**VENTSPILS AUGSTSKOLA**

**INFORMĀCIJAS TEHNOLOĢIJU FAKULTĀTE**

**BAKALAURA DARBS**

**Docētāju noslodzes pārvaldības sistēmas izveide.**

Autors Ventspils Augstskolas

Informācijas tehnoloģiju fakultātes

bakalaura studiju programmas „Datorzinātnes”

3. kursa students

**Endijs Bertāns**

Matr.nr. 22020008

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(paraksts)

Fakultātes dekāns doc. Dr.sc.comp. Vairis Caune

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(paraksts)

Zinātniskais vadītājs \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ieņemamais amats, zinātniskais nosaukums, vārds, uzvārds)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(paraksts)

**ANOTĀCIJA**

**Darba nosaukums:** Docētāju noslodzes pārvaldības sistēmas izveide.

**Darba autors:** Endijs Bertāns

**Darba vadītājs:** doc. Dr.sc.comp. Vairis Caune

**Darba apjoms:** \* lpp., \* tabulas, \* attēli, \* bibliogrāfiskie norādes, \* pielikumi.

**Atslēgas vārdi:** SPRING BOOT, SPRING SECURITY, JWT, ANGULAR, AWS.

Bakalaura darba mērķis ir izstrādāt docētāju slodzes pārvaldības sistēmu pēc Informācijas tehnoloģiju fakultātes dekāna prasībām.

Sistēmas izstrādei izmantots Java Spring Boot ietvars kopā ar Spring Data JPA, Spring Security un Spring Boot Starter Validation bibliotēkām, kā arī datu bāzes sistēma MySQL. Lietotāja saskarne izstrādāta, izmantojot Angular. Sistēmas darbība palaista Docker konteinerizētā vidē, tādejādi sistēmu var palaist uz jebkuras operētājsistēmas, kurā atrodas docker.

Darba gaitā izstrādāta tīmekļa lietotne, kas ļauj plānot un pārvaldīt docētāju slodzes.

**ABSTRACT**

**Title:** Development of a workload management system for teaching staff.

**Author:** Endijs Bertāns

**Academic Advisor:** doc. Dr.sc.comp. Vairis Caune

**Volume of the work:** \* pages, \* tables, \* images, \* equations, \* literature

**Keywords: S**PRING BOOT, SPRING SECURITY, JWT, ANGULAR, AWS.

The aim of the bachelor thesis is to develop a workload management system for teaching staff according to the requirements of the Dean of the Faculty of Information Technology.

The system is developed using the Java Spring Boot framework with Spring Data JPA, Spring Security and Spring Boot Starter Validation libraries, as well as the MySQL database system. The user interface was developed using Angular. The system runs in a Docker containerised environment.

A web application has been developed to allow the scheduling and management of lecturers' workloads.

Saturs

[Ievads 5](#_Toc196129858)

[1. Esošās sistēmas analīze 6](#_Toc196129859)

[2. Prasību analīze. 8](#_Toc196129860)

[3. Tehnoloģiju salīdzinājums un izvēle sistēmas izstrādei 9](#_Toc196129861)

[4. Jaunās sistēmas izstrāde 10](#_Toc196129862)

[4.1 Datu bāzes modeļa izstrāde 10](#_Toc196129863)

[4.2 Aizmugursistēmas izveide 13](#_Toc196129864)

[IZMANTOTĀS LITERATŪRAS UN AVOTU SARAKSTS 15](#_Toc196129865)

[Pielikumi 16](#_Toc196129866)

[GALVOJUMS 18](#_Toc196129867)

# Ievads

Mūsdienās valsts un privātām iestādēm arvien būtiskāka kļūst efektīva, viegla un droša procesu pārvaldība, lai šos procesus atvieglotu un paātrinātu. Ventspils Augstskolas semestra plānu apstiprināšanas process notiek relatīvi manuāli – semestra plānus veido excel izklājlapā, kur atrodas tabula ar 41 kolonu, katru mācību gadu veidojot jaunu izklājlapu, kas var novest pie aizkavējumiem un lielas darbinieku slodzes. Tā rezultātā palielinās kļūdu risks, un izmaksas. Šīs problēmas risināšanai ir plānots izstrādāt docētāju noslodzes pārvaldības sistēmu, kuras mērķis ir automatizēt semestra plānu veidošanas procesu, nodrošināt labāku informācijas apmaiņu starp sistēmas lietotājiem, piemēram, docētājiem un dekāniem, kā arī nodrošināt datu saglabāšanu vienā sistēmā, lai varētu viegli pārskatīt citu mācību gadu datus.

Darba mērķis ir docētāju noslodzes pārvaldības sistēmas izveide, kura tiktu ieviesta vismaz vienā fakultātē. Šī sistēma sniegs iespēju digitalizēt un automatizēt semestra plānu veidošanas procesu, tādējādi ievērojami atvieglojot administratīvo darbu un samazinot kļūdu iespējamību.

Lai sasniegtu izvirzīto mērķi, pētījumā tiks veikti sekojoši uzdevumi:

* Analizēt esošo semestra plānu izveides procesu.
* Modelēt un izstrādāt datubāzi, pēc esošo excel izklājlapu, piemēriem.
* Analizēt dažādas datu bāzes, lai secinātu kuru šai sistēmai izmantot būs vispiemērotāk.
* Realizēt projektēto datubāzi.
* Manuāli ievadīt paragdatus no izklajlapām, lai pārliecinātos, ka tiek saglabāta esošā funkcionalitāte un visiem datiem no izklajlapas ir vieta datubāzē.
* Analizēt dažādas aizmugursistēmas.
* Izveidot aizmugursistēmu saskarnei ar datubāzes tabulām - datu izveidei, lasīšanai, rediģēšanai un dzēšanai.
* Analizēt pieejamos drošības risinājumus izvēlētajām sistēmām.
* Ieviest sistēmā drošības risinājumus.
* Analizēt dažādus risinājumus, sistēmas saskarnes izveidei.
* Izmantojot atlasīto saskarnes ietvaru, veidot pieslēgšanos sistēmai, izmantojot izvēlēto drošības risinājumu
* Izveidot datu izvades / ievades saskarni.
* Izveidot sistēmai automātisku palaišanu uz serveri, pēc jaunāko izmaņu nosūtīšanu versiju kontroles rīkam.

