palīdz paātrināt izstrādes procesu.
   2. React – interfeisa komponentu sistēma, lai labu interfeisu, ir nepieciešama laba komponentu sistēma, protams, varēja būvēt savu komponentu sistēmu, bet tas aizņemtu pārāk daudz laika, un resursu, pietam, vēl, nevar paredzē, kādas grūtības rastos, kas tieši būtu nepieciešams, lai sistēma funkcionētu atbilstoši vajadzībām, tāpēc, ir labāk izmantot jau gatavu, pārbaudītu, sistēmu.
   3. Material UI – bāzes stils, būtu labi, lai jau uzreiz būtu kaut kāds mājaslapas stils, kuru pēc tām pēc konkrēta lietotāja nepieciešamībām varētu modificēt un pielāgot tā specifiskām vajadzībām.
   4. React Admin – administratorā daļas abstrakcija, nodrošina resursu sasaisti starp Backend un Frontend, konrkēti adminstrātora daļai, izvēle, ir pamatota, ar to, ka ir nepieciešama, kaut kāda bāze, uz kuras būvēt adminstrātora daļu, jo lielāka daļa no operācijām atkārtojās, tāpēc arī šīs komponents ir ļoti lietderīgs.
2. Backend:
   1. Node – JavaScript bāzēta Backend sistēma, ar to palīdzību tiek nodrošināta visas servera operācijas: GraphQL API un PostgreSQL. Alternatīas izvēles varētu būt PHP, bet šajā gadījumā, lai ērtāk saskaņotu Backend ar frontend, tieši Node der ļoti labi, neskatoties uz tā trūkumiem.
   2. Express -
   3. PostgreSQL – datu bāzes vadības sistēma, lai nodrošinātu datu uzglabāšanu un manipulācijas. Tieši izvēlējos PostgreSQL un ne MySQL, vai citu SQL tipa datu bāzes vadības sistēmu, tāpēc, ka PostgreSQL satur specifisku funkcionalitāti, kura noderētu projekta realizācijas, konrkēti, JSONB datu tips, ar kura palīdzību var realizēt efektīvu produktu atribūtu sistēmu, protams arī izmantojot MySQL, varētu to iegūt ar EAV paņiemienu.
   4. GraphQL – pieprasījumu valoda, kura nodrošina efektīvu datu pieprasījumu rezolūciju. Klients pieprasa konkrētus datus, un severis tieši tos tadus, kurus klients ir pierasījis, nevairāk, nemazāk. Atšķībā no alternatīvas – Rest API, šī sistēma ir krietni elastīgāka un datu pieprasījums tiek veikts no viena endpoint.

# **PROGRAMMATŪRAS PRODUKTA MODELĒŠANA UN PROJEKTĒŠANA**

## Sistēmas arhitektūra

Manā sistēmā ir divi lietotāju tipi: administrators un pircējs, tāpēc kopumā interneta veikala sistēma sastāvēs no 2 daļām: administratora daļa un klienta daļa. (skatīt 3. pielikumā).

### Administratora daļa.

Administrators var apskatīt un rediģēt interneta veikala datus. Tam ir pieeja pie produktiem, kategorijām, atribūtiem, atribūtu kopām, CMS lapām, CMS blokiem, ierobežotiem pircēja datiem (kartes dati un paroles tiek šifrētas), pasūtījumiem, rēķiniem, produktu atsauksmēm un citiem datiem.

### Klienta daļa.

Klienta nevar redzēt un manipulēt interneta veikala datus, tas redz tikai tā priekšpusi, kurā tas spēj atrast sev nepieciešamos produktus un pievienot tos savā grozā. Lietotājs var būt kā viesis, vai arī kā reģistrēts lietotājs, abos gadījumos ir iespējams veikt pirkumu. Abu tipu lietotāji var meklēt produktu ievadot kaut kādus datus par to, piemēram, produkta nosaukumu, vai arī izvēlēties noteiktus filtrus, un balstoties uz tiem, izlasīt noteiktus produktus. Var pievienot produktu pie groza, rediģēt tā daudzumu, vai arī noņemt to. Lietotājs var piereģistrēties. Reģistrējoties, lietotājs var saglabāt savas adreses, kartes datus, kā arī pievienot, rediģēt un dzēst produktu atsauksmes.

