

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

Izglītības programma: Programmēšana

KVALIFIKĀCIJAS DARBS

Kopbraukšanas sistēma

Paskaidrojošais raksts 129 lpp.

Audzēknis:

R.Lagzdiņš

Vadītājs:

N.Pauders

Normu kontrole:

O. Sabanska

Rīga

2020

ANOTĀCIJA

Kvalifikācijas darbā ir aprakstīts kopbraukšanas sistēmas izstrādāšanas process. Sistēma dod iespēju lietotājiem pievienot savu braucienu, meklēt, apskatīt un pieteikties citu lietotāju izveidotajiem braucieniem. Kā arī ir iespējams pārvaldīt lietotāju sistēmas izmantošanas tiesības. Taču galvenais uzsvars ir likts uz sistēmas izmantošanu autorizētiem lietotājiem. Sistēma tika izstrādāta ar PHP programmēšanas valodu, izmantojot Yii2 ietvaru, MVC programmēšanas principu, un MySQL datu bāzu pārvaldības sistēmu.

Kvalifikācijas darbs ietver sevī ievadu, uzdevuma nostādni, prasību specifikāciju, uzdevuma risināšanas līdzekļu izvēles pamatojumu, programmatūras produkta modelēšanas un projektēšanas aprakstu, datu struktūru aprakstu, lietotāja ceļvedi, nobeigumu un pielikumus. Ievadā ir aprakstīta šīs sistēmas aktualitāte un pamatota tās vajadzība. Uzdevumu nostādnē ir galvenās sistēmas funkcionalitātes, kuras sistēmai jāveic. Prasību specifikācijā sastāv no ieejas informācijas, izejas informācijas un arēja informācijas. Aprakstītas sistēmas funkcionālās un nefunkcionālās prasības. Uzdevumu risināšanas līdzekļu izvēles pamatojumā ir apraksts par izstrādes rīkiem tika izmantoti sistēmas izstrādē, un iemesls kādēļ tie tika izvēlēti. Programmatūras produkta modelēšanas un projektēšanas apraksts sastāv no sistēmas struktūras modeļa, kas ietver sistēmas arhitektūru, sistēmas ER modeli, un funkcionālās sistēmas modeli, kas satur detalizētos komponenta projektējumus. Datu struktūru aprakstā tiek paskaidrota un parādīta datu bāzes relācijas shēma un tabulu struktūra. Lietotāja ceļvedī ir norādītas nepieciešamās sistēmas prasības aparatūrai un programmatūrai, sistēmas instalācija un palaišana, kas paskaidro, kā pareizi jāizmanto sistēma. Testa piemērā ir dots brauciena pievienošanas apraksts, un apraksts par to kā atrast un pieteikties sev vēlamajam braucienam.

Kvalifikācijas darbs sastāv no 129 lapaspusēm, kurā ietilpst 20 attēli 7 tabulas, un 3 pielikumi. Pielikumi satur ER diagrammu, tabulu saišu shēmu un programmas pirmkodu.

ANOTATION

The qualification work describes the process of developing a ridesharing system. The system allows users to add their own trip, search, view and log in to trips created by other users. It is also possible to manage user rights to use the system. However, the main emphasis is on using the system for authorized users. The system was developed with PHP programming language using Yii2 framework, MVC programming principle, and MySQL database management system.

The qualification work includes an introduction, task statement, requirements specification, task solution means selection justification, software product modeling and design description, data structure description, user guide, conclusion and appendices. The introduction describes the topicality of this system and substantiates its need. The task statement contains the main system functionalities that the system must perform. The requirements specification consists of input information, output information and external information. The functional and non-functional requirements of the system are described. The choice of problem-solving tools is based on a description of the development tools used in the development of the system and the reason why they were chosen. The software product modeling and design description consists of a system structure model that includes a system architecture, a system ER model, and a functional system model that contains detailed component designs. The description of data structures explains and shows the database relational scheme and table structure. The user guide provides the necessary hardware and software system requirements, system installation and startup, and explains how to use the system properly. The test example provides a description of adding a trip and a description of how to find and apply for your desired trip.

The qualification paper consists of 124 pages, which includes 20 figures, 7 tables and 3 appendices. The appendices contain the ER diagram, the table link diagram and the program source code.

SATURS

IEVADS	5
1. UZDEVUMA NOSTĀDNE.....	6
2. PRASĪBU SPECIFIKĀCIJA.....	7
2.1. Ieejas un izejas informācijas apraksts.....	7
2.1.1. Ieejas informācijas apraksts	7
2.1.2. Izejas informācijas apraksts	8
2.1.3. Arējās informācijas apraksts	8
2.2. Funkcionālās prasības	9
2.3. Nefunkcionālās prasības	11
3.UZDEVUMU RISINĀŠANAS LĪDZEKĻU IZVĒLES PAMATOJUMS.....	12
4. PROGRAMMATŪRAS PRODUKTA MODELĒŠANA UN PROJEKTĒŠANA	14
4.1. Sistēmas struktūras modelis.....	14
4.1.1. Sistēmas arhitektūra.....	14
4.1.2. Sistēmas ER modelis	18
4.2. Funkcionālais sistēmas modelis	19
5. DATU STRUKTŪRU APRAKSTS.....	25
6. LIETOTĀJA CEĻVEDIS.....	29
6.1. Sistēmas prasības aparatūrai un programmatūrai.....	29
6.2. Sistēmas instalācija un palaišana.....	30
6.3. Programmas apraksts.....	31
6.4. Testa piemērs	34
7. NOBEIGUMS.....	38
8. INFORMĀCIJAS AVOTI	39
PIELIKUMI	40

IEVADS

Mūsdienu sabiedrībā cilvēki vienmēr ir kustībā un tiem ir vēlme vai vajadzība kaut kur nonākt. Diemžēl ne visiem ir pieejams savs transportlīdzeklis, tāpēc tiem nākas izmantot sabiedrisko transportu.

Bet, kā jau gandrīz visam, sabiedriskajam transportam ir savi mīnusi. Tie mēdz būt pārpildīti, kursēt ļoti reti, vai arī mēdz būt situācijas, ka sabiedriskais transports uz noteiktām vietām nekursē vispār. Taču tiem, kam ir pieejams savs transportlīdzeklis, mēdz saskarties ar salīdzinoši lielām izmaksām, kas saistās ar transportlīdzekļa uzturēšanu un ceļa izmaksām. Šīs sistēmas mērķis ir apvienot šīs divas cilvēku grupas, lai atvieglotu transportlīdzekļa atrašanu uz kādu no vajadzīgajiem galamērķiem, un, iespējams, samazinātu ceļa izmaksas tiem, kam ir savs transportlīdzeklis.

Sistēma būs pieejama reģistrētiem lietotājiem, gan arī viesiem. Taču izmantojot sistēmu kā viesim būs pieejama tikai daļa no sistēmas. Pieteikties braucienam un tos izveidot būs iespējams tikai reģistrētam lietotājam, bet apskatīt braucienus varēs ikviens.

1. UZDEVUMA NOSTĀDNE

Darba uzdevums ir izveidot kopbraukšanas sistēmu. Ar tās palīdzību lietotājs var apskatīt pieejamos braucienus, pievienot pats savu braucienu un pieteikties kādam braucienam.

Kopbraukšanas sistēmai ir jāizpilda vairākas funkcionalitātes:

- Datu – lietotāja, transportlīdzekļa un brauciena – pievienošana, rediģēšana, apstrāde un dzēšana;
- Brauciena izveide;
- Brauciena dzēšana
- Pieteikšanās braucienam.
- Atteikšanās no brauciena.
- Lietotāju administrēšana.

2. PRASĪBU SPECIFIKĀCIJA

2.1. Ieejas un izejas informācijas apraksts

2.1.1. Ieejas informācijas apraksts

Sistēmā tiks nodrošināta šāda ieejas informāciju apstrāde.

1. Informācija par lietotāju sastāvēs no šādiem datiem
 - a. Lietotājvārds - burtu un ciparu teksts ar izmēru līdz 40 rakstzīmēm
 - b. Vārds – burtu un ciparu teksts ar izmēru līdz 40 rakstzīmēm.
 - c. uzvārds- burtu un ciparu teksts līdz 40 rakstzīmēm.
 - d. epasts – burtu un ciparu teksts ar izmēru līdz 50 rakstzīmēm.
 - e. Telefona numurs – ciparu un simbolu virkne līdz 25 rakstzīmēm
 - f. Parole - burtu un ciparu teksts ar izmēru līdz 200 rakstzīmēm.
2. Informācija par transportlīdzekli sastāvēs no šādiem datiem.
 - a. Reģistrācijas numurs – burtu un ciparu teksts ar izmēru līdz 11 rakstzīmēm.
 - b. Krāsa – burtu un ciparu teksts ar izmēru līdz 20 rakstzīmēm
 - c. Ražotājs - burtu un ciparu teksts ar izmēru līdz 40 rakstzīmēm
 - d. Modelis – burtu un ciparu teksts ar izmēru līdz 40 rakstzīmēm
 - e. Vietu skaits – cipars ar izmēru 1
 - f. Attēls – Attēla fails
3. Informācija par braucienu sastāvēs no šādiem datiem
 - a. Datums – datums un laiks
 - b. Piezīmes - burtu teksts ar izmēru līdz 400 rakstzīmēm.
4. Meklēšanā
 - a. Pieejamais datums – datums
 - b. Ietilpība – skaitlis
 - c. Cena par īri – skaitlis ar 2 zīmēm aiz komata

2.1.2. Izejas informācijas apraksts

1. Kopējā brauciena informācija, kas sevī iekļauj izbraukšanas laiku, datumu un vietu, un datus, lai iespējams sakontaktēties.
2. Epasta vēstule ar līdzbraucēju kontaktinformāciju.
3. Pievienoto braucienu saraksts ar iespējām to filtrēt

2.1.3. Arējās informācijas apraksts

Sistēmai ir informācijas apmaiņa ar “<https://cloud.google.com/maps-platform/>”. Sistēma saņem kartes datus – Sākuma punktu, galapunktu, aptuveno ceļu.

Sistēmas savienojās ar automātisku pieprasījuma izveidošanu. Pieprasījuma rezultātā tiek veidota autorizēšanās, kartes datu iegūšana. Tiek padoti šādi sistēmas dati – sākumpunkta koordinātes, galapunkta koordinātes, API atslēga.

2.2. Funkcionālās prasības

1. Jānodrošina klienta informācijas apstrāde.
 - 1.1. Sistēmai jānodrošina klienta reģistrācija tikai tad, ja visi lauki ir korekti aizpildīti.
 - 1.2. Sistēmai ir jānodrošina lietotājvārda maiņa.
 - 1.3. Sistēmai ir jānodrošina lietotājvārda unikalitāte.
 - 1.4. Sistēmai ir jānodrošina lietotājvārda saglabāšana tikai ar mazajiem burtiem un cipariem.
 - 1.5. Sistēmai ir jānodrošina paroles maiņa.
 - 1.6. Sistēmai ir jānodrošina paroles maiņa izmantojot unikālu saiti, kas klientam tiek nosūtīta uz epastu.
 - 1.7. Paroles maiņas saite drīkst būt derīga tikai 10 minūtes
 - 1.8. Sistēmai ir jānodrošina epasta maiņa.
 - 1.9. Sistēmai ir jānodrošina telefona numura maiņa.
 - 1.10. Sistēmai ir jānodrošina klientu un administratoru datu saglabāšana.
2. Jānodrošina klienta un administratoru autorizācija.
 - 2.1. Sistēmai ir jānodrošina autorizācija, ievadot lietotājvārdu un paroli.
 - 2.2. Sistēmai ir jānodrošina autorizācija, ja lietotājs nav bloķēts.
 - 2.3. Sistēmai ir jālīdzekļve nepareizas datu ievadīšanas rezultātā.
 - 2.4. Sistēmai ir jāizvada paziņojums nepareizas datu ievadīšanas rezultātā.
 - 2.5. Sistēmai ir jālīdzekļve, ja lietotājs ir bloķēts.
 - 2.6. Sistēmai ne administratoru lietotājiem jālīdzekļve pie administratoru paneļa.
 - 2.7. Sistēmai ir jānodrošina lauku aizpildes validācija.
 - 2.8. Sistēmai jānodrošina administratoru piekļuve pie lietotāju informācijas.
3. Jānodrošina transportlīdzekļa informācijas apstrāde.
 - 3.1. Sistēmai jānodrošina transportlīdzekļa pievienošana tikai tad, ja visi lauki ir korekti aizpildīti.
 - 3.2. Sistēmai jāizvada kļūdas paziņojums, nepareizu lauku aizpildes gadījumā.
 - 3.3. Sistēmai jānodrošina transportlīdzekļa reģistrācijas numura rediģēšana.
 - 3.4. Sistēmai jānodrošina transportlīdzekļa krāsas rediģēšana.
 - 3.5. Sistēmai jānodrošina transportlīdzekļa attēla maiņa.

- 3.6. Sistēmai jānodrošina transportlīdzekļa dzēšana, ja tas netiek izmantots nevienā braucienā.
4. Jānodrošina brauciena informācijas apstrāde.
 - 4.1. Braucienus var izveidot tikai tad, ja visi lauki ir korekti aizpildīti.
 - 4.2. Braucienus var pievienot tikai tad, ja ir pievienots vismaz viens transportlīdzeklis.
 - 4.3. Iespēja dzēst braucienus tikai tad, ja nevienš cits lietotājs nav pieteicies.
 - 4.4. Sistēmai jānodrošina iespēja apskatīt cik brīvas vietas ir braucienam.
 - 4.5. Jāsaglabā informācija par braucieniem.
 - 4.6. Sistēmai jānodrošina brauciena pievienošana vismaz 3 stundas uz priekšu no sistēmas laika
 - 4.7. Sistēmai jānodrošina braucienus meklēšana
 - 4.8. Sistēmai jānodrošina braucienus meklēšana pēc datuma
 - 4.9. Sistēmai jānodrošina braucienus meklēšana pēc pilsētām
 - 4.10. Sistēmai jānodrošina braucienus kārtošana pēc izbraukšanas laika
5. Jānodrošina datu attēlošana tabulās.
 - 5.1. Sistēmai jānodrošina datu attēlošana tabulā.
 - 5.2. Ja ir vairāk kā 10 ieraksti, sistēmai dati jāsadala pa vairākām lapām
 - 5.3. Vienā lapaspusē attēlo līdz 10 ierakstiem
6. Jānodrošina paziņojumu nosūtīšana
 - 6.1. Sistēmai jānodrošina paziņojums uz lietotāja epastu, ja braucienam pieteicies pasažieris
 - 6.2. Sistēmai jānodrošina paziņojums uz lietotāja epastu, ja pasažieris atteicies no brauciena
 - 6.3. Sistēmai jānodrošina paziņojums uz epastu sistēmas uzturētājam, ja notikusi kāda kļūda

2.3. Nefunkcionālās prasības

1. Sistēmas saskarnes valodai ir jābūt latviešu valodai
2. Jānodrošina programmas pielāgošanos ekrāna izmēram
3. Jānodrošina tīmekļa lietojumprogrammas pielāgošanos ekrāna izmēram(responsīvs dizains), lai to varētu lietot no dažādām ierīcēm.
4. Izvēlnei jāatrodas lapas loga augšā.
5. Izvēlnei uz mazākiem ekrāniem jāparādās nolaižamā sarakstā.

3.UZDEVUMU RISINĀŠANAS LĪDZEKĻU IZVĒLES PAMATOJUMS

Lai izveidotu kursa darbu tika izmantotas daudz un dažādas tehnoloģijas.

Lai veiktu izstrādi tika izmantota izstrādes vide PhpStorm(2019.3.3), ko izstrādājis JetBrains. Šo izstrādes vidi izvēlējies, jo tai ir daudz lietderīgu izstrādes rīku, kas atvieglo darbu, un, protams, studentiem ir pieejams bez maksas.

Programmatūras saglabāšanai tika izmantota viena no populārākajām atvērtā koda versiju kontroles sistēmām – Git(2.26.2) un Github, kas piedāvā koda repozitoriju uzturēšanu.

Pašas sistēmas izveidē tika izmantots atvērtā koda PHP ietvars “Yii2”. Ar šo ietvaru mājaslapa tiek izveidota pēc populāra programmatūras izstrādes veida - MVC(Model-View-Controller), jeb modelis-skats-kontrolieris. Šis ietvars nodrošina drošību pret vairākiem populāriem uzbrukumu veidiem, piemēram SQL injekcija, starpvietņu skriptošanu un citiem.

Mājaslapas darbības nodrošināšanai tika izmantots bezmaksas atvērtā koda programmatūra XAMPP(v3.2.4). Šajā programmā ir iekļauts PHP(7.4.2) interpeters un vairāki moduļi, taču izmantoju tikai šos divus – Apache HTTP servers(2.4.41) un MySQL datubāzu pārvaldības sistēma.

Datu saglabāšanai tika izmantota MySQL(7.4.2). Tā ir bezmaksas, atvērtā koda relāciju datubāzes pārvaldības sistēma. Datubāzes izveidošanai, rediģēšanai un aplūkošanai tika izmantots “phpMyAdmin(5.0.1)”

Lai vieglāk būtu izmantot citu izstrādātāju radītos modulus tika izmantota programmatūra Composer(1.10.6)

Lai cilvēki varētu piekļūt mājaslapai tiek izmantots AWS(Amazon Web Services) virtuālais t3.micro windows servers.

IZMANTOTIE AIZGULTIE MODUĻI

Datuma un laika atlasītājs(Date Time Picker). Aizgultais kods, lai izveidotu datuma un laika izvēles logu.

Datņu pievienošanas logs(widget-fileinput). Lai vieglāk pievienotu attēlus, tos vienkārši ievielkot attiecīgajā logā.

Google kartes apstrāde(google-maps-library) Vienkāršākai kartes attēlošanai un apstrādei.

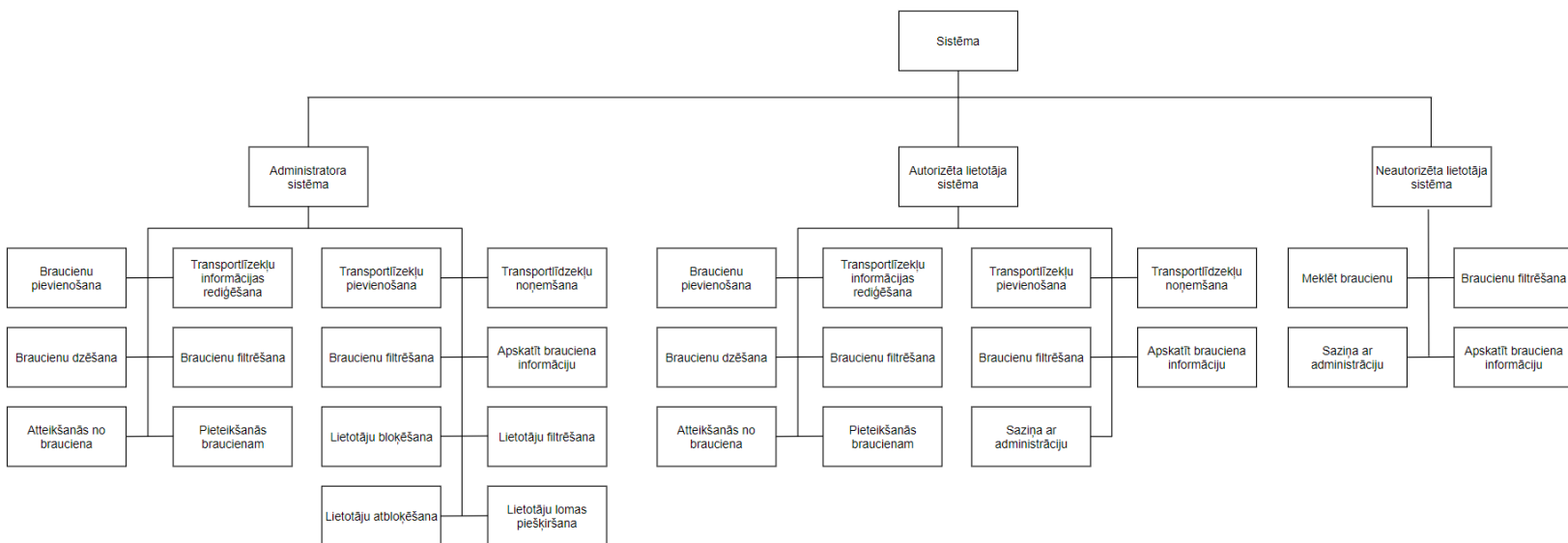
Meklētāja lauks(select2) Ļauj savienot tekstu ar citām vērtībām un vieglākai datu meklēšanai.