Darba struktūra ir loģiski sadalīta vairākās daļās, sākot ar teorētisko daļu, kurā apskatīti semestra plānu izveides procesa esošie risinājumi un to ierobežojumi, turpmāk detalizēti izklāstīta jaunās sistēmas projektēšana un izstrāde, un beidzoties ar praktisko sistēmas izveidi. Pētījuma temats ir norobežots ar konkrētām Ventspils augstskolas vajadzībām, taču metodes un pieejas var tikt paplašinātas arī citās iestādēs ar līdzīgi strukturētām administratīvajām prasībām.

Šajā darbā izmantotās pētījuma realizācijas metodes ietver esošās sistēmas procesu analīzi, jaunu risinājumu projektēšanu un prototipa izstrādi, kā arī visaptverošu testēšanu, lai nodrošinātu sistēmas uzticamību un lietojamību reālās darba situācijās, tādejādi tiks nodrošināta administratīvo procesu digitalizācija un automatizācija Ventspils Augstskolā.

# Esošās sistēmas analīze

Lai izprastu nepieciešamās prasības un funkcionalitāti jaunajai centralizētai sistēmai nepieciešams analizēt esošo sistēmu, lai nepazaudētu funkcionalitāti, risinātu esošās problēmas un automatizētu iespējami vairāk funkciju, tādejādi uzlabojot un paātrinot semestra plānu izveides procesu.

Pašreizējā procesa ietvaros datu ievade notiek excel izklājlapā ar 41 kolonu, kur katrā rindā ir ieraksts specifiskam priekšmetam, vai tā priekšmeta teorijas daļai un, vai praktiskai daļai, kur tiek ievadīta sekojoša informācija:

* docenta vārds,
* docenta uzvārds,
* vārds un uzvārds kopā,
* amats un vārds, un uzvārds,
* kredītpunkti pilnai slodzei, kuri iegūti no docenta amatu grupas, piemēram, lektoriem kredītpunkti pilnai slodzei ir 12
* Slodzes daļa, kura nosaka formula (esošās rindas kredītpunkti \* grupu skaitu \* koeficentu) / kredītpunktiem pilnai slodzei
* Docenta amata nosaukums, piemēram, ja profesors ir ievēlēts:
  + profesors
  + asoc. Profesors
  + docents
  + asistents
  + lektors

bet, ja profesors nav ievēlēts

* viesprofesors
* viesasoc. Profesors
* viesdocents
* viesasistents
* vieslektors
* Amata grupa, piemēram,
  + profesors
  + asoc. Profesors
  + docents
  + asistents
  + lektors
* Statuss, piemēram,
  + Neievēlētie
  + Ievēlētie
* Iekļaut budžetā, piemēram,
  + 1
  + Nenotiks
  + 0
* Pasniedzēja fakultāte, piemēram,
  + ITF
  + TSF
* Semestris, piemēram,
  + Rudens,
  + Pavasaris
* Priekšmeta nosaukums, piemēram,
  + Mehānika,
  + Sensori un robotika
* Programmas koeficents, kuru aprēķina iegūstot attiecību starp kredītpunktiem un kontaktstundām (maģistriem 0.75, bakalauriem 1)
* Priekšmeta kredītpunkti
* LAIS kods
* Reģistrācija, piemēram,
  + Automātiska
  + Izvēles
* Priekšmeta fakultāte
* Programma, piemēram,
  + KNE
  + EIB
* Grupa semestra grafikam, piemēram,
  + 1KNE
  + 1EIB
* Grupa, piemēram,
  + 1EIB+1KNE+LiepU
* Grupu skaits
* KP skaits grupai
* KpxGrupasxKoef kuru aprēķina reizinot kredītpunktus ar grupu skaitum un ar koeficentu
* Komentāri, kur bieži vien norādīts vai ieraksts domāts praktiskām vai teorijas lekcijām.
* Akad. h./ nedēļā, kurš netiek aizpildīts
* Programmas daļa, piemēram,
  + Nozares profesionālās specializācijas kursi
  + Nozares (profesionālās darbības jomas) teorētiskie pamatkursi un informācijas tehnoloģiju kursi
  + Izvēles daļas kursi

Tālāk seko budžeta plānošanas kolonas

* Alga, kuru ņem no docenta amatu grupas.
* Nozares koeficents – palielina vieslektoru algu līdz 30%
* Alga mēnesī
* Vai atvaļinājums ieskaitās – uz vieslektoriem neattiecas
* Mēnešu skaits – atvaļinājuma mēneši + nostrādātie mēneši
* Algai paredzēts – mēnešu skaits \* alga mēnesī
* Budžeta pozīcija, kura nosaka budžetu no kura lektoram maksās algu.

Katra mācību gada sākumā šīs kolonas tiek manuāli aizpildītas, izmantojot iepriekšējā gada datus kā paraugu, kas prasa ievērojamu laika ieguldījumu un nav viegli pārredzams.

Šajā tabulā atrodas dati par visiem docentiem un priekšmetiem attiecīgajā mācību gadā. Tā kā dati ir daudz un tie tiek ievadīti manuāli, pastāv augsta kļūdu ievades iespēja, kas var novest pie grūti pamanāmām kļūdām. Turklāt, vēlos piebilst velreiz, ka šajā tabulā atrodas informācija par visiem docentiem, tādejādi izklājlapu publicēšana ir ierobežota, un dot iespēju docentiem pārskatīt savu individuālo mācību gada plānu ir pārāk laikietilpīgi, jo pārskats katram docentam būtu jataisa atsevišķi un manuāli, taču tas ļautu docentiem novērtēt plāna atbilstību savām prasībām un dotu iespēju ieteikt uzlabojumus, kas samazinātu kļūdainību risku un palielinātu procesa caurredzamību.