## Sistēmas ER modelis

ER diagramma (skatīt 1. pielikumā) sastāv no 12 entitijām, kas atspoguļo datu apriti sistēmā.

* “Product” – produkts,
* “AttributeSet” – atribūtu sets ir sets, kas apvieno kaut kādu atribūtu daudzumu.
* “Attribute” – atribūts, kas apraksta konkrētas produkta īpašības
* “Category” – kategorija, ir grupa, kas apvieno vairākas citas kategorijas (apakškategorijas), vai/un produktus.
* “Customer” – pircējs – persona, kura meklē produktus, liek tos grozā un veic pasūtījumums.
* “Address” – adrese, kuru var pievienot pircējs pirkuma laikā, un kas ir nepieciešama pasūtījumam, lai nodrošinātu piegādi un apmaksu
* “Cart” – grozs, struktūra, kas satur sevī kopsavilkumu (cenas) par tā saturu, un sastāv tas no groza priekšmetiem.
* “CartItem” – groza priekšmets, ir produkts, kas tika pievienots grozām, ar papildus datiem - daudzums.
* “Order” – pasūtījums ir pircēja darbības galīgais rezultāts, tas ietver sevī visu nepieciešamo informāciju: kopējie dati: numurs, datums, cenas: cena par piegāi, nodokļi, kopējā cena bez nodokļiem, kopējā cena ar nodokļiem, un informāciju par visiem pasūtītiem produktiem un to daudzumu.
* “OrderItem” – pasutījuma priekšmets, kas pieder noteiktam pasūtījuma, kas saglabā sevī uz to brīdi aktuālo informāciju par produktu
* Invoice - rēķins, ir dokuments, kas satur visu galīgo informāciju par pasūtījuma datiem: kopējie dati: numurs, datums, cenas: cena par piegāi, nodokļi, kopējā cena bez nodokļiem, kopējā cena ar nodokļiem, un referenci uz konkrētu pasūtījumu un tā pasūtījuma priekšmetiem.
* Review – atsauksme, kuru uzraksta pircējs par produktu.

Datu bāzes relācijas parada kā savstarpēji ir savienotas divas vai vairākas entītijas:

* Starp produktiem un kategorijām ir attiecība daudzi pret daudziem, jo vienam produktam var būt vairākas kategorijas, un vienai kategorijai var būt vairāki produkti.
* Starp produktiem un atribūtu setiem ir attiecība daudzi pret vienu, jo produktam var būt tikai viens atribūtu sets, bet atribūtu setiem var būt vairāki produkti.
* Starp produktu un atsauksmi ir attiecība viens pret daudziem, jo vienam produktam var būt vairākas atsaukmes, bet vienai atsaukmei var būt tikai viens produkts.
* Starp pircēju un adresi ir attiecība viens pret daudziem, jo vienam pircējam var būt vairākas adreses, bet vienai adreses var būt tikai viens pircējs.
* Starp pircēju un produktu ir attiecība daudzi pret daudziem, jo viens pircējs var nopirkt vairākus produktus, un viens produkts var būt pārdots vairākiem pircējiem.
* Starp pircēju un pasūtījumu ir attiecība viens pret daudziem, jo vienam pircējam var būt vairāki pasūtījumi, bet pasūtījumam var būt tikai viens pircējs.
* Starp pircēju un atsauksmi ir attiecība viens pret daudziem, jo vienam pircējam var būt vairākas atsauksmes, bet vienai atsauksmei var būt tikai viens pircējs.
* Starp pircēju un pircēja grupu ir attiecība daudzi pret daudziem, jo vienam pircējam var būt vairākas pircēju grupas, un vienai pircēju grupai var būt vairāki pircēji.
* Starp kategoriju un kategoriju ir attiecība daudzi viens pret daudziem, jo vienai kategorijai var būt vairākas apakškategorijas, bet vienai apakškategorija, var būt tikai viena kategorija.
* Starp pasūtījumu un rēķinu ir attiecība viens pret viens, jo vienam pasūtījumam var būt tikai viens rēķins un vienam rēķinam var būt tikai viens pasūtījums.
* Starp atribūtu setu un atribūtu ir attiecība daudzi pret daudziem, jo vienam atribūtu setam var būt vairāki atribūti, un vienam atribūtam var būt vairāki atribūtu seti.

