LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

SCORO CRM SISTĒMAS DATU SINHRONIZĀCIJA IZMANTOJOT REST API

KVALIFIKĀCIJAS DARBS

Autore: **Anna Gronska**
Studenta apliecības Nr.: ag14052
Darba vadītājs: M. dat. Matīss Rikters

RĪGA 2016

ANOTĀCIJA

Autores uzdevums šajā kvalifikācijas darbā bija sinhronizēt esilideris.lv, bvk.lv un SIA ,,BCLS" sistēmu datus ar Scoro CRM sistēmu. Datu sinhronizācijas mērķis bija iekšējā darba optimizācija esilideris.lv, bvk.lv un SIA ,,BCLS" sistēmās, lai turpmāk sistēmas savā darbā var izmantot šo cilvēkresursu pārvaldības sistēmu izmantojot sinhronizētos datus.

Projekts veidots pēc spējā programmatūras izstrādes modeļa. Sistēmas datu sinhronizācijas risinājumi izstrādāti galvenokārt izmantojot API savienojumus un PHP programmēšanas valodu.

Atslēgas vārdi: CRM, PHP, SQL, datu sinhronizācija, API.

ABSTRACT

The author's task in “Scoro CRM system data synchronization using REST API” paper was to synchronize esilideris.lv, bvk.lv and SIA “BCLS” systems data with Scoro system. Aim of data synchronization was to improve internal work processes in esilideris.lv, bvk.lv and SIA “BCLS” systems so that these systems could continue their work using Scoro human relationship management system using synchronized data.

The project was created using the agile software development model. System data synchronization solutions were developed primarily using API connections and PHP programming language.

Keywords: CRM, PHP, SQL, data synchronization, API.

SATURS

1. APZĪMĒJUMU SARAKSTS

Lietotie saīsinājumi un apzīmējumi	Skaidrojums
SQL	Structured Query Language. Datubāzu vaicājumu valoda.
CRM	Customer Relationship Management. Veids kā pārvaldīt uzņēmuma savstarpējo mijiedarbību ar darbiniekiem un klientiem.
PHP	Hypertext Preprocessor. Servera puses programmēšanas valoda.
XML	Extensible Markup Language. Paplašināmā iezīmēšanas valoda, kas strukturē tekstu informācijas koplietošanai internetā.
REST	Representational State Transfer. Interneta programmatūras arhitektūras veids.
mdb	Faila formāts, kas tiek izmantots Microsoft Access programmatūrā, lai glabātu datubāzes.
API	Application Programming Interface. Sekvenču un protokolu rīks, kas paredzēts programmatūru izstrādei.
PDO	PHP Data Objects. PHP programmēšanas metode, kas nodrošina drošu datu plūsmu.
SSL	Secure Sockets Layer. Drošības protokols, kas veic datu šifrēšanu.
HTTPS	Secure Hyper Text Transfer Protocol. Tīmekļa drošības protokols, kas nodrošina datu plūsmu tīmeklī šifrējot pārsūtāmos datus.
JSON	JavaScript Object Notation. Datu apmaiņai paredzēts uzglabāšanas formāts.
HTML	Hypterext Markup Language. Standartizēta teksta failu iezīmēšanas valoda, kas paredzēta tīmekļa vietņu veidošanai.
CSS	Cascading Style Sheets. Īpaša stila valoda, kuru izmanto, lai aprakstītu iezīmēšanas valodā rakstītu dokumentu izskatu un formātu.
JavaScript	Skriptu valoda, kas tiek plaši izmantota tīmekļa lapu veidošanā, lai izstrādātu interaktīvus efektus.

Bootstrap	Atvērtā pirmkoda rīku kopums tīmekļa lietotņu izstrādei, kas sevī ietver dažādus dizaina šablonus, formas, pogas un citus saskarnes elementus.
Scrum	Spējās izstrādes modeļa metodoloģija, kas izmantota izstrādājot šo kvalifikācijas darbu.
Scheduled task	Ieplānots uzdevums. Izveidots automātisks datu atjauninātājs.

IEVADS

Scoro CRM ir inovatīva cilvēkresursu pārvaldības sistēma, kas tiek integrēta vairāku iestāžu darbā. Sistēmas ieviešanai iestādēs bieži vien nepieciešama datu sinhronizācija, lai būtu iespējams pilnvērtīgi izmantot visas Scoro sniegtās iespējas.

Kvalifikācijas darbā aprakstīti Scoro CRM sistēmas datu sinhronizācijas procesi ar bvk.lv, esilideris.lv un SIA ,,BCLS" sistēmu rīcībā esošajiem datiem.

CRM ir sistēmas veids, kas fokusējas uz klientu attiecību pārvaldību. Šīs sistēmas mērķis ir piedāvāt ērtāku veidu, kā organizēt darbu un pārvaldīt cilvēkresursus gan mazās, gan lielākās iestādēs.

Autore kvalifikācijas darbu veica prakses ietvaros, kur darbā tika izmantota Scoro sistēma, lai pārvaldītu uzņēmuma darba procesus. Prakses uzņēmums sniedz datu sinhronizācijas pakalpojumus citām elektroniskām sistēmām, un prakses ietvaros attiecīgie datu sinhronizācijas projekti tika veikti. Izstrādāto datu sinhronizāciju mērķis ir Scoro CRM sistēmas implementēšana bvk.lv, esilideris.lv un SIA ,,BCLS" sistēmu turpmākajā darbā, kā arī iepriekšējo datu pārnese uz Scoro, lai šis process nebūtu jādara manuāli.

Datu sinhronizāciju izstrādes procesā tika izmantota Scrum plānošanas metodoloģija, lai optimāli organizētu laiku un projektu izstrādes tempu. Izstrādājot šāda tipa datu sinhronizācijas, kur procesā ir vairāki algoritma izstrādes soļi, Scrum izstrādes metodoloģija ir ļoti atbilstoša. Scrum metodoloģija iedalās attiecīgos sprintos un lietotāju stāstos, ko var attiecināt uz izstrādes procesa algoritma soļiem. Strādājot pēc Scrum metodoloģijas, izstrādes procesā radušos traucēkļus ir vieglāk analizēt un risināt, jo projekta posmi ir optimāli sadalīti.

Veidojot datu sinhronizācijas, īpaša uzmanība un vērība tika pievērsta, lai visi esošie sistēmas dati tiktu pārnesti uz Scoro CRM, jo darbs ir ar lielām un būtiskām sistēmām, kuras nevar atļauties nekādu datu zudumu.

Darbs ir izstrādāts patstāvīgi un neatkarīgi no citiem līdzīgiem projektiem.

2. PROGRAMMAS PRASĪBU SPECIFIKĀCIJA

2.1. Ievads

2.1.1. Nolūks

Programmatūras prasību specifikācija ir paredzēta izstrādājamās Scoro CRM sistēmas datu sinhronizācijas prasību aprakstīšanai. Atbilstoši programmas prasību specifikācijai tiks izstrādāts arī programmas projektējuma apraksts un funkcionējošs programmkods. Šis dokuments ir paredzēts datu sinhronizācijas projektētājam un pasūtītājam, lai saskaņotu prasības pret veicamo datu sinhronizācijas rezultātu un turpmāko funkcionalitāti.

2.1.2. Darbības sfēra

Programmas produkta mērķis ir nodrošināt veiksmīgu datu sinhronizāciju starp Scoro CRM sistēmu un pasūtītāja sistēmu, kas spētu sasaistīt visus sistēmas datus ar Scoro CRM sistēmu, lai pasūtītājs var pilnvērtīgi izmantot Scoro savā darba organizācijā. Produkta paredzētais pielietojums ir turpmākais darbs ar Scoro CRM sistēmu un visiem sasinhronizētajiem datiem.

Sinhronizējot Scoro CRM sistēmu ar pasūtītāja sistēmas datiem, iespējams optimizēt un uzlabot līdzšinējo sistēmas darba kultūru, kā arī atvieglot darbu pasūtītāja sistēmas lietotājiem.

2.1.3. Definīcijas un saīsinājumi

Šajā dokumentā lietotie projektam specifiskie jēdzieni, saīsinājumi un apzīmējumi ir definēti darba 1. nodaļā - „Apzīmējumu saraksts”.

2.1.4. Saistība ar citiem dokumentiem

Programmatūras prasību specifikācija ir kvalifikācijas darba „Scoro CRM sistēmas datu sinhronizācija izmantojot REST API” sastāvdaļa. Šis dokuments ir izstrādāts saskaņā ar standartu LVS 68:1996 “Programmatūras prasību specifikācijas ceļvedis” [5].

2.1.5. Dokumenta pārskats

Dokumentā ir iekļautas šādas nodaļas:

- Ievads

- Vispārējs apraksts
- Funkcionālās prasības
- Nefunkcionālās prasības

2.2. Vispārējs apraksts

2.2.1. Produkta perspektīva

Izstrādātais programmkods veic sinhronizāciju ar pasūtītāja sistēmu, un ir neatkarīga pirmkoda bibliotēka, kas nodrošinās savstarpējo funkcionālo datu apmaiņu un sinhronizāciju. Katrā no sistēmām, ar kurām tiks veikta datu sinhronizācija, ir nepieciešams savs risinājums un pirmkods, lai tiktu veikta sinhronizācija ar Scoro CRM sistēmu.

2.2.2. Produkta funkcijas

Programmatūras kodam ir jāspēj sinhronizēt Scoro CRM sistēmu ar pasūtītāja sistēmas datiem. Programmkodam ir jāimplementē visi pasūtītāja sistēmas dati bez jebkādu datu zuduma. Pēc veiksmīgas datu sinhronizācijas, pasūtītāja darbinieki un sistēmas lietotāji veiksmīgi strādā izmantojot sinhronizētos datus.