Datu pārskatīšanai un analīzei tiek izmantotas izklājlapu iebūvētās filtrēšanas funkcijas (piemēram, “Sākas ar”, “Beidzas ar”, “Ietilpst”, “Ir vienāds”), kas ļauj atlasīt tikai tās rindas, kurām atbilst noteikts kritērijs.

Kopsavilkumā, esošā semestra plāna izveides procesa galvenās problēmas saistītas ar manuālu datu ievadi, informācijas fragmentāciju un nepietiekamu validācijas mehānismu. Automatizēta, centralizēta un lietotājam draudzīga sistēma varētu ievērojami uzlabot procesa caurspīdību, samazināt kļūdu iespējamību un optimizēt resursu izmantošanu, tādējādi nodrošinot efektīvāku administratīvo procesu izpildi.

# Prasību analīze.

Pamatojoties uz esošās sistēmas analīzi, varu identificēt un formulēt galvenās prasības jaunās centralizētās sistēmas izstrādei. Mērķis ir izveidot risinājumu, kas automatizē datu ievadi, nodrošina datu validāciju pirms ievades un piedāvā ievades laukiem piemērus, lai lietotājam būtu skaidrs, kādi dati ir jāievada. Pēc datu ievades sistēmai jāspēj automātiski veikt aprēķinus un aizpildīt attiecīgos laukus, tādējādi samazinot manuālo darbu un potenciālo kļūdu iespējamību.

Sistēmai jānodrošina iespēja importēt datus no izklājlapām (piemēram, Excel formātā), kas ļaus nodrošināt vienkāršu un ātru pāreju no pašreizējās sistēmas uz jauno risinājumu.

Jaunajai sistēmai jābūt pielāgotai dažādu lietotāju grupu vajadzībām:

**Administratīvie darbinieki** – var ievadīt, dzēst un rediģēt datus, kā arī pievienot docentus.

**Docenti** – tiesības pārskatīt savu plānu, ar ierobežotu informācijas daudzumu, bet bez iespējas veikt izmaiņas pašā plānā.

Svarīgs nosacījums ir spēja katra mācību gada sākumā izveidot jaunu semestri, izmantojot iepriekšējo gada semestra datus kā bāzi. Kā rezultātā iesākt jaunu mācību semestra plānu būs ātri un vienārši.

Lietotāja saskarnei jābūt intuitīvai un vizuāli līdzīgai izklājlapai, kas ļaus lietotājiem viegli pielāgoties jaunajai sistēmai. Tāpat jānodrošina iespēja filtrēt un sakārtot datus līdzīgos principos kā esošajā sistēmā.

Jāievēro industrijas labās prakses drošības aspekti, tostarp datu drošā dzēšana – izdzēstie dati netiek pilnībā dzēsti no sistēmas, bet gan paslēpti, lai tos nepieciešamības gadījumā varētu atgriezt. Tas ļaus izvairīties no nejaušas datu dzēšanas tādējādi sabojājot esošos ierakstus.

Analizējot pašreizējo mācību gada plāna izveides procesu, secināms, ka esošo pieeju, kuras pamatā ir manuāla datu ievade un Excel izklājlapu izmantošana, ir nepieciešams uzlabot. Galvenie uzdevumi ir:

* Nodrošināt datu pārbaudi,
* Nodrošināt automātisku aprēķināmo lauku aizpildi,
* Nodrošināt pieeju docentiem, ar iespēju apskatīt savu mācību gada plānu,
* Nodrošināt datu ievadi no izklājlapas
* Nodrošināt iepriekšējā mācību gada plānu kopēšanu uz esošo mācību gadu.

Jaunās centralizētās sistēmas izstrāde ļaus apvienot vairākas būtiskas funkcijas: automatizētu datu apstrādi, uzlabotu datu validāciju, lietotāju lomu diferenciāciju un dinamisku datu atjaunošanu. Tas veicinās caurredzamību un lietotāju iesaisti, dodot docentiem iespēju aktīvi piedalīties mācību gada plāna veidošanā.

# Tehnoloģiju salīdzinājums un izvēle sistēmas izstrādei

Izstrādājamās sistēmas darbībai ir nepieciešama datubāze, kur glabāt un saņemt datus. Lai identificētu populārākās un plašāk pielietotās datubāzes, izmantotoju DB-Engines Ranking [1], kas apkopo dažādu datubāzu popularitātes rādītājus, balstoties uz, meklējumu skaitu, tehnisko dokumentāciju un darba piedāvājumiem.

Saskaņā ar šīs statistikas datiem populārākās datubāzes ir:

* “Oracle”
* “MySQL”
* “Microsoft SQL Server”
* “PostgreSQL”
* “MongoDB”

Neraugoties uz Oracle datubāzes popularitāti un izplatību korporatīvajā vidē, tās ieviešana prasa ievērojamus resursus un licencēšanas izmaksas, kas šajā projektā nav pamatoti. Līdzīga situācija attiecas uz Microsoft SQL Server, kurš gan nodrošina plašu funkcionalitāti, bet ir komerciāls risinājums “MongoDB” kā “NoSQL” datubāze ir labi piemērota dokumentu tipa datiem. Tomēr, ņemot vērā šī projekta prasības, piemērotākas ir relāciju datubāzes, tātad izvēle ir starp “MySQL” un “PostgreSQL”, izvēlēšos populārāko jeb “MySQL”.

Lai nodrošinātu drošu datu lasīšanu, rakstīšanu, rediģēšanu un dzēšanu. nepieciešama aizmugursistēma, jeb saskarne ar datubāzes datiem. Lai gan nav viena universāla reitinga, kas precīzi atspoguļotu aizmugursistēmu ietvaru popularitāti 2025. gadā, izplatītākie ietvari ir [3]:

* Spring Boot (Java),
* Express.js (Node.js),
* Django (Python),
* Ruby on Rails (Ruby)
* ASP.NET Core (C#).