## Funkcionālais sistēmas modelis

# DATU STRUKTŪRAS APRAKSTS

Datu bāzes fiziskā struktūra (skatīt 2. pielikumā) sastāv no 16 tabulām, kas satur informāciju par sistēmas lietotājiem, klientiem, kategorijām, produktiem, pirkumiem, atribūtiem un citiem objektiem. Ir divas galvenās tabulas, kurām ir visvairāk savienojumu, tas ir produktu tabula, un klientu tabula, un, acīmredzami, šie divi objekti, kopumā arī definē lielāko daļu no sistēmas, jo klients pērk produktus. Datubāzes realizēšanai izmanto PostgreSQL. Shēma tika ģenerēta.

Tabulā “Address” glabājas informācija par klienta izveidotajām adresēm.

Šī tabula attiecās uz “Customer” tabulu ar “customer\_id” ārējo atslēgu, un viedo “daudzi pret vienu” attiecību – vairākas adreses var piederēt vienam klientam.

1. tabula

#### Tabulas “**Address**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numurs | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | 255 | Primāra atslēga |
| 2. | firstName | varchar | 255 | Vārds |
| 3. | lastName | varchar | 255 | Uzvārds |
| 4. | phoneNumber | varchar | 255 | Telefona numurs |
| 5. | country | varchar | 255 | Valsts |
| 6. | city | varchar | 255 | Pilsēta |
| 7. | province | varchar | 255 | Reģions |
| 8. | street1 | varchar | 255 | Iela |
| 9. | street2 | varchar | 255 | Iela |
| 10. | postalCode | varchar | 255 | Pasta kods |
| 11. | customer\_id | integer | - | Ārēja atslēga uz klientu |

Tabulā “Attribute” glabājas informācija par produkta atribūtiem.

Šī tabula attiecās uz “AttributeSet” tabulu un viedo “daudzi pret daudziem” attiecību – vairāki atribūti var piederēt vairākiem atribūtu setiem, tāpēc ir nepieciešama atsevišķā tabula “AttributeSetAttribute”, kur ir divas ārēja atslēgas, kas savieno šīs tabulas: “attribute\_set\_id”, “attribute\_id”.

2. tabula

#### Tabulas “**Attribute**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numurs | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | - | Primāra atslēga |
| 2. | code | varchar | 255 | Unikāls kods |
| 3. | isEnabled | boolean | - | Vai ir ieslēgts |
| 4. | label | varchar | - | Nosaukums |
| 5. | type | varchar | 255 | Datu tips |
| 6. | attributeOptions | jsonb | - | Iespējamie vērtību varianti |
| 7. | isFilter | boolean | - | Vai tiek izmantots, kā filtrs meklējot produktu |

Tabulā “AttributeSet” glabājas informācija par atribūtu setiem, tās ir kopas, kas apvieno vairākus atribūtus, un produktam var būt viena atribūtu kopa, un uz to kopu bāzēsies tā iespējamie atribūti.

Šī tabula attiecās uz “Attribute” tabulu un viedo “daudzi pret daudziem” attiecību – vairāki atribūti var piederēt vairākiem atribūtu setiem, tāpēc ir nepieciešama atsevišķā tabula “AttributeSetAttribute”, kur ir divas ārēja atslēgas, kas savieno šīs tabulas: “attribute\_set\_id”, “attribute\_id”.

3. tabula

#### Tabulas “**AttributeSet**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numurs | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | - | Primāra atslēga |
| 2. | code | varchar | 255 | Unikāls kods |
| 3. | isEnabled | boolean | - | Vai ir ieslēgts |
| 4. | name | varchar | 255 | Nosaukums |

Tabulā “AttributeSetAttribute” ir tabula, lai realizēta “daudzi pret daudziem” attiecību starp “AttributeSet” un “Attribute” tabulām.

4. tabula

#### Tabulas “**AttributeSetAttribute**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numurs | Nosaukums | Tips | Garums | Piezīme |
| 1. | attribute\_set\_id | integer | - | Ārēja atslēga uz atribūtu setu |
| 2. | attribute\_id | integer | - | Ārēja atslēga uz atribūtu |

Tabulā “Cart” (nepieciešams papildināt tabulas laukus) glabājas informācija par klienta grozu.

Šī tabula attiecās uz “Customer” tabulu ar “customer\_id” ārējo atslēgu, un veido “viens pret vienu”, jo vienam klientam var būt tikai viens grozs.