2.2.3. Lietotāja raksturiezīmes

Lietotāji, kas izmantos veiktās datu sinhronizācijas, būs Scoro sistēmā reģistrētie lietotāji. Lietotājiem nepieciešamas pamata iemaņas datora un interneta pārlūkprogrammas lietošanā, kā arī pamatzināšanas Scoro CRM sistēmas lietošanā. Darbs ar veiktajām datu sinhronizācijām ir paredzēts konkrētā projekta pasūtītāja sistēmas administratoriem, kuri pārvaldīs sistēmā sniegtos datus, kā arī pašiem sistēmas lietotājiem, kuri izmantos Scoro CRM sistēmas sniegtās iespējas.

2.2.4. Pieņēmumi un atkarības

1. Lai funkcionāli piekļūtu veikto datu sinhronizāciju datiem un pašai Scoro sistēmai, nepieciešama interneta pārlūkprogramma un nodrošināts interneta pieslēgums.

2. Lai pilnvērtīgi lietotājs darbā izmantotu sinhronizētos datus, nepieciešamas pieejas Scoro sistēmai, kā arī zināšanas kā strādāt ar sistēmu.

2.3. Funkcionālās prasības

Scoro CRM datu veiksmīgai sinhronizēšanai ir vairāki algoritma soļi, kuriem jāizpildās pareizi, lai katrs nākamais solis tiktu uzsākts un lai gala rezultātā ir notikusi pareiza datu sinhronizācija. Katrā no sistēmām, ar kurām autore veica datu sinhronizāciju, ir atšķirīgi algoritma soļi un atbilstoši atšķirīgas funkcionālās prasības.

2.3.1. Esilideris.lv sistēmas datu sinhronizācijas funkcionālās prasības

Funkcionējošai Scoro sistēmas datu sinhronizācijai ar esilideris.lv sistēmu tiek definēti 5 galvenie funkcionālie soļi:

1. Datu konvertēšana uz SQL formātu.
2. Datu bāzes tabulu un lauku izveide.
3. SQL skripta pārveidošana.
4. SQL skripta augšupielāde.
5. REST API savienojuma izveidošana un datu sinhronizācija.

2.3.1.1. Datu konvertēšana uz SQL formātu

Sākumā esošajiem datiem, kas sniegti .mdb formātā un glabājas Microsoft Access programmatūrā, ir nepieciešama konvertēšana uz SQL formātu. Datu konvertēšanai tiks izmantota atvērtā pirmkoda programmatūra BullZip[1] . Rezultātā ar BullZip programmatūras palīdzību tiks ģenerēts SQL skripts ar nepieciešamajiem datiem un struktūru, kas ir šī algoritma soļa funkcionālā prasība.

2.3.1.2. Datu bāzes tabulu un lauku izveide

Lai ģenerētais SQL skripts tiktu veiksmīgi augšupielādēts datubāzē, nepieciešama attiecīga datu bāzes tabulu un lauku izveide phpMyAdmin rīkā. Funkcionālā prasība izveidotajām tabulām ir, lai tabulas un lauki būtu izveidoti precīzi un saskaņoti ar SQL ģenerēto skriptu, citādi datu augšupielāde nav iespējama.

[1] Bullzip http://www.bullzip.com/products/a2m/info.php

2.3.1.3. SQL skripta pārveidošana

BullZip programmatūras veidotajam SQL skriptam ir nepieciešami pārveidojumi, lai būtu iespējams skriptu ielādēt datubāzē. Lai funkcionālā prasība tiktu izpildīta, pārveidošanas procesos ietilpst noteiktu koda rindiņu pievienošana un dzēšana, lai phpMyAdmin rīks akceptētu skripta augšupielādi datubāzē.

2.3.1.4. SQL skripta augšupielāde

Ģenerētais un pārveidotais SQL skripts tiek augšupielādēs phpMyAdmin rīkā izveidotajās datubāzu tabulās. Funkcionālā prasība šim algoritma solim ir SQL skripta atbilstība datubāzu tabulām, lai dati veiksmīgi tiktu augšupielādēti phpMyAdmin rīkā un būtu pieejami turpmākam datu sinhronizācijas procesam no izveidotajām datu bāzu tabulām.

2.3.1.5. REST API savienojuma izveidošana un datu sinhronizācija

Izveidoto datubāzes failu veiksmīgai sinhronizācijai tiks izveidots API savienojums ar Scoro sistēmu. Savienojuma izveide un sinhronizācija noris ar PHP koda palīdzību, un soli var uzskatīt par izpildītu, ja Scoro sistēmā tiek izvadīti vēlamie sinhronizācijas dati.

2.3.2. Bvk.lv sistēmas datu sinhronizācijas funkcionālās prasības

Funkcionējošai Scoro sistēmas datu sinhronizācijai ar bvk.lv sistēmu tiek definēti 3 galvenie funkcionālie soļi:

1. Datu ieguve no ,,Valis" sistēmas.
2. Iegūto XML failu nolasīšana, pārveidošana un sinhronizēšana ar Scoro.
3. Atbilstoša ieplānotā uzdevuma izveidošana.

2.3.2.1. Datu ieguve no „Valis'' sistēmas

No skolu pārvaldības sistēmas ,,Valis" tiks iegūti XML faili ar datu sinhronizācijai nepieciešamajiem skolnieku personīgajiem datiem. Soli var uzskatīt par funkcionāli izpildītu, ja dati tiek veiksmīgi iegūti un uzglabāti izmantotajos izstrādes konfigurācijas rīkos.

2.3.2.2. Iegūto XML failu nolasīšana, pārveidošana un sinhronizēšana ar Scoro

Izmantojot Scoro izveidoto „Modify" PHP funkciju, izstrādātais programmkods nolasa noglabātos XML failus no „Valis" sistēmas. Ar PHP koda palīdzību, sinhronizācijas datu struktūra tiks izmainīta, uzglabājot datus masīvos, lai tie tālāk tiktu sinhronizēti ar Scoro. Funkcionālā prasība šim solim ir veiksmīga failu nolasīšana, failu pārveidošana Scoro sistēmai sinhronizējamā veidā, lai tie tiktu izvadīti Scoro CRM sistēmā.

2.3.2.3. Atbilstoša ieplānotā uzdevuma izveidošana

Tiek izveidots ieplānotais uzdevums atbilstoši pasūtītāja datu sinhronizācijas funkcionālajām prasībām, un iestrādātajam ieplānotajam uzdevumam ir jāsinhronizē jauniegūtie dati katru diennakti pulksten 1:00, kad sistēmai ir mazs noslogojuma procents.

2.3.3. SIA „BCLS" sistēmas datu sinhronizācijas funkcionālās prasības

Funkcionējošai Scoro sistēmas datu sinhronizācijai ar SIA „BCLS" sistēmu tiek definēti 5 galvenie funkcionālie soļi:

1. Datu ielasīšana ārējā saskarnē no Scoro sistēmas.
2. Atbilstoša ieplānotā uzdevuma izveidošana.
3. Iegūto datu nolasīšana, pārveidošana un sinhronizēšana ar Scoro.
4. Manuāli pievienoto datu funkcionāla sinhronizācija.

2.3.3.1. Datu ielasīšana ārējā saskarnē no Scoro sistēmas

Lai dati tiktu ielasīti, ir jāizveido veiksmīgs API savienojums starp Scoro sistēmu un autores izveidoto lietotāja ārējo saskarni. Funkcionāli, turpmāk strādājot ar saskarni, tajā tiek izvadīti dati, kuri tika iepriekš ievadīti Scoro no SIA „BCLS" puses.

2.3.3.2. Atbilstoša ieplānotā uzdevuma izveidošana

Tiek izveidots ieplānotais uzdevums atbilstoši pasūtītāja datu sinhronizācijas funkcionālajām prasībām, un šim iestrādātajam ieplānotajam uzdevumam ir jāsinhronizē

jauniegūtos datus no Scoro ar izveidoto ārējo saskarni. Ieplānotais uzdevums tiks darbināts reizi diennaktī, pulksten 00:00, lai sinhronizētu darba dienā savāktos datus.

2.3.3.3. Iegūto datu nolasīšana, pārveidošana un sinhronizēšana ar Scoro

Izmantojot Scoro izveidoto ,,Request" PHP funkciju, izstrādātais programmkods nolasa noglabātos datus. Funkcionālā prasība šim solim ir veiksmīgi nolasīti dati, kas tālāk tiek izvadīti izstrādātajā sistēmas ārējā saskarnē izmantojot izveidoto ieplānoto uzdevumu.

2.3.3.4. Manuāli pievienoto datu funkcionāla sinhronizācija

Ārējā saskarnē izveidotajai funkcionālajai pogai, kas uz klikšķa sūta lietotāja ievadītos datus uz Scoro, nepieciešama funkcionāla ievadīto datu saņemšana, datu plūsmu veikšana un korekta datu ielasīšana Scoro projekta pasūtītāja prasītajās sadaļās.

2.4. Nefunkcionālās prasības

2.4.1. Drošības prasības

Drošai datu apmaiņai jābūt aizsargātai un izstrādātai, izmantojot drošas programmēšanas metodes. Ir jābūt nodrošinājumam pret SQL injekcijām datubāzēs, kā arī ir jānodrošina datu šifrēšana. Nepieciešams izstrādāt drošas datu plūsmas tīmeklī, kas novērstu datu zagšanas iespējamību.

Scoro CRM sistēmai jānodrošina datu sinhronizācijas drošību un pārsūtāmo datu šifrēšanu tīmeklī un pašā Scoro sistēmā.