4. PROGRAMMATŪRAS PRODUKTA MODELĒŠANA UN PROJEKTĒŠANA

4.1. Sistēmas struktūras modelis

4.1.1. Sistēmas arhitektūra

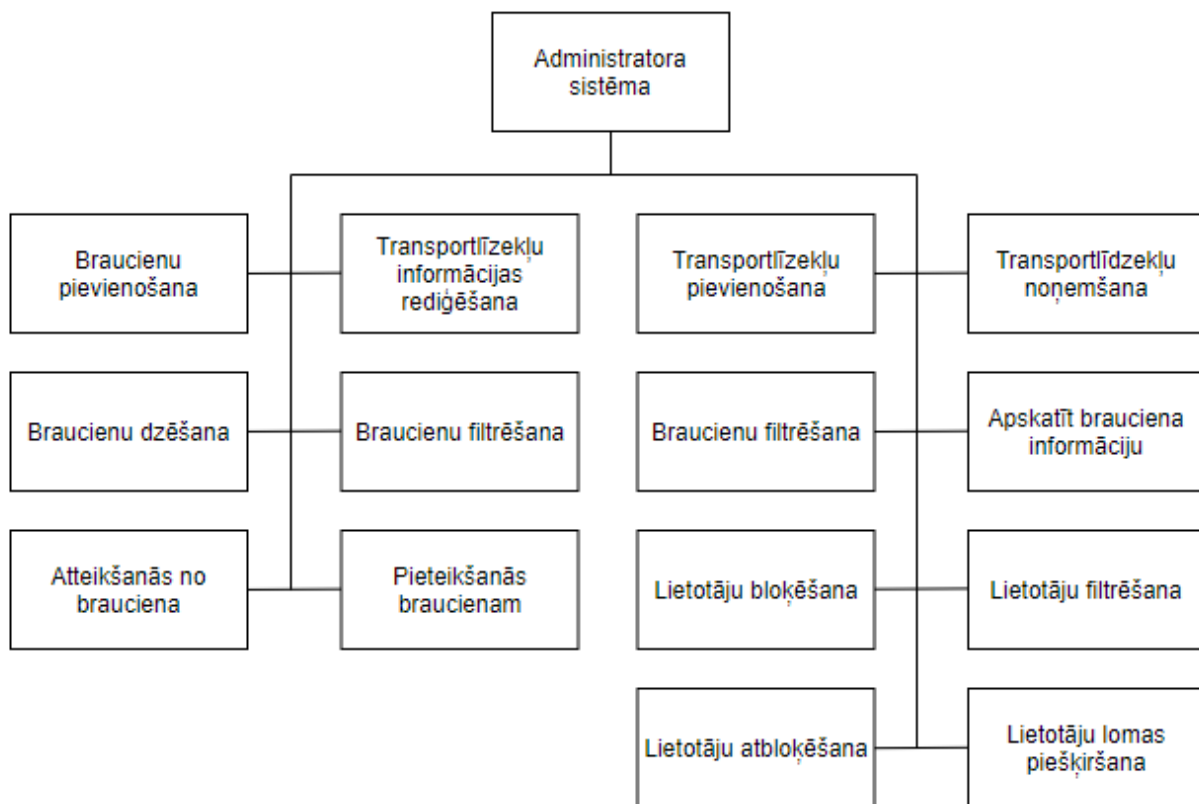
Sistēmai ir 3 daļas. Neautorizētu lietotāju, autorizētu lietotāju un administratora. Šīm 3 sistēmām nav lielu atšķirību, vienīgi atšķiras pieejas līmeņi pie dažādām sistēmas daļām.



4.1 att. Sistēmas struktūras modeļa attēlojums

Administrators sistēma

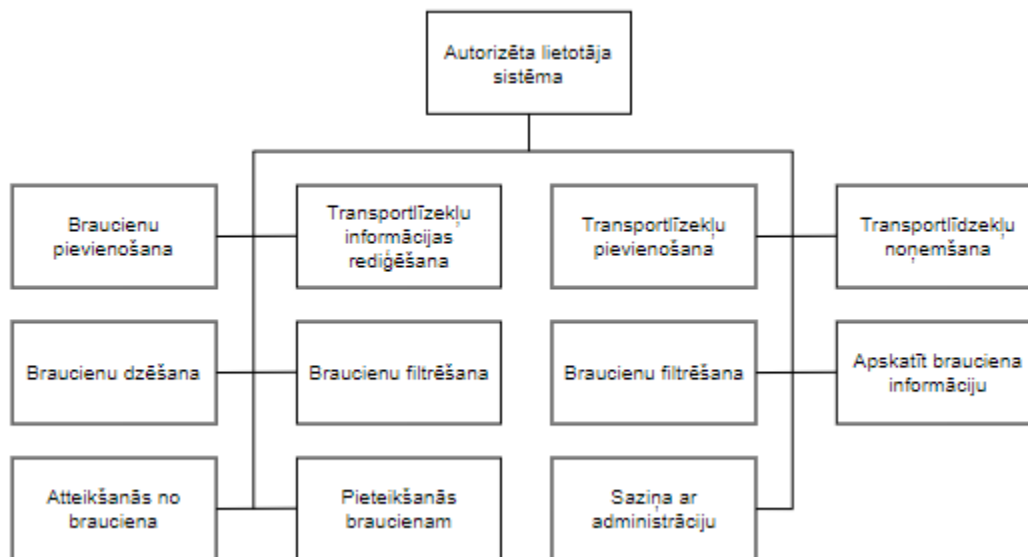
Administratoriem ir visvairāk pieeju, tie var apskatīt visus reģistrētos lietotājus, liegt un atjaunot pieeju reģistrētiem lietotājiem. Kā arī administratoriem var veikt tādas pašas darbības kā autorizēti lietotāji.



4.2 att. Administrators sistēmas struktūras modeļa attēlojums

Autorizēta lietotāja sistēma

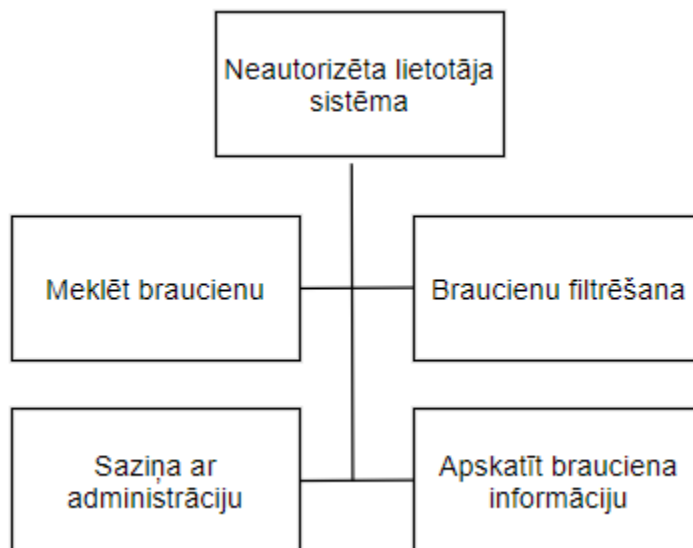
Autorizēta lietotāja sistēma ir daudz vairāk pieeju kā neautorizētam lietotājam. Autorizētam lietotājam ir iespēja pievienot transportlīdzekli, izveidot savu braucienu, kā arī meklēt un pievienoties citu cilvēku izveidotiem braucieniem.



4.3 att. Autorizēta lietotāja sistēmas struktūras modeļa attēlojums

Neautorizēta lietotāja sistēma

Neautorizētam lietotājam pieeja un funkcionalitāte ir ļoti limitēta. Neautorizēts lietotājs drīkst meklēt un apskatīt braucienus.



4.4 att. Neautorizēta lietotāja sistēmas struktūras modeļa attēlojums

4.1.2. Sistēmas ER modelis

Sistēma darbojas ar iepriekš pievienotiem, uzstādītiem datiem, un lietotāju pievienotiem datiem. (Skat 1. pielikums)

Datu kopai lietotājs ir 2 saites.

Pirmā saite ir ar lietotāja braucieniem, viens pret daudziem, jo viens lietotājs var pieteikties vai izveidot vairākus braucienus.

Otrā saite ir ar transportlīdzekli, viens pret daudziem, jo vienam lietotājam var piederēt vairāki transportlīdzekļi, bet vienam transportlīdzeklī var būt viens īpašnieks.

Datu kopai braucieni ir 1 saite.

Saite ir ar lietotāja braucieniem, viens pret daudziem, jo vienam braucienam var būt vairāki braucēji.

Datu kopai transportlīdzeklis ir 1 saite.

Saite ir ar braucieniem, viens pret daudziem, jo viens transportlīdzeklis var tikt izmantots vairākos braucienos.

Datu kopai pilsētas ir 2 saites.

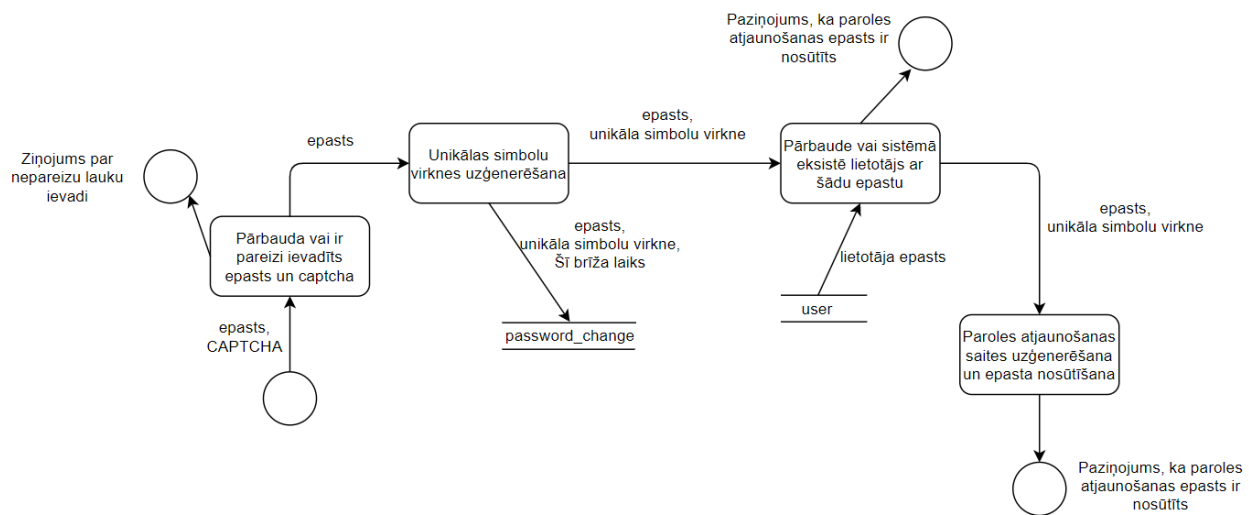
Pirmā saite ir ar braucieniem, viens pret daudziem, jo no vienas pilsētas var notikt vairāki braucieni.

Otrā saite ir ar braucieniem, viens pret daudziem, jo no uz vienu pilsētu var notikt vairāki braucieni.

4.2. Funkcionālais sistēmas modelis

Paroles atjaunošanas pieprasīšana

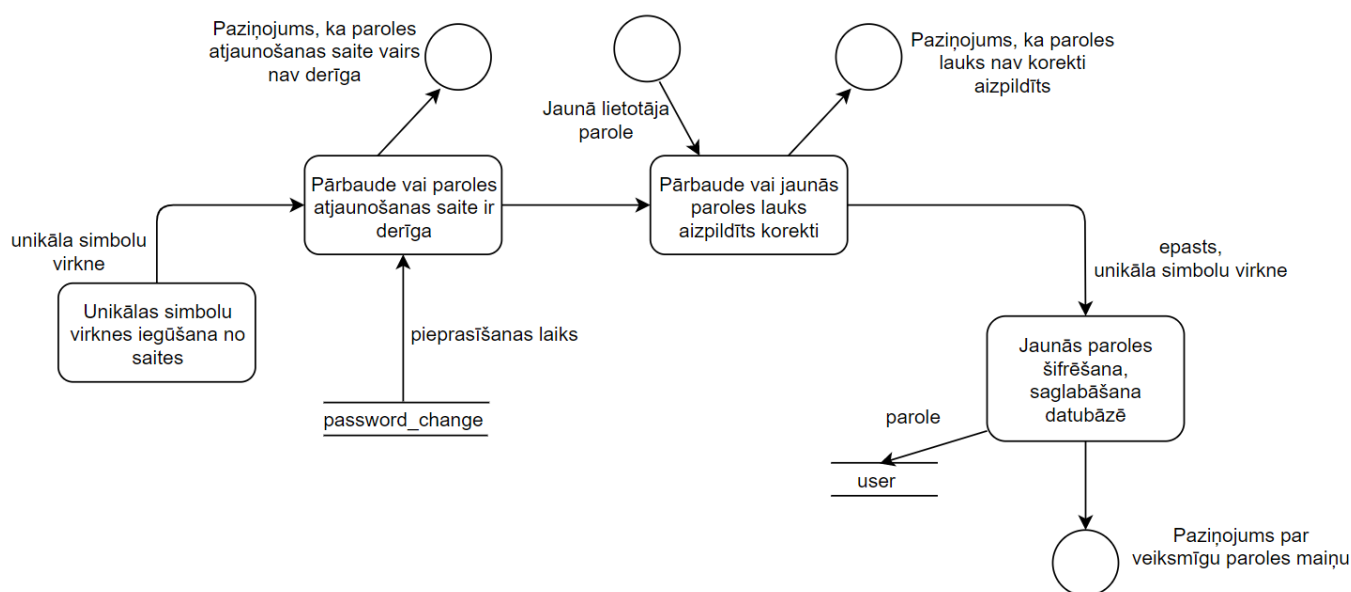
Paroles atjaunošanas pieprasīšana (Skat att. 4.5.) Atverot paroles atjaunošanas pieprasīšanas skatu lietotājam ir jāievada savs epasts un CAPTCHA. Pirmais process pārbauda vai lauki ir pareizi aizpildīti. Gadījumā, ja kāda no pārbaudēm nav sekmīga, tiek izvadīts kļūdas paziņojums. Otrais process veic simbolu virknes uzģenerēšanu un lietotāja epasta, kā arī šī brīža laika saglabāšanu datubāzē. Trešais process veic pieprasījumu uz datubāzi un pārbauda vai eksistē lietotājs ar šādu epasta adresi. Drošības nolūkos tiek izvadīts paziņojums, ka epasts ir nosūtīts, pat ja šāds lietotājs neeksistē. Bet gadījumā, ja lietotājs eksistē, tad tiek uzģenerēta paroles atjaunošanas saite un tiek nosūtīts epasts lietotājam.



4.5 att. Paroles atjaunošanas pieprasīšanas detalizētais komponentu attēlojums

Paroles atjaunošana

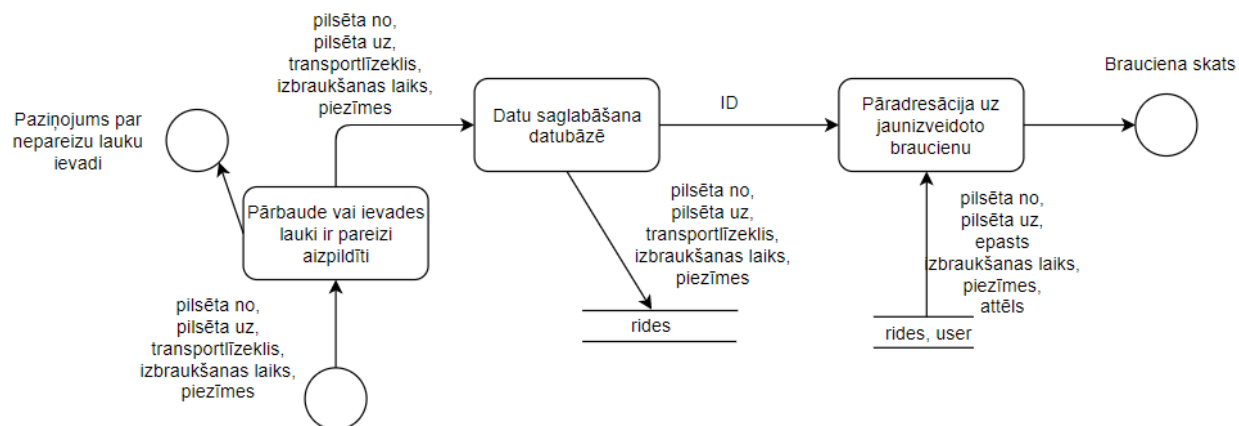
Paroles atjaunošana (Skat att. 4.6.) Pirmajā procesā sistēma no saites iegūst paroles atjaunošanas unikālo simbolu virkni. Otrais process pārbauda vai paroles atjaunošanas saite ir derīga. Ja kopš paroles atjaunošanas pieprasīšanas ir pagājušas 10 minūtes, vai arī saite ir sabojāta, tad sistēma izvada paziņojumu, ka paroles atjaunošanas saite vairs nav derīga. Ja pārbaude ir veiksmīga, tad lietotājam tiek parādīts paroles atjaunošanas skats, kurā lietotājs var ievadīt savu jauno paroli. Trešajā procesā tiek pārbaudīts, vai lietotājs ir pareizi aizpildījis paroles ievades lauku, ja tas nav izdarīts korekti, lietotājam tiek izvadīts kļūdas paziņojums. Pēdējā procesā tiek šifrēta un saglabāta jaunā lietotāja parole un beigās izvadīts paziņojums, ka paroles maiņa ir notikusi veiksmīgi.



4.6 att. Paroles atjaunošanas detalizētais komponentu attēlojums

Brauciena izveide

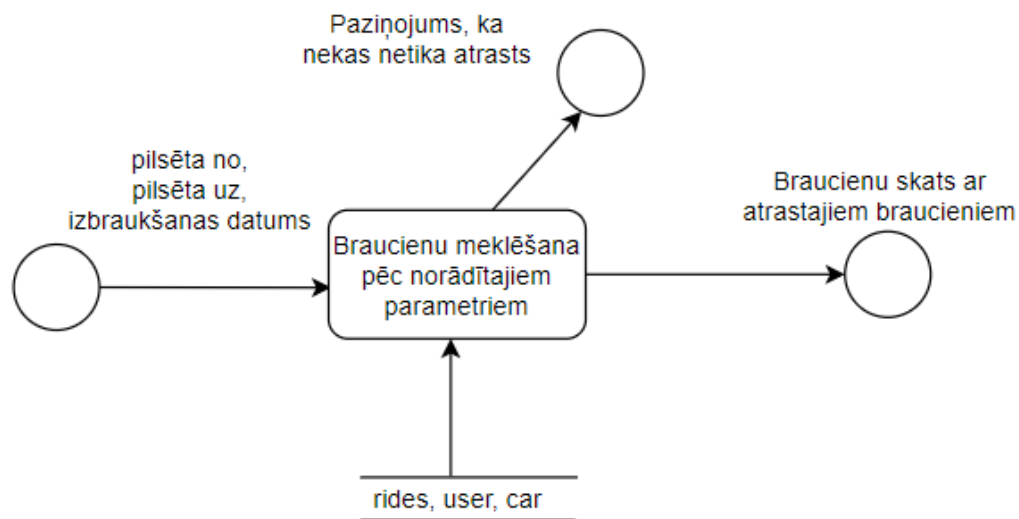
Brauciena pievienošana (Skat att. 4.7.) Pirmajā procesā tiek pārbaudīts, vai ievadītie lauki ir aizpildīti pareizi, ja nav, tiek izvadīts paziņojums, par nepareizu lauku aizpildi. Ja visi lauki ir pareizi aizpildīti, brauciena dati tiek saglabāti datubāzē, un lietotājs tiek pārvirzīts uz tikko pievienoto brauciena lapu.



4.7 att. Brauciena izveides detalizētais komponentu attēlojums

Braucienu meklēšana

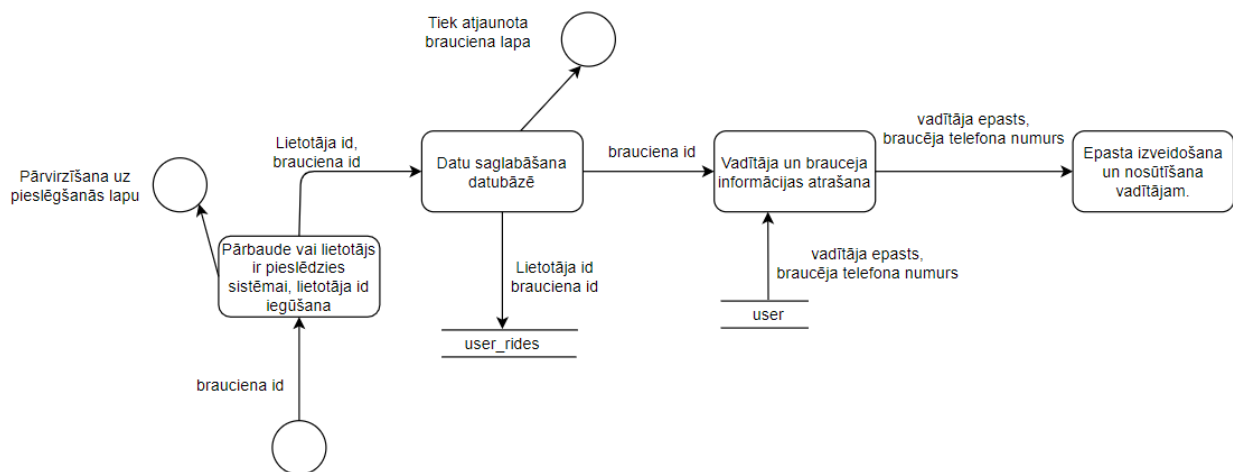
Braucienu meklēšana (Skat att. 4.8) Lietotājam ir iespēja ievadīt vēlamu pilsētu no, pilsētu uz un brauciena dienu. Ja šie parametri nav ievadīti, tad tiek meklēti visi braucieni, kas ir pieejami. Ja nav izdevies atrast nevienu pieejamu braucienu, vietā, kur jābūt braucienu sarakstam tiek parādīts kļūdas paziņojums, ka nav neviena brauciena. Tomēr, ja ir atrasts kāds brauciens, tad visi braucieni, kas ir atrasti tiek parādīti tabulā.