5. tabula

#### Tabulas “**Cart**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Numurs | Nosaukums | Tips | Garums | Piezīme |
| 1. | totalTax | double | - |  |
| 2. | subtotal | double | - |  |
| 3. | total | double | - |  |
| 4. | customer\_id | integer | - | Ārēja atslēga uz klientu |

Tabulā “CartItem” glabājas informācija par katru pievienotu produktu pie klienta groza.

Šī tabula ir nepieciešama, lai realizētu “daudzi pret daudziem” attiecību starp “Cart” un “Product” tabulām.

6. tabula

#### Tabulas “**CartItem**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | quantity | integer | - | Produkta daudzums |
| 2. | cart\_id | integer | - | Ārēja atslēga uz grozu |
| 3. | product\_id | integer | - | Ārēja atslēga uz produktu |

Tabulā “Category” glabājas informācija par kategorijām.

7. tabula

#### Tabulas “**Category**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | - | Primāra atslēga |
| 2. | urlKey | varchar | 255 | Unikāls lapas identifikators |
| 3. | isEnabled | boolean | - | Vai ir ieslēgta |
| 4. | name | varchar | 255 | Nosaukums |
| 5. | isInMenu | boolean | - | Vai tiek pievienota pie “menu” |
| 6. | category\_id | integer | - | Ārēja atslēga uz pašu sevi |

Tabulā “Customer” glabājas informācija par klientiem.

8. tabula

#### Tabulas “**Customer**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | - | Primāra atslēga |
| 2. | email | varchar | 255 | Unikāls epasts |
| 3. | password | varchart | 255 | Šifrēta parole |
| 4. | firstName | varchart | 255 | Vārds |
| 5. | lastName | varchart | 255 | Uzvārds |
| 6. | token | text | - | Autorizācijas tokens |
| 7. | isGuest | boolean | - | Vai ir viesis |

Tabulā “CustomerCustomerGroup” glabājas informācija par klientu grupu grupām.

Šī tabula ir nepieciešama, lai realizētu “daudzi pret daudziem” attiecību starp “Customer” un “CustomerGroup” tabulām.

9. tabula

#### Tabulas “**CustomerCustomerGroup**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | customer\_id | integer | - | Ārēja atslēga uz klientu |
| 2. | customer\_group\_id | integer | - | Ārēja atslēga uz klienta grupu |

Tabulā “CustomerGroup” glabājas informācija par klientu grupām.

10. tabula

#### Tabulas “CustomerGroup” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | customer\_id | integer | - | Ārēja atslēga uz klientu |
| 2. | customer\_group\_id | integer | - | Ārēja atslēga uz klienta grupu |

Tabulā “Invoice” glabājas informācija par klienta rēķiniem.

11. tabula

#### Tabulas “**Invoice**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | - | Primāra atslēga |
| 2. | date | date | - | Datums |
| 3. | totalDelivery | double | - | Pārvešanas izmaksas |
| 4. | totalTax | double | - | Nodokļi |
| 5. | subtotal | double | - | Kopējā cena bez nodokļiem |
| 6. | total | double | - | Kopējā cena |
| 7. | order\_id | integer | - | Ārēja atslēga uz pirkumu |

Tabulā “Order” glabājas informācija par pirkumiem.

12. tabula

#### Tabulas “**Order**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | - | Primāra atslēga |
| 2. | reference | varchar | 255 | Unikāla eference |
| 3. | date | date | - | Datums |
| 4. | status | varchar | 255 | Status |
| 5. | isReturned | boolean | - | Vai ir atgriezsts |
| 6. | totalDelivery | double | - | Pārvešanas izmaksas |
| 7. | totalTax | double | - | Nodokļi |
| 8. | subtotal | double | - | Kopēja cena bez nodokļiem |
| 9. | total | double | - | Kopēja cena |
| 10. | customer\_id | integer | - | Ārēja atslēga uz klientu |
| 11. | address\_id | integer | - | Ārēja atslēga uz adresi |

Tabulā “OrderItem” glabājas informācija par pirkumiem.

13. tabula

#### Tabulas “**OrderItem**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | - | Primāra atslēga |
| 2. | quantity | integer | - | Produkta daudzums |
| 3. | totalTax | dobule | - | Nodokļi |
| 4. | subtotal | double | - | Kopēja cena bez nodokļiem |
| 5. | total | double | - | Kopēja cena |
| 6. | order\_id | integer | - | Ārēja atslēga uz pirkumu |
| 7. | product\_id | integer | - | Ārēja atslēga uz produktu |

Tabulā “Product” glabājas informācija par produktiem.