“Spring Boot” ir Java ietvars, kas izstrādāts, lai vienkāršotu sarežģītu, uzņēmuma līmeņa lietojumprogrammu izstrādi. Tam ir automatizēta konfigurācija un iebūvētās funkcionalitātes, kā drošība ar “Spring security”, datubāžu saskarne ar “Spring data JPA” uc., kā arī man patīk “Spring” anotāciju pieeja, lai samazinātu koda daudzumu. Taču salīdzinot ar citiem ietvariem, “Spring Boot” prasa lielākus sistēmas resursus, un to ir grūtāk izmantot.

Express.js piedāvā ātru un vienkāršu risinājumu JavaScript vidē. Šī ietvara elastība ļauj izstrādātājiem brīvi pielāgot funkcijas atbilstoši specifiskām prasībām, taču šī vienkāršība rada problēmas lielākos projektos. Express.js nav noteikta projekta struktūras, kods ir liels, kā arī man nepatīk, ka javascript programmēšanas valodai nav stingri noteikti mainīgo tipi.

Ruby on Rails, balstīts uz "konvencijas pār konfigurāciju" principu, paātrina izstrādes procesu un vienlaikus sniedz spēcīgu kopienas atbalstu, kas ir vērtīgs lielāku projektu ietvaros. Taču šai aizmugursistēmai ir jābūt ļoti labi izstrādātam plānam pirms sistēmas izveides, jo veikt izmaiņas izveidotā arhitektūrā ir ļoti laikietilpīgi.

Ņemot vērā to, ka esmu iepriekš veidojis aizmugursistēmas ar Spring, kā arī esmu izgājis “CodeWithMosh” kursus šim ietvaram un salīdzinot ietvarus esmu secinājis, ka citām aizmugursistēmām, nekas ļoti labāks nav šīs sistēmas integrācijai izvēlēšos “SpringBoot”.

Sistēmas saskarnei ar lietotāju veidosim mājaslapu, un lai to viegli un droši savientu ar izvēlēto aizmugursistēmu izmantošu kādu no simtiem ietvariem, no kuriem izplatītākie ir:

* Angular (Typescript),
* React (Javascript / Typescript),
* Vue.js (Javascript / Typescript),

Angular, izstrādā un uztur Google, tas piedāvā pilnvērtīgu un strukturizētu platformu – izstrādes faili ir sadalīti, un failu nosaukumi standartizēti [3. 5.lpp.]. Angular ir paredzēts lieliem, mērogojamiem projektiem, kur komponenšu atkārtota izmantošana un failu struktūras standartizācija ir būtiska, kā arī angular ir ļoti laba dokumentācija.

React, izstrādā un uztur Facebook, funkcijas ir minimalistiskas, un pakļaujas uz cilvēku izveidotām bibliotekām, lai papildinātu funkcionalitāti, kā piemēram “React Router” vai “react-redux” react ir elastīgs un tam nav strikta failu struktūra projektam , kā arī typescript kods tiek rakstīts kopā ar html kodu vienā failā [3. 5.lpp.]. Atšķirībā no Angular kurš html kodā ievieš typescript, React ievieš typescript kodā html kodu.

Savukārt, Vue.js, izstrādā un uztur Evan You piedāvā vienkāršu mācīšanās līkni un ir līdzīgs Angular taču ar atvieglotu sintaksi. Šajā ietvarā typescript kods nav atdalīts un ir rakstīts “<script>” blokā html stila failā [3. 5.lpp.].

Ņemot vērā ka šo projektu, ja viņu implementēs, uzturēs citi cilvēki, angular ar savu strikto projekta struktūru un Angular izstrādātām bibliotekām ir visspiemērotākais vieglai attīstībai un uzturēšanai.

Tātad esmu nolēmis jaunās sistēmas izveidei izmantot MySQL datubāzi, Java Spring aizmugursistēmai, un lietotāja saskarnei Angular. Ar MySQL un Spring esmu jau strādājis, taču Angular ir kas jauns, tāpēc pirms jaunās sistēmas izveides būs nepieciešams apgūt Angular.