Veicot sistēmās personas datu sinhronizāciju, kurā ir tādi sensitīvi dati kā dzīvesvietas adrese, personas kods, finanšu dati un cita būtiska informācija, ir jānodrošina, lai sinhronizējamie dati nekādā veidā nenokļūst trešās personas rīcībā.

2.4.2. Uzturamība

Izstrādātajai datu sinhronizācijai un programmkodam ir jābūt funkcionējošam arī pēc projekta nodošanas, jo bvk.lv un SIA ,,BCLS" sistēmās datu sinhronizācija nav vienreizējs process un datu sinhronizācijas norise ir paredzēta katru dienu atbilstoši izveidotajiem ieplānotajiem uzdevumiem. Programmkodam ir jāpilda savas funkcijas, lai turpmāk projektu pasūtītājs var izmantot darbā pastāvīgu jauno datu sinhronizāciju.

Programmatūras kodam un funkcijām ir jābūt pēc iespējas saprotamākām un papildināmām, jo datu sinhronizācija ir pastāvīgs process, un var radīties nepieciešamība mainīt datu sinhronizācijas struktūru.

2.4.3. Lietotāja saskarne

Veicamajos datu sinhronizācijas projektos, ir nepieciešams izstrādāt ārējo saskarni tikai SIA ,,BCLS" projektam, kurā saskarnei ir jābūt viegli lietojamai un pārredzamai, kurā ir tikai datu ievades lauki un būtiskākā informācija.

Esilideris.lv un bvk.lv sistēmu datu sinhronizācijā ar Scoro, var uzskatīt, ka ārējā saskarne ir pati Scoro CRM sistēma, kas ir ērti un funkcionāla pielāgota lietotājam, lai lietotu sinhronizētos datus.

3. PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS

3.1. Ievads

3.1.1. Dokumenta nolūks

Programmatūras projektējuma nolūks ir aprakstīt kā Scoro CRM sistēmas datu sinhronizācijas programmatūras prasību specifikācijā minētās prasības tiks realizētas projektā.

Dokuments ir izstrādāts, lai pārveidotu prasības izstrādei parocīgākā formā un lai atvieglotu plānošanu un projekta implementēšanu, kas nodrošinātu datu sinhronizācijas pirmkoda pārskatāmību un lasāmību, paplašināmību, kā arī testējamību. Mērķauditorija ir izstrādātāji, uzturētāji un testētāji.

3.1.2. Darbības sfēra

Izstrādātā projekta pirmkods ir neatkarīgs un individuāls kods katrai pasūtītāja sistēmai, kas nodrošina veiksmīgu datu sinhronizāciju. Izstrādātās datu sinhronizācijas galvenais mērķis ir veiksmīga sinhronizācija ar Scoro CRM sistēmu, lai pasūtītājs varētu strādāt ar datiem Scoro sistēmā.

3.1.3. Definīcijas un saīsinājumi

Šajā dokumentā lietotie projektam specifiskie jēdzieni, saīsinājumi un apzīmējumi ir definēti darba 1. nodaļā - „Apzīmējumu saraksts”.

3.1.4. Saistība ar citiem dokumentiem

Programmatūras projektējuma apraksts ir kvalifikācijas darba „Scoro CRM sistēmas datu sinhronizācija izmantojot REST API " sastāvdaļa.

Programmatūras projektējuma apraksts ir izstrādāts balstoties uz “LVS 72:1996 Ieteicamā prakse programmatūras projektējuma aprakstīšanai” [6] vadlīnijām.

3.1.5. Dokumenta pārskats

Dokumentā ir iekļautas šādas nodaļas:

- Ievads

- Detalizēts projektējuma apraksts
- Nefunkcionālo prasību realizācija

3.2. Detalizēts projektējuma apraksts

3.2.1. Datubāzes tabulas

Zemāk aprakstītas nepieciešamās datu bāzu tabulas, lai sinhronizētu datus starp Scoro CRM un bvk.lv, esilideris.lv un SIA „BCLS" sistēmām.

3.2.1.1. Datubāzes tabulas esilideris.lv sistēmas datu sinhronizācijai

Tabulas „Ligumi", „Skolnieki", „Skolnieki_beidza'' un „Visi_skolnieki'' glabā lielu skaitu lauku (15-25 lauki), tādēļ tiks attēloti būtiskākie 10 lauki no katras tabulas.

3.2.1.1.1. Datu bāzes tabula „Atzimes''

Tabula paredzēta personu atzīmju uzglabāšanai.

3.1.tabula

Kolonna	Datu tips	Cita informācija	Apraksts
SkolnieksID	int(11)	PRIMARY KEY, NOT NULL	Unikāls personas identifikators.
Pr_kods	varchar(4)	NOT NULL	Mācību priekšmeta identificējošs kods.
Atzime	varchar(20)	NOT NULL	Iegūtā atzīme mācību priekšmetā.
Datums_atz	datetime	NOT NULL	Datums, kad iegūta atzīme.

3.2.1.1.2. Datu bāzes tabula „Ligumi''

Tabula paredzēta personu mācību līgumu informācijas uzglabāšanai.

3.2. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
ID	int(11)	PRIMARY KEY, NOT NULL, AUTO_INCREMENT	Unikāls personas identifikators.

SkolnieksID	int(11)	NULL	Skolnieka identificējošs kods.
Ligumsnr	int(11)	NULL	Mācību līguma numurs.
Ligreg	timestamp	NOT NULL, CURRENT_TIMESTAMP	Datums, kad parakstīts līgums.
Periods	int(11)	NULL	Mācību periods gadu izteiksmē.
Sakums	datetime	NULL	Mācību sākuma datums.
Beigas	datetime	NULL	Mācību beigu datums.
Summa	float	NULL	Mācību maksa.
Aplnr	varchar(20)	NULL	Studenta apliecības numurs.
Scoro_id	int(11)	NOT NULL	Studenta ID Scoro sistēmā.

3.2.1.1.3. Datu bāzes tabula „Samaksa''

Tabula paredzēta, lai uzglabātu informāciju par katru maksājumu, kas paredzēts, lai segtu mācību maksu.

3.3. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
SkolnieksID	int(11)	PRIMARY KEY, NOT NULL	Unikāls personas identifikators.
Datums	datetime	NULL	Datums līdz kuram jāveic apmaksa par mācībām.
Summa	float	NULL	Summa, kas jāmaksā.
Izsutits	tinyint(1)	NULL	Identifikators, vai mācību rēķins ir ticis izsūtīts.
Konts	varchar(4)	NULL	Studenta bankas konts.
Ieskaitits	tinyint(1)	NULL	Identifikators, vai mācību maksa ir veikta.

3.2.1.1.4. Datu bāzes tabula „Samaksa_beidza''

Tabula paredzēta, lai uzglabātu vispārīgu informāciju par mācību apmaksu pēc mācību perioda.

3.4. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
SkolnieksID	int(11)	PRIMARY KEY, NOT NULL	Unikāls personas identifikators.
Datums	datetime	NULL	Datums, kurā veikta pilna apmaksa par mācībām.
Summa	float	NULL	Summa, kas jāmaksā.

3.2.1.1.5. Datu bāzes tabula „Skolas''

Tabula paredzēta, lai uzglabātu informāciju par personu iepriekšējo izglītību.

3.5. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
SkolasID	int(11)	PRIMARY KEY, NOT NULL	Skolas identifikators.
Nosaukums	varchar(100)	NOT NULL	Skolas nosaukums
Adrese1	varchar(100)	NULL	Ielas nosaukums, kur skola atrodas.
Adrese2	varchar(100)	NULL	Pagasta un/vai pilsētas nosaukums, kur skolas atrodas.
Indekss	varchar(7)	NULL	Pasta indekss attiecīgi skolas atrašanās vietai.
Atlase	tinyint(1)	NULL	Identifikators, kas norāda, vai skolā ir notikusi atlase.
Komentars	varchar(100)	NULL	Komentāri, kas saistīti ar skolu.
Grupa_sk	varchar(1)	NULL	Skolas grupas identifikators.
scoro_id	int(11)	NOT NULL	Skolas identifikators Scoro sistēmā.

3.2.1.1.6. Datu bāzes tabula „Atzimes_beidza''

Tabula paredzēta atzīmju uzglabāšanai ar kādām tika pabeigta iepriekšējā izglītība.

3.6. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
SkolnieksID	int(11)	PRIMARY KEY, NOT NULL	Unikāls personas identifikators.
Pr_kods	varchar(4)	NOT NULL	Mācību priekšmeta identificējošs kods.
Atzime	varchar(20)	NOT NULL	Iegūtā atzīme mācību priekšmetā.
Datums_atz	datetime	NOT NULL	Datums, kad iegūta atzīme.

3.2.1.1.7. Datu bāzes tabula „Skolnieki''

Tabula paredzēta, lai uzglabātu skolnieku persondatus.

3.7. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
SkonieksID	int(11)	PRIMARY KEY, NOT NULL	Unikāls personas identifikators.
SkolasID	int(11)	NULL	Skolas identifikators.
Uzv_v	varchar(50)	NULL	Personas uzvārds un vārds.
Aizbildnis	varchar(50)	NULL	Personas aizbildņa uzvārds un vārds.
PK	varchar(12)	NULL	Skolnieka personas kods.
Adrese1	varchar(50)	NULL	Personas dzīvesvietas ielas nosaukums.
Adrese2	varchar(50)	NULL	Personas dzīvesvietas pagasta/pilsētas/mājas nosaukums.
Indeks	varchar(7)	NULL	Pasta indekss attiecīgi personas dzīvesvietai.

ATelefons	varchar(50)	NULL	Aizbildņa telefona numurs.
scoro_id	int(11)	NOT NULL	Skolnieka identifikators Scoro sistēmā.