4.8 att. Brauciena meklēšanas detalizētais komponentu attēlojums

Pievienošanās braucienam

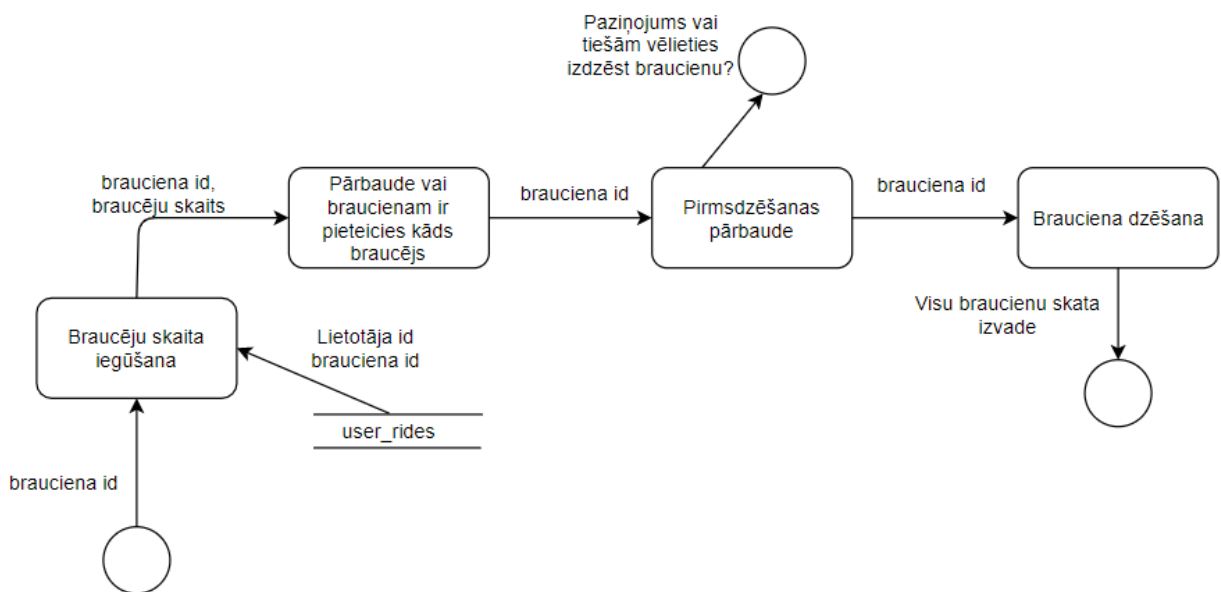
Pievienošanās braucienam, (Skat att. 4.9.) Kad lietotājs ir atradis braucienu, kas viņam ir iepaticies, tam ir iespēja pieteikties. Tā kā tajā brīdī lietotājs atrodas brauciena lapā, tad sistēma iegūst brauciena id. Pirmajā procesā tiek pārbaudīts vai lietotājs ir pieslēdzies sistēmai, jo braucienam var pievienoties tikai autorizēti lietotāji. Ja tas nav izdarīts, tad lietotājs tiek pārvirzīts uz pievienošanās lapu. Ja lietotājs ir pieslēdzies, tad sistēma iegūst lietotāja id. Nākamajā procesā dati tiek saglabāti datu bāzē un lietotājam tiek atjaunota brauciena lapā. Trešajā procesā pēc brauciena id tiek atrasti lietotāju dati.



4.9 att. Pievienošanās braucienam
detalizētais komponentu attēlojums

Braucienu dzēšana

Brauciena dzēšana, (Skat att. 4.10.) Tā kā lietotājs atrodas sava brauciena skatā, sistēma iegūst brauciena id. Pirmais process iegūst braucēju skatu attiecīgajam braucienam. Otrais process pārbauda vai braucienam ir pieteicies kāds lietotājs, ja braucienam ir kāds pieteicies, tad lietotājam dzēšanas poga nebūs pieejama. Tomēr, ja neviens nav pieteicies, tad lietotājam ir iespēja izdzēst braucienu. Trešajā procesā lietotājam ir jāapstiprina vai tiešām dzēst braucienu. Ja lietotājs apstiprina, tad brauciens tiek izdzēsts un lietotājs tiek pārvirzīts uz visu braucienu lapu.



4.10 att. Pievienošanās braucienam detalizētais komponentu attēlojums

5. DATU STRUKTŪRU APRAKSTS

Datubāze sastāv no 6 tabulām, kas sevī iekļauj informāciju par lietotājiem, braucieniem, transportlīdzekļiem, pilsētām un paroles maiņu

- 5.1. Tabula **User** iekļauj datus par lietotāju
- 5.2. Tabula **Car** iekļauj datus par transportlīdzekli
- 5.3. Tabula **Cities** iekļauj datus par pilsētām
- 5.4. Tabula **Rides** iekļauj datus par braucieniem
- 5.5. Tabula **User_rides** iekļauj datus par lietotāju braucieniem
- 5.6. Tabula **Password_change** iekļauj datus par paroles atjaunošanu

Lietotāja dati “User”

Lietotāja tabula. Šajā tabulā tiek saglabāti dati par Reģistrētu lietotāju.

5.1. tabula

Tabulas “User” struktūra

Nr.	Nosaukums	Tips	Garums	Piezīme
1.	ID	Autonumber	11	Lietotāja id
2.	USERNAME	varchar	40	Lietotājvārds
3.	NAME	varchar	40	Vārds
4.	SURNAME	varchar	40	Uzvārds
5.	PASSWORD	varchar	200	Parole
6.	EMAIL	varchar	50	Epasts
7.	REG_DATE	timestamp	-	Reģistrēšanās datums
8.	ACCESSTOKEN	varchar	255	Pieejas tokens
9.	AUTHKEY	varchar	255	Autorizācijas atslēga
10.	ROLE	integer	4	Loma
11.	BLOCKED	integer	1	Pazīme vai lietotājs ir blokēts
12.	PHONE	varchar	30	Telefona numurs

Transportlīdzekļa dati “Car”

Transportlīdzekļu tabula. Šajā tabulā tiek saglabāti dati par lietotāja pievienoto transportlīdzekli. Ārējā atslēga owner_id norāda uz lietotāju tabulu “user”

5.2. tabula

Tabulas “Car” struktūra

Nr.	Nosaukums	Tips	Garums	Piezīme
1.	ID	Autonumber	11	Transportlīdzekļa id
2.	reg_nr	varchar	11	Reģistrācijas numurs
3.	color	varchar	20	Krāsa
4.	make	varchar	20	Ražotājs
5.	model	varchar	20	Modelis
6.	seats	integer	1	Vietu skaits
7.	owner_id	integer	11	Īpašnieka id
8.	image_name	varchar	100	Bildes nosaukums

Pilsētu dati “Cities”

Pilsētu tabula. Šajā tabulā tiek saglabāti dati par Latvijas pilsētām, kuri pēc tam tiek izmantoti braucienu pievienošanai un meklēšanai.

5.3. tabula

Tabulas “Cities” struktūra

Nr.	Nosaukums	Tips	Garums	Piezīme
1.	name	varchar	22	Pilsētas nosaukums
2.	id	Autonumber	11	Pilsētas id

Brauciena dati “Rides”

Braucienų tabula. Šajā tabulā tiek saglabāti dati par lietotāja izveidotajiem. Ārējā atslēga CITY_FROM_ID un CITY_TO_ID norāda uz pilsētu tabulu “cities”. Ārējā atslēga DRIVER_ID norāda uz lietotāju tabulu “User”. Ārējā atslēga CAR_ID norāda uz transportlīdzekļu tabulu “Car”

5.4. tabula

Tabulas “Rides” struktūra

Nr.	Nosaukums	Tips	Garums	Piezīme
1.	ID	Autonumber	11	Brauciena id
2.	CITY_FROM_ID	integer	11	Pilsētas no id
3.	CITY_TO_ID	integer	11	Pilsētas uz id
4.	DRIVER_ID	integer	11	Vadītāja id
5.	DATE	datetime	-	Izbraukšanas laiks
6.	CAR_ID	integer	11	Transportlīdzekļa id
7.	NOTES	varchar	400	Piezīmes

Lietotāja braucienų dati “User_rides”

Lietotāju braucienų tabula. Šajā tabulā tiek saglabāti dati par braucieniem, kuriem lietotājs ir pieteicies. Ārējā atslēga USER_ID norāda uz lietotāju tabulu “User”. Ārējā atslēga RIDE_ID norāda uz braucienų tabulu “Rides”.

5.5. tabula

Tabulas “User_rides” struktūra

Nr.	Nosaukums	Tips	Garums	Piezīme
1.	USER_ID	integer	11	Lietotāja id
2.	RIDE_ID	integer	11	Brauciena id
3.	JOIN_DATE	datetime	-	Pievienošanās laiks
4.	CANCELLED	integer	1	Pazīme, vai atteicies no brauciena

Paroles maiņas informācijas dati “Password_change”

Paroles atjaunošanas tabula. Šajā tabulā tiek saglabāti dati par paroles atjaunošanu.

5.6. tabula

Tabulas “Password_change” struktūra

Nr.	Nosaukums	Tips	Garums	Piezīme
1.	ID	integer	11	Paroles atjaunošanas id
2.	EMAIL	varchar	50	Epasts
3.	PASSKEY	varchar	30	Uzģenerētā simbolu virkne
4.	TIME_CREATED	datetime	1	Laiks, kad izveidots pieprasījums

6. LIETOTĀJA CEĻVEDIS

6.1. Sistēmas prasības aparatūrai un programmatūrai

Lai sistēma spētu darboties, un to būtu iespējams apskatīt ir nepieciešams:

- Webservers, kurš nodrošina programmas darbību,
- Interneta pārlūkprogramma

Sistēmas minimālās prasības:

Operētājsistēma	Windows 7 vai jaunāka Mac OS X Yosemite 10.10 vai jaunāka. 64-bit Ubuntu 12.04+, Debian 8+, openSUSE 12.2++, vai Fedora Linux 17
Processors	Intel Pentium 4 vai labāks Athlon 64 vai labāks
Brīva vieta uz diska	Vismaz 5 GB
Operatīvā atmiņa	1 GB minimums, vismaz 2 GB ieteicams

6.2. Sistēmas instalācija un palaišana

- 6.2.1. Izveidot Virtuālo serveri, piemēram, XAMPP
- 6.2.2. Lejupielādēt programmas failus, vai iegūt tos izmantojot git programmatūru
- 6.2.3. Novietot programmas failus mapē htdocs
- 6.2.4. Kādā no pārlūkprogrammām ievadīt (localhost/phpmyadmin/)
- 6.2.5. Importēt datubāzes kopbrauksana.sql failu no programmas failu mapes
- 6.2.6. Pārlūkprogrammā ierakstīt “http://localhost/d41-RaimondsLagzdins-kopbrauksana/web/”, lai pārbaudītu vai programma darbojās

6.3. Programmas apraksts

Atverot mājaslapu pirmais, ko lietotājs ieraudzīs ir sākumlapa. Tajā ir izvēlne, ar vairākām pogām. Un divas pogas ekrāna vidū “Atrodi braucienu” un “Pievieno braucienu” (Skat 6.1. att.)



6.1 att. Sākuma ekrāna attēls

Lai atrastu braucienu, lietotājam ir pieejams šāds logs. Šeit lietotājs var ievadīt pilsētu no kuras vēlas doties, uz kuru vēlas doties, kā arī vēlamo dienu. Bet ir arī iespēja atstāt šos visus laukus tukšus un apskatīt visus pieejamos braucienus (Skat 6.2. att.)

6.2 att. Braucienu meklēšanas attēls

Šādi izskatās skats ar visiem pieejamajiem braucieniem. Šeit lietotājs var sakārot braucienus pēc izbraukšanas laika gan augošā secībā, gan dilstošā secībā. Kā arī lietotājam ir iespēja aiziet uz katra brauciena lapu, lai iegūtu vairāk informāciju. (Skat 6.3. att.)

Kopbraukšana						
Sākums Saziņa Mani braucieni Mani transportlīdzekļi Atslēgties (vadītājs1)						
Braucieni						
Piedāvā braucienu Atrast citu						
Tiek rādīti ieraksti 1-6 no 6.						
#	Pilsēta no	Pilsēta uz	Vārds	Izbraukšanas laiks	Vietu skaits	
1	Rīga	Ogres pilsēta	vadītājs3	2020-06-06 02:50:00	3	Apskatīt
2	Jūrmala	Rīga	vadītājs3	2020-06-10 13:45:00	3	Apskatīt
3	Liepāja	Rīga	vadītājs4	2020-06-11 04:50:00	3	Apskatīt
4	Rīga	Daugavpils	vadītājs4	2020-06-02 01:50:00	3	Apskatīt
5	Jūrmala	Rīga	vadītājs4	2020-06-04 14:25:00	3	Apskatīt
6	Siguldas pilsēta	Rīga	vadītājs1	2020-06-09 13:45:00	3	Apskatīt

6.3 att. Braucienų meklēšanas attēls


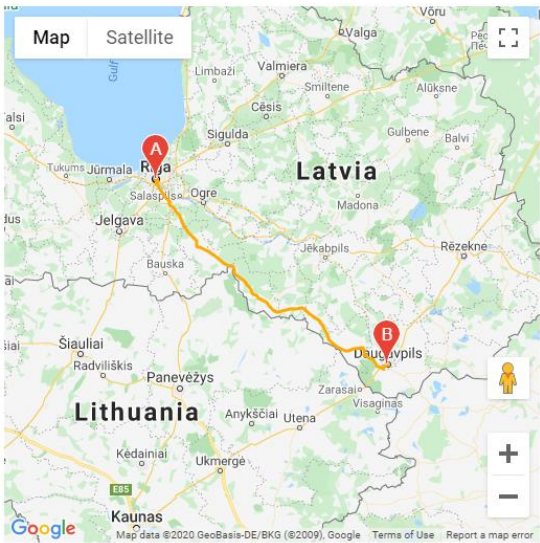
Šeit iespējams apskatīt kā izskatās brauciena lapa, un iespējams pievienoties braucienam. Tiek parādīts kartē aptuvenais ceļš kas tiks mērots, kā arī transportlīdzeklis ar kuru notiks brauciens. (Skat 6.4. att.)

Kopbraukšana

Sākums Saziņa Mani braucieni Mani transportlīdzekļi Atslēgties (vadītajs1)

Pieteikties

Pilsēta no	Rīga
Pilsēta uz	Daugavpils
Epasts	vaditajs4@vaditajs4.lv
Izbraukšanas laiks	2020-06-02 01:50:00
Piezīmes	Izbraukšana no centra

6.4 att. Brauciena lapas attēls

6.4. Testa piemērs

Šajā testa piemērā aplūkosim kā pievienot braucienu, un kā var pieteikties braucienam. No sākuma būs brauciena pievienošana.



Sākumā jāatver brauciena pievienošana. Šajā piemērā izveidosim braucienu no Rīgas uz Ventspili. Lai to izdarītu sākumā jāaizpilda visi lauki. (Skat 6.5. att.).

Pilsēta no

Pilsēta uz

Mašīna

Izbraukšanas laiks

		06-06-2020 10:30
---	---	------------------

Piezīmes

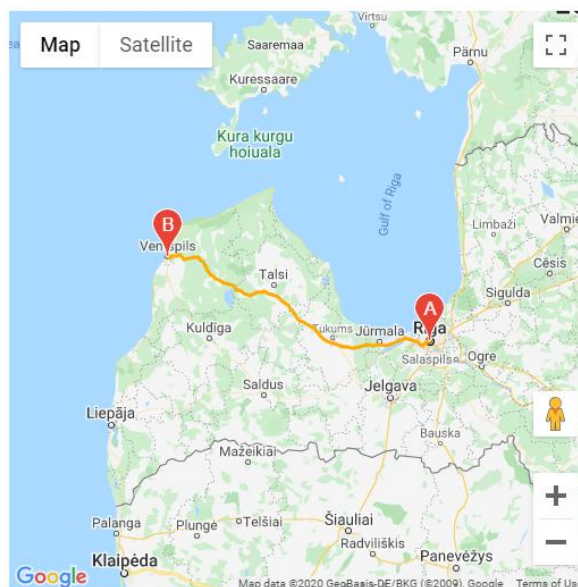
Pievienot

6.5 att. Brauciena pievienošanas formas attēls

Kad tas ir izdarīts, un brauciens ir izveidots. Tagad ir redzams brauciena skats. Un ja gadījumā ir radusies kāda kļūda, tad vēl ir iespēja šo braucienu izdzēst.(Skat 6.6. att.)

Dzēst braucienu

Pilsēta no	Rīga
Pilsēta uz	Ventspils
Epasts	vaditajs1@vaditajs1.lv
Izbraukšanas laiks	2020-06-06 10:30:00
Piezīmes	Izbraukšana no spices



6.6 att. Brauciena informācijas lapas attēls

Lai atrastu šo braucienu, un tam varētu pieteikties. Aizpildīsim meklēšanas formas laukus tā, lai atrastu iepriekš izveidoto braucienu(Skat 6.7. att.)

Pilsēta no

Rīga

Pilsēta uz

Ventspils

Izbraukšanas laiks

Atrast braucienu

6.7 att. Brauciena meklēšanas formas attēls

Kad vēlmais brauciens ir atrasts var spiest pogu apskatīt, lai redzētu vairāk informāciju par braucienu, un tam varētu pievienoties.(Skat 6.8. att.)

Braucieni

[Piedāvā braucienu](#) [Atrast citu](#)

Tiek rādīti ieraksti 1-1 no 1.

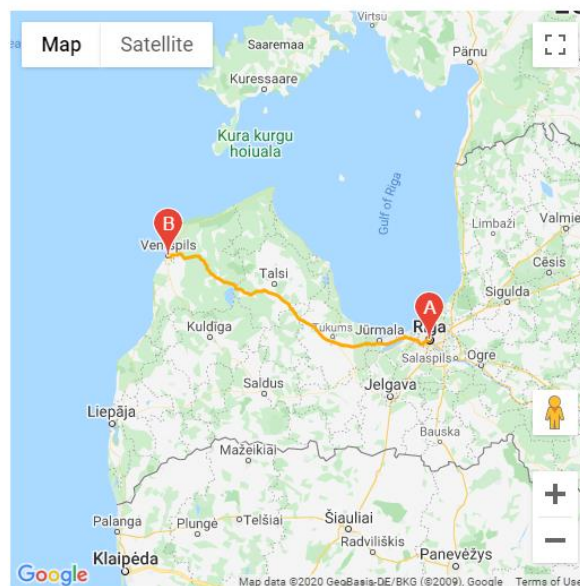
#	Pilsēta no	Pilsēta uz	Vārds	Izbraukšanas laiks	Vietu skaits	
1	Rīga	Ventspils	vaditajs1	2020-06-06 10:30:00	3	Apskatīt

6.8 att. Atrasto braucienu tabulas attēls

Ja redzamā brauciena informācija apmierina, tad var spiest pogu pieteikties. (Skat 6.9. att.)

[Pieteikties](#)

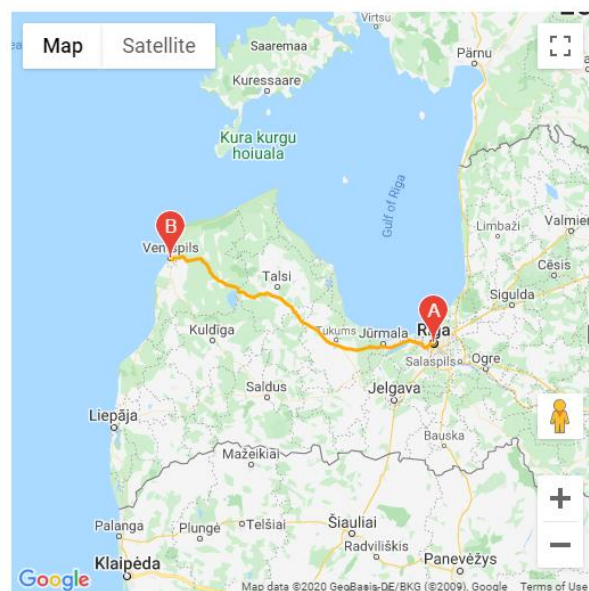
Pilsēta no	Rīga
Pilsēta uz	Ventspils
Epasts	vaditajs1@vaditajs1.lv
Izbraukšanas laiks	2020-06-06 10:30:00
Piezīmes	Izbraukšana no spices



6.9. att. Izvēlēta brauciena lapas attēls

Ja ir nospiesta poga pieteikties, tad skats tiek atjaunināts un ir vēl iespēja atteikties no brauciena. (Skat 6.10. att.)

Atteikties	
Pilsēta no	Rīga
Pilsēta uz	Ventspils
Epasts	vaditajs1@vaditajs1.lv
Izbraukšanas laiks	2020-06-06 10:30:00
Piezīmes	Izbraukšana no spices



6.10. att. Izvēlētā brauciena lapas attēls pēc pievienošanās braucienam

7. NOBEIGUMS

Kvalifikācijas darbā izvirzītais mērķis ir gandrīz sasniegts. Ir izveidota funkcionējoša mājaslapa, ko lietotāji varētu arī veiksmīgi sākt izmantot. Mājaslapa atvieglos starppilsētu transporta atrašanu tiem cilvēkiem, kas iepriekš izmantojuši dažādās sociālo tīklu grupas, lai atrastu sev vēlamu transportu, to visu apvienojot vienā vietā.

Tomēr šīs sistēmas darbība ir ļoti atkarīga no lietotāju aktivitātes, tāpēc labākus rezultātus varēs iegūt tikai pēc kāda laika.