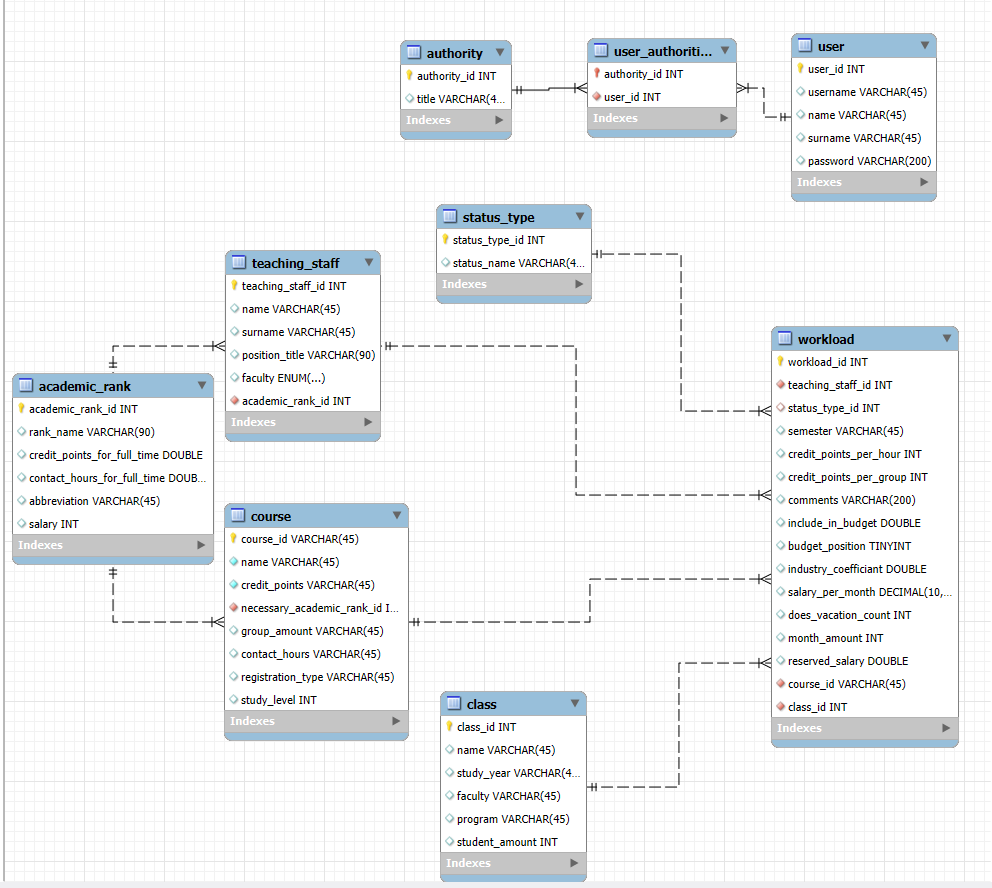
# Jaunās sistēmas izstrāde

Lai izstrādātu jauno sistēmu, vispirms izveidošu datubāzes modeli. Pēc tā izveides datubāzē ievietošu testa datus un izstrādāšu skriptu, kas nolasīs un attēlos šos datus līdzīgi kā esošajā sistēmā. Kad datubāze būs veiksmīgi notestēta, turpināšu ar aizmugursistēmas modeļu izstrādi un vienkāršu kontrolieru izveidi, lai iespējami ātri varētu ieviest pamata drošības risinājumu. Pēc drošības funkcionalitātes ieviešanas savienošu lietotāja saskarni ar aizmugursistēmu, lai varētu pieslēgties sistēmai. Kad lietotāja autentifikācija būs veiksmīgi ieviesta, turpināšu sistēmas attīstību, vienlaikus izstrādājot gan aizmugursistēmas, gan lietotāja saskarnes komponentes. Kā pirmo funkcionalitāti izstrādāšu tabulu, kas atgādina esošās sistēmas risinājumu. Pēc tam pievienošu iespēju izveidot jaunus objektus. Nākamajā solī uzlabosim datu attēlojumu, ieviešot filtrēšanas iespējas, kolonnu paslēpšanu un kārtošanu. Visbeidzot izstrādāšu funkcionalitāti jauna semestra izveidei, kas ļaus ērti pārnest datus no iepriekšējiem semestriem.

## Datu bāzes modeļa izstrāde

Datubāzes pirmais modelis sastāv no 6 tabulām (4.1 att):

* Noslodzes plāns (Workload) – kura sastāv no 16 kolonām un ir domāta, kā galvenā tabula, kura atspoguļu esošās sistēmas izklajlapu, šajā tabulā atrodas atslēgas no studiju priekšmetu tabulas (course), statusa tipu (status\_type) tabulas, docētāju (teaching staff) tabulas un kursa (class) tabula, visas tabulas savienotas ar viens pret daudziem attiecību.
* Studiju priekšmeta (course) tabula sastāv no 8 kolonām, kur aprakstīts studiju priekšmets, un vienu atslēgu uz amatu grupas (academic rank) tabulu, lai norādītu kāda ir nepieciešamā amatu grupa, lai pasniegtu šo studiju priekšmetu, tabulas savienotas ar viens pret daudziem attiecību.
* Docētāju (teaching staff) tabula sastāv no 6 kolonām, kur atrodas informācija par docētāju, kā arī viena atslēga uz amatu grupas tabulu (academic rank), lai norādītu docētāja amatu, tabulas savienotas ar viens pret daudziem attiecību.
* Statusa tipa (status type) tabula, kura sastāv no 2 kolonām un neietver atslēgas no citām tabulām. Šī tabula ir, lai norādītu pasniedzēja statusu, kā, piemēram, ievēlēts, kas nozīmē, ka pasniedzējs nav vieslektors.
* Kursa (class) tabula, kura sastāv no 6 kolonām un neietver atslēgas no citām tabulām. Šī tabula norāda informāciju par kursu – kursa gadu, kursa vārdu, programmu, studentu skaitu
* Amatu grupu (academic rank) tabula sastāv no 6 kolonām, un neietver atslēgas no citām tabulām. Šī tabula norāda informāciju par amatu grupu – amata nosaukumu, kredītpunkti pilnai slodzei, kontaktstundas pilnai slodzei, abreviatūra un alga.



*4.1. att.* Pirmais datu bāzes modelis

Lai nodrošinātu lietotāju autentifikāciju sistēmā izveidoju arī papildus 3 tabulas:

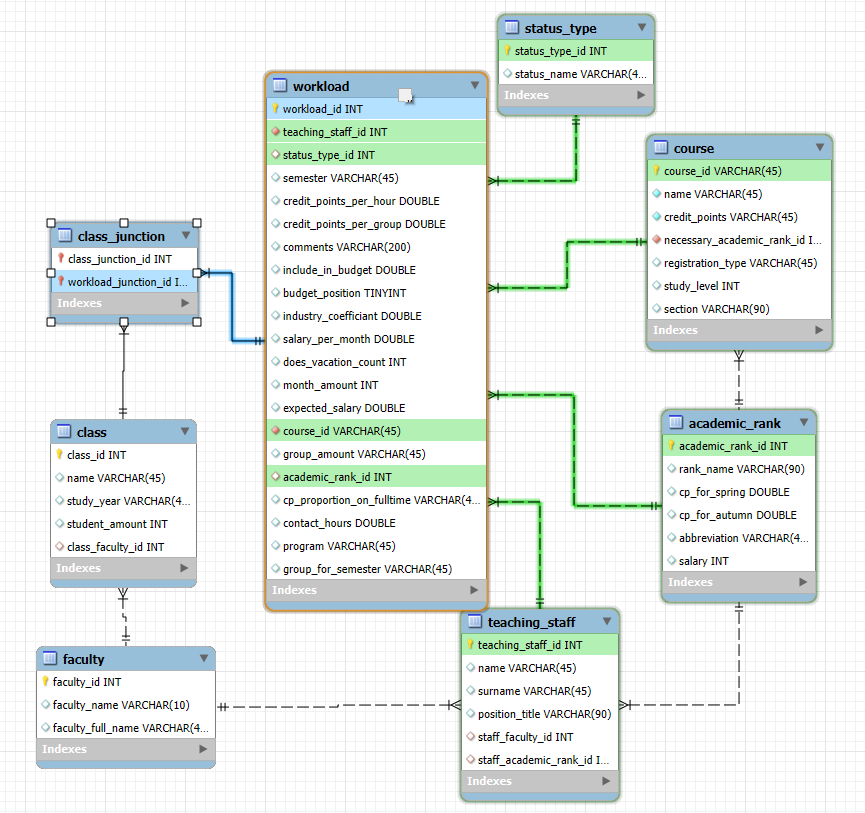
* Lomas (authority)
* Lietotāja (user)
* Un lietotāju lomu (user authorities) tabula lai savienotu Lomas un lietotājus ar many to many attiecību.