14. tabula

#### Tabulas “**Product**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | - | Primāra atslēga |
| 2. | urlKey | varchar | 255 | Unikāls lapas identifikators |
| 3. | sku | varchar | 255 | Krājumu glabāšanas vienība |
| 4. | isEnabled | boolean | - | Vai ir ieslēgts |
| 5. | type | varchar | 255 | Tips (vienkāršs, konfigurēts) |
| 6. | price | double | - | Pamata cena |
| 7. | stockQuantity | integer | - | Daudzums noliktavā |
| 8. | specialDiscountType | varchar | 255 | Atlaides tips (daudzums, procentuāls) |
| 9. | specialDiscountValue | double | - | Atlaides vērtība |
| 10. | specialTaxRate | double | - | Nodokļu procents |
| 11. | shortDescription | varchar | 255 | Īss apraksts |
| 12. | longDescription | text | - | Garš apraksts |
| 13. | media | jsonb | - | Attēli un video |
| 14. | attributeValues | jsonb | - | Atribūtu vērtības |
| 16. | attribute\_set\_id | integer | - | Ārēja atslēga uz atribūtu |

Tabulā “ProductCategory” glabājas informācija par produktu kategorijām.

Šī tabula ir nepieciešama, lai realizētu “daudzi pret daudziem” attiecību starp “Product” un “Category” tabulām.

15. tabula

#### Tabulas “**ProductCategory**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | product\_id | integer | - | Ārēja atslēga uz produktu |
| 2. | category\_id | integer | - | Ārēja atslēga uz kategoriju |

Tabulā “Review” glabājas informācija par klienta atsauksmēm uz produktu.

16. tabula

#### Tabulas “**ProductCategory**” struktūra

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr. | Nosaukums | Tips | Garums | Piezīme |
| 1. | id | integer | - | Primāra atslēga |
| 2. | status | varchar | 255 | Status (pieņemta, apstrādā, noliegta) |
| 3. | date | date | - | Datums |
| 4. | title | varchar | 255 | Nosaukums |
| 5. | content | text | - | Saturs |
| 6. | rating | integer | - | Reitings no 1 līdz 10 |
| 7. | customer\_id | integer | - | Ārēja atslēga uz klientu |
| 8. | product\_id | integer | - | Ārēja atslēga uz produktu |

# **LIETOTĀJA CEĻVEDI**

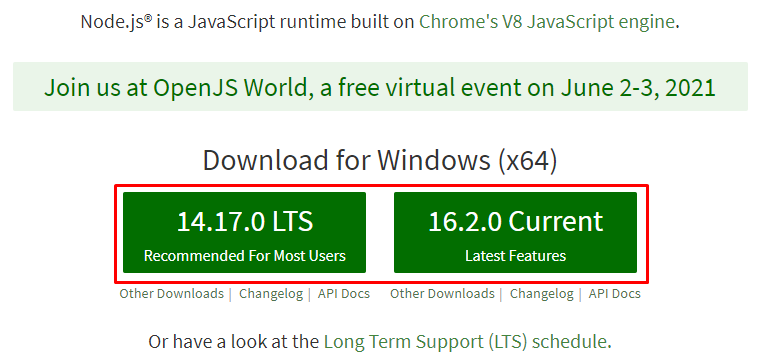
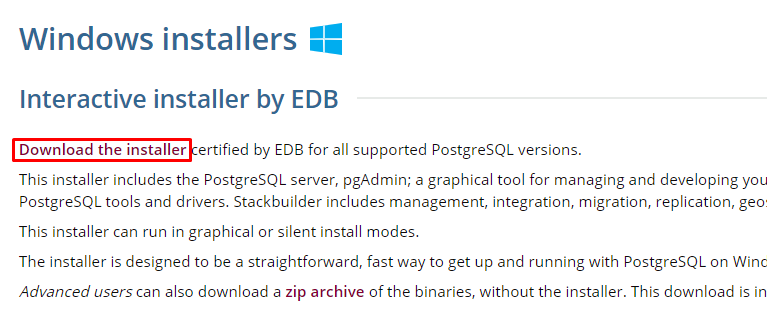
## Sistēmas prasības aparatūrai un programmatūrai