Izstrādājot šo kvalifikācijas darbu tika iegūtas ļoti daudz jaunu zināšanas un prasmes. Tika apgūts jauns php programmēšanas valodas ietvars “Yii2” Kā arī tika apgūts programmēšanas veids MVC. Sistēmas izstrāde nebija viegla, tā aizņēma diezgan daudz laika, jo bija daudz nezināmā ko vajadzēja apgūt

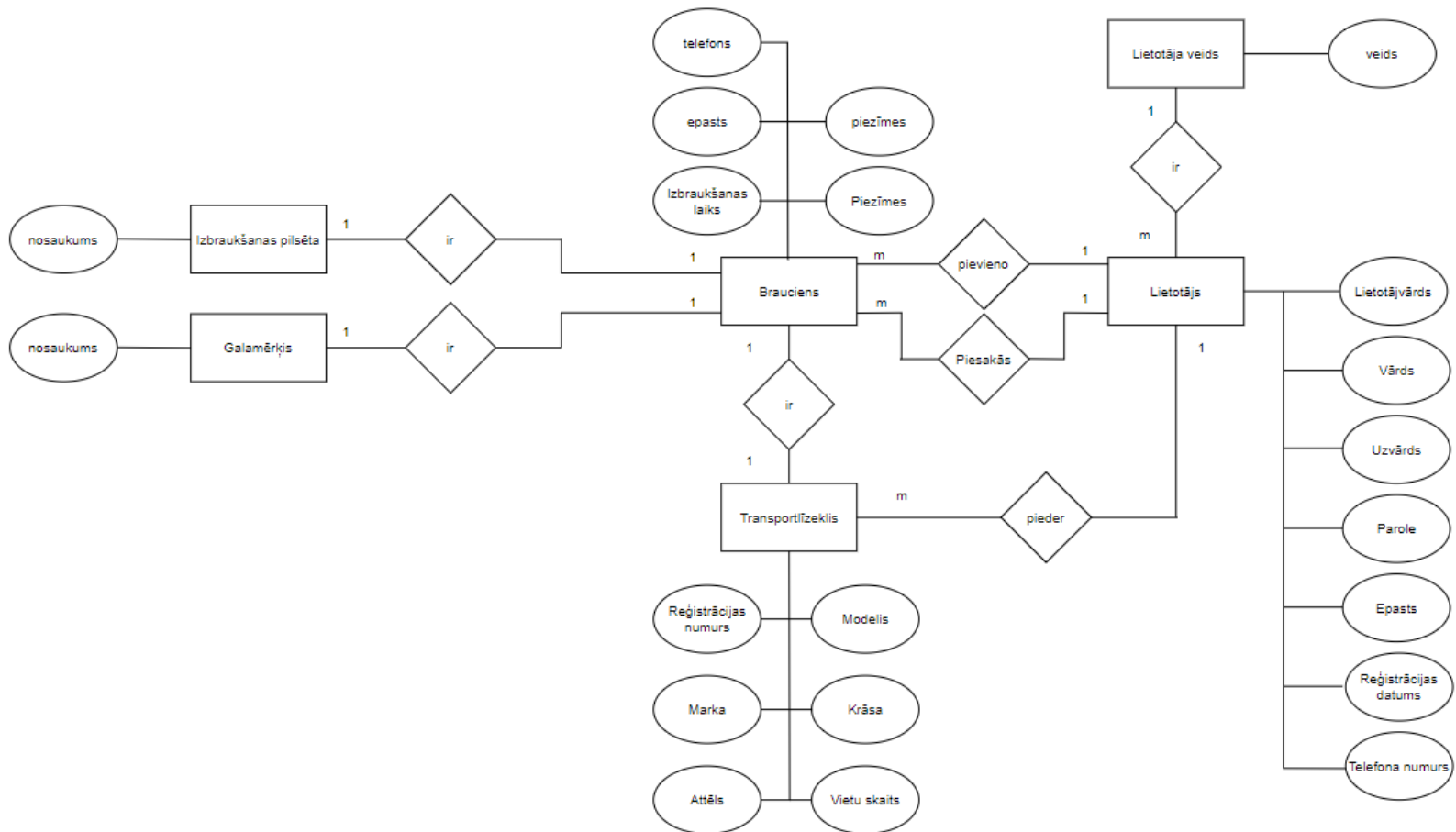
8. INFORMĀCIJAS AVOTI

1. Oficiālā Yii2 dokumentācija <https://www.yiiframework.com/doc/guide/2.0/en> Materiāls apskatīts internetā(05.02.2020)
2. jQuery iekš Yii2 <https://github.com/yiisoft/yii2-jui> Materiāls apskatīts internetā(15.04.2020)
3. Pamācība kā izveidot lietotāju reģistrāciju <https://www.youtube.com/watch?v=oU-4tHaommk> Materiāls apskatīts internetā (15.02.2020)
4. Pamācība kā pievienot savu pogu iekš grid-view <https://stackoverflow.com/questions/39716563/add-a-button-to-grid-view-in-yii2> Materiāls apskatīts internetā(29.04.2020)
5. Date time picker izmantošana <https://github.com/kartik-v/yii2-widget-datetimepicker> Materiāls apskatīts internetā (29.04.2020)
6. Failu augšupielāde <https://www.yiiframework.com/doc/guide/2.0/en/input-file-upload> Materiāls apskatīts (20.04.2020)
7. Skaistāks failu pievienošanas logs <https://github.com/kartik-v/yii2-widget-fileinput> Materiāls apskatīts (20.04.2020)
8. Google maps apstrāde <https://github.com/2amigos/yii2-google-maps-library> Materiāls apskatīts (07.05.2020)
9. Labāks dropdown lists <https://github.com/kartik-v/yii2-widget-select2> Materiāls apskatīts (07.05.2020)
10. Koordināšu iegūšana no pilsētas nosaukuma <https://github.com/2amigos/yii2-google-maps-library/issues/54> Materiāls apskatīts(09.05.2020.)

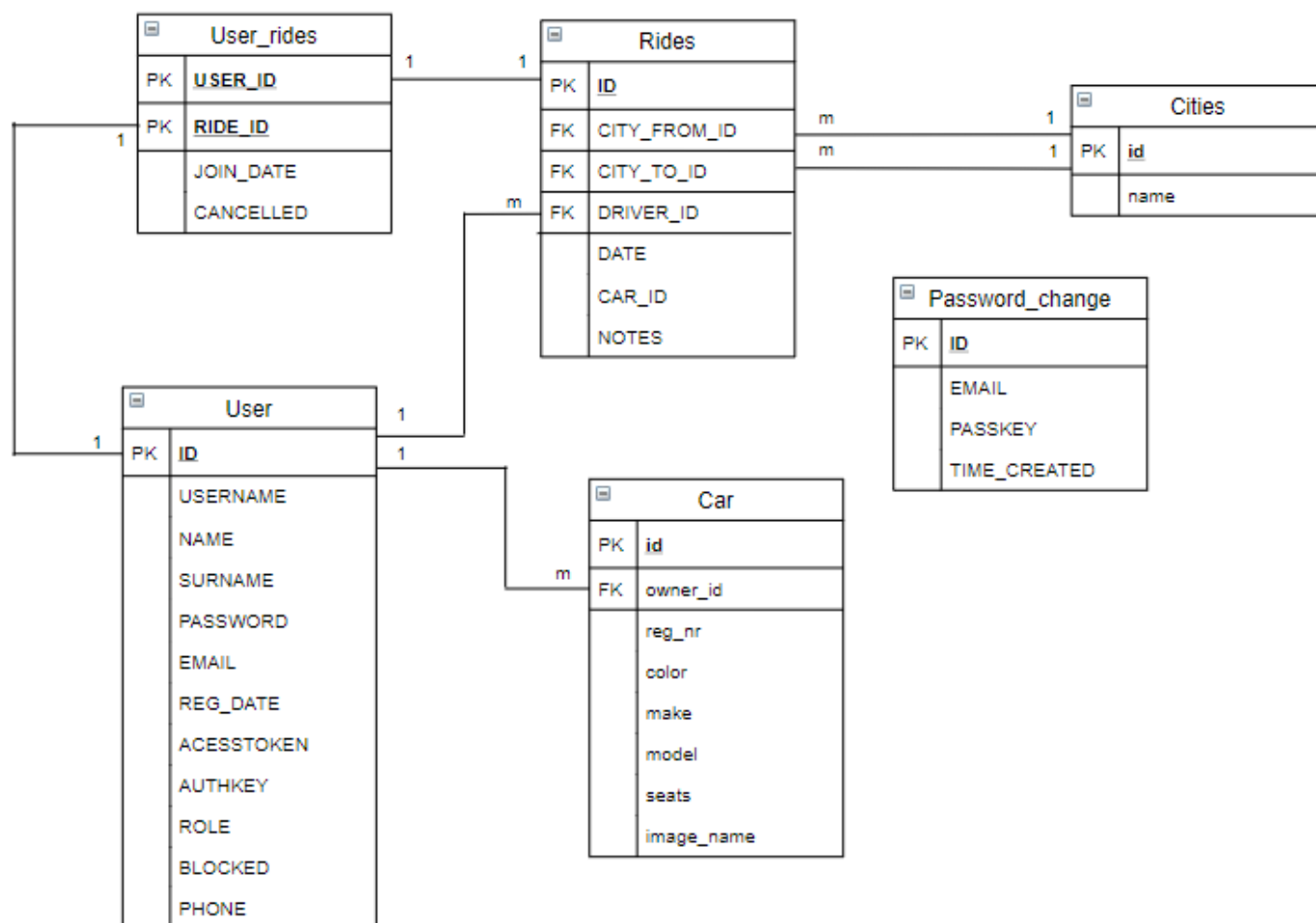
PIELIKUMI

1. Pielikums

Sistēmas ER modelis



Tabulu relāciju shēma



Programmas pirmkods**AdminController.php**

<?php

namespace app\controllers;

use Yii;

use yii\web\Controller;

use yii\data\ActiveDataProvider;

use app\models\USER;

use yii\filters\AccessControl;

use yii\filters\VerbFilter;

class AdminController extends Controller

{

public function behaviors()

{

return [

'access' => [

'class' => AccessControl::className(),

'only' => ['index','update','delete','view'],

'rules' => [

[

'actions' => ['index','update','delete','view'],

'allow' => true,

'roles' => ['@'],

'matchCallback' => function (\$rule, \$action) {

return User::isUserAdmin(Yii::\$app->user->identity->USERNAME);

}

],

],

],

'verbs' => [

'class' => VerbFilter::className(),

'actions' => [

'logout' => ['post'],

],

],

];

}

public function actionIndex()

{

\$dataProvider = new ActiveDataProvider([

```

        'query' => USER::find(),
    ]);

    return $this->render('index', [
        'dataProvider' => $dataProvider,
    ]);
}

public function actionUpdate($id)
{
    $model = $this->findModel($id);

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['view', 'id' => $model->ID]);
    }
    return $this->render('update', [
        'model' => $model,
        'btnText' => 'aktualizēt'
    ]);
}

protected function findModel($id)
{
    if (($model = USER::findOne($id)) !== null) {
        return $model;
    }

    throw new NotFoundHttpException('The requested page does not exist.');
```

```

}

public function actionView($id)
{
    return $this->render('view', [
        'model' => $this->findModel($id),
    ]);
}

public function actionDelete($id)
{
    $this->findModel($id)->delete();

    return $this->redirect(['index']);
}

```

```

        public function actionBlock($id)
        {
            $model = $this->findModel($id);
            $model->BLOCKED = 1;
            $model->save();

            return $this->render('view', [
                'model' => $model
            ]);
        }
        public function actionUnblock($id)
        {
            $model = $this->findModel($id);
            $model->BLOCKED = 0;
            $model->save();

            return $this->render('view', [
                'model' => $model
            ]);
        }
    }
}

```

CarController.php

```
<?php
```

```
namespace app\controllers;
```

```

use app\models\UploadForm;
use app\models\Images;
use Yii;
use app\models\Car;
use yii\data\ActiveDataProvider;
use yii\filters\AccessControl;
use yii\web\Controller;
use yii\web\NotFoundHttpException;
use yii\filters\VerbFilter;
use yii\web\UploadedFile;

```

```
/**
```

```
 * CarController implements the CRUD actions for Car model.
```

```
*/
```

```
class CarController extends Controller
```

```
{
```

```
/**
```

```
 * { @inheritdoc }
```

```
*/
```

```
    public function behaviors()
```

```

{
    return [
        'access' => [
            'class' => AccessControl::className(),
            'only' => ['create','index','update','view'],
            'rules' => [
                [
                    'actions' => ['create','index','update','view'],
                    'allow' => true,
                    'roles' => ['@'],
                    'matchCallback' => function ($rule, $action) {
                        return !Yii::$app->user->isGuest;
                    }
                ],
            ],
        ],
        'verbs' => [
            'class' => VerbFilter::className(),
            'actions' => [
                'delete' => ['POST'],
            ],
        ],
    ];
}

/**
 * Lists all Car models.
 * @return mixed
 */
public function actionIndex()
{
    $dataProvider = new ActiveDataProvider([
        'query' => Car::find()->where(['owner_id' => Yii::$app->user->getId()]),
        'pagination' => [
            'pageSize' => 10,
        ],
    ]);

    return $this->render('index', [
        'dataProvider' => $dataProvider,
    ]);
}

/**
 * Displays a single Car model.
 * @param integer $id

```

```

* @return mixed
* @throws NotFoundHttpException if the model cannot be found
*/
public function actionView($id)
{
    return $this->render('view', [
        'model' => $this->findModel($id),
    ]);
}

/**
 * Creates a new Car model.
 * If creation is successful, the browser will be redirected to the 'view' page.
 * @return mixed
 */
public function actionCreate()
{
    $model = new Car();
    $upload = new UploadForm();

    $timestamp = Yii::$app->formatter->asTimestamp(date('Y-m-d h:i:s'));
    if ($model->load(Yii::$app->request->post())) {

        $model->owner_id = Yii::$app->user->getId();

        if($upload->imageFile = UploadedFile::getInstance($upload, 'imageFile')){
            $upload->imageName = $upload->imageFile->baseName . $timestamp . '.' . $upload-
>imageFile->extension;
            $model->image_name = $upload->imageName;

            $upload->upload();
        }
        $model->save();

        return $this->redirect(['view', 'id' => $model->id]);
    }

    return $this->render('create', [
        'model' => $model,
        'upload' => $upload,
    ]);
}

/**
 * Updates an existing Car model.
 * If update is successful, the browser will be redirected to the 'view' page.

```

```

* @param integer $id
* @return mixed
* @throws NotFoundHttpException if the model cannot be found
*/
public function actionUpdate($id)
{
    $model = $this->findModel($id);
    $upload = new UploadForm();
    $timestamp = Yii::$app->formatter->asTimestamp(date('Y-m-d h:i:s'));

    if ($model->load(Yii::$app->request->post())) {
        $upload->imageFile = UploadedFile::getInstance($upload, 'imageFile');
        if(isset($upload->imageFile->baseName)) {
            $upload->imageName = $upload->imageFile->baseName . $timestamp . '.' . $upload-
>imageFile->extension;
            $model->image_name = $upload->imageName;
        }
        $model->save();
        return $this->redirect(['view', 'id' => $model->id]);
    }

    return $this->render('update', [
        'model' => $model,
        'upload' => $upload,
    ]);
}

/**
 * Deletes an existing Car model.
 * If deletion is successful, the browser will be redirected to the 'index' page.
 * @param integer $id
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionDelete($id)
{
    $this->findModel($id)->delete();

    return $this->redirect(['index']);
}

/**
 * Finds the Car model based on its primary key value.
 * If the model is not found, a 404 HTTP exception will be thrown.
 * @param integer $id
 * @return Car the loaded model

```

```

    * @throws NotFoundHttpException if the model cannot be found
    */
    protected function findModel($id)
    {
        if (($model = Car::findOne($id)) !== null) {
            return $model;
        }

        throw new NotFoundHttpException('The requested page does not exist.');
    }
}

```

FileController.php

```

<?php
namespace app\controllers;

use Yii;
use yii\web\Controller;
use app\models\UploadForm;
use yii\web\UploadedFile;

class FileController extends Controller
{
    public function actionUpload()
    {
        $model = new UploadForm();

        if (Yii::$app->request->isPost) {
            $model->imageFile = UploadedFile::getInstance($model, 'imageFile');

            if ($model->upload()) {
                // file is uploaded successfully
                return;
            }
        }

        return $this->render('upload', ['model' => $model]);
    }
}

```

PassChController.php

```

<?php

namespace app\controllers;

use app\models\USER;
use Yii;
use app\models\PassCh;

```



```

use yii\data\ActiveDataProvider;
use yii\web\Controller;
use yii\web\NotFoundHttpException;
use yii\filters\VerbFilter;
//use yii\base\Security;

/**
 * PassChController implements the CRUD actions for PassCh model.
 */
class PassChController extends Controller
{
    /**
     * {@inheritdoc}
     */
    public function behaviors()
    {
        return [
            'verbs' => [
                'class' => VerbFilter::className(),
                'actions' => [
                    'delete' => ['POST'],
                ],
            ],
        ];
    }

    public function actionRecover($passkey)
    {
        $model = PassCh::findByPassKey($passkey);
        $user = USER::findByEmail($model->EMAIL);

        if (!$model->validatePassKey()){
            Yii::$app->session->setFlash('linkExpired');
        }
        if ($user->load(Yii::$app->request->post())) {
            $user->updatePassword();
            Yii::$app->session->setFlash('successfulPasswordChange');
        }
        $user->PASSWORD = "";
        return $this->render('recover', [
            'model' => $user,
        ]);
    }

    /**
     * Creates a new PassCh model.

```

```

* If creation is successful, the browser will be redirected to the 'view' page.
* @return mixed
*/
public function actionCreate()
{
    $model = new PassCh();

    if ($model->load(Yii::$app->request->post())) {

        $model->PASSKEY = Yii::$app->security->generateRandomString(30);

        $emailBody = 'http://raimondsl.id.lv/pass-ch/recover?passkey='.$model->PASSKEY;

        $model->save();
        if($this->checkUser($model->EMAIL)){
            $this->contact($model->EMAIL, $emailBody);
        }

        Yii::$app->session->setFlash('passwordChangeRequest');
    }

    return $this->render('create', [
        'model' => $model,
    ]);
}

/**
 * Finds the PassCh model based on its primary key value.
 * If the model is not found, a 404 HTTP exception will be thrown.
 * @param integer $id
 * @return PassCh the loaded model
 * @throws NotFoundHttpException if the model cannot be found
 */
protected function findModel($id)
{
    if (($model = PassCh::findOne($id)) !== null) {
        return $model;
    }

    throw new NotFoundHttpException('The requested page does not exist.');
```

```

}

public function contact($email, $body)
{
    Yii::$app->mailer->compose()

```

```

        ->setTo($email)
        ->setFrom(Yii::$app->params['senderEmail'])
        ->setReplyTo(Yii::$app->params['senderEmail'])
        ->setSubject('Paroles atjaunošana')
        ->setTextBody($body)
        ->send();

    return true;
}
public function checkUser($email)
{
    $userCount = USER::find()->where(['EMAIL' => $email])->count();
    return $userCount;
}
}
RegisterController.php
<?php

```

```

namespace app\controllers;

```

```

use Yii;
use app\models\USER;
use yii\data\ActiveDataProvider;
use yii\web\Controller;
use yii\web\NotFoundHttpException;
use yii\filters\VerbFilter;

```

```

/**
 * RegisterController implements the CRUD actions for USER model.
 */
class RegisterController extends Controller
{
    /**
     * {@inheritdoc}
     */
    public function behaviors()
    {
        return [
            'verbs' => [
                'class' => VerbFilter::className(),
                'actions' => [
                    'delete' => ['POST'],
                ],
            ],
        ];
    }
}

```

```

/**
 * Lists all USER models.
 * @return mixed
 */
// public function actionIndex()
// {
//     $dataProvider = new ActiveDataProvider([
//         'query' => USER::find(),
//     ]);
//
//     return $this->render('index', [
//         'dataProvider' => $dataProvider,
//     ]);
// }

/**
 * Displays a single USER model.
 * @param integer $id
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionView($id)
{
    return $this->render('view', [
        'model' => $this->findModel($id),
    ]);
}

/**
 * Creates a new USER model.
 * If creation is successful, the browser will be redirected to the 'view' page.
 * @return mixed      actionCreate
 */
public function actionIndex()
{
    $model = new USER();
    $btnText = 'Reģistrēties';

    if ($model->load(Yii::$app->request->post()) ) {
        $model->REG_DATE = date("Y-m-d");

        if($model->validate()) {
            // paroles saglabāšana hash formātā

```

```

        $model->PASSWORD = Yii::$app->getSecurity()->generatePasswordHash($model-
>PASSWORD);
        $model->save();

        return $this->redirect(['site/login']);
    }
    return $this->render('create', [
        'model' => $model,
        'btnText' => $btnText
    ]);
}
return $this->render('create', [
    'model' => $model,
    'btnText' => $btnText
]);
}

/**
 * Updates an existing USER model.
 * If update is successful, the browser will be redirected to the 'view' page.
 * @param integer $id
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionUpdate($id)
{
    $session = Yii::$app->session;

    $model = $this->findModel($id);
    $model->PASSWORD = null;

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        $model->PASSWORD = Yii::$app->getSecurity()->generatePasswordHash($model-
>PASSWORD);

        $model->save();

        return $this->redirect(['view', 'id' => $model->ID]);
    }
    $btnText = 'Aktualizēt';
    return $this->render('update', [
        'model' => $model,
        'btnText' => $btnText
    ]);
}

```

```

/**
 * Deletes an existing USER model.
 * If deletion is successful, the browser will be redirected to the 'index' page.
 * @param integer $id
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionDelete($id)
{
    $this->findModel($id)->delete();

    return $this->redirect(['index']);
}

/**
 * Finds the USER model based on its primary key value.
 * If the model is not found, a 404 HTTP exception will be thrown.
 * @param integer $id
 * @return USER the loaded model
 * @throws NotFoundHttpException if the model cannot be found
 */
protected function findModel($id)
{
    if (($model = USER::findOne($id)) !== null) {
        return $model;
    }

    throw new NotFoundHttpException('The requested page does not exist.');
}
}

```

RidesController.php

```

<?php

```

```

namespace app\controllers;

use app\models\USER;
use app\models\UserRides;
use Yii;
use app\models\Rides;
use app\models\RidesSearch;
use yii\filters\AccessControl;
use yii\web\Controller;
use yii\web\NotFoundHttpException;
use yii\filters\VerbFilter;
use yii\helpers\ArrayHelper;