3.2.1.1.8. Datu bāzes tabula „Skolnieki_beidza''

Tabula paredzēta, lai uzglabātu persondatus par skolniekiem, kuri jau mācības beiguši.

3.8. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
SkonieksID	int(11)	PRIMARY KEY, NOT NULL	Unikāls personas identifikators.
SkolasID	int(11)	NULL	Skolas identifikators.
Uzv_v	varchar(50)	NULL	Personas uzvārds un vārds.
Aizbildnis	varchar(50)	NULL	Personas aizbildņa uzvārds un vārds
PK	varchar(12)	NULL	Skolnieka personas kods.
Klase	int(11)	NULL	Klase, kurā tika pabeigtas mācības.
Grupa	varchar(50)	NULL	Personas piederošā mācību grupa.
Gads	int(11)	NULL	Mācību pēdējais gads.
Komentari	varchar(100)	NULL	Komentāri par personu.
scoro_id	int(11)	NOT NULL	Skolnieka identifikators Scoro sistēmā.

3.2.1.1.9. Datu bāzes tabula „Visas_skolas”

Tabula paredzēta Latvijas izglītības iestāžu glabāšanai, lai tās varētu indeksēt Scoro sistēmā.

3.9. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
id	int(11)	PRIMARY KEY, NOT NULL, AUTO_INCREMENT	Unikāls skolas identifikators.
scoro_id	int(11)	NOT NULL	Skolas identifikators Scoro sistēmā.
skola	varchar(255)	NOT NULL	Skolas pilnais nosaukums.

3.2.1.1.10. Datu bāzes tabula „Visi_skolnieki’’

Tabula paredzēta, lai uzglabātu visu skolnieku persondatus.

3.10. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
id	int(11)	PRIMARY KEY, NOT NULL, AUTO_INCREMENT	Unikāls personas identifikators.
scoro_id	int(11)	NOT NULL	Skolnieka identifikators Scoro sistēmā.
vards	varchar(255)	NOT NULL	Personas vārds.
uzvards	varchar(255)	NOT NULL	Personas uzvārds.
personas_kods	varchar(255)	NOT NULL	Skolnieka personas kods.
prog_nosaukums	varchar(255)	NOT NULL	Mācību programmas nosaukums, kurā skolnieks mācās.
dok_nosaukums	varchar(255)	NOT NULL	Personas dzīvesvietas pagasta/pilsētas/mājas nosaukums.
numurs	varchar(255)	NOT NULL	Pasta indekss attiecīgi personas dzīvesvietai.

skolas_id	int(11)	NOT NULL	Skolas identifikators, kuru skolnieks pabeidza.
statuss	varchar(255)	NOT NULL	Skolnieka pašreizējais statuss.

3.2.1.2. Datubāzes tabula BVK sistēmai

BVK sistēmas datu sinhronizācijai ir atšķirīgs risinājums no esilideris.lv un SIA „BCLS" sistēmu sinhronizācijām, un visi dati tiks glabāti vienā tabulā.

Tabula „bvk_savienojuma_lauki" glabā lielu skaitu lauku (vairāk nekā 70 lauki), tādēļ tiks attēloti būtiskākie 10 lauki.

3.2.1.2.1. Datu bāzes tabula „bvk_savienojuma_lauki"

Tabula paredzēta, lai uzglabātu visus nepieciešamos persondatus datu sinhronizācijai.

3.11. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
c_zinasparpersonu	varchar(255)	NULL	Informācija par personu tekstuālā formātā.
c_personaslietasnumurs	text	NULL	Personas lietas numurs BVK sistēmā.
c_faktiskaadrese	text	NULL	Personas adrese.
c_pasesserijasnr	text	NULL	Personas pases numurs.
c_pasesizdosanasdatums	datetime	NULL	Personas pases izdošanas datums.
c_pasestermins	datetime	NULL	Datums, līdz kuram derīga personas pases.
c_pasesizdosanasvieta	text	NULL	Pasta nodaļa un tās adrese, kur izdota personas pase.
c_idkartesnr	text	NULL	Personas identifikācijas kartes numurs.
c_pilsonba	varchar(255)	NULL	Personas pilsonība.

c_statussvalsti	varchar(255)	NULL	Personas valstiskais statuss.

3.2.1.3. Datubāzes tabulas SIA „BCLS" sistēmai

Tabulas „gg_order_new" glabā lielu skaitu lauku (21 lauki), tādēļ tiks attēloti būtiskākie 10 lauki no tabulas.

3.2.1.3.1. Datu bāzes tabula „gg_contacts"

Tabula paredzēta, lai uzglabātu visus nepieciešamos SIA „BCLS" sadarbības kompāniju datus.

3.12. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
id	int(11)	PRIMARY KEY, NOT NULL, AUTO_INCREMENT	Unikāls kompānijas identifikators.
contact_id	int(11)	NOT NULL	Kompānijas identifikators SIA „BCLS" sistēmā.
name	text	NOT NULL	Kompānijas nosaukums.
contact_type	text	NOT NULL	Kompānijas tips.
id_code	int(11)	NOT NULL	Kompānijas identifikators.
manager_id	int(11)	NOT NULL	Kompānijas direktora identifikators SIA „BCLS" sistēmā.

3.2.1.3.2. Datu bāzes tabula „gg_orders"

Tabula paredzēta, lai uzglabātu visus nepieciešamos SIA „BCLS" pasūtījumu datus.

3.13. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
id	int(11)	PRIMARY KEY, NOT NULL	Pasūtījuma apmaksātāja identifikators.

doc_id	int(11)	NOT NULL, AUTO_INCREMENT	Pasūtījuma dokumenta identifikators.
doc_num	int(11)	NOT NULL	Sagatavoto pasūtījuma dokumentu skaits.
sum	double	NOT NULL	Pasūtījuma summa.
pvn	int(11)	NOT NULL	Aprēķinātā pievienotās vērtības nodokļa summa.
date	date	NOT NULL	Pasūtījuma veikšanas datums.
prod	text	NOT NULL	Pasūtītā produkta identificējošs kods.
price	double	NOT NULL	Pasūtītā produkta cena gabalā.
amount	int(11)	NOT NULL	Pasūtīto produktu daudzums.
status	text	NOT NULL	Informācija, vai pasūtījums ticis apmaksāts.

3.2.1.3.3. Datu bāzes tabula „gg_order_new”

Tabula paredzēta, lai uzglabātu visus nepieciešamos SIA „BCLS” jauno pasūtījumu datus.

3.14. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
id	int(11)	PRIMARY KEY, NOT NULL	Pasūtījuma identifikators.
cont	int(11)	NOT NULL	Pasūtītāja kompānijas identifikators.
cont_nam	varchar(255)	NOT NULL	Pasūtītāja kompānijas nosaukums.
doc_id	int(11)	NOT NULL, AUTO_INCREMENT	Pasūtījuma dokumenta numurs.
machine	varchar(255)	NOT NULL	SIA „BCLS” sistēmas specializēts lauks.
tircount	varchar(255)	NOT NULL	SIA „BCLS” sistēmas specializēts lauks.
sum	varchar(255)	NOT NULL	Jaunā pasūtījuma naudas summa.

codes	varchar(255)	NOT NULL	Loģistikas pārvadājumu karnešu saīsinājumi.
clientprice	varchar(255)	NOT NULL	Pasūtītāja piedāvātā pakalpojuma summa.
added_date	timestamp	NOT NULL, CURRENT_TIMESTAMP	Laiks, kurā pasūtījums pievienots datubāzei.

3.2.1.3.4. Datu bāzes tabula „gg_produkts”

Tabula paredzēta, lai uzglabātu visus nepieciešamos SIA „BCLS” piedāvāto produktu datus.

3.15. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
id	int(11)	PRIMARY KEY, NOT NULL	Tabulas rindu identifikators.
unit	text	NOT NULL	Produkta nosaukums.
price	double	NOT NULL	Produkta fiksētā cena.
comment	text	NOT NULL	Piezīmes par produktu.
product_id	int(11)	NOT NULL	Produkta identifikators.

3.2.1.3.5. Datu bāzes tabula „gg_users”

Tabula paredzēta, lai uzglabātu SIA „BCLS” sistēmas lietotāju lomas.

3.16. tabula

Kolonna	Datu tips	Cita informācija	Apraksts
id	int(11)	PRIMARY KEY, NOT NULL	Tabulas rindu identifikators.
user	text	NOT NULL	Sistēmas lietotāja nosaukums.
pass	text	NOT NULL	Sistēmas lietotāja parole.
role	text	NOT NULL	SIA „BCLS” sistēmas noteikta lietotāja loma.

3.3. Nefunkcionālo prasību realizācija

3.3.1. Drošības prasību realizācija

Drošība tiks nodrošināta, apgūstot un pielietojot PHP PDO un Base64[2] drošās programmēšanas metodes. PHP PDO metode nodrošina datubāzes pret SQL injekcijām, un Base64 veic bināro datu šifrēšanu, lai sinhronizējamo datu plūsmas tīmeklī būtu drošas un nerastos datu zagšanas iespējas.

Scoro CRM sistēmā pārsūtamo datu drošība tiks nodrošināta, izmantojot HTTPS un SSL drošības protokolus, kas veic pārsūtāmo datu šifrēšanu tīmeklī un Scoro sistēmā.

Personas sensitīvo sinhronizējamo datu drošība tiks nodrošināta, aizsargājot datus ar drošām parolēm, gan konfigurācijas rīkos, gan datorā no kura tika veikta projekta izstrāde.

3.3.2. Uzturamības prasību realizācija

Programmkoda uzturamība tiks nodrošināta rakstot atbilstošos un pietiekamus komentārus, piešķirot mainīgajiem loģiskus nosaukumus, komentējot izmantotās iebūvētās funkcijas, kā arī tiks izveidota dokumentācija, lai būtu iespējams mainīt un papildināt datu sinhronizācijas programmkodu, ja tas kādreiz būs nepieciešams.