Kad modelis izveidots, datubāzei pievienoju datus no esošās sistēmas (skat. pielikumā nr.1) pēc kā jau datus liekot, atklāju sekojošas nepilnības:

* Esošais datu bāzes modelis neatbalsta pievienot noslodzes plānam vairākus kursus.
* Esošajā datu bāzes modelī mainot docētājiem amatu grupu, tā tiktu nomainīta visos ierakstos, taču amata grupa docētājam var mainīties starp mācību gadiem.
* Fakultātes kolona atkārtojas starp tabulām.

Nepilnības salabošanas rezultātā izveidojās 2 papildu tabulas (4.2 att):

* Klases savienošanas (class junction), lai spētu noslodzes plāna ierakstam pievienot vairākas klases
* Fakultāšu (faculty) tabulu, lai tabulās, kur atradās kolonas par fakultāti, tiktu izmantoti dati no šīs tabulas.

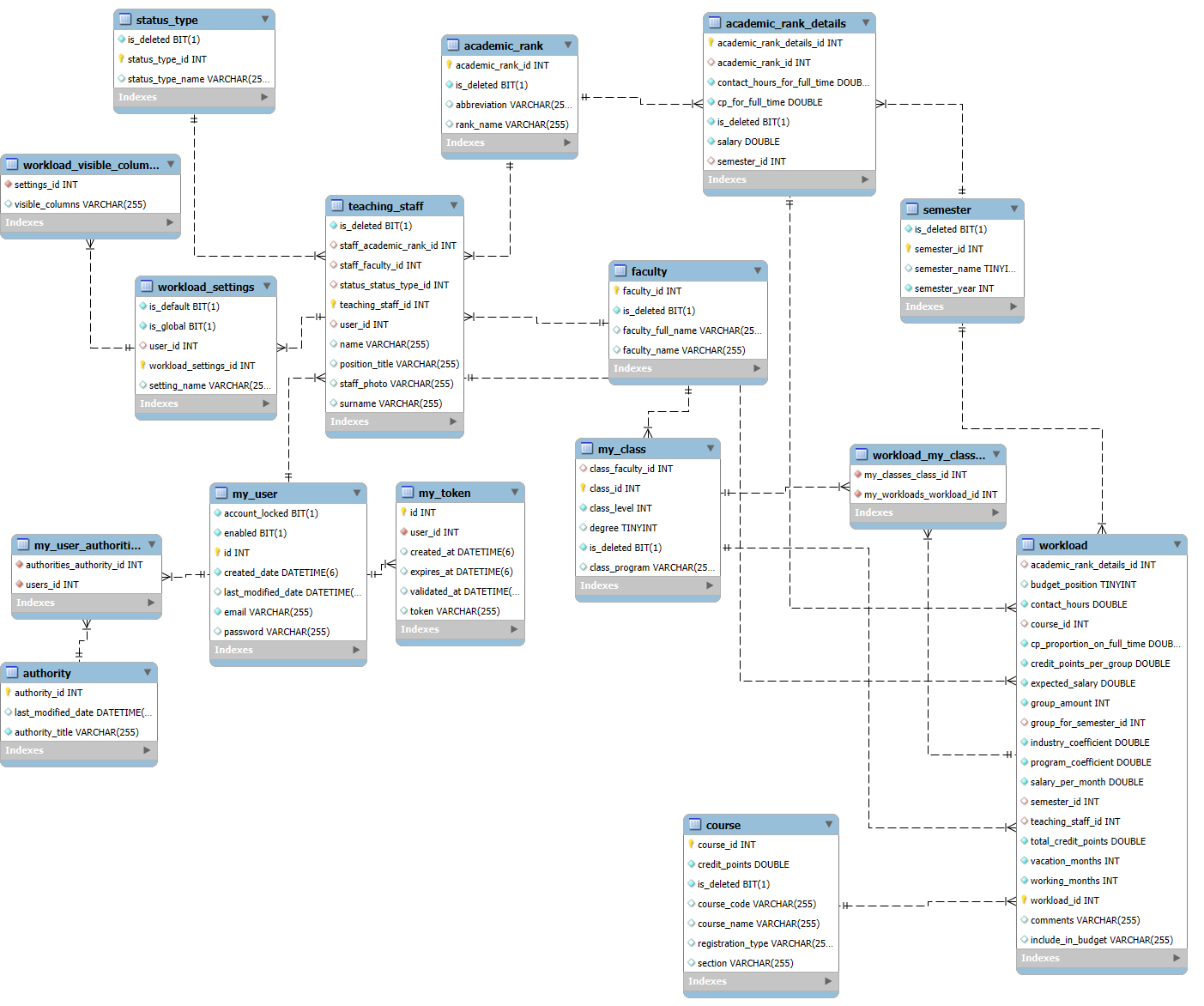
Kā arī pievienojo noslodzes plāna (workload) tabulai kolonu ar atslēgu amatu grupai, tādejādi ļaujot docētājiem mainīt amata grupu, dažādos mācību gados, neietekmējot citu gadu datus.