```

```

use app\models\Cities;
use app\models\Car;
/**
 * RidesController implements the CRUD actions for Rides model.
 */
class RidesController extends Controller
{
    /**
     * {@inheritdoc}
     */
    public function behaviors()
    {
        return [
            'access' => [
                'class' => AccessControl::className(),
                'only' => ['create', 'userrides','join'],
                'rules' => [
                    [
                        'actions' => ['create','userrides', 'join', 'view'],
                        'allow' => true,
                        'roles' => ['@'],
                        'matchCallback' => function ($rule, $action) {
                            return !Yii::$app->user->isGuest;
                        }
                    ],
                ],
            ],
            'verbs' => [
                'class' => VerbFilter::className(),
                'actions' => [
                    'logout' => ['post'],
                ],
            ],
        ];
    }

    /**
     * metode atgriež, vai ir vadītājs
     */
    public function isDriver($rideId)
    {
        $rides = Rides::find()->where(['ID' => $rideId])->one();

        if(Yii::$app->user->getId() == $rides->DRIVER_ID){
            return 1;
        }else{

```

```

        return 0;
    }
}

/**
 * Opens rides search form
 * @return mixed
 */
public function actionSearch()
{
    $searchModel = new RidesSearch();
    $dataProvider = $searchModel->search(Yii::$app->request->queryParams);

    $cityArray = ArrayHelper::map(Cities::find()->orderBy("name ASC")->all(), 'id', 'name');

    return $this->render('_search', [
        'model' => $searchModel,
        'dataProvider' => $dataProvider,
        'cityArray' => $cityArray,
    ]);
}

public function actionIndex()
{
    $searchModel = new RidesSearch();
    $filter = Yii::$app->request->post();

    if(!isset(Yii::$app->session['_RideFilter']) || Yii::$app->request->post()) {
        Yii::$app->session['_RideFilter'] = $filter;
    }

    $dataProvider = $searchModel->search(Yii::$app->session['_RideFilter']);

    return $this->render('index', [
        'searchModel' => $searchModel,
        'dataProvider' => $dataProvider,
        'filter' => $filter,
    ]);
}

public function actionUserrides()
{
    $searchModel = new RidesSearch();

```



```

        $dataProvider = $searchModel->searchUserRides();
        return $this->render('myrides', [
            'searchModel' => $searchModel,
            'dataProvider' => $dataProvider,
        ]);
    }

    /**
     * Displays a single Rides model.
     * @param integer $id
     * @return mixed
     * @throws NotFoundHttpException if the model cannot be found
     */
    public function actionView($id)
    {
        $model = $this->findModel($id);
        $car = $model->cAR;

        return $this->render('view', [
            'model' => $model,
            'car' => $car,
        ]);
    }

    /**
     * Creates a new Rides model.
     * If creation is successful, the browser will be redirected to the 'view' page.
     * @return mixed
     */
    public function actionCreate()
    {
        $model = new Rides();
        $cities = ArrayHelper::map(Cities::find()->orderBy("name ASC")->all(), 'id', 'name');
        $cars = ArrayHelper::map(Car::find()->where(['owner_id' => Yii::$app->user->getId()])->all(), 'id', 'reg_nr');

        if(empty($cars)){
            Yii::$app->session->setFlash('needCar');
            return $this->redirect(['car/create']);
        }

        if ($model->load(Yii::$app->request->post())) {
            $model->DRIVER_ID = Yii::$app->user->getId();
            $model->DATE = date("Y-m-d H:i", strtotime($model->DATE));
        }
    }

```

```

        $model->save();
        return $this->redirect(['view', 'id' => $model->ID]);
    }

    return $this->render('create', [
        'model' => $model,
        'cities' => $cities,
        'cars' => $cars
    ]);
}

/**
 * Pievieno cilvēku braucienam
 * join_date nenorādu, jo datubāzē tiek izmantota default vērtība
 */
public function actionJoin($id)
{
    $model = new UserRides();

    $rideModel = new Rides();

    $riderModel = new USER();
    $driverModel = new USER();

    $model->RIDE_ID = $id;
    $model->USER_ID = Yii::$app->user->getId();

    $model->save();

    $rideModel = $rideModel->findById($id);
    $riderModel = $riderModel->findById(Yii::$app->user->getId());
    $driverModel = $driverModel->findById($rideModel->DRIVER_ID);

    $emailBody = $this->setEmailBody($rideModel, $riderModel);
    $this->contact($driverModel->EMAIL, $emailBody);

    return $this->redirect(['view', 'id' => $model->RIDE_ID]);
}

/**
 * Updates an existing Rides model.
 * If update is successful, the browser will be redirected to the 'view' page.
 * @param integer $id
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found

```

```

*/
public function actionUpdate($id)
{
    $model = $this->findModel($id);

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['view', 'id' => $model->ID]);
    }

    return $this->render('update', [
        'model' => $model,
    ]);
}

/**
 * Deletes an existing Rides model.
 * If deletion is successful, the browser will be redirected to the 'index' page.
 * @param integer $id
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionDelete($id)
{
    $this->findModel($id)->delete();

    return $this->redirect(['index']);
}

/**
 * Finds the Rides model based on its primary key value.
 * If the model is not found, a 404 HTTP exception will be thrown.
 * @param integer $id
 * @return Rides the loaded model
 * @throws NotFoundHttpException if the model cannot be found
 */
protected function findModel($id)
{
    if (($model = Rides::findOne($id)) !== null) {
        return $model;
    }

    throw new NotFoundHttpException('The requested page does not exist.');
```

```

}

public function contact($email, $body)

```

```

{
    Yii::$app->mailer->compose()
        ->setTo($email)
        ->setFrom(Yii::$app->params['senderEmail'])
        ->setReplyTo(Yii::$app->params['senderEmail'])
        ->setSubject('Jauna informācija par braucienu!')
        ->setTextBody($body)
        ->send();
    return true;
}
public function setEmailBody(Rides $rideModel, USER $userModel)
{
    $emailBody = 'Tavam braucienam no ' . $rideModel->cITYFROM->name . ' uz ' .
    $rideModel->cITYTO->name . ' ir jauns līdzbraucējs!
    Ja vēlies sakontaktēties, tava līdzbraucēja telefona numurs - ' . $userModel->PHONE;
    return $emailBody;
}
}
SiteController.php
<?php

```

```

namespace app\controllers;

use Yii;
use yii\filters\AccessControl;
use yii\web\Controller;
use yii\web\Response;
use yii\filters\VerbFilter;
use app\models\LoginForm;
use app\models\ContactForm;
use app\models\USER;
class SiteController extends Controller
{
    /**
     * { @inheritdoc }
     */
    public function behaviors()
    {
        return [
            'access' => [
                'class' => AccessControl::className(),
                'only' => ['logout'],
                'rules' => [
                    [
                        'actions' => ['logout'],
                        'allow' => true,

```

```

        'roles' => ['@'],
    ],
],
'verbs' => [
    'class' => VerbFilter::className(),
    'actions' => [
        'logout' => ['post'],
    ],
],
];
}

/**
 * { @inheritdoc }
 */
public function actions()
{
    return [
        'error' => [
            'class' => 'yii\web\ErrorAction',
        ],
        'captcha' => [
            'class' => 'yii\captcha\CaptchaAction',
            'fixedVerifyCode' => YII_ENV_TEST ? 'testme' : null,
        ],
    ];
}

/**
 * Displays homepage.
 *
 * @return string
 */
public function actionIndex()
{
    return $this->render('index');
}

/**
 * Login action.
 *
 * @return Response|string
 */
public function actionLogin()
{

```

```

        if (!Yii::$app->user->isGuest) {
            return $this->goHome();
        }

        $model = new LoginForm();
        if ($model->load(Yii::$app->request->post()) && $model->login()) {
            return $this->goBack();
        }

        $model->password = "";

        return $this->render('login', [
            'model' => $model,
        ]);
    }

    /**
     * Logout action.
     *
     * @return Response
     */
    public function actionLogout()
    {
        Yii::$app->user->logout();

        return $this->goHome();
    }

    /**
     * Displays contact page.
     *
     * @return Response|string
     */
    public function actionContact()
    {
        $model = new ContactForm();
        if ($model->load(Yii::$app->request->post()) && $model->contact($model->email)) {
            Yii::$app->session->setFlash('contactFormSubmitted');

            return $this->refresh();
        }
        return $this->render('contact', [
            'model' => $model,
        ]);
    }
}

```

```

/**
 * Displays about page.
 *
 * @return string
 */
public function actionAbout()
{
    return $this->render('about');
}

public function actionRegister()
{
    $model = new USER();

    if (!Yii::$app->user->isGuest) {
        return $this->goHome();
    }

    if ($model->load(Yii::$app->request->post()) ) {

        $model->REG_DATE = date('Y-m-d H:i:s');
        if($model->validate()) {

            // paroles saglabāšana hash formātā
            $this->hashPassword($model);

            if($model->save()){
                return $this->redirect(['login']);
            }

        }

        $model->PASSWORD = null;
        return $this->render('register', [
            'model' => $model,
        ]);
    }

    public function hashPassword($model)
    {
        $model->PASSWORD = password_hash($model->PASSWORD,
        PASSWORD_ARGON2I);
        $model->AUTHKEY = md5(random_bytes(5));
    }
}

```

```
        $model->ACCESSTOKEN =  
password_hash(random_bytes(10),PASSWORD_DEFAULT);
```

```
        return $model;  
    }  
}
```

UserRidesController.php

```
<?php
```

```
namespace app\controllers;
```

```
use Yii;  
use app\models\UserRides;  
use yii\data\ActiveDataProvider;  
use yii\web\Controller;  
use yii\web\NotFoundHttpException;  
use yii\filters\VerbFilter;
```

```
/**
```

```
 * UserRidesController implements the CRUD actions for UserRides model.
```

```
*/
```

```
class UserRidesController extends Controller
```

```
{
```

```
    /**
```

```
     * { @inheritdoc }
```

```
    */
```

```
    public function behaviors()
```

```
    {
```

```
        return [
```

```
            'verbs' => [
```

```
                'class' => VerbFilter::className(),
```

```
                'actions' => [
```

```
                    'delete' => ['POST'],
```

```
                ],
```

```
            ],
```

```
        ];
```

```
    }
```

```
    /**
```

```
     * Lists all UserRides models.
```

```
     * @return mixed
```

```
    */
```

```
    public function actionIndex()
```

```
    {
```

```
        $dataProvider = new ActiveDataProvider([
```

```
            'query' => UserRides::find(),
```



```

    );

    return $this->render('index', [
        'dataProvider' => $dataProvider,
    ]);
}

/**
 * Displays a single UserRides model.
 * @param integer $USER_ID
 * @param integer $RIDE_ID
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionView($USER_ID, $RIDE_ID)
{
    return $this->render('view', [
        'model' => $this->findModel($USER_ID, $RIDE_ID),
    ]);
}

/**
 * Creates a new UserRides model.
 * If creation is successful, the browser will be redirected to the 'view' page.
 * @return mixed
 */
public function actionCreate()
{
    $model = new UserRides();

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['view', 'USER_ID' => $model->USER_ID, 'RIDE_ID' => $model->RIDE_ID]);
    }

    return $this->render('create', [
        'model' => $model,
    ]);
}

/**
 * Updates an existing UserRides model.
 * If update is successful, the browser will be redirected to the 'view' page.
 * @param integer $USER_ID
 * @param integer $RIDE_ID
 * @return mixed

```

```

* @throws NotFoundHttpException if the model cannot be found
*/
public function actionUpdate($USER_ID, $RIDE_ID)
{
    $model = $this->findModel($USER_ID, $RIDE_ID);

    if ($model->load(Yii::$app->request->post()) && $model->save()) {
        return $this->redirect(['view', 'USER_ID' => $model->USER_ID, 'RIDE_ID' => $model-
>RIDE_ID]);
    }

    return $this->render('update', [
        'model' => $model,
    ]);
}

/**
 * Deletes an existing UserRides model.
 * If deletion is successful, the browser will be redirected to the 'index' page.
 * @param integer $USER_ID
 * @param integer $RIDE_ID
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionDelete($USER_ID, $RIDE_ID)
{
    $this->findModel($USER_ID, $RIDE_ID)->delete();

    return $this->redirect(['rides/index']);
}
public function actionCancel($USER_ID, $RIDE_ID)
{
    $model = $this->findModel($USER_ID, $RIDE_ID);
    $model->CANCELLED = 1;
    $model->save();

    return $this->redirect(['rides/index']);
}

/**
 * Finds the UserRides model based on its primary key value.
 * If the model is not found, a 404 HTTP exception will be thrown.
 * @param integer $USER_ID
 * @param integer $RIDE_ID

```

```

    * @return UserRides the loaded model
    * @throws NotFoundHttpException if the model cannot be found
    */
    protected function findModel($USER_ID, $RIDE_ID)
    {
        if (($model = UserRides::findOne(['USER_ID' => $USER_ID, 'RIDE_ID' => $RIDE_ID]))
        !== null) {
            return $model;
        }

        throw new NotFoundHttpException('The requested page does not exist.');
```

ProfileController.php

```

<?php

namespace app\controllers;

use Yii;
use app\models\USER;
use yii\data\ActiveDataProvider;
use yii\filters\AccessControl;
use yii\web\Controller;
use yii\web\NotFoundHttpException;
use yii\filters\VerbFilter;

/**
 * ProfileController implements the CRUD actions for USER model.
 */
class ProfileController extends Controller
{
    /**
     * {@inheritdoc}
     */
    public function behaviors()
    {
        return [
            'access' => [
                'class' => AccessControl::className(),
                'only' => ['create', 'index', 'update', 'view'],
                'rules' => [
                    [
                        'actions' => ['create', 'index', 'update', 'view'],
                        'allow' => true,
                        'roles' => ['@'],
                    ],
                ],
            ],
        ];
    }
}
```

```

        'matchCallback' => function ($rule, $action) {
            return !Yii::$app->user->isGuest;
        }
    ],
],
],
'verbs' => [
    'class' => VerbFilter::className(),
    'actions' => [
        'delete' => ['POST'],
    ],
],
];
}

/**
 * Displays a single USER model.
 * @param integer $id
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionView($id)
{
    if($id != Yii::$app->user->getId()){
        return $this->redirect(['view', 'id' => Yii::$app->user->getId()]);
    }
    return $this->render('view', [
        'model' => $this->findModel($id),
    ]);
}

/**
 * Updates an existing USER model.
 * If update is successful, the browser will be redirected to the 'view' page.
 * @param integer $id
 * @return mixed
 * @throws NotFoundHttpException if the model cannot be found
 */
public function actionUpdate($id)
{
    if($id != Yii::$app->user->getId()){
        return $this->redirect(['view', 'id' => Yii::$app->user->getId()]);
    }
    $model = $this->findModel($id);

    if ($model->load(Yii::$app->request->post()) && $model->save()) {

```

```

        return $this->redirect(['view', 'id' => $model->ID]);
    }

    return $this->render('update', [
        'model' => $model,
    ]);
}

public function actionUpdatepassword($id)
{
    if($id != Yii::$app->user->getId()){
        return $this->redirect(['view', 'id' => Yii::$app->user->getId()]);
    }
    $model = $this->findModel($id);

    if ($model->load(Yii::$app->request->post())) {

        if(!$model->validatePassword($model->oldPassword)){
            return $this->redirect(['view', 'id' => $model->ID]);
        }

        $model->PASSWORD = $model->newPassword;
        $model->updatePassword();

        $model->save();
        return $this->redirect(['view', 'id' => $model->ID]);
    }

    return $this->render('updatepassword', [
        'model' => $model,
    ]);
}

/**
 * Finds the USER model based on its primary key value.
 * If the model is not found, a 404 HTTP exception will be thrown.
 * @param integer $id
 * @return USER the loaded model
 * @throws NotFoundHttpException if the model cannot be found
 */
protected function findModel($id)
{
    if (($model = USER::findOne($id)) !== null) {
        return $model;
    }
}

```

```
        throw new NotFoundException("The requested page does not exist.");  
    }  
}
```

Modeli

Car.php

<?php

```
namespace app\models;

use Yii;

/**
 * This is the model class for table "car".
 *
 * @property int $id
 * @property string $reg_nr
 * @property string $color
 * @property string $make
 * @property string $model
 * @property int $seats
 * @property int $owner_id
 * @property string $image_name
 */
class Car extends \yii\db\ActiveRecord
{
    /**
     * {@inheritdoc}
     */
    public static function tableName()
    {
        return 'car';
    }

    /**
     * {@inheritdoc}
     */
    public function rules()
    {
        return [
            [['reg_nr', 'color', 'make', 'model', 'seats'], 'required', 'message' => '{attribute} nedrīkst būt tukšs'],
            [['seats'], 'integer'],
            [['reg_nr', 'color'], 'string', 'max' => 11],
            [['make', 'model'], 'string', 'max' => 20],
        ];
    }

    /**
```

```

        * { @inheritdoc }
        */
    public function attributeLabels()
    {
        return [
            'id' => 'ID',
            'reg_nr' => 'Reģistrācijas numurs',
            'color' => 'Krāsa',
            'make' => 'Ražotājs',
            'model' => 'Modelis',
            'seats' => 'vietu skaits',
        ];
    }
}
Cities.php
<?php

```

```

namespace app\models;

use Yii;

/**
 * This is the model class for table "cities".
 *
 * @property string $name
 * @property int $id
 * @property int $level
 */
class Cities extends \yii\db\ActiveRecord
{
    /**
     * { @inheritdoc }
     */
    public static function tableName()
    {
        return 'cities';
    }

    /**
     * { @inheritdoc }
     */
    public function rules()
    {
        return [
            [['name', 'level'], 'required'],
            [['level'], 'integer'],

```



```

        [['name'], 'string', 'max' => 22],
    ];
}

/**
 * {@inheritdoc}
 */
public function attributeLabels()
{
    return [
        'name' => 'Name',
        'id' => 'ID',
        'level' => 'Level',
    ];
}
}
ContactForm.php
<?php

namespace app\models;

use Yii;
use yii\base\Model;

/**
 * ContactForm is the model behind the contact form.
 */
class ContactForm extends Model
{
    public $name;
    public $email;
    public $subject;
    public $body;
    public $verifyCode;

    /**
     * @return array the validation rules.
     */
    public function rules()
    {
        return [
            // name, email, subject and body are required
            [['name', 'email', 'subject', 'body'], 'required', 'message' => '{attribute} lauks nevar būt tukšs'],
            // email has to be a valid email address

```

```

        ['email', 'email', 'message' => '{attribute} nav pareiza epasta adrese'],
        // verifyCode needs to be entered correctly
        ['verifyCode', 'captcha', 'message' => '{attribute} CAPTCHA kods nav aizpildīts pareizi'],
    ];
}

/**
 * @return array customized attribute labels
 */
public function attributeLabels()
{
    return [
        'verifyCode' => 'Verifikācijas kods',
        'name' => 'Vārds',
        'email' => 'e-pasts',
        'subject' => 'Temats',
        'body' => 'Ziņa',
    ];
}

/**
 * Sends an email to the specified email address using the information collected by this model.
 * @param string $email the target email address
 * @return bool whether the model passes validation
 */
public function contact($email)
{
    if ($this->validate()) {
        Yii::$app->mailer->compose()
            ->setTo(Yii::$app->params['adminEmail'])
            ->setFrom($email)
            ->setReplyTo($email)
            ->setSubject($this->subject)
            ->setTextBody($this->body)
            ->send();

        return true;
    }
    return false;
}
}
LoginForm.php
<?php

namespace app\models;

```

```

use Yii;
use yii\base\Model;

/**
 * LoginForm is the model behind the login form.
 *
 * @property User1|null $user This property is read-only.
 */
class LoginForm extends Model
{
    public $username;
    public $password;
    public $rememberMe = true;

    private $_user = false;

    public $hash;
    /**
     * @return array the validation rules.
     */
    public function rules()
    {
        return [
            // username and password are both required
            [['username', 'password'], 'required', 'message' => 'Lauks {attribute} nedrīkst būt tukšs'],
            // rememberMe must be a boolean value
            ['rememberMe', 'boolean'],
            // password is validated by validatePassword()
            ['password', 'validatePassword'],
            [['username'], 'filter', 'filter' => 'strtolower'],
        ];
    }
    public function attributeLabels()
    {
        return [
            'username' => 'Lietotājevārds',
            'password' => 'Parole',
            'rememberMe' => 'atcerēties',
        ];
    }
    /**
     * Validates the password.
     * This method serves as the inline validation for password.
     *
     * @param string $attribute the attribute currently being validated

```

```

* @param array $params the additional name-value pairs given in the rule
*/
public function validatePassword($attribute)
{
    if (!$this->hasErrors()) {

        $user = $this->getUser();

        if (!$user || !$user->validatePassword($this->password)) {
            $this->addError($attribute, 'Nepareizs lietotājvārds vai parole');
        }
        if($user->isBlocked()){
            $this->addError($attribute, 'Lietotājs bloķēts, lūdzu sazinieties ar administrāciju!');
        }
    }
}

/**
 * Logs in a user using the provided username and password.
 * @return bool whether the user is logged in successfully
 */
public function login()
{
    if ($this->validate()) {

        return Yii::$app->user->login($this->getUser(), $this->rememberMe ? 3600*24*30 : 0);
    }
    return false;
}

/**
 * Finds user by [[username]]
 *
 * @return User1|null
 */
public function getUser()
{
    if ($this->_user === false) {
        $this->_user = USER::findByUsername($this->username);
    }

    return $this->_user;
}

public function loginAdmin()
{

```

```

        if ($this->validate() && User::isUserAdmin($this->username)) {
            return Yii::$app->user->login($this->getUser(), $this->rememberMe ? 3600 * 24 * 30 :
0);
        } else {
            return false;
        }
    }
}