3.3.3. Lietotāja saskarnes prasību realizācija

SIA „BCLS” datu sinhronizācijas projektā ārējās saskarnes izveidei tiks izmantots Bootstrap [9], ar kura palīdzību tiks izveidoti divi ārējās saskarnes skati, kas ir viegli lietojami, pārredzami un kuros ir atrodama tikai datu ievade un to pievienošana Scoro sistēmai.

4. TESTĒŠANAS DOKUMENTĀCIJA

4.1. Ievads

4.1.1. Nolūks

Šajā dokumentācijas nodaļā aprakstīts izstrādātās Scoro CRM datu sinhronizācijas testēšanas darbu ieskats. Testēšanas dokumentācija sniedz ieskatu datu sinhronizācijas testēšanas ievērotajos principos, praktiskajos procesos, testēšanas gaitā un gala rezultātos.

4.1.2. Saistība ar citiem dokumentiem

Testēšanas darbi veikti un testēšanas dokumentācija, žurnāls izstrādāti, balstoties uz programmatūras prasību specifikācijas nodaļā minētajām Scoro datu sinhronizācijas funkcionālajām prasībām.

4.2. Testēšanas apraksts

Testēšana tika veikta atsevišķi katrā no esilideris.lv, bvk.lv un SIA „BCLS" sistēmām, un testēšanas darbi tika veikti pēc principa, ka katrs no datu sinhronizācijas veiktajiem algoritma soļiem tika testēts atsevišķi un uzreiz pēc algoritma soļa izstrādes pabeigšanas. Katras iterācijas beigās tika veikts pārbaudes veida pārskats, vai algoritma soļi un to testēšana ir noritējuši pareizi. Tika aprakstīts testēšanas žurnāls pēc sinhronizācijas soļiem un gaidāmās funkcionalitātes.

4.3. Žurnāls

4.3.1. Esilideris.lv sistēmas datu sinhronizācijas testēšana

4.3.1.1. Datu konvertēšana uz SQL formātu

4.1. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
1.	Tiek salīdzināts .mdb fails ar izveidoto SQL failu, un tiek pārbaudīts vai datu daudzums sakrīt.	Visi .mdb dati ir veiksmīgi konvertēti SQL skriptā bez jebkādiem datu zudumiem.	Sekmīgs.

4.3.1.2. Datu bāzes tabulu un lauku izveide

4.2. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
2.	Tiek salīdzināts izveidotais SQL fails ar izveidotajām datu bāzes tabulām un to laukiem, un tiek pārbaudīts, vai visi tabulu lauki tiek nosaukti atbilstoši SQL failiem.	Pareizi izveidotas datu bāzu tabulas ar precīzi nosauktiem tabulu laukiem.	Sekmīgs.

4.3.1.3. SQL skripta pārveidošana

4.3. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
3.	Lai pārliecinātos par SQL koda pareizību, pārveidotais SQL fails tiek pievienots phpMyAdmin datubāzes rīkā.	PhpMyAdmin datubāzes rīks nesniedz kļūdu paziņojumu, un pārveidotais SQL fails ir veiksmīgi pievienots.	Sekmīgs.

4.3.1.4. SQL skripta augšupielāde

4.4. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
4.	Tiek salīdzināta SQL koda datu bāzes tabulas struktūra un datu rindu skaits starp phpMyAdmin rīku un datu uzglabāšanas .mdb failu.	PhpMyAdmin datubāzes rīkā ir tik pat daudz kolonu un datu rindu kā konkrētajai datu bāzes tabulai .mdb formātā.	Sekmīgs.

4.3.1.5. REST API savienojuma izveidošana un datu sinhronizācija

4.5. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
5.	Testēšana noris caur Scoro sistēmu, kurā tiek pārbaudīts, vai savienojums tika veikts un visi nepieciešamie dati ir sinhronizēti.	Ievadot kādus no sinhronizētajiem datiem Scoro, parādās logs, kas uzrāda ievadītos datus pilnīgi.	Sekmīgs.

4.3.2. Bvk.lv sistēmas datu sinhronizācijas testēšana

4.3.2.1. Datu ieguve no „Valis'' sistēmas

4.6. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
1.	Tiek salīdzināti visi iegūtie dati no „Valis" sistēmas ar datiem Cpanel konfigurācijas rīkā.	Iegūtie dati no „Valis" ir tādā pašā apjomā kā izstrādes konfigurācijas rīkā.	Sekmīgs.

4.3.2.2. Iegūto XML failu nolasīšana, pārveidošana un sinhronizēšana ar Scoro

4.7. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
2.	Tiek pārskatīts vai darbībās ar XML failiem nav radušies datu zudumi, un vai iegūtie dati no „Valis" sistēmas sakrīt ar datiem Scoro sistēmā.	XML failu pārstrāde ir noritējusi bez datu zudumiem, un dati tiek izvadīti Scoro sistēmā.	Sekmīgs.

4.3.2.3. Atbilstoša ieplānotā uzdevuma izveidošana

4.8. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
3.	Tiek izveidots testa ieplānotais uzdevums, kas sinhronizēs datus tuvākajā brīdī. (Pēc tam ieplānotā uzdevuma darbības laiks tiek pārmainīts uz definēto laiku, kas konkrētajā projektā ir vienreiz diennaktī pulksten 01:00, un tiek pārbaudīts vēlreiz.)	Dati veiksmīgi sinhronizēti, un tiek izvadīti Scoro attiecīgi pēc ieplānotā uzdevuma darbības laika.	Sekmīgs.

4.3.3. SIA „BCLS" sistēmas datu sinhronizācijas testēšana

4.3.3.1. Datu ielasīšana ārējā saskarnē no Scoro sistēmas

4.9. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
1.	Tiek salīdzināti sinhronizācijas dati starp Scoro un autores izstrādāto ārējo saskarni.	Dati veiksmīgi sinhronizēti, un Scoro esošie datie tiek izvadīti saskarnē.	Sekmīgs.

4.3.3.2. Atbilstoša ieplānotā uzdevuma izveidošana

4.10. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
2.	Tiek izveidots testa ieplānotais uzdevums, kas sinhronizēs datus tuvākajā brīdī. (Pēc tam ieplānotā uzdevuma darbības laiks tiek pārmainīts uz definēto laiku, kas konkrētajā projektā ir vienreiz diennaktī pulksten 00:00, un tiek pārbaudīts vēlreiz.)	Dati veiksmīgi sinhronizēti, un tiek izvadīti ārējā saskarnē attiecīgi pēc ieplānotā uzdevuma darbības laika.	Sekmīgs.

4.3.3.3. Iegūto datu nolasīšana, pārveidošana un sinhronizēšana ar Scoro

4.11. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
3.	Tiek salīdzināti dati starp Scoro un autores izstrādāto ārējo saskarni, kas sinhronizēti pēc ieplānotā uzdevuma darbības laika.	Dati veiksmīgi sinhronizēti starp Scoro un ārējo saskarni bez datu zudumiem.	Sekmīgs.

4.3.3.4. Manuāli pievienoto datu funkcionāla sinhronizācija

4.12. tabula

Nr.	Testa scenārijs	Sagaidāmais rezultāts	Testa rezultāts
4.	Tiek pārbaudīts no Scoro sistēmas puses, vai pēc lietotāja pieprasījuma dati no ārējās saskarnes tika nosūtīti uz Scoro.	Lietotāja nosūtītie dati parādās Scoro sistēmas atbilstošajās sadaļās.	Sekmīgs.

5. PROJEKTA ORGANIZĀCIJA

Projekti tika izstrādāti sabiedrības ar ierobežotu atbildību Stratcom projektu ietvaros, izmantojot spējās programmatūras izstrādes modeli. Izstrādes sākumā tika apgūti teorētiskie un praktiskie materiāli, lai iepazītu Scoro sistēmas un API savienojumu darbību, kā arī lai apgūtu PHP programmēšanas valodu un SQL vaicājumus. Izstrādes sākumā projekta vadītājs definēja vēlamos rezultātus, iespējamos sarežģījumus un lietas, kurām pievērst īpašu uzmanību. Tika definēti arī konfidencialitātes aspekti, jo sinhronizētie dati ir marķējami kā sensitīvi, kurus nedrīkst nodot trešajām personām. Prasības definēja SIA Stratcom Scoro projektu vadītājs, bet tās realizēja darba autore.

Izstrādes process norisinājās pēc attiecīgajiem projekta algoritma soļiem, kurus definēja projekta vadītājs. Testēšana norisinājās paralēli izstrādes procesam – katrs algoritma solis tika testēts atsevišķi, jo projekta algoritma soļi tika veikti un izstrādāti dažādās vidēs, kā arī bija nepieciešams vērst uzmanību, lai neradītos nekādi sinhronizējamo datu zudumi starp algoritma soļiem. Gala testēšana tika veikta pašā Scoro sistēmā ar jau sinhronizētajiem datiem.

Pēc gala testēšanas norisinājās projekta dokumentācijas galīgās versijas izstrāde, kur tika apkopoti materiāli un informācija, kas saistīti ar projekta izstrādi.

Projekti tika izstrādāti individuāli un dažādās vidēs, tādēļ netika izmantota neviena versiju kontroles sistēma. Projekta darbu organizēšanai tika izmantota Scoro sistēma, kas tiek plaši izmantota pašā prakses uzņēmumā, lai uzdotu uzdevumus un pārvaldītu projektā notiekošos procesus.