Nav specifiskas sistēmas aparatūras prasības, un programmu ir iespējams palaist gandrīz uz jebkuras modernās sistēmas, tāpēc, visas prasības attiecās tikai uz nepieciešamo programmatūru. Programma var būt instalēta gan uz Linux, gan uz Windows, taču MacOS netiek atbalstīs šobrīd, lai instalētu un palaistu to ir nepieciešama noteikta programmatūra:

* Node / NPM
* PostgreSQL

### Windows

Lai šo programmu varētu palaist ir nepieciešams uzinstalēt

1. Node / NPM
   1. Pāriet uz šo adresi: <https://nodejs.org/en/>
   2. Tad lejupielādēt 14.17.0 LTS versiiju, taču programma spēj strādāt arī ar visjaunāko Node versiju
   3. Palaist instalāciju un izpildīt visas darbības pēc noklusējuma
   4. Pēc šo darbību izpildes, programma būs veiksmīgi instalēta
2. PostgreSQL
   1. Pāriet uz šo adresi: <https://www.postgresql.org/download/windows/>
   2. Tad lejupielādē jaunāko PostgreSQL versiju
   3. Tad izpildīt visas darbības pēc noklusējuma.
   4. Pēc šo darbību izpildes, programma būs veiksmīgi instalēta.

### Linux (Ubuntu)

Lai uzinstalētu visu uz Linux (ubuntu), ir nepieciešams palaist tikai dažas komandas:

1. Node / NPM
   1. curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
   2. nvm install 14.17.0
   3. Pēc šo darbību izpildes, programma būs veiksmīgi instalēta
2. PostgreSQL
   1. sudo apt update
   2. sudo apt install postgresql postgresql-contrib
   3. Pēc šo darbību izpildes, programma būs veiksmīgi instalēta

## Sistēmas instalācija un palaišana

Vispirms ir nepieciešams strādājošs Node un PostgreSQL setups, kad tas ir sagatavots, tad ir nepieciešams palaist šīs komandas:

1. git clone https://github.com/rvtprog-kval-21/d41-DenissStrombergs-InternetaVeikals.git
2. cd d41-DenissStrombergs-InternetaVeikals
3. cd backend npm start
4. cd ..
5. cd frontend npm run