```

PassCh.php

```
<?php
```

```
namespace app\models;
```

```
use Yii;
```

```
use yii\db\Expression;
```

```
/**
```

```
 * This is the model class for table "password_change".
```

```
 *
```

```
 * @property int $ID
```

```
 * @property int $EMAIL
```

```
 * @property string $PASSKEY
```

```
 * @property string $TIME_CREATED
```

```
 */
```

```
class PassCh extends \yii\db\ActiveRecord
```

```
{
```

```
    public $verifyCode;
```

```
    /**
```

```
     * { @inheritdoc }
```

```
    */
```

```
    public static function tableName()
```

```
    {
```

```
        return 'password_change';
```

```
    }
```

```
    /**
```

```
     * { @inheritdoc }
```

```
    */
```

```
    public function rules()
```

```
    {
```

```
        return [
```

```
            [['EMAIL'], 'string', 'max' => 50],
```

```
            [['TIME_CREATED'], 'safe'],
```

```
            [['PASSKEY'], 'string', 'max' => 30],
```

```
            [['EMAIL'], 'required', 'message' => '{attribute} nedrīkst būt tukšs'],
```

```
            [['EMAIL', 'email', 'message' => 'Nav norādīta korekta epasta adrese'],
```

```

        // verifyCode needs to be entered correctly
        ['verifyCode', 'captcha', 'message' => '{attribute} CAPTCHA kods nav aizpildīts pareizi'],
        [['EMAIL'], 'filter', 'filter' => 'strtolower'],
    ];
}

/**
 * { @inheritdoc }
 */
public function attributeLabels()
{
    return [
        'ID' => 'ID',
        'EMAIL' => 'Epasts',
        'PASSKEY' => 'Passkey',
        'TIME_CREATED' => 'Time Created',
    ];
}

/**
 * @param $passkey
 * @return int
 * Pārbaude vai paroles maiņas saite ir vēl aktīva, ja laiks beidzies, returno 0
 */
public function validatePassKey()
{
    return PassCh::find()->where(['PASSKEY' => $this->PASSKEY])-
>andWhere(['>', 'TIME_CREATED', new Expression('timestamp(DATE_SUB(NOW(),
INTERVAL 10 MINUTE))')]
    )->count();
}

public static function findByPassKey($passKey)
{
    return self::findOne(['PASSKEY' => $passKey]);
}
}
Rides.php
<?php

namespace app\models;

use Yii;

```

```

/**
 * This is the model class for table "rides".
 *
 * @property int $ID
 * @property int $CITY_FROM_ID
 * @property int $CITY_TO_ID
 * @property int $DRIVER_ID
 * @property string $DATE
 * @property int $CAR_ID
 * @property string $NOTES
 *
 * @property Cities $cITYFROM
 * @property Cities $cITYTO
 * @property Car $cAR
 * @property UserRides[] $userRides
 * @property User $uSER
 */
class Rides extends \yii\db\ActiveRecord
{
    /**
     * { @inheritdoc }
     */
    public static function tableName()
    {
        return 'rides';
    }

    /**
     * { @inheritdoc }
     */
    public function rules()
    {
        return [
            [['CITY_FROM_ID', 'CITY_TO_ID', 'DRIVER_ID', 'DATE', 'CAR_ID', 'NOTES'],
            'required', 'message' => '{attribute} lauks nedrīkst būt tukšs'],
            [['CITY_FROM_ID', 'CITY_TO_ID', 'DRIVER_ID', 'CAR_ID'], 'integer'],
            [['DATE'], 'safe'],
            [['NOTES'], 'string', 'max' => 400],
            [['CITY_FROM_ID'], 'exist', 'skipOnError' => true, 'targetClass' => Cities::className(),
            'targetAttribute' => ['CITY_FROM_ID' => 'id']],
            [['CITY_TO_ID'], 'exist', 'skipOnError' => true, 'targetClass' => Cities::className(),
            'targetAttribute' => ['CITY_TO_ID' => 'id']],
            [['CAR_ID'], 'exist', 'skipOnError' => true, 'targetClass' => Car::className(),
            'targetAttribute' => ['CAR_ID' => 'id']],
        ];
    }
}

```

```

/**
 * { @inheritdoc }
 */
public function attributeLabels()
{
    return [
        'ID' => 'ID',
        'CITY_FROM_ID' => 'Pilsēta no',
        'CITY_TO_ID' => 'Pilsēta uz',
        'DRIVER_ID' => 'Vadītājs',
        'DATE' => 'Izbraukšanas laiks',
        'CAR_ID' => 'Mašīna',
        'NOTES' => 'Piezīmes',
    ];
}

/**
 * Atgriež brīvo vietu skaitu mašīnā
 */
public function getSeatCount()
{
    $car = Car::find()->where(['id' => $this->CAR_ID])->one();

    $availableSeatCount = $car->seats;
    $takenSeatCount = UserRides::find()->where(['RIDE_ID' => $this->ID, 'CANCELLED' =>
0])->count();
    return $availableSeatCount - $takenSeatCount;
}

/**
 * Atgriež vai braucienam ir pieteicies kāds braucējs
 */
public function hasRiders()
{
    $takenSeatCount = UserRides::find()->where(['RIDE_ID' => $this->ID, 'CANCELLED' =>
0])->count();
    if($takenSeatCount == 0){
        return 0;
    }else{
        return 1;
    }
}

/**
 * @return \yii\db\ActiveQuery

```



```

    */
    public function getCITYFROM()
    {
        return $this->hasOne(Cities::className(), ['id' => 'CITY_FROM_ID']);
    }

    /**
     * @return \yii\db\ActiveQuery
     */
    public function getCITYTO()
    {
        return $this->hasOne(Cities::className(), ['id' => 'CITY_TO_ID']);
    }

    public function getCAR()
    {
        return $this->hasOne(Car::className(), ['id' => 'CAR_ID']);
    }
    /**
     * @return \yii\db\ActiveQuery
     */
    public function getUserRides()
    {
        return $this->hasMany(UserRides::className(), ['RIDE_ID' => 'ID']);
    }

    /**
     * @return \yii\db\ActiveQuery
     */
    public function getUser()
    {
        return $this->hasOne(User::className(), ['ID' => 'DRIVER_ID']);
    }
    /**
     * atgriež vadītāja vārdu
     */
    public function getDriverName()
    {
        return $this->hasOne(USER::className(), ['ID' => 'DRIVER_ID']);
    }

    public static function findById($id)
    {
        return self::findOne(['ID' => $id]);
    }
}

```

RidesSearch.php

<?php

```
namespace app\models;
```

```
use yii\base\Model;
```

```
use yii\data\ActiveDataProvider;
```

```
use app\models\Rides;
```

```
use yii\db\Expression;
```

```
use Yii;
```

```
/**
```

```
 * RidesSearch represents the model behind the search form of `app\models\Rides`.
```

```
 */
```

```
class RidesSearch extends Rides
```

```
{
```

```
    /**
```

```
     * { @inheritdoc }
```

```
    */
```

```
    public function rules()
```

```
    {
```

```
        return [
```

```
            [['ID', 'CITY_FROM_ID', 'CITY_TO_ID', 'DRIVER_ID'], 'integer'],
```

```
            [['DATE'], 'safe'],
```

```
        ];
```

```
    }
```

```
    /**
```

```
     * { @inheritdoc }
```

```
    */
```

```
    public function scenarios()
```

```
    {
```

```
        // bypass scenarios() implementation in the parent class
```

```
        return Model::scenarios();
```

```
    }
```

```
    /**
```

```
     * Creates data provider instance with search query applied
```

```
     *
```

```
     * @param array $params
```

```
     *
```

```
     * @return ActiveDataProvider
```

```
     */
```

```
    public function search($params)
```

```
    {
```

```
        // $query = Rides::find();
```

```
        $query = Rides::find()->where(['>', 'DATE', new Expression('NOW()')]);
```

```

// add conditions that should always apply here

$dataProvider = new CActiveDataProvider([
    'query' => $query,
    'pagination' => [
        'pageSize' => 10,
    ],
]);

$this->load($params);

if (!$this->validate()) {
    // uncomment the following line if you do not want to return any records when validation
fails
    // $query->where('0=1');
    return $dataProvider;
}
// grid filtering conditions
if(!empty($this->DATE)){
    $this->DATE = date("Y-m-d", strtotime($this->DATE));
}

$query->andWhere([
    'ID' => $this->ID,
    'CITY_FROM_ID' => $this->CITY_FROM_ID,
    'CITY_TO_ID' => $this->CITY_TO_ID,
    'DRIVER_ID' => $this->DRIVER_ID,
    'date(DATE)' => $this->DATE,
]);

return $dataProvider;
}

public function searchUserRides($params = null)
{
    $query = Rides::find()->where(
        ['DRIVER_ID' => Yii::$app->user->getId()])
        ->andWhere(['>','DATE', new Expression('NOW()')])
        );

    // add conditions that should always apply here

    $dataProvider = new CActiveDataProvider([
        'query' => $query,
        'pagination' => [
            'pageSize' => 5,

```

```

        ],
    );

    $this->load($params);

    if (!$this->validate()) {
        // uncomment the following line if you do not want to return any records when validation
fails
        // $query->where('0=1');
        return $dataProvider;
    }

    // grid filtering conditions
    $query->andFilterWhere([
        'ID' => $this->ID,
        'CITY_FROM_ID' => $this->CITY_FROM_ID,
        'CITY_TO_ID' => $this->CITY_TO_ID,
        'DRIVER_ID' => $this->DRIVER_ID,
        'date(DATE)' => $this->DATE,
    ]);

    return $dataProvider;
}
}

```

UploadForm.php

```

<?php

namespace app\models;
use Yii;
use yii\base\Model;
use yii\web\UploadedFile;

class UploadForm extends Model
{
    /**
     * @var UploadedFile
     */
    public $imageFile;
    public $imageName;

    public function rules()
    {
        return [
            [['imageFile'], 'file', 'skipOnEmpty' => true, 'extensions' => 'png, jpg, jpeg'],
        ];
    }
}

```

```

public function upload()
{

    if ($this->validate()) {
        $this->imageFile->saveAs('../web/uploads/' . $this->imageName);
        return true;
    } else {
        return false;
    }
}

public function attributeLabels()
{
    return [
        'imageFile' => 'Attēls',
    ];
}

}
USER.php
<?php

namespace app\models;

use Yii;

/**
 * This is the model class for table "USER".
 *
 * @property int $ID
 * @property string $USERNAME
 * @property string $NAME
 * @property string $SURNAME
 * @property string $PASSWORD
 * @property string $EMAIL
 * @property string $REG_DATE
 * @property string $AUTHKEY
 * @property string $ACCESSTOKEN
 * @property int $ROLE
 * @property int $BLOCKED
 * @property int $PHONE
 */
class USER extends \yii\db\ActiveRecord implements \yii\web\IdentityInterface
{
    public $email;

```

```

public $user;
public $id;
public $username;
public $password;
public $authKey;
public $accessToken;

public $oldPassword;
public $newPassword;
public $repeatPassword;

const ROLE_USER = 10;
const ROLE_ADMIN = 20;

/**
 * { @inheritdoc }
 */
public static function tableName()
{
    return 'USER';
}

/**
 * { @inheritdoc }
 */
public function rules()
{
    return [
        [['USERNAME', 'NAME', 'SURNAME', 'EMAIL', 'PASSWORD', 'REG_DATE',
'PHONE'], 'required', 'message' => 'Lauks {attribute} nedrīkst būt tukšs'],
        [['REG_DATE'], 'safe'],
        [['EMAIL', 'USERNAME'], 'unique', 'message' => '{attribute} ir jau aizņemts, lūdzu
mēģiniet citu'],
        [['USERNAME', 'NAME', 'SURNAME'], 'string', 'max' => 40],
        [['PASSWORD', 'oldPassword', 'newPassword', 'repeatPassword'], 'string', 'min' => 8,
'tooShort' => '{attribute} lauks nedrīkst būt īsāks par 8 simboliem'],
        [['EMAIL'], 'email', 'message' => '{attribute} nav pareiza epasta adrese'],
        [['AUTHKEY', 'ACCESSTOKEN'], 'string', 'max' => 255],
        ['ROLE', 'default', 'value' => 10],
        ['ROLE', 'in', 'range' => [self::ROLE_USER, self::ROLE_ADMIN]],
        [['EMAIL', 'USERNAME'], 'filter', 'filter' => 'strtolower'],
        ['USERNAME', 'match', 'pattern' => '/^[a-z0-9]+$/', 'message' => 'Izmantot tikai mazos
burtus bez garuma un mīkstinājuma zīmēm'],
        ['newPassword', 'compare', 'compareAttribute' => 'repeatPassword', 'message' =>
'parolēm jābūt vienādām'],

```

```

        ['repeatPassword', 'compare', 'compareAttribute' => 'newPassword', 'message' =>
'parolēm jābūt vienādām'],
    ];
}

/**
 * { @inheritdoc }
 */
public function attributeLabels()
{
    return [
        'ID' => 'ID',
        'USERNAME' => 'Lietotājvārds',
        'NAME' => 'Vārds',
        'SURNAME' => 'Uzvārds',
        'PASSWORD' => 'Parole',
        'EMAIL' => 'Epasts',
        'REG_DATE' => 'Reģistrācijas datums',
        'PHONE' => 'Telefona numurs',
        'ROLE' => 'Lietotāja loma 10 - lietotājs 20- administrators',
        'oldPassword' => 'Vecā parole',
        'newPassword' => 'Jaunā parole',
        'repeatPassword' => 'Atkārtojiet paroli',
    ];
}

/**
 * { @inheritdoc }
 * @return USERQuery the active query used by this AR class.
 */
public static function find()
{
    return new USERQuery(get_called_class());
}

public static function findByUsername($username)
{
    return self::findOne(['USERNAME' => $username]);
}

/**
 * { @inheritdoc }
 */
public static function findIdentity($id)
{

```

```

        return self::findOne($id);
    }

    /**
     * { @inheritdoc }
     */
    public static function findIdentityByAccessToken($token, $type = null)
    {

        return self::findOne(['ACCESSTOKEN' => $token]);
    }

    /**
     * { @inheritdoc }
     */
    public function getId()
    {
        return $this->ID;
    }

    /**
     * { @inheritdoc }
     */
    public function getAuthKey()
    {
        return $this->AUTHKEY;
    }

    /**
     * { @inheritdoc }
     */
    public function validateAuthKey($authKey)
    {
        return $this->authKey === $authKey;
    }

    public function validatePassword($password){
        return password_verify($password, $this->PASSWORD);
    }

    public static function isUserAdmin($username)
    {
        if (static::findOne(['USERNAME' => $username, 'ROLE' => self::ROLE_ADMIN])){

            return true;
        }
    }

```



```

        } else {

            return false;
        }
    }
    public static function findByEmail($email)
    {
        return self::findOne(['EMAIL' => $email]);
    }

    public function updatePassword()
    {
        $this->PASSWORD = password_hash($this->PASSWORD, PASSWORD_ARGON2I);
        $this->save();

        return 1;
    }

    public function isBlocked()
    {
        return $this->BLOCKED;
    }

    public static function findById($id)
    {
        return self::findOne(['ID' => $id]);
    }
}
USERQuery.php
<?php

namespace app\models;

/**
 * This is the ActiveQuery class for [[USER]].
 *
 * @see USER
 */
class USERQuery extends \yii\db\ActiveQuery
{
    /**public function active()
    {
        return $this->andWhere('[[status]]=1');
    }*/

    /**
     * { @inheritdoc }

```

```

    * @return USER[]|array
    */
    public function all($db = null)
    {
        return parent::all($db);
    }

    /**
     * { @inheritdoc}
     * @return USER|array|null
     */
    public function one($db = null)
    {
        return parent::one($db);
    }
}
UserRides.php
<?php

namespace app\models;

use Yii;

/**
 * This is the model class for table "user_rides".
 *
 * @property int $USER_ID
 * @property int $RIDE_ID
 * @property string $JOIN_DATE
 * @property int $CANCELLED
 *
 * @property User $uSER
 * @property Rides $rIDE
 */
class UserRides extends \yii\db\ActiveRecord
{
    /**
     * { @inheritdoc}
     */
    public static function tableName()
    {
        return 'user_rides';
    }

    /**
     * { @inheritdoc}

```

```

*/
public function rules()
{
    return [
        [['USER_ID', 'RIDE_ID'], 'required'],
        [['USER_ID', 'RIDE_ID', 'CANCELLED'], 'integer'],
        [['JOIN_DATE'], 'safe'],
        [['USER_ID', 'RIDE_ID'], 'unique', 'targetAttribute' => ['USER_ID', 'RIDE_ID']],
        [['USER_ID'], 'exist', 'skipOnError' => true, 'targetClass' => User::className(),
'targetAttribute' => ['USER_ID' => 'ID']],
        [['RIDE_ID'], 'exist', 'skipOnError' => true, 'targetClass' => Rides::className(),
'targetAttribute' => ['RIDE_ID' => 'ID']],
    ];
}

/**
 * {@inheritdoc}
 */
public function attributeLabels()
{
    return [
        'USER_ID' => 'User ID',
        'RIDE_ID' => 'Ride ID',
        'JOIN_DATE' => 'Join Date',
        'CANCELLED' => 'Atteicies',
    ];
}

/**
 * @return \yii\db\ActiveQuery
 */
public function getUser()
{
    return $this->hasOne(User::className(), ['ID' => 'USER_ID']);
}

/**
 * @return \yii\db\ActiveQuery
 */
public function getRIDE()
{
    return $this->hasOne(Rides::className(), ['ID' => 'RIDE_ID']);
}

/**
 * atgriež vai lietotājs ir pievienojies

```

```

    */
    public function isJoined()
    {
        return UserRides::find()->where(['RIDE_ID' => $this->RIDE_ID, 'USER_ID' => $this->USER_ID])->count();
    }
    /**
     * atgriež vai lietotājs ir iepriekš atteicies no brauciena
     */
    public function hasCancelled()
    {
        $model = UserRides::find()->where(['RIDE_ID' => $this->RIDE_ID, 'USER_ID' => $this->USER_ID])->one();
        if(isset($model)){
            return $model->CANCELLED;
        }
        return 0;
    }
}

```

Skati

ADMIN

_form.php

```
<?php
```

```

use yii\helpers\Html;
use yii\widgets\ActiveForm;

```

```

/* @var $this yii\web\View */
/* @var $model app\models\USER */
/* @var $form yii\widgets\ActiveForm */
?>

```

```
<div class="user-form">
```

```
    <?php $form = ActiveForm::begin(); ?>
```

```
    <?= $form->field($model, 'ROLE')->textInput(['maxlength' => true]) ?>
```

```
    <div class="form-group">
```

```
        <?= Html::submitButton($btnText, ['class' => 'btn btn-success', 'name'=>'register']) ?>
```

```
    </div>
```

```
    <?php ActiveForm::end(); ?>
```

```
</div>
```

index.php

```
<?php

use yii\helpers\Html;
use yii\grid\GridView;

/* @var $this yii\web\View */
/* @var $dataProvider yii\data\ActiveDataProvider */

$this->title = 'Admin panel';
$this->params['breadcrumbs'][] = $this->title;
?>
<div class="user-index">

    <h1><?= Html::encode($this->title) ?></h1>

    <?= GridView::widget([
        'dataProvider' => $dataProvider,
        'columns' => [
            ['class' => 'yii\grid\SerialColumn'],

            'USERNAME',
            'NAME',
            'SURNAME',
            'EMAIL:email',
            'REG_DATE',
            [
                'class' => 'yii\grid\ActionColumn',
                'template' => '{myButton}',
                'buttons' => [
                    'myButton' => function($url, $dataProvider, $key) {

                        return Html::a('Apskatīt', ['view','id' => $dataProvider->ID], [
                            'class' => 'btn btn-success']);
                    }
                ]
            ],
        ],
    ]); ?>

</div>
update.php
<?php
```

```

use yii\helpers\Html;

/* @var $this yii\web\View */
/* @var $model app\models\USER */

$this->title = 'Update User: ' . $model->NAME;

?>
<div class="user-update">

    <h1><?= Html::encode($this->title) ?></h1>

    <?= $this->render('_form', [
        'model' => $model,
        'btnText' => $btnText
    ]) ?>

</div>
view.php
<?php

use yii\helpers\Html;
use yii\widgets\DetailView;

/* @var $this yii\web\View */
/* @var $model app\models\USER */

$this->title = $model->NAME;
$this->params['breadcrumbs'][] = ['label' => 'Users', 'url' => ['index']];
$this->params['breadcrumbs'][] = $this->title;
\yii\web\YiiAsset::register($this);
?>
<div class="user-view">

    <h1><?= Html::encode($this->title) ?></h1>

    <p>
        <?= Html::a('Aktualizēt', ['update', 'id' => $model->ID], ['class' => 'btn btn-primary']) ?>
        <?= Html::a('Dzēst', ['delete', 'id' => $model->ID], [
            'class' => 'btn btn-danger',
            'data' => [
                'confirm' => 'Vai jūs tiešām vēlieties dzēst lietotāju?',
                'method' => 'post',
            ],
        ]) ?>

```

```

<?php
    if($model->isBlocked()){
        echo Html::a('Atbloķēt!', ['unblock', 'id' => $model->ID], [
            'class' => 'btn btn-success',
            'data' => [
                'confirm' => 'Vai jūs tiešām vēlieties atbloķēt lietotāju?',
                'method' => 'post',
            ],
        ]);
    }else{
        echo Html::a('Bloķēt!', ['block', 'id' => $model->ID], [
            'class' => 'btn btn-danger',
            'data' => [
                'confirm' => 'Vai jūs tiešām vēlieties bloķēt lietotāju?',
                'method' => 'post',
            ],
        ]);
    }
?>
</p>