Projektu vadībai, jautājumiem, konsultāciju sniegšanai un problēmu risinājumiem Scoro datu sinhronizācijas projektos parasti tiek atvēlēti aptuveni 15% no kopējā projekta izstrādes laika, kas tika realizēts un ņemts vērā arī autores izstrādātajās datu sinhronizācijās.

Projektu izstrāde tika veikta izmantojot:

1) Microsoft Access esilideris.lv sistēmas datu uzglabāšanai.
2) Bullzip rīku esilideris.lv sistēmas datu konvertācijai no .mdb uz SQL formātu.
3) Bootstrap atvērtā pirmkoda rīku kopumu SIA „BCLS" ārējās saskarnes izstrādei.
4) phpMyAdmin rīku tabulu veidošanai un datu glabāšanai datu bāzē.
5) Cpanel uzturēšanas paneli, lai uzglabātu nepieciešamos datus un programmkoda failus.

6) PHP programmēšanas valodu, lai tiktu izveidoti API savienojumi un lai noritētu funkcionāla datu sinhronizācija.

7) Scoro CRM sistēmu, lai organizētu projekta norisi, kā arī lai veiktu datu implementēšanu, testēšanu un datu sinhronizāciju ar pašu sistēmu.

6. KONFIGURĀCIJAS PĀRVALDĪBA

Izstrādātajiem datu sinhronizācijas projektiem ir dažādi algoritma soļi, kuru vides un rīki atšķiras, tādēļ projektā nebija vienotas versiju kontroles sistēmas. Veicamie projekta izstrādes uzdevumi tika pārvaldīti ar Scoro CRM sistēmu – projekta vadītājs sniedza veicamos uzdevumus un informāciju darba autorei ar Scoro cilvēkresursu pārvaldības sistēmas iespējām. Viss nepieciešamais programmkods un dati tika glabāti uzturēšanas platformā Cpanel, kurā var redzēt pēdējo izmaiņu veikšanas datumu, savukārt datu bāzes tabulas un faili tika glabāti phpMyAdmin rīkā, kurā arī ir iespējams sekot līdzi veiktajām izmaiņām. Projekti tika izstrādāti individuāli, un sekošana līdzi izmaiņu veikšanai nebija tik būtiska. Neskaitot tiešsaistes vides un rīkus, kur tika glabāti projekta izpildes faili, rezerves kopijas tika glabātas uz prakses uzņēmuma lokālā servera.

Šādas konfigurācijas pārvaldības metodes tika izvēlētas, jo projekts tika izstrādāts prakses ietvaros, un šāda ir autores prakses uzņēmuma veicamo darbu un projektu kultūra.

7. KVALITĀTES NODROŠINĀŠANA

Lai nodrošinātu kvalitatīva projekta produkta izstrādi tika veiktas sekojošas darbības:

1) Projektu pavadošā dokumentācija tika izstrādāta, vadoties pēc atbilstošajiem Latvijas Valsts standartiem, attiecīgi "Programmatūras prasību specifikācijas ceļvedis, LVS 72:1996" un "LVS72:1996 Ieteicamā prakse programmatūras projektējuma aprakstīšanai".

2) Izstrādes laikā tika ievērots vienots programmēšanas koda stils, un visi mainīgie tika nosaukti izmantojot "underscore" stilu, kas atdala mainīgā nosaukuma vārdus un uzlabo tā lasāmību.

3) Programmkods tika atbilstoši komentēts, un pirms koda funkcijām tika aprakstīts, kam funkcija paredzēta. Programmatūras koda izstrādes laikā tika ievēroti arī DRY (don't repeat yourself) principi.

4) Katram projekta algoritma solim tika veikta testēšana, lai pārliecinātos par programmkoda un algoritma soļu pareizu izpildi, kā arī nodrošinātu to, lai nerastos datu zudumi.

5) Gan komentāri, gan mainīgo un failu nosaukumi tika atbilstoši un loģiski nosaukti, kas palīdz vieglāk uztvert kodu, failu nozīmi un mērķi.

6) Programmatūras izstrādes laikā tika rīkotas regulāras sapulces ar Scoro projektu vadītāju, kurās tika apspriesta projekta gaita, apskatīti rezultāti, pārrunātas problēmsituācijas, kā arī novērtēti paveiktie un plānoti turpmākie uzdevumi.

7) Pēc algoritma soļu testēšanas un gala testēšanas Scoro sistēmā var secināt, ka Scoro CRM sistēmas sinhronizācija ir noritējusi veiksmīgi. Testēšanā piedalījās arī pieredzējušie uzņēmuma kolēģi, kas pārliecinājās, ka iesniegtā koda kvalitāte un dati ir sinhronizēti bez kļūdām un zudumiem.

8. DARBIETILPĪBAS NOVĒRTĒJUMS

Autorei nebija iepriekšējās pieredzes API savienojumu izveidē un datu sinhronizācijā, tāpēc izstrādājamā projekta darbietilpības novērtēšana tika veikta pēc ekspertu un pieredzējušo prakses uzņēmuma kolēģu metodes. Prakses uzņēmuma visi projekti un veicamie darbi tiek veikti pēc spējās programmēšanas modeļa izmantojot Scrum metodoloģiju. Šādā veidā tika izstrādāti arī kvalifikācijas darba veicamās datu sinhronizācijas un pārējie nepieciešamie procesi, un tādēļ arī darbietilpība tika novērtēta pēc šīs metodoloģijas.

Veicamie projekti tika sadalīti 5 iterācijās (pēc Scrum metodoloģijas – sprintos), ar iterācijas garumu 2 nedēļas. Katrā iterācijā bija atšķirīgs lietotāju stāstu daudzums, un katrs no šiem stāstiem tika novērtēts ar sarežģītības punktu skaitu, izmantojot Fibonači skaitļu sekvenci (0, 1, 2, 3, 5, 8, 13, 21).

Plānojot minētās 5 iterācijas (ieskaitot testēšanu), pirms izstrādes kopā tika aprēķināti 186 sarežģītības punkti, kas atbilst 2,5 personmēnešiem. Laika plānošana šādos inovatīvos projektos kā API sinhronizācija ar salīdzinoši jaunieviestu sistēmu kā Scoro, var kļūt kompleksa, jo izstrādes laikā rodas traucēkļi, kurus nav iespējams tik ātri atrisināt, jo problēmu risināšanas atbalsts nav plaši pieejams.

Pēc plānoto 5 iterāciju veikšanas, tika analizēts, ka nepieciešama vēl viena iterācija jeb Scrum sprints izstrādei, kā arī nepieciešama atsevišķa dokumentācijas izstrādes iterācija. Pēc laika pārplānošanas, kopumā tika aprēķināti 255 sarežģītības punkti un 7 projekta iterācijas katra divu nedēļu garumā, kas rezultējas 3,5 personmēnešos.

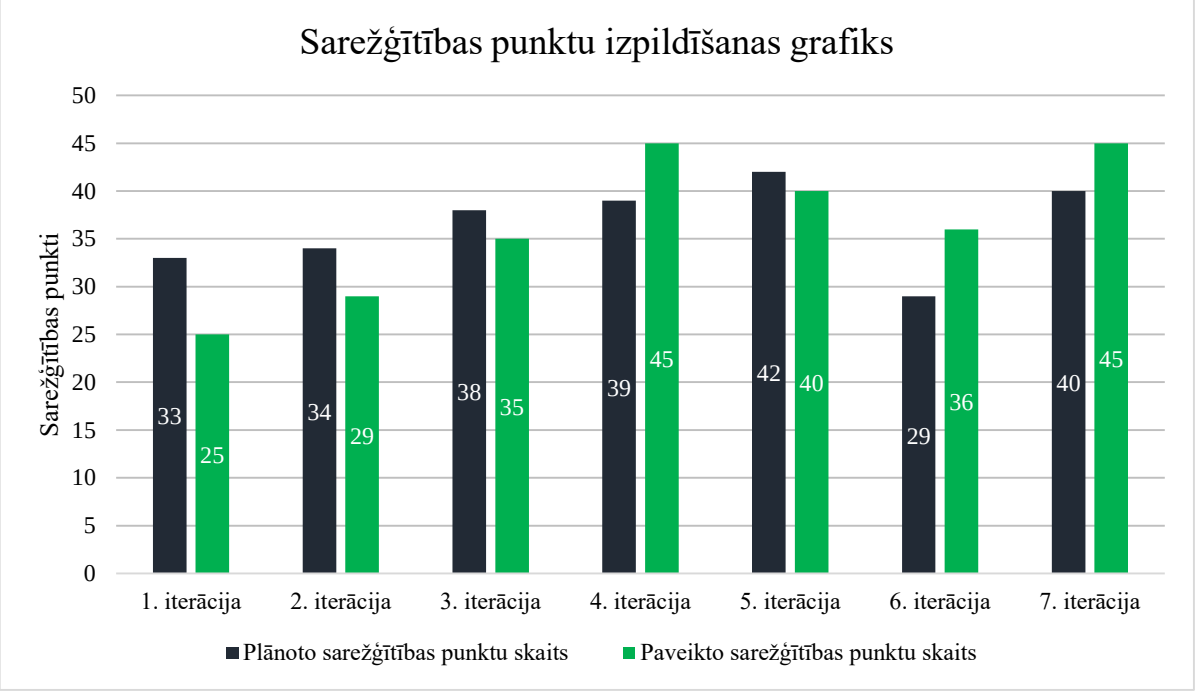
Izstrādes ātrums (velocity):

- 255/7 = 36,4 vidējais sarežģītības punktu skaits iterācijā
- 36,4/10 = 3,64 vidējais sarežģītības punktu skaits dienā
- 3,64/8 = 0,46 vidējais sarežģītības punktu skaits stundā