## **Programmas** apraksts

## **Testa piemērs**

# NOBEIGUMS

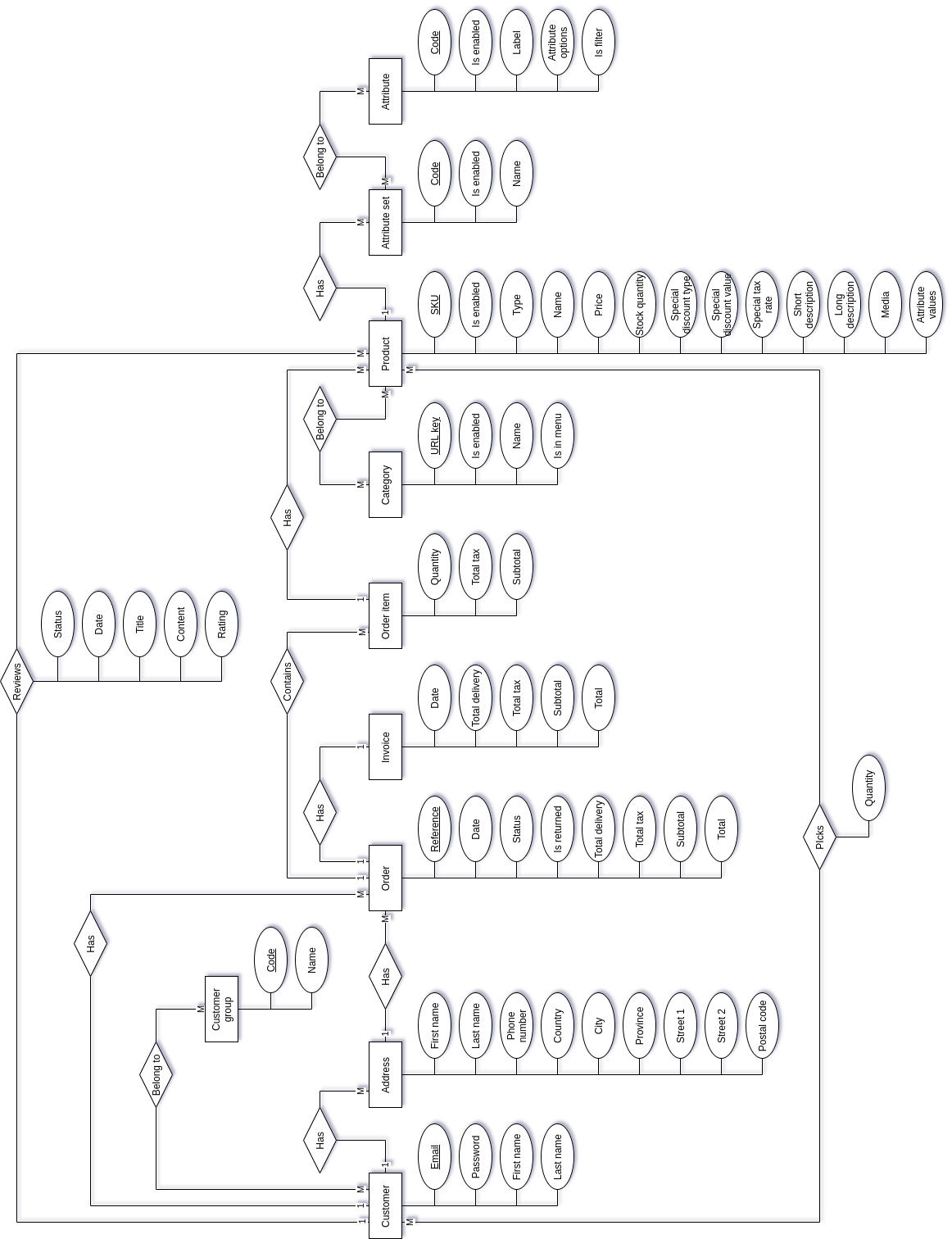
Secinājums

# INFORMĀCIJAS AVOTI

# PIELIKUMI

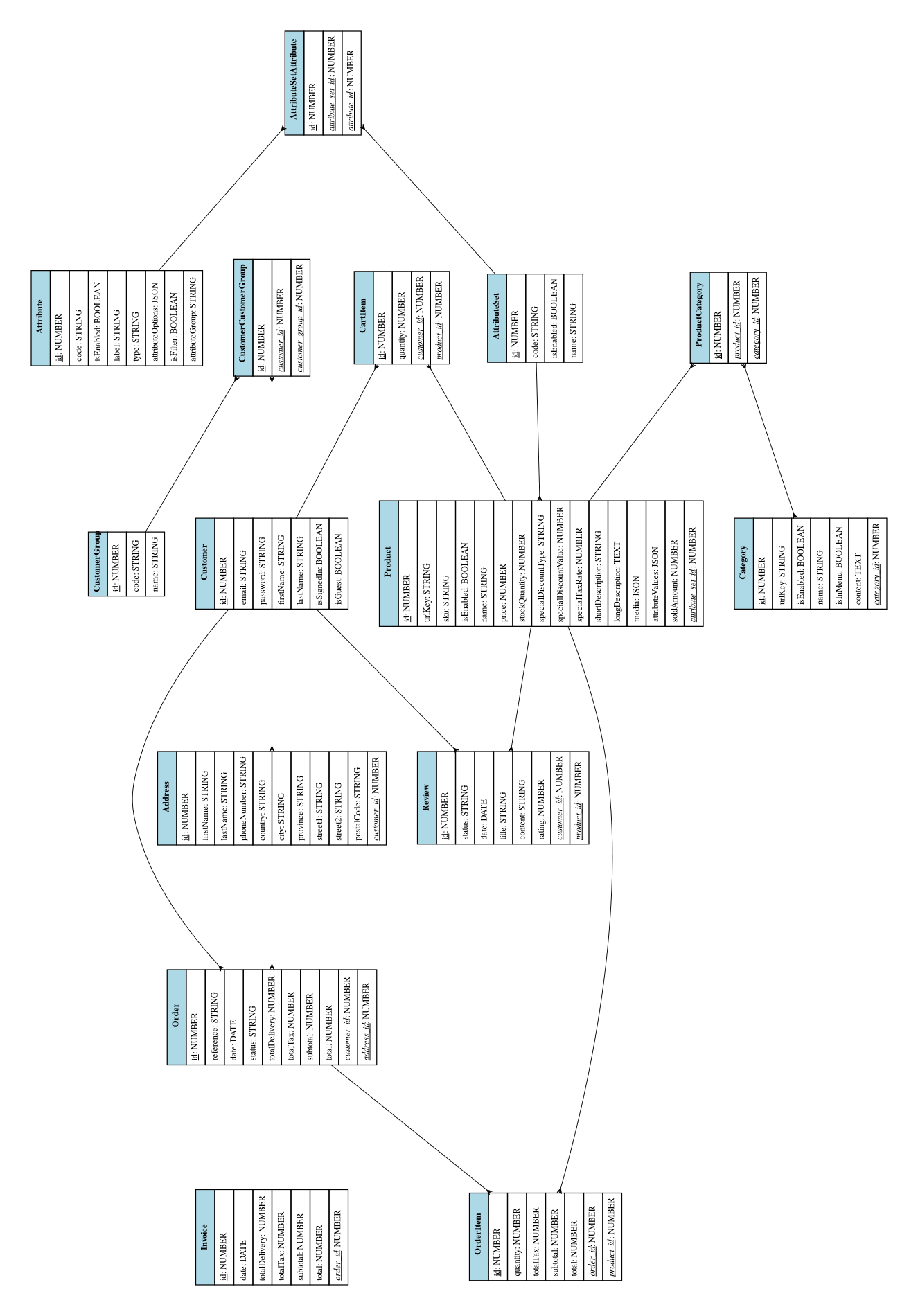
1. pielikums