```

```

<?= DetailView::widget([
    'model' => $model,
    'attributes' => [
        'ID',
        'USERNAME',
        'NAME',
        'SURNAME',
        'EMAIL:email',
        'REG_DATE',
    ],
]) ?>

```

</div>

CAR

_form.php

```

<?php

```

```

use yii\helpers\Html;
use yii\widgets\ActiveForm;
use kartik\file\FileInput;

```

```

/* @var $this yii\web\View */
/* @var $model app\models\Car */
/* @var $form yii\widgets\ActiveForm */
?>

```

```

<div class="car-form">
    <?php $form = ActiveForm::begin(); ?>
    <div class="col-lg-12">
        <div class="col-lg-6">
            <?= $form->field($model, 'reg_nr')->textInput(['maxlength' => true]) ?>

            <?= $form->field($model, 'color')->textInput(['maxlength' => true]) ?>

            <?= $form->field($model, 'make')->textInput(['maxlength' => true]) ?>

            <?= $form->field($model, 'model')->textInput(['maxlength' => true]) ?>

            <?= $form->field($model, 'seats')->textInput(['maxlength' => true]) ?>
        </div>

        <div class="col-lg-6">
            <?= $form->field($upload, 'imageFile')->widget(FileInput::classname(), [
                'options' => [
                    'accept' => 'image/*',
                    'multiple'=>false,
                ],
                'pluginOptions' => [
                    'showUpload' => false,
                ]
            ]);?>
        </div>
    </div>
    <div class="form-group">
        <?= Html::submitButton('Saglabāt', ['class' => 'btn btn-success', 'style' => 'margin-left:
30px;']) ?>
    </div>

```

```

<?php ActiveForm::end(); ?>

```

```

</div>

```

Create.php

```

<?php

```

```

use yii\helpers\Html;

```

```

/* @var $this yii\web\View */

```

```

/* @var $model app\models\Car */

```

```

$this->title = 'Pievienot transportlīdzekli';

```



```
?>
<div class="car-create">

    <h1><?= Html::encode($this->title) ?></h1>
    <?php if (Yii::$app->session->hasFlash('needCar')): ?>

        <div class="alert alert-danger">
            Pirms brauciena pievienošanas lūdzu pievienojiet mašīnu ar kuru veiksiet braucienu!
        </div>
    <?php endif;?>
    <?= $this->render('_form', [
        'model' => $model,
        'upload' => $upload,
    ]) ?>

```

```
</div>
```

Index.php

```
<?php
```

```
use yii\helpers\Html;
use yii\grid\GridView;
```

```
/* @var $this yii\web\View */
/* @var $dataProvider yii\data\ActiveDataProvider */
```

```
$this->title = 'Mani transportlīdzekļi';
?>
```

```
<div class="car-index">
```

```
    <h1><?= Html::encode($this->title) ?></h1>
```

```
    <p>
```

```
        <?= Html::a('Pievienot transportlīdzekli', ['create'], ['class' => 'btn btn-success']) ?>
```

```
    </p>
```

```
    <?= GridView::widget([
        'dataProvider' => $dataProvider,
        'columns' => [
            ['class' => 'yii\grid\SerialColumn'],
            'reg_nr',
            'color',
            'make',
            'model',
            'seats',
            ['class' => 'yii\grid\ActionColumn'],
        ],
    ])
```

```

],
'emptyText' => 'Ups, izskatās, ka neesi pievienojis nevienu transportlīdzekli',
]); ?>

```

```

</div>

```

Update.php

```

<?php

```

```

use yii\helpers\Html;

```

```

/* @var $this yii\web\View */

```

```

/* @var $model app\models\Car */

```

```

/* @var $update app\models\Car */

```

```

$this->title = 'Update Car: ' . $model->reg_nr;

```

```

$this->params['breadcrumbs'][] = ['label' => 'Cars', 'url' => ['index']];

```

```

$this->params['breadcrumbs'][] = ['label' => $model->id, 'url' => ['view', 'id' => $model->id]];

```

```

$this->params['breadcrumbs'][] = 'Update';

```

```

?>

```

```

<div class="car-update">

```

```

    <h1><?= Html::encode($this->title) ?></h1>

```

```

    <?= $this->render('_form', [

```

```

        'model' => $model,

```

```

        'upload' => $upload,

```

```

    ]) ?>

```

```

</div>

```

View.php

```

<?php

```

```

use yii\helpers\Html;

```

```

use yii\widgets\DetailView;

```

```

/* @var $this yii\web\View */

```

```

/* @var $model app\models\Car */

```

```

$this->title = $model->reg_nr;

```

```

\yii\web\YiiAsset::register($this);

```

```

?>

```

```

<div class="car-view">

```

```

    <h1><?= Html::encode($this->title) ?></h1>

```

```

<p>
  <?= Html::a('Aktualizēt', ['update', 'id' => $model->id], ['class' => 'btn btn-primary']) ?>
  <?= Html::a('Dzēst', ['delete', 'id' => $model->id], [
    'class' => 'btn btn-danger',
    'data' => [
      'confirm' => 'Vai jūs tiešām vēlaties dzēst šo mašīnu?',
      'method' => 'post',
    ],
  ]) ?>
</p>

```

```

<?= DetailView::widget([
  'model' => $model,
  'attributes' => [
    'reg_nr',
    'color',
    'make',
    'model',
    'seats',
  ],
]) ?>
<?php echo Html::img('@web/uploads/'.$model->image_name, [
  'alt' => 'pic not found',
  'width' => '800px',
  'height' => '600px'
])
);?>
</div>

```

FILE

Upload.php

```

<?php
use yii\widgets\ActiveForm;
use kartik\file\FileInput;
?>

<?php $form = ActiveForm::begin(['options' => ['enctype' => 'multipart/form-data']]) ?>

<?= $form->field($model, 'imageFile')->fileInput() ?>

<button>Submit</button>

<?= $form->field($model, 'imageFile')->widget(FileInput::classname(), [
  'options' => ['accept' => 'image/*'],
]);?>
<?php ActiveForm::end() ?>

```

LAYOUTS

Main.php

```
<?php
```

```
/* @var $this \yii\web\View */
/* @var $content string */
```

```
use app\widgets\Alert;
use yii\helpers\Html;
use yii\bootstrap\Nav;
use yii\bootstrap\NavBar;
use yii\widgets\Breadcrumbs;
use app\assets\AppAsset;
```

```
AppAsset::register($this);
```

```
?>
```

```
<?php $this->beginPage() ?>
```

```
<!DOCTYPE html>
```

```
<html lang="<?= Yii::$app->language ?>">
```

```
<head>
```

```
    <meta charset="<?= Yii::$app->charset ?>">
```

```
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
    <?php $this->registerCsrfMetaTags() ?>
```

```
    <title><?= Html::encode('Kopbraukšana') ?></title>
```

```
    <?php $this->head() ?>
```

```
<!-- Global site tag (gtag.js) - Google Analytics -->
```

```
<script async src="https://www.googletagmanager.com/gtag/js?id=UA-167443930-1"></script>
```

```
<script>
```

```
    window.dataLayer = window.dataLayer || [];
```

```
    function gtag(){dataLayer.push(arguments);}
```

```

        'options' => [
            'class' => ' navbar-inverse navbar-fixed-top',
        ],
    );
    $labels = [
        ['label' => 'Sākums', 'url' => ['/site/index']],
        ['label' => 'Saziņa', 'url' => ['/site/contact']],
    ];
    if(Yii::$app->user->isGuest){
        array_push($labels,['label' => 'Pieslēgties', 'url' => ['/site/login']],['label' => 'Reģistrēties',
'url' => ['/site/register']]);
    }else{
        array_push($labels, ['label' => 'Mani braucieni', 'url' => ['/rides/userrides']]);
        array_push($labels, ['label' => 'Mani transportlīdzekļi', 'url' => ['/car/index']]);
        array_push($labels, ['label' => 'Profils', 'url' => ['/profile/view', 'id' => Yii::$app->user-
>getId()]);
        if(Yii::$app->user->identity->ROLE == 20){
            array_push($labels,['label' => 'ADMIN PANELIS', 'url' => ['/admin/index']]);
        }
        array_push($labels,<li>
            . Html::beginForm(['/site/logout'], 'post')
            . Html::submitButton(
                'Atslēgties (' . Yii::$app->user->identity->USERNAME . ')',
                ['class' => 'btn btn-link logout']
            )
            . Html::endForm()
            . '</li>');
    }

    echo Nav::widget([
        'options' => ['class' => 'navbar-nav navbar-right'],
        'items' => $labels
    ])
    );
    NavBar::end();

?>

<div class="container">
    <?= Breadcrumbs::widget([
        'links' => isset($this->params['breadcrumbs']) ? $this->params['breadcrumbs'] : [],
    ]) ?>
    <?= Alert::widget() ?>
    <?= $content ?>
</div>

```

```

</div>

<footer class="footer">
    <div class="container">
        <p class="pull-left">&copy; Raimonds Lagzdīns <?= date('Y') ?></p>
    </div>
</footer>

<?php $this->endBody() ?>
</body>
</html>
<?php $this->endPage() ?>
PASS_CH
_form.php
<?php

use yii\helpers\Html;
use yii\widgets\ActiveForm;
use yii\captcha\Captcha;

/* @var $this yii\web\View */
/* @var $model app\models\PassCh */
/* @var $form yii\widgets\ActiveForm */
?>

<div class="pass-ch-form col-lg-5">
    <?php if (Yii::$app->session->hasFlash('passwordChangeRequest')): ?>

        <div class="alert alert-success">
            Paroles atjaunošanas saite nosūtīta jums uz epastu
        </div>

    <?php else: ?>

    <?php $form = ActiveForm::begin(); ?>

    <?= $form->field($model, 'EMAIL')->textInput() ?>

    <?= $form->field($model, 'verifyCode')->widget(Captcha::className(), [
        'template' => '<div class="row"><div class="col-lg-3">{image}</div><div class="col-lg-6">{input}</div></div>',
    ]) ?>
    <div class="form-group">
        <?= Html::submitButton('Atjaunot paroli', ['class' => 'btn btn-success']) ?>
    </div>

```

```

        <?php ActiveForm::end(); ?>
    <?php endif; ?>
</div>
Create.php
<?php

use yii\helpers\Html;

/* @var $this yii\web\View */
/* @var $model app\models\PassCh */

$this->title = 'Paroles atjaunošana';

?>
<div class="pass-ch-create">

    <h1><?= Html::encode($this->title) ?></h1>
    <?= $this->render('_form', [
        'model' => $model,
    ]) ?>

</div>
Index.php
<?php

use yii\helpers\Html;
use yii\grid\GridView;

/* @var $this yii\web\View */
/* @var $dataProvider yii\data\ActiveDataProvider */

$this->title = 'Pass Ches';
$this->params['breadcrumbs'][] = $this->title;
?>
<div class="pass-ch-index">

    <h1><?= Html::encode($this->title) ?></h1>

    <p>
        <?= Html::a('Create Pass Ch', ['create'], ['class' => 'btn btn-success']) ?>
    </p>

    <?= GridView::widget([
        'dataProvider' => $dataProvider,

```

```

        'columns' => [
            ['class' => 'yii\grid\SerialColumn'],

            'ID',
            'USER_ID',
            'PASSKEY',
            'TIME_CREATED',

            ['class' => 'yii\grid\ActionColumn'],
        ],
    ); ?>

```

</div>

Recover.php

<?php

```
use yii\helpers\Html;
```

```
/* @var $this yii\web\View */
```

```
/* @var $model app\models\PassCh */
```

```
$this->title = 'Paroles atjaunošana';
```

?>

```
<div class="pass-ch-recover">
```

```
    <h1><?= Html::encode($this->title) ?></h1>
```

```
    <?= $this->render('recoveryForm', [
        'model' => $model,
    ]) ?>
```

</div>

recoveryForm.php

<?php

```
use yii\helpers\Html;
```

```
use yii\widgets\ActiveForm;
```

```
use yii\captcha\Captcha;
```

```
/* @var $this yii\web\View */
```

```
/* @var $model app\models\PassCh */
```

```
/* @var $form yii\widgets\ActiveForm */
```

?>

```
<div class="pass-ch-form col-lg-5">
```



```

<?php if (Yii::$app->session->hasFlash('linkExpired')): ?>

    <div class="alert alert-danger">
        Šī saite vairs nav derīga, lūdzu pieprasiet paroles atjaunošanu vēlreiz!
    </div>

<?php else: ?>
    <?php if (Yii::$app->session->hasFlash('successfulPasswordChange')): ?>

        <div class="alert alert-success">
            Parole veiksmīgi nomainīta, varat pieteikties sitēmā!
        </div>

    <?php else: ?>
        <?php $form = ActiveForm::begin(); ?>

        <?= $form->field($model, 'PASSWORD')->passwordInput() ?>

        <div class="form-group">
            <?= Html::submitButton('Saglabāt', ['class' => 'btn btn-success']) ?>
        </div>

        <?php ActiveForm::end(); ?>
    <?php endif; ?>
<?php endif; ?>
</div>

```

RIDES

_form.php

```

<?php

use app\models\Cities;
use yii\helpers\ArrayHelper;
use yii\helpers\Html;
use yii\widgets\ActiveForm;
use yii\jui\DatePicker;

use kartik\base\Module;
use kartik\datetime\DateTimePicker;

use dosamigos\google\maps\LatLng;
use dosamigos\google\maps\services\DirectionsWayPoint;
use dosamigos\google\maps\services\TravelMode;
use dosamigos\google\maps\overlays\PolylineOptions;
use dosamigos\google\maps\services\DirectionsRenderer;
use dosamigos\google\maps\services\DirectionsService;
use dosamigos\google\maps\overlays\InfoWindow;

```

```

use dosamigos\google\maps\overlays\Marker;
use dosamigos\google\maps\Map;
use dosamigos\google\maps\services\DirectionsRequest;
use dosamigos\google\maps\overlays\Polygon;
use dosamigos\google\maps\layers\BicyclingLayer;

use kartik\select2\Select2;

/* @var $this yii\web\View */
/* @var $model app\models\Rides */
/* @var $form yii\widgets\ActiveForm */
/* @var $cities */
/* @var $cars */
?>

<div class="rides-form col-lg-6">

    <?php $form = ActiveForm::begin(); ?>

    <?= $form->field($model, 'CITY_FROM_ID')->widget(Select2::classname(), [
        'data' => $cities,
        'options' => ['placeholder' => 'Pilsēta no kuras dosies braucienā'],
    ]); ?>

    <?= $form->field($model, 'CITY_TO_ID')->widget(Select2::classname(), [
        'data' => $cities,
        'options' => ['placeholder' => 'Pilsēta uz kuru brauksi'],
    ]); ?>

    <?= $form->field($model, 'CAR_ID')->dropDownList($cars) ?>

    <?= $form->field($model, 'DATE')->widget(DateTimePicker::classname(), [
        'name' => 'dp_1',
        //'type' => DateTimePicker::DATE_,
        'options' => [
            'class' => 'form-control',
            'autocomplete' => 'off',
            'readonly' => true
        ],
        'language' => 'lv',
        'pluginOptions' => [
            'autoclose' => true,
            'startDate' => date('d-m-Y H:i', time() + 7200), // + 2 stundas
            'autocomplete' => 'off',
            'format' => 'dd-mm-yyyy hh:ii',
            'timeZone' => 'Europe/Riga',
        ],
    ]); ?>

```

```

],

])?>

<?= $form->field($model, 'NOTES')->textarea(['maxlength' => true, 'placeholder' => 'Ieraksti
precīzu izbraukšanas vietu, ja vēlies, tad arī atlīdzību par braucienu']) ?>

<div class="form-group">
    <?= Html::submitButton('Pievienot', ['class' => 'btn btn-success']) ?>
</div>

<?php ActiveForm::end(); ?>

</div>
_search.php
<?php

use yii\helpers\Html;
use yii\widgets\ActiveForm;

use kartik\datetime\DateTimePicker;
use kartik\select2\Select2;
/* @var $this yii\web\View */
/* @var $model app\models\RidesSearch */
/* @var $form yii\widgets\ActiveForm */
/* @var $cityArray yii\widgets\ActiveForm */
?>

<div class="rides-search">

    <?php $form = ActiveForm::begin([
        'action' => ['index'],
        'method' => 'post',
    ]); ?>

    <?= $form->field($model, 'CITY_FROM_ID')->widget(Select2::classname(), [
        'data' => $cityArray,
        'options' => ['placeholder' => 'Pilsēta no kuras vēlies doties (Atstāj tukšu, lai redzētu visus
braucienus)'],
    ]); ?>

    <?= $form->field($model, 'CITY_TO_ID')->widget(Select2::classname(), [
        'data' => $cityArray,

```

```

        'options' => ['placeholder' => 'Pilsēta uz kuru vēlies doties (Atstāj tukšu, lai redzētu visus
braucienus)'],
    ); ?>

```

```

<?= $form->field($model, 'DATE')->widget(DateTimePicker::classname(), [
    'options' => [
        'readonly' => true
    ],
    'language' => 'lv',
    'pluginOptions' => [
        'autoclose' => true,
        'startDate' => '+0d',
        'autocomplete' => 'off',
        'format' => 'dd-mm-yyyy',
        'timeZone' => 'Europe/Riga',
        'minView' => 2,
    ],
])?>

```

```

<div class="form-group">
    <?= Html::submitButton('Atrast braucienu', ['class'=>'btn btn-primary']) ?>
</div>

```

```

<?php ActiveForm::end(); ?>

```

```

</div>

```

Create.php

```

<?php

```

```

use yii\helpers\Html;
use kartik\datetime\DateTimePicker;
/* @var $this yii\web\View */
/* @var $model app\models\Rides */

```

```

$this->title = 'Pievieno braucienu';
?>

```

```

<div class="rides-create">

```

```

    <?= $this->render('_form', [
        'model' => $model,
        'cities' => $cities,
        'cars' => $cars,
    ]) ?>

```

```

</div>

```

Index.php

```

<?php

```

```

use yii\helpers\Html;
use yii\grid\GridView;
use app\models\UserRides;
use yii\web\UrlManager;
use yii\helpers\Url;

/* @var $this yii\web\View */
/* @var $searchModel app\models\RidesSearch */
/* @var $dataProvider yii\data\ActiveDataProvider */

$this->title = 'Braucieni';
?>
<div class="rides-index">

    <h1><?= Html::encode($this->title) ?></h1>

    <p>
        <?= Html::a('Piedāvā braucienus', ['create'], ['class' => 'btn btn-success']) ?>
        <?= Html::a('Atrast citu', ['search'], ['class' => 'btn btn-primary']) ?>
    </p>

    <?php // echo $this->render('_search', ['model' => $searchModel]); ?>

    <?= GridView::widget([
        'dataProvider' => $dataProvider,
        // 'filterModel' => $searchModel,
        'columns' => [
            ['class' => 'yii\grid\SerialColumn'],
            ['header' => 'Pilsēta no', 'attribute' => 'cITYFROM.name'],
            ['header' => 'Pilsēta uz', 'attribute' => 'cITYTO.name'],
            'DRIVER_ID' => 'driverName.NAME',
            'DATE',
            [
                'header' => 'Vietu skaits',
                'value' => function($model){
                    return $model->getSeatCount();
                }
            ],
            [
                'class' => 'yii\grid\ActionColumn',
                'template' => '{myButton}',
                'buttons' => [
                    'myButton' => function($url, $dataProvider, $key) {

                        return Html::a('Apskatīt', ['view', 'id' => $dataProvider->ID], [

```

```

        'class' => 'btn btn-success']);
    }
    ]
    ],
    ],
    ); ?>

```

</div>

Myrides.php

```

<?php

use yii\helpers\Html;
use yii\grid\GridView;
use yii\web\UrlManager;
use yii\helpers\Url;
/* @var $this yii\web\View */
/* @var $searchModel app\models\RidesSearch */
/* @var $dataProvider yii\data\ActiveDataProvider */

$this->title = 'Braucieni';
?>
<div class="rides-index">

    <h1><?= Html::encode($this->title) ?></h1>

    <p>
        <?= Html::a('Piedāvā braucienu', ['create'], ['class' => 'btn btn-success']) ?>
    </p>

    <?php // echo $this->render('_search', ['model' => $searchModel]); ?>

    <?= GridView::widget([
        'dataProvider' => $dataProvider,
        //filterModel' => $searchModel,
        'columns' => [
            ['class' => 'yii\grid\SerialColumn'],
            ['header' => 'pilsēta no', 'attribute' => 'cITYFROM.name'],
            ['header' => 'pilsēta uz', 'attribute' => 'cITYTO.name'],
            'DATE',
            [
                'class' => 'yii\grid\ActionColumn',
                'template' => '{myButton}',
                'buttons' => [
                    'myButton' => function($url, $dataProvider, $key) {
                        return Html::a('Apskatīt', ['view', 'id' => $dataProvider->ID], [

```

```

        'class' => 'btn btn-success']);
    }
]
],
'emptyText' => 'Ups, izskatās, ka neesi pievienojis nevienu braucienu!',
]); ?>

```

</div>

Update.php

<?php

```
use yii\helpers\Html;
```

```
/* @var $this yii\web\View */
```

```
/* @var $model app\models\Rides */
```

```
$this->title = 'Update Rides: ' . $model->ID;
```

```
$this->params['breadcrumbs'][] = ['label' => 'Rides', 'url' => ['index']];
```

```
$this->params['breadcrumbs'][] = ['label' => $model->ID, 'url' => ['view', 'id' => $model->ID]];
```

```
$this->params['breadcrumbs'][] = 'Update';
```

```
?>
```

```
<div class="rides-update">
```

```
    <h1><?= Html::encode($this->title) ?></h1>
```

```
    <?= $this->render('_form', [
```

```
        'model' => $model,
```

```
    ]) ?>
```