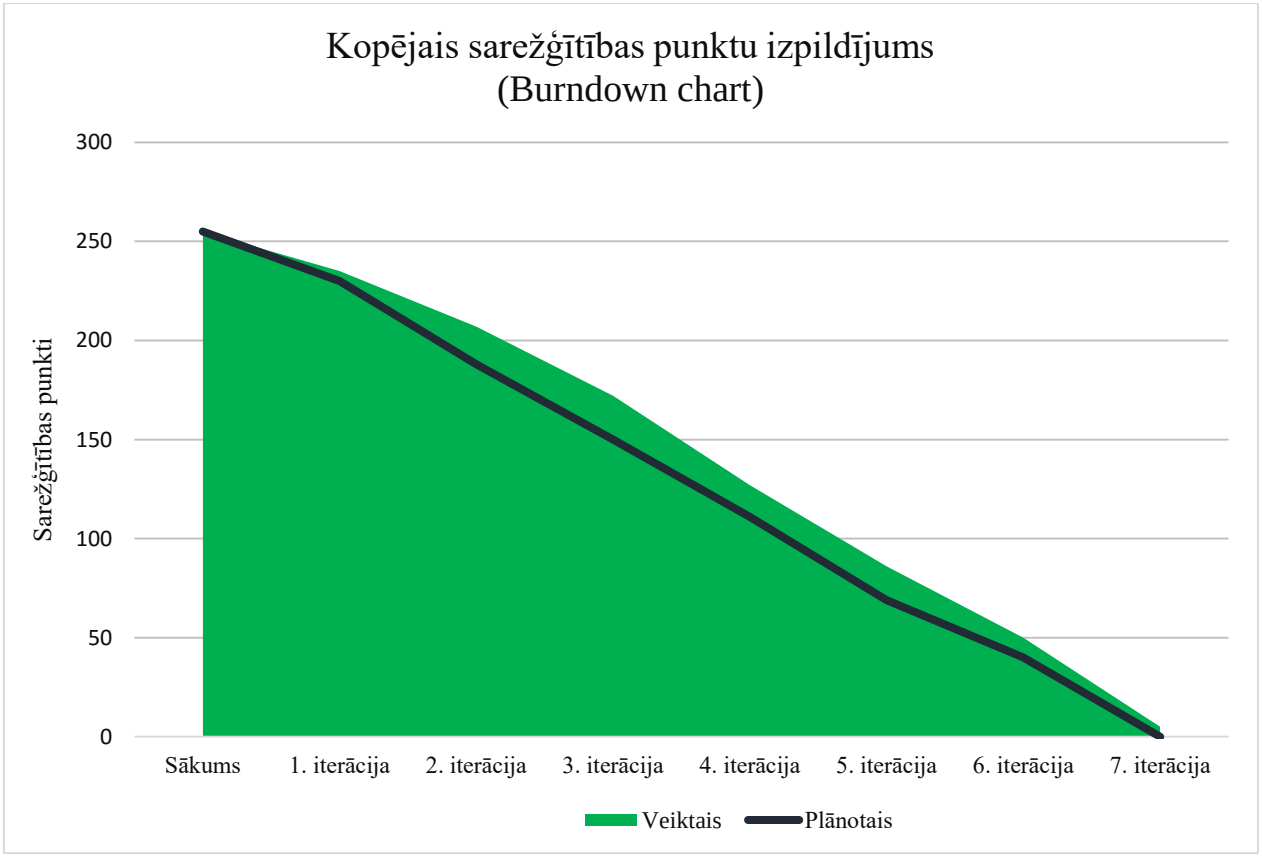
8.1. tabula

	Esilideris.lv	**Bvk.lv**	**SIA „BCLS”**	**Dokumentācija**
Sarežģītības punkti	72	55	83	45
Stundas	157	120	180	98

Kopā tika atvēlētas 555 stundas, kas rezultējas 69,4 dienās jeb 3,47 personmēnešos.



8.1. att. **Sarežģītības punktu plānojums un izpildījums katrai iterācijai**



8.2. att. **Sarežģītības punktu kopējais plānojums un izpildījums**

Koda rindu skaits esilideris.lv sistēmas datu sinhronizācijai

8.2. tabula

Valoda	Failu skaits	Koda rindu skaits	Tukšas rindas	Komentāru rindas	Kopējais rindu skaits
PHP	12	1020	230	127	1397
SQL	10	59688	55	88	59831

Koda rindu skaits bvk.lv sistēmas datu sinhronizācijai

8.3. tabula

Valoda	Failu skaits	Koda rindu skaits	Tukšas rindas	Komentāru rindas	Kopējais rindu skaits
PHP	2	204	32	18	257
SQL	1	280	10	13	303
XML	1	310	8	15	333

Koda rindiņu skaits SIA „BCLS” sistēmas datu sinhronizācijai

8.4. tabula

Valoda	Failu skaits	Koda rindu skaits	Tukšas rindas	Komentāru rindas	Kopējais rindu skaits
PHP	4	148	38	11	199
SQL	5	186	30	53	266
HTML	2	418	85	42	545
CSS	8	530	23	50	593
Javascript	8	415	10	18	443

Koda rindiņu skaitīšanai tika izmantots atvērtā pirmkoda rīks LineTally[2]. Atšķirības sistēmu SQL rindiņu skaitā skaidrojams ar faktu, ka esilideris.lv sistēmas datu sinhronizācija ir vienreizēji veicams darbs, un visiem iepriekš sistēmā sakrātajiem datiem bija nepieciešama pārnese uz Scoro CRM sistēmu, kamēr bvk.lv un SIA „BCLS" datu sinhronizācijām tika definētas citas prasības un risinājumi.

SIA „BCLS" sistēmas ārējās saskarnes izstrādei tika izmantots Bootstrap, kas ietver HTML, CSS un Javascript, un lielākā daļa no koda rindiņām tika ģenerētas izmantojot Bootstrap tehnoloģiju.

[2] LineTally http://www.freewarefiles.com/LineTally_program_36090.html

9. SECINĀJUMI

Kvalifikācijas darba izstrādes rezultātā ir veikta veiksmīga datu sinhronizācija starp Scoro CRM un bvk.lv, esilideris.lv un SIA „BCLS" sistēmām, kā arī tika izstrādāta atbilstoša dokumentācija. Izstrādātā programmkoda rezultātā tika izpildīti sākotnējie izvirzītie projekta mērķi un tika izpildīta prasītā funkcionalitāte.

Izstrādājot projektu pēc spējās programmēšanas metodoloģijas, tika veiksmīgi organizēts laiks un projekta virzība, kā arī tika savlaicīgi apzināta problēmsituāciju rašanās, kas ļāva optimāli aprēķināt papildu nepieciešamo izstrādes laika iterāciju.

Ievērojot labo programmēšanas praksi, rakstot saprotamus un loģiskus komentārus un ievērojot atbilstošus koda funkciju nosaukumus, bija pārvaldība pār PHP programmkodu, kas ļāva tajā labāk orientēties, uzlabojot izstrādes laiku.

Ļoti liela nozīme projekta izstrādē bija izmantotajiem papildrīkiem, sistēmām un Scrum metodoloģijai, kas nodrošināja ļoti labu savstarpējo sadarbību un komunikāciju starp izstrādātāju un projekta vadītāju, kas palīdzēja arī risināt projektos radušās problēmsituācijas.

Scoro CRM sistēmas savstarpēja sinhronizācija ar API ir relatīvi inovatīvs risinājums, kas brīžiem radīja sarežģījums, jo šādiem risinājumiem nav pieejami pietiekami plašs atbalsts un informācijas daudzums, lai risinātu projektā laikā radušos izstrādes traucēkļus.

Autore, izstrādājot kvalifikācijas darbu, guva ievērojamu zināšanas un prasmes API savienojumu veidošanā un PHP programmēšanas valodā.

PATEICĪBAS

Autore izsaka pateicību prakses uzņēmuma SIA „Stratcom'' izpilddirektoram Ivaram Bandonim par prakses vietas piešķiršanu un iespēju došanu.

Autore izsaka pateicību kvalifikācijas darba vadītājam Matīsam Rikteram par nozīmīgiem ieteikumiem darba rakstīšanas procesā kā arī viņa ieguldīto laiku.

Autore izsaka pateicību SIA „Stratcom'' projektu vadītājam un vecākajam programmētājām Ērikam Zvaigznem par kvalifikācijas darba tēmas ieteikšanu un palīdzību būtiskajā projekta izstrādes procesā.

IZMANTOTĀ LITERATŪRA UN AVOTI

1. Vispārīga informācija ar Scoro CRM sistēmu [tiešsaiste]. Pieejams: https://www.scoro.lv/ [Pārbaudīts 29.05.2016].
2. PHP programmēšanas valodas vadlīnijas [tiešsaiste]. Pieejams: http://php.net/manual/en/index.php [Pārbaudīts 29.05.2016].
3. Scoro CRM API savienojumu vadlīnijas [tiešsaiste]. Pieejams: https://api.scoro.lv/api/ [Pārbaudīts 29.05.2016].
4. XML strukturēšanas un veidošanas vadlīnijas [tiešsaiste]. Pieejams: http://www.xmlobjective.com/create-an-xml-document-using-php/ [Pārbaudīts 29.05.2016].
5. LVS 68:1996 Programmatūras prasību specifikācijas ceļvedis [tiešsaiste]. Pieejams: http://estudijas.lu.lv/mod/resource/view.php?id=131427 [Pārbaudīts 29.05.2016].
6. LVS 72:1996 Ieteicamā prakse programmatūras projektējuma aprakstīšanai [tiešsaiste]. Pieejams: http://estudijas.lu.lv/mod/resource/view.php?id=131428 [Pārbaudīts 29.05.2016].
7. Kvalifikācijas darba izstrādes un aizstāvēšanas metodiskie norādījumi 2014 [tiešsaiste]. Pieejams: http://estudijas.lu.lv/mod/resource/view.php?id=178220 [Pārbaudīts 29.05.2016].
8. Prasības noslēgumu darbu izstrādei un saturam (LU rīkojums 2012.gads) [tiešsaiste]. Pieejams: http://estudijas.lu.lv/mod/resource/view.php?id=125421 [Pārbaudīts 29.05.2016].
9. Bootstrap dokumentācija [tiešsaiste] Pieejams: http://getbootstrap.com/ [Pārbaudīts 29.05.2016].