*4.2. att.* Labotais datu bāzes modelis

Lai viegli salīdzinātu un analizētu jaunās sistēmas datu modeli, izveidoju SQL skriptu, kurš pēc iespējas tuvāk esošajam datu modelim izvada datus. (skatīt pielikumā nr. 2). Laboto modeli piepildīju ar datiem, un viss izskatījās ļoti labi.

Taču aizmugursistēmas izveides laikā datu bāzes modelim tika veikta sekojoši uzlabojumi (4.3 att) kā rezultātā datubāzes modelis sastāv no 16 tabulām:

* Autorizācijas detaļas (my\_user) tika savienots ar docētāju (teaching staff) tabulu, lai docētāji varētu piekļūt sistēmai un redzēt savu noslodzes plānu, attiecīgajā semestrī
* Noslodzes plānam ir nepieciešams sadalījums pa semestriem, tāpēc tika izveidota semestra (semester) tabula, kura ir savienota ar noslodzes plānu.
* Lai semestra sadalījums tiešām būtu loģisks, sadalīju amatu grupas tabulu (academic\_rank) 2 daļās amatu grupu tabulā un amatu grupu detaļu (academic rank details) tabulā, amatu grupu tabulā atstājos informāciju par amata nosaukumu un abreviatūru, taču informāciju par kontaktstundām, algu, kredītpunktiem pilnai slodzei un galvenokārt atslēga uz semestra tabulu, tika ielikta jaunajā amatu grupu detaļu tabulā, tādejādi nodrošinot, ka ja jaunā semestrī ir cita alga, alga iepriekšējiem datiem nemainītos.
* Kā arī tika izveidota pieejas atslēgu tabula, kurā tiek saglabāti profila aktivizēšanas kodi, kuri tiek nosūtīti uz epastu nesen izveidotiem docētājiem.
* Tika izveidotas2 jaunas tabulas (workload\_settings, workload\_visible columns), lai nodrošinātu lietotājiem iespēju saglabāt tabulu iestatījumus savā profilā.
* Visām tabulām kuru dati ir izmantoti citā tabulā ar atslēgas palīdzību ieviesu drošu dzēšanu, jeb papildus kolonu, kura nosaka vai ieraksts ir dzēsts vai nē, tādejādi dati netiek dzēsti, bet gan paslēpti.
* Statusa tips (status type) tika noņemts no noslodzes plāna un savienots tieši ar docētāju (teaching staff)



*4.3. att.* Gala DB modelis.

## 4.2 Aizmugursistēmas izveide

Aizmugursistēma ļaus droši piekļūt datiem no datubāzes caur “REST API” galapunktiem. Lai izveidotu jaunajai sistēmai aizmugursistēmu izmantoju “Spring initializer” [4], kurš ļauj mājaslapā konfigurēt projektu – izvēlēties programmēšanas valodu, izvēlēties versiju, mainīt nosaukumu, pievienot bibliotēkas uc. Šim projektam izmantoju sekojošas bibliotēkas:

* “Spring-boot-starter-data-jpa” – ļauj veidot vai atjaunot savienoto datubāzi, ņemot vērā projekta klases.
* “Spring-boot-starter-mail” – ļauj savienot aizmugursistēmu ar epasta serveri, lai sūtītu epastus.
* “Spring-boot-starter-validation” – atvieglo un samazina kodu datu validācijai.
* “Mysql-connector-j” – ļauj savienot aizmugursistēmu ar datubāzi.
* “Lombok” – aizstāj bieži izmantotus un vieglus koda fragmentus ar anotāciju.
* “spring-boot-starter-security” – ievieš autorizāciju, ierobežo pieeju aizmugursistēmai uz noteiktām ip adresēm, kā arī ierobežo atļauto pieprasijumu veidus.
* “jjwt-api” – ļauj izveidot šifrētu atslēgu, kurā atrodas informācija par to cik ilgi viņa derīga, kam viņa pieder uc. Šo atslēgu pēctam var aizsūtīt lietotāja saskarnei, lai tā varētu autentificēt sevi aizmugursistēmai.
* “springdoc-openapi-starter-webmvc-ui” – grafiski attēlo aizmugursistēmas galapunktus, kā arī ļauj pārbāudīt, vai tie strādā.
* “opencsv” – bibliotēka, kas palīdz veidot, rediģēt, lasīt csv failus.
* “spring-boot-starter-thymeleaf” – strādā kopā ar aizmugursistēmu lai papildinātu html funkcionalitāti, piemēram, dot iespēju html kodā iespraust datus no aizmugursistēmas, šo izmantoju, lai veidotu paraugu epasta vēstulei.

Pēc projekta izveides konfigurēju aplikācijas iestatījumus, lai katru reizi palaižot aplikāciju, tā nemainītu datubāzes struktūru un tikai pārbaudītu esošo, tādejādi pasargājot datubāzes modeli no nejaušām izmaiņām izstrādes laikā, kā arī pievienoju adresi, caur kuru var piekļūt datubāzei, un iedevu autorizācijas informāciju.

|  |
| --- |
| spring:  datasource:  url: jdbc:mysql://localhost:3306/mydatabase  username: ${DB\_NAME}  password: ${DB\_PASSWORD}  driver-class-name: com.mysql.cj.jdbc.Driver  jpa:  hibernate:  ddl-auto: validate  show-sql: true |

*4.1 koda fragments.* ***Spring* satvara konfigurācija**

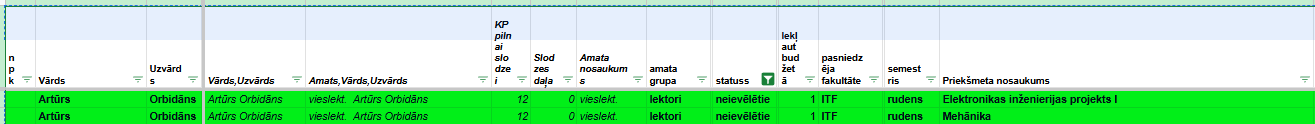
sāku veidot un konfigurēt “spring security” bibliotēku Java modeļus, ņemot izveidoto datubāzes modeli, kā piemēru.