</div>

View.php

<?php

```
use yii\helpers\Html;
```

```
use yii\widgets\DetailView;
```

```
use app\models\UserRides;
```

```
use dosamigos\google\maps\LatLng;
```

```
use dosamigos\google\maps\services\TravelMode;
```

```
use dosamigos\google\maps\overlays\PolylineOptions;
```

```
use dosamigos\google\maps\services\DirectionsRenderer;
```

```
use dosamigos\google\maps\services\DirectionsService;
```

```

use dosamigos\google\maps\overlays\InfoWindow;
use dosamigos\google\maps\overlays\Marker;
use dosamigos\google\maps\Map;
use dosamigos\google\maps\services\DirectionsRequest;

/* @var $this yii\web\View */
/* @var $model app\models\Rides */
/* @var $car app\models\Car */

\yii\web\YiiAsset::register($this);
?>
<div class="rides-view col-lg-6">

    <h1><?= Html::encode($this->title) ?></h1>

    <p>
        <?php
            $userRides = new UserRides();
            $userRides->USER_ID = Yii::$app->user->getId();
            $userRides->RIDE_ID = $model->ID;

            if($model->DRIVER_ID != Yii::$app->user->getId()){
                if ($userRides->isJoined() == 0){
                    if($model->getSeatCount() > 0){
                        echo Html::a('Pieteikties', ['join','id' => $model->ID], [
                            'class' => 'btn btn-success']);
                    }
                }else{
                    if(!$userRides->hasCancelled()){
                        echo Html::a('Atteikties', ['/user-rides/cancel','RIDE_ID' => $model->ID,
'USER_ID' => Yii::$app->user->getId()], [
                            'class' => 'btn btn-danger',
                            'data' => [
                                'confirm' => 'Vai tiešām vēlies atteikties no brauciena?
Pēc tam vairs nevarēsi pievienoties!',
                                'method' => 'post',
                            ],
                        ]);
                    }
                }else{
                    if(!$model->hasRiders()){
                        echo Html::a('Dzēst braucienu', ['delete', 'id' => $model->ID], [
                            'class' => 'btn btn-danger',
                            'data' => [
                                'confirm' => 'Are you sure you want to delete this item?',

```



```

        'method' => 'post',
    ],
    );
}

}

?>
</p>
<?= DetailView::widget([
    'model' => $model,
    'attributes' => [
        [
            'label' => 'Pilsēta no',
            'attribute' => 'cITYFROM.name',
        ],
        [
            'label' => 'Pilsēta uz',
            'attribute' => 'cITYTO.name',
        ],
        [
            'label' => 'Epasts',
            'attribute' => 'uSER.EMAIL',
        ],
        'DATE',
        'NOTES',
    ],
    );
] ?>

<?php echo Html::img('@web/uploads/'.$scar->image_name, [
    'alt' => 'pic not found',
    'width' => '500px',
    'height' => '400px'
    ]
    );?>
</div>

```

```

<?php

```

```

$geo_coding_client = new \dosamigos\google\maps\services\GeocodingClient();

```

```

$map = new \dosamigos\google\maps\Map([
    'center' => new \dosamigos\google\maps\LatLng(['lat' => 52.1326, 'lng' => 5.2913]),
    'zoom' => 7,
]);

```

```

$lookup_response = $geo_coding_client->lookup([
    'address' => $model->cITYFROM->name,
    'region' => 'Latvia',
]);

$lat_from = isset($lookup_response['results'][0]['geometry']['location']['lat']) ?
$lookup_response['results'][0]['geometry']['location']['lat'] : null;
$lng_from = isset($lookup_response['results'][0]['geometry']['location']['lng']) ?
$lookup_response['results'][0]['geometry']['location']['lng'] : null;

$lookup_response = $geo_coding_client->lookup([
    'address' => $model->cITYTO->name,
    'region' => 'Latvia',
]);

$lat_to = isset($lookup_response['results'][0]['geometry']['location']['lat']) ?
$lookup_response['results'][0]['geometry']['location']['lat'] : null;
$lng_to = isset($lookup_response['results'][0]['geometry']['location']['lng']) ?
$lookup_response['results'][0]['geometry']['location']['lng'] : null;

$lat = isset($lookup_response['results'][0]['geometry']['location']['lat']) ?
$lookup_response['results'][0]['geometry']['location']['lat'] : null;

$coord = new LatLng(['lat' => $lat_from, 'lng' => $lat_from]);
$map = new Map([
    'center' => $coord,
    'zoom' => 7,
]);

// lets use the directions renderer
$from = new LatLng(['lat' => $lat_from, 'lng' => $lng_from]);
$to = new LatLng(['lat' => $lat_to, 'lng' => $lng_to]);

$directionsRequest = new DirectionsRequest([
    'origin' => $from,
    'destination' => $to,
    'travelMode' => TravelMode::DRIVING
]);

// Lets configure the polyline that renders the direction
$polylineOptions = new PolylineOptions([
    'strokeColor' => '#FFAA00',
    'draggable' => true
]);

```

```
// Now the renderer
$directionsRenderer = new DirectionsRenderer([
    'map' => $map->getName(),
    'polylineOptions' => $polylineOptions
]);

// Finally the directions service
$directionsService = new DirectionsService([
    'directionsRenderer' => $directionsRenderer,
    'directionsRequest' => $directionsRequest
]);

// Thats it, append the resulting script to the map
$map->appendScript($directionsService->getJs());

// Lets add a marker now
$marker = new Marker([
    'position' => $coord,
    'title' => 'My Home Town',
]);

// Display the map -finally :)
echo $map->display();
?>
```

```
</div>
```

SITE

Contact.php

```
<?php
```

```
/* @var $this yii\web\View */
/* @var $form yii\bootstrap\ActiveForm */
/* @var $model app\models\ContactForm */

use yii\helpers\Html;
use yii\bootstrap\ActiveForm;
use yii\captcha\Captcha;

$this->title = 'Saziņa';
// $this->params['breadcrumbs'][] = $this->title;
?>
<div class="site-contact">
    <h1><?= Html::encode($this->title) ?></h1>
```

```

<?php if (Yii::$app->session->hasFlash('contactFormSubmitted')): ?>

    <div class="alert alert-success">
        Paldies, ka uzrakstījāt. Mēs drīz sazināsimies
    </div>

<?php else: ?>

<?php
    if(!Yii::$app->user->isGuest){
        $model->name = Yii::$app->user->identity->NAME;
        $model->email = Yii::$app->user->identity->EMAIL;
    }
?>

<p>
    Ja jums ir kādi jautājumi lūdzu aizpildiet šo formu, lai sazinātos
</p>

<div class="row">
    <div class="col-lg-5">

        <?php $form = ActiveForm::begin(['id' => 'contact-form']); ?>

        <?= $form->field($model, 'name')->textInput(['autofocus' => true]) ?>

        <?= $form->field($model, 'email') ?>

        <?= $form->field($model, 'subject') ?>

        <?= $form->field($model, 'body')->textarea(['rows' => 6]) ?>

        <?= $form->field($model, 'verifyCode')->widget(Captcha::className(), [
            'template' => '<div class="row"><div class="col-lg-3">{image}</div><div
class="col-lg-6">{input}</div></div>',
        ]) ?>

        <div class="form-group">
            <?= Html::submitButton('Nosūtīt', ['class' => 'btn btn-primary', 'name' =>
'contact-button']) ?>
        </div>

    <?php ActiveForm::end(); ?>

```

```

        </div>
    </div>

    <?php endif; ?>
</div>
Error.php
<?php

/* @var $this yii\web\View */
/* @var $name string */
/* @var $message string */
/* @var $exception Exception */

use yii\helpers\Html;

$this->title = $name;
?>
<div class="site-error">
    <h1><?= Html::encode($this->title) ?></h1>

    <div class="alert alert-danger">
        <?= nl2br(Html::encode($message)) ?>
    </div>
    <?php
    Yii::$app->mailer->compose()
        ->setTo('raimondsl99@gmail.com')
        ->setFrom(Yii::$app->params['senderEmail'])
        ->setReplyTo(Yii::$app->params['senderEmail'])
        ->setSubject('Kļūda produkcijā')
        ->setTextBody('Kļūda produkcijā '.$this->title)
        ->send();
    ?>

    <div class="text-center">
        <?php echo Html::img('@web/error.png', [
            'alt' => 'pic not found',
            'width' => '400px',
            'height' => '400px'
        ])
        ?>
    </div>

</div>
Index.php
<?php

```

```

/* @var $this yii\web\View */
use yii\helpers\Html;
$this->title = 'My Yii Application';
?>
<div class="site-index background-image">

    <div class="jumbotron main-font">
        <h1>Kopbraukšana!</h1>

    </div>

    <div class="body-content main-font">

        <div class="row">
            <div class="col-lg-4">
            </div>
            <div class="col-lg-4">
                <h2 class="main-font-heading">Atrodi braucienu!</h2>

                <p>Vēlies ietaupīt naudu un laiku, lai nokļūtu no vienas pilsētas uz citu? Spied uz
                pogas zemāk, lai atrastu braucienu.</p>

                <?= Html::a('Atrodi braucienu', ['/rides/search'], ['class'=>'btn btn-primary']) ?>
            </div>
        </div>
        <div class="row">
            <div class="col-lg-4">
            </div>
            <div class="col-lg-4">
                <h2 class="main-font-heading">Piedāvā braucienu!</h2>

                <p>Vēlies atrast kopbraucēju, lai
                samazinātu sava ceļa izmaksas? </p>
                <?= Html::a('Piedāvā braucienu', ['/rides/create'], ['class'=>'btn btn-primary']) ?>
            </div>
        </div>

    </div>
</div>
Login.php
<?php

/* @var $this yii\web\View */
/* @var $form yii\bootstrap\ActiveForm */
/* @var $model app\models\LoginForm */

```

```

use yii\helpers\Html;
use yii\bootstrap\ActiveForm;

$this->title = 'Pieslēgties';
?>
<div class="site-login">

    <h1><?= Html::encode($this->title) ?></h1>

    <p>Lūdzu aizpildiet redzamos laukus, lai pieslēgtos:</p>

    <?php $form = ActiveForm::begin([
        'id' => 'login-form',
        'layout' => 'horizontal',
        'fieldConfig' => [
            'template' => "{label}\n<div class=\"col-lg-3\">{input}</div>\n<div class=\"col-lg-8\">{error}</div>",
            'labelOptions' => ['class' => 'col-lg-1 control-label'],
        ],
    ]); ?>

    <?= $form->field($model, 'username')->textInput(['autofocus' => true]) ?>

    <?= $form->field($model, 'password')->passwordInput() ?>

    <?= $form->field($model, 'rememberMe')->checkbox([
        'template' => "<div class=\"col-lg-offset-1 col-lg-3\">{input} {label}</div>\n<div class=\"col-lg-8\">{error}</div>",
    ]) ?>

    <div class="form-group">
        <div class="col-lg-offset-1 col-lg-11">
            <?= Html::a('Aizmirsi paroli?', ['/pass-ch/create']) ?>
        </div>
        <div class="col-lg-offset-1 col-lg-11">
            <?= Html::submitButton('Pieslēgties', ['class' => 'btn btn-success', 'name' => 'login-button']) ?>
            <?= Html::a('Reģistrēties', ['/site/register'], ['class' => 'btn btn-primary']) ?>
        </div>
    </div>

    <?php ActiveForm::end(); ?>

</div>
Register.php
<?php

```

```

use yii\helpers\Html;
use yii\widgets\ActiveForm;

/* @var $this yii\web\View */
/* @var $model app\models\USER */
/* @var $form yii\widgets\ActiveForm */
?>

<div class="user-form">

    <?php $form = ActiveForm::begin(); ?>

    <?= $form->field($model, 'USERNAME')->textInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'NAME')->textInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'SURNAME')->textInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'PASSWORD')->passwordInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'PHONE')->textInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'EMAIL')->textInput(['maxlength' => true]) ?>

    <div class="form-group">
        <?= Html::submitButton('Reģistrēties', ['class' => 'btn btn-success', 'name'=>'register']) ?>
    </div>

    <?php ActiveForm::end(); ?>

</div>

```

USER-RIDES

_form.php

```

<?php

use yii\helpers\Html;
use yii\widgets\ActiveForm;

/* @var $this yii\web\View */
/* @var $model app\models\UserRides */
/* @var $form yii\widgets\ActiveForm */
?>

<div class="user-rides-form">

```



```

<?php $form = ActiveForm::begin(); ?>

<?= $form->field($model, 'USER_ID')->textInput() ?>

<?= $form->field($model, 'RIDE_ID')->textInput() ?>

<?= $form->field($model, 'JOIN_DATE')->textInput() ?>

<div class="form-group">
    <?= Html::submitButton('Save', ['class' => 'btn btn-success']) ?>
</div>

<?php ActiveForm::end(); ?>

```

```

</div>

```

Create.php

```

<?php

use yii\helpers\Html;

/* @var $this yii\web\View */
/* @var $model app\models\UserRides */

$this->title = 'Create User Rides';
$this->params['breadcrumbs'][] = ['label' => 'User Rides', 'url' => ['index']];
$this->params['breadcrumbs'][] = $this->title;
?>
<div class="user-rides-create">

    <h1><?= Html::encode($this->title) ?></h1>

    <?= $this->render('_form', [
        'model' => $model,
    ]) ?>

```

```

</div>

```

Index.php

```

<?php

use yii\helpers\Html;
use yii\grid\GridView;

/* @var $this yii\web\View */
/* @var $dataProvider yii\data\ActiveDataProvider */

$this->title = 'User Rides';

```

```

$this->params['breadcrumbs'][] = $this->title;
?>
<div class="user-rides-index">

    <h1><?= Html::encode($this->title) ?></h1>

    <p>
        <?= Html::a('Create User Rides', ['create'], ['class' => 'btn btn-success']) ?>
    </p>

    <?= GridView::widget([
        'dataProvider' => $dataProvider,
        'columns' => [
            ['class' => 'yii\grid\SerialColumn'],

            'USER_ID',
            'RIDE_ID',
            'JOIN_DATE',

            ['class' => 'yii\grid\ActionColumn'],
        ],
    ]); ?>

</div>
Update.php
<?php

use yii\helpers\Html;

/* @var $this yii\web\View */
/* @var $model app\models\UserRides */

$this->title = 'Update User Rides: ' . $model->USER_ID;
$this->params['breadcrumbs'][] = ['label' => 'User Rides', 'url' => ['index']];
$this->params['breadcrumbs'][] = ['label' => $model->USER_ID, 'url' => ['view', 'USER_ID' =>
$model->USER_ID, 'RIDE_ID' => $model->RIDE_ID]];
$this->params['breadcrumbs'][] = 'Update';
?>
<div class="user-rides-update">

    <h1><?= Html::encode($this->title) ?></h1>

    <?= $this->render('_form', [
        'model' => $model,

```

```

    }) ?>

</div>
View.php
<?php

use yii\helpers\Html;
use yii\widgets\DetailView;

/* @var $this yii\web\View */
/* @var $model app\models\UserRides */

$this->title = $model->USER_ID;
$this->params['breadcrumbs'][] = ['label' => 'User Rides', 'url' => ['index']];
$this->params['breadcrumbs'][] = $this->title;
\yii\web\YiiAsset::register($this);
?>
<div class="user-rides-view">

    <h1><?= Html::encode($this->title) ?></h1>

    <p>
        <?= Html::a('Update', ['update', 'USER_ID' => $model->USER_ID, 'RIDE_ID' => $model-
>RIDE_ID], ['class' => 'btn btn-primary']) ?>
        <?= Html::a('Delete', ['delete', 'USER_ID' => $model->USER_ID, 'RIDE_ID' => $model-
>RIDE_ID], [
            'class' => 'btn btn-danger',
            'data' => [
                'confirm' => 'Are you sure you want to delete this item?',
                'method' => 'post',
            ],
        ]) ?>
    </p>

    <?= DetailView::widget([
        'model' => $model,
        'attributes' => [
            'USER_ID',
            'RIDE_ID',
            'JOIN_DATE',
        ],
    ]) ?>

</div>

CSS

```

Site.css

```
@import
url('https://fonts.googleapis.com/css2?family=Bitter:wght@700&family=Roboto+Mono:wght@
700&display=swap');
html,
body {
    height: 100%;
}

.wrap {
    min-height: 100%;
    height: auto;
    margin: 0 auto -60px;
    padding: 0 0 60px;
}

.wrap > .container {
    padding: 70px 15px 20px;
}

.footer {
    height: 60px;
    background-color: #f5f5f5;
    border-top: 1px solid #ddd;
    padding-top: 20px;
}

.jumbotron {
    text-align: center;
    background-color: transparent;
}

.jumbotron .btn {
    font-size: 21px;
    padding: 14px 24px;
}

.not-set {
    color: #c55;
    font-style: italic;
}

/* add sorting icons to gridview sort links */
a.asc:after, a.desc:after {
    position: relative;
    top: 1px;
```

```

display: inline-block;
font-family: 'Glyphicons Halflings';
font-style: normal;
font-weight: normal;
line-height: 1;
padding-left: 5px;
}

a.asc:after {
  content: /*"\e113"*/ "\e151";
}

a.desc:after {
  content: /*"\e114"*/ "\e152";
}

.sort-numerical a.asc:after {
  content: "\e153";
}

.sort-numerical a.desc:after {
  content: "\e154";
}

.sort-ordinal a.asc:after {
  content: "\e155";
}

.sort-ordinal a.desc:after {
  content: "\e156";
}

.grid-view th {
  white-space: nowrap;
}

.hint-block {
  display: block;
  margin-top: 5px;
  color: #999;
}

.error-summary {
  color: #a94442;
  background: #fdf7f7;
  border-left: 3px solid #eed3d7;

```

```

padding: 10px 20px;
margin: 0 0 15px 0;
}

/* align the logout "link" (button in form) of the navbar */
.nav li > form > button.logout {
padding: 15px;
border: none;
}

@media(max-width:767px) {
.nav li > form > button.logout {
display: block;
text-align: left;
width: 100%;
padding: 10px 15px;
}
}

.nav > li > form > button.logout:focus,
.nav > li > form > button.logout:hover {
text-decoration: none;
}

.nav > li > form > button.logout:focus {
outline: none;
}

.navbar-custom{
background-color: green !important;
}

.navbar-inverse > li > a{
color: green !important;
}

.background-image{
background-image: url("../background.jpg");
position: absolute;
top: 0;
left: 0;
right: 0;
height: 100%;
width: auto;
background-repeat: no-repeat;
background-size: cover;
background-position:center;
}

```

```
.main-font, main-font-heading{
    font-family: 'Bitter', serif;
    font-family: 'Roboto Mono', monospace;
    font-size: 20px;
    color: white;
    text-shadow: -1px -1px 0 #000, 1px -1px 0 #000, -1px 1px 0 #000, 1px 1px 0 #000;
    text-align: center;
}
.main-font-heading{
    font-size: 35px;
}
```

PROFILE

_form.php

```
<?php

use yii\helpers\Html;
use yii\widgets\ActiveForm;

/* @var $this yii\web\View */
/* @var $model app\models\USER */
/* @var $form yii\widgets\ActiveForm */
?>

<div class="user-form">

    <?php $form = ActiveForm::begin(); ?>

    <?= $form->field($model, 'USERNAME')->textInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'NAME')->textInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'SURNAME')->textInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'EMAIL')->textInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'PHONE')->textInput(['maxlength' => true]) ?>

    <div class="form-group">
        <?= Html::submitButton('Saglabāt', ['class' => 'btn btn-success']) ?>
        <?= Html::a('Mainīt paroli', ['updatepassword', 'id' => Yii::$app->user->getId()], ['class' =>
'btn btn-primary']) ?>
    </div>

    <?php ActiveForm::end(); ?>
```

```

</div>
_passwordForm.php
<?php

use yii\helpers\Html;
use yii\widgets\ActiveForm;

/* @var $this yii\web\View */
/* @var $model app\models\USER */
/* @var $form yii\widgets\ActiveForm */
?>

<div class="user-form">

    <?php $form = ActiveForm::begin(); ?>

    <?= $form->field($model, 'oldPassword')->passwordInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'newPassword')->passwordInput(['maxlength' => true]) ?>

    <?= $form->field($model, 'repeatPassword')->passwordInput(['maxlength' => true]) ?>

    <div class="form-group">
        <?= Html::submitButton('Saglabāt', ['class' => 'btn btn-success']) ?>
    </div>

    <?php ActiveForm::end(); ?>

</div>
Update.php
<?php

use yii\helpers\Html;

/* @var $this yii\web\View */
/* @var $model app\models\USER */

$this->title = 'Profils: ' . $model->NAME;
?>
<div class="user-update">

    <h1><?= Html::encode($this->title) ?></h1>

    <?= $this->render('_form', [
        'model' => $model,
    ]) ?>

```



```

    ]) ?>

</div>
updatePassword.php
<?php

use yii\helpers\Html;

/* @var $this yii\web\View */
/* @var $model app\models\USER */

$this->title = 'Paroles maina';
?>
<div class="user-update">

    <h1><?= Html::encode($this->title) ?></h1>

    <?= $this->render('_passwordForm', [
        'model' => $model,
    ]) ?>

</div>
view.php
<?php

use yii\helpers\Html;
use yii\widgets\DetailView;

/* @var $this yii\web\View */
/* @var $model app\models\USER */

$this->title = $model->NAME;

\yii\web\YiiAsset::register($this);
?>
<div class="user-view">

    <h1><?= Html::encode($this->title) ?></h1>

    <p>
        <?= Html::a('Labot profila datus', ['update', 'id' => $model->ID], ['class' => 'btn btn-
primary']) ?>
    </p>

    <?= DetailView::widget([
        'model' => $model,

```

```
'attributes' => [  
  'USERNAME',  
  'NAME',  
  'SURNAME',  
  'EMAIL:email',  
  'PHONE',  
],  
) ?>  
</div>
```