## ER diagramma

  
1. attēls. ER diagramma

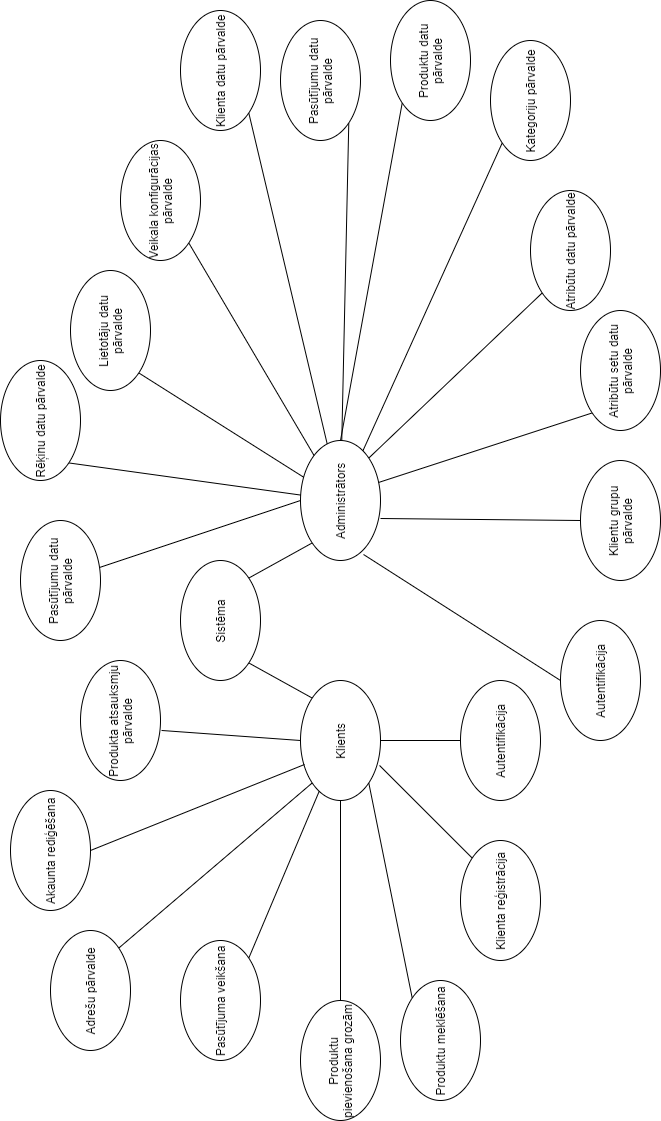
1. pielikums

## Fiziska struktūra

  
2. attēls. Fiziska struktūra

1. pielikums

## Sistēmas arhitektūra

  
3. attēls. Sistēmas arhitektūra