# IZMANTOTĀS LITERATŪRAS UN AVOTU SARAKSTS

1. DB-Engines Ranking [Tiešsaistē]. Pieejams: <https://db-engines.com>
2. PAWEŁ DYMORA 1 , MIROSŁAW MAZUREK 2 , MARIUSZ NYCZ , Comparison of Angular, React, and Vue Technologies in the Process of Creating Web Applications on the User Interface Side (2023) [Tiešsaistē]. Pieejams: [10.15584/jetacomps.2023.4.21](https://dx.doi.org/10.15584/jetacomps.2023.4.21) [Skatīts Apr. 10, 2025]
3. Stack overflow developer survey [Tiešsaistē]. Pieejams: <https://survey.stackoverflow.co/2024/technology/>
4. Spring initializer [Tiešsaistē]. Pieejams: <https://start.spring.io/>

# Pielikumi

#### Pielikums Nr. 1





A screenshot of a computer

AI-generated content may be incorrect.

#### Pielikums Nr. 2

SELECT t.name as 'Vārds',

t.surname as 'Uzvārds',

CONCAT(t.name, ' ', t.surname) AS 'Vārds, Uzvārds',

CONCAT(t.position\_title, ' ', t.name, ' ', t.surname) AS 'Amats,Vārds, Uzvārds',

aStaff.cp\_for\_autumn AS 'KP pilnai slodzei',

w.cp\_proportion\_on\_fulltime AS 'Slodzes daļa',

t.position\_title AS 'Amata nosaukums',

aStaff.rank\_name AS 'amata grupa',

s.status\_type\_name AS 'statuss',

w.include\_in\_budget AS 'Iekļaut budžetā',

fstaff.faculty\_name AS 'pasniedzēja fakultāte',

w.semester AS 'semestris',

co.name AS 'Priekšmeta nosaukums',

w.credit\_points\_per\_hour AS 'Progr. koef. KP/stundas',

co.credit\_points AS 'Priekšmeta kp',

w.credit\_points\_per\_group AS 'KP skaits grupai',

w.group\_amount AS 'grupu skaits',

w.contact\_hours AS 'Kontaktstundas',

co.course\_id AS 'LAIS kods',

co.section AS 'Programmas daļa',

co.registration\_type AS 'Reģistrācija',

fCourse.faculty\_name AS 'Priekšmeta fakultāte',

fClass.faculty\_name AS 'kursa fakultāte',

program AS 'Programma',

co.study\_level AS 'Studiju līmenis',

w.group\_for\_semester AS 'Grupa semestra grafikam',

-- grupa MANY TO MANY????

GROUP\_CONCAT(class.name SEPARATOR ', ') AS classes,

w.comments AS 'Komentāri',

w.budget\_position AS 'budžeta pozīcija',

GROUP\_CONCAT(class.student\_amount SEPARATOR ', ') AS 'Studentu skaits',

aStaff.salary AS 'Alga',

w.industry\_coefficiant AS 'Nozares koef.',

w.salary\_per\_month AS 'Alga mēnesī',

w.include\_in\_budget AS 'Vai atvaļinājums ieskaitās',

w.month\_amount AS 'Mēnešu skaits',

w.expected\_salary AS 'Algai paredzētais'

FROM workload AS w

JOIN teaching\_staff AS t ON w.teaching\_staff\_id = t.teaching\_staff\_id

JOIN status\_type AS s ON s.status\_type\_id = w.status\_type\_id

JOIN course AS co ON co.course\_id = w.course\_id

JOIN academic\_rank AS aCourse ON aCourse.academic\_rank\_id = co.necessary\_academic\_rank\_id

JOIN academic\_rank AS aStaff ON aStaff.academic\_rank\_id = w.academic\_rank\_id

JOIN faculty AS fStaff ON fstaff.faculty\_id = t.staff\_faculty\_id

JOIN faculty AS fCourse ON fCourse.faculty\_id = co.necessary\_academic\_rank\_id

JOIN class\_junction AS classId ON classId.workload\_junction\_id = w.workload\_id

JOIN class AS class ON class.class\_id = classId.class\_junction\_id

JOIN faculty AS fClass ON fClass.faculty\_id = class.class\_faculty\_id

GROUP BY

w.workload\_id

# GALVOJUMS

Ar šo es, Endijs Bertāns*,* galvoju, ka šis bakalaura darbs ir manis paša patstāvīgi izpildīts oriģināls darbs. Visi informācijas avoti, kā arī no tiem ņemtie dati un definējumi ir norādīti darbā. Šis darbs tādā vai citādā veidā nav iesniegts nevienai citai pārbaudījumu komisijai un nav nekur publicēts.

Esmu informēts (-a), ka mans bakalaura darbs tiks ievietots un apstrādāts Vienotajā datorizētajā plaģiāta kontroles sistēmā plaģiāta kontroles nolūkos.

202\_\_. gada \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(paraksts)

Es, Endijs Bertāns, atļauju Ventspils Augstskolai savu bakalaura darbu bez atlīdzības ievietot un uzglabāt Latvijas Nacionālās bibliotēkas pārvaldītā datortīklā Academia (www.academia.lndb.lv), kurā tie ir pieejami gan bibliotēkas lietotājiem, gan globālajā tīmeklī tādā veidā, ka ikviens tiem var piekļūt individuāli izraudzītā laikā, individuāli izraudzītā vietā.

Piekrītu \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Nepiekrītu \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

202\_\_. gada \_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_