PIELIKUMI

1. Pielikums – PHP koda fragments no esilideris.lv datu sinhronizācijas

```php
    <?php
/* *** Skolu pievienošana *** */

//error_reporting(E_ALL);
/** Savienojuma detaļas **/
define('API_KEY', 'ScoroAPI_XXXXXXXXXXXXXXX');
define('BASE_URL', 'https://esilideris.scoro.lv/api/contacts/');
ini_set('default_charset', 'UTF-8');
$skolas = array();
$user = 'paraugsa';
$pass = 'xxxxxxxxxxxxx';
/** Savienojuma detaļu beigas **/

/** nodrošinājums pret sql injekcijām **/
$conn = new
PDO('mysql:host=localhost;dbname=paraugsa_esilideris;charset=utf8;',
$user, $pass);
$sql = 'SELECT * FROM Skolas';
$q = $conn->query($sql);
$q->setFetchMode(PDO::FETCH_ASSOC);

$skolas_data = array();

/** Scoro definētā API savienojuma informācija **/

//foreach($skolas as $skola){
    $fields = array(
        'apiKey' => API_KEY,
        'lang' => 'lat',
        'company_account_id' => 'skola',
        'per_page' => 100,
        'page' => 10
    );
$fieldsJson = json_encode($fields);

$ch = curl_init();
```

```
    curl_setopt($ch, CURLOPT_URL, BASE_URL);
    curl_setopt($ch, CURLOPT_POST, true);
    curl_setopt($ch, CURLOPT_POSTFIELDS, $fieldsJson);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

    $result = curl_exec($ch);
    if ($result) {
        $contacts = json_decode($result, true);

  /** Scoro definētās API savienojuma informācijas beigas **/

        foreach($contacts['data'] as $skola_scoro){
            $id = $skola_scoro['name'];
            $sql0 = "Select * FROM Skolas WHERE Nosaukums = '$id'";
            $q0 = $conn->query($sql0);
            $q0->setFetchMode(PDO::FETCH_ASSOC);
            while ($r0 = $q0->fetch()){
                $id_y = $r0['SkolasID'];
            }
            $skolas_data[$id_y] = $skola_scoro['contact_id'];
            //echo $skola_scoro['contact_id'];
        }

        foreach($skolas_data as $sk => $key){
            echo $key.' - '.$sk.'<br/>';

    $sql1 = "UPDATE Skolas SET scoro_id = '$key' WHERE SkolasID = '$sk'";
            $q1 = $conn->query($sql1);
            $q1->execute();
        }

    <!--    ?><pre><?print_r($contacts);?></pre><? -->

    }
//}
?>
```

2. Pielikums – PHP koda fragments no bvk.lv datu sinhronizācijas

```php
<?php

/** Savienojuma detaļas **/
error_reporting(E_ALL);
define('API_KEY', 'ScoroAPI_XXXXXXXXXXXXX');
define('BASE_URL', 'https://bvk.scoro.lv/api/contacts/');
/** Savienojuma detaļas beigas **/

$array = array();

/** Scoro definētā API savienojuma informācija **/
$fields = array(
    'apiKey' => API_KEY,
    'lang' => 'eng',
    'company_account_id' => 'skola',
    'per_page' => '99'
    );

$fieldsJson = json_encode($fields);

$ch = curl_init();
/** Scoro definētās API savienojuma informācijas beigas **/


/** funkcija XML faila izveidošanai **/
function array_to_xml( $contacts, &$xml_data ) {
    foreach( $contacts as $key => $value ) {
        if( is_array($value) ) {
            if( is_numeric($key) ){
                $key = 'DAT'; //dealing with <0/>..<n/> issues
            }
            $subnode = $xml_data->addChild($key);
            array_to_xml($value, $subnode);
        } else {
            $xml_data->addChild("$key",htmlspecialchars("$value"));
      }
   }
```

```
}
/** XML faila izveidošanas funkcijas beigas **/

/** Scoro dokumentācijas definētās savienojuma detaļas **/
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
curl_setopt($ch, CURLOPT_URL, BASE_URL);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $fieldsJson);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$result = curl_exec($ch);
if ($result) {
    $contacts = json_decode($result, true);
     $custom_fields = array ();
$count = count($contacts);
echo $count;
foreach($contacts['data'] as $contact){
    foreach($contact as $contacts => $key){
/** Scoro dokumentācijas definētās savienojuma detaļu beigas **/

$vertiba_1 = array_keys($key);
$vertiba_2 = array_values($key);
$vertiba_3 = array_keys($key['custom_fields']);
$vertiba_4 = array_values($key['custom_fields']);

foreach(array_combine($vertiba_1, $vertiba_2) as $nos => $vert){

$custom_fields[$nos] = $vert;

}
}
echo'<pre>';
print_r($custom_fields);
echo'</pre>';

/** Nepieciešamo lauku ievietošana masīvā **/

$data = array (
'name' => $contact['name'],
```

```
'lastname' => $contact['lastname'],
'contact_type' => $contact['contact_type'],
'id_code' => $contact['id_code'],
'bankaccount' => $contact['bankaccount'],
'birthday' => $contact['birthday'],
'position' => $contact['position'],
'comments' => $contact['comments'],
'sex' => $contact['sex'],
'timezone' => $contact['timezone'],
'manager_id' => $contact['manager_id'],
'is_supplier' => $contact['is_supplier'],
'is_client' => $contact['is_client'],
'modified_date' => $contact['modified_date'],
'addresses' => $contact['addresses'],
'means_of_contact' => $contact['means_of_contact'],
'tags' => $contact['tags'],

'custom_fields' => $custom_fields

);

$array[] = $data;
/** Nepieciešamo lauku ievietošana masīvā beigas **/

?>

<!-- <pre><?print_r($contacts);?></pre> -->

<?

}}
/** XML faila veidošana **/

$xml_data = new SimpleXMLElement('<?xml version="1.0"?><DATA></DATA>');
array_to_xml($array,$xml_data);
$result = $xml_data->asXML('xml/test2.xml');
?>
```

3. Pielikums – PHP koda fragments no SIA „BCLS" datu sinhronizācijas

```php
<?php

/** Savienojuma detaļas **/
$servername = "XXXXXXXX";
$username = "paraugsa";
$password = "xxxxxxxxxxxx";
$dbname = "paraugsa_logistika";

/** nodrošinājums pret sql injekcijām **/
try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$password);
    <!-- set the PDO error mode to exception -->
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully";
    }
catch(PDOException $e)
    {
    echo "Connection failed: " . $e->getMessage();
    }
/** nodrošinājums pret sql injekcijām beigas **/
/** Savienojuma detaļu beigas **/

/** Scoro dokumentācijas definētās savienojuma detaļas **/
error_reporting(E_ALL);
define('API_KEY', 'ScoroAPI_XXXXXXXXXXXXXXX');
define('BASE_URL', 'https://bcls.scoro.lv/api/');

$destination = array('contacts');

$fields = array(
    'apiKey' => API_KEY,
    'lang' => 'eng',
    'company_account_id' => 'DSD'
);
$fieldsJson = json_encode($fields);
```

```
$ch = curl_init();

foreach($destination as $dest){
curl_setopt($ch, CURLOPT_URL, BASE_URL.$dest);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $fieldsJson);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$result = curl_exec($ch);
if ($result) {
    $contacts = json_decode($result, true);
    ?><pre><?print_r($contacts);?></pre><?
/** Scoro dokumentācijas definētās savienojuma detaļu beigas **/

/** datu ievietošana masīvos un datubāzē **/
    foreach($contacts['data'] as $data){
    $contact_id = $data['contact_id'];
    $name = $data['name'];
    $lastname = $data['lastname'];
    $contact_type = $data['contact_type'];
    $id_code = $data['id_code'];
    $manager_id = $data['manager_id'];

    var_dump($contact_id);

        $sql = "INSERT INTO gg_contacts (contact_id, name, lastname,
contact_type, id_code, manager_id) VALUES ('$contact_id', '$name',
'$lastname', '$contact_type', '$id_code', '$manager_id')";
        $conn->exec($sql);
    }
/** datu ievietošana masīvos un datubāzē beigas **/

}
}
$conn = null;
?>
```

DOKUMENTĀRĀ LAPA

Kvalifikācijas darbs „***Scoro CRM sistēmas datu sinhronizācija izmantojot REST API***” izstrādāts Latvijas Universitātes Datorikas fakultātē.

Ar savu parakstu apliecinu, ka darbs izstrādāts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: ***Anna Gronska*** _____ **___.05.2016**.

Rekomendēju darbu aizstāvēšanai
Darba vadītājs: ***M. dat. Matīss Rikters*** _____ **___.05.2016.**

Recenzents: M. dat. Jānis Vempers

Darbs iesniegts 30.05.2016.
Kvalifikācijas darbu pārbaudījumu komisijas sekretārs: *Darja Solodovņikova* _____

Darbs aizstāvēts kvalifikācijas darbu pārbaudījuma komisijas sēdē
___.06.2016. prot. Nr. _____

Komisijas sekretārs(-e): _____