LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

REKLĀMAS SISTĒMAS IZSTRĀDE

BAKALAURA DARBS

Autors: **Anastasija Gončarova**
Studenta apliecības Nr.: ag13124
Darba vadītājs: Dr.dat. Asociētā profesore Darja Solodovņikova

RĪGA 2018

Anotācija

Šī darba mērķis ir apskatīt specifiskās sistēmas izstrādi un noteikt tai vispiemērotāko arhitektūru un atbilstošāko satvaru.

Tiek izstrādāta tīmekļa vietne, kas nodrošina daudzpusīgu darbu ar tīmekļa reklāmu dažādām lietotāju grupām - reklāmas firmas darbiniekiem (dizaineriem, statistiem, projektu vadītājiem utt.), mājaslapu apmeklētājiem, īpašniekiem utt.

Darba hipotēze, kas tiek apskatīta un pierādīta pētījuma laikā, ir sekojoša: šāda tipa tīmekļa programmatūrai optimāls satvars ir Laravel.

Atslēgvārdi: Laravel, PHP, tīmekļa programma, Internets

Abstract

ADVERTISING SYSTEM DEVELOPMENT

The purpose of this work is to look at the development of a specific system and identify the most appropriate architecture and framework for it.

Within this research, a website is developed. It provides multi-faceted web-based advertising for different user groups - the advertising companies employees (designers, statisticians, project managers, etc.), website visitors, owners, etc.

The work hypothesis that is examined and demonstrated during the study is as follows: the optimal framework for this kind of web-based software is Laravel.

Keywords: Laravel, PHP, web application, Internet

Satura radītājs

Apzīmējumu saraksts

Ar izstrādes procesu saistītie tehniskie apzīmējumi

Apzīmējums	**Skaidrojums**
Laravel	Satvars ar atvērtu pirmkodu, kas ir piemērots tīmekļa programmu izstrādei ar MVC arhitektūru
PHP	Programmēšanas valoda, uz kuras ir rakstīts un kurai piemērots Laravel satvars.
MVC	Datu sadalīšanas shēma tīmekļa programmā, arhitektūras veids, kur loģika un dati tiek sadalīti starp Modeļiem (Model), Skatiem (View) un Kontrolleriem (Controller)
Satvars	Platforma, kas nosaka programmas struktūru
HTML	HyperText Markup Language, standartā iezīmējuma valoda Interneta lapām
CSS	Cascading Style Sheets, formālā valoda lapas iezīmējuma formatēšanai, parasti tiek izmantota kopā ar HTML elementiem
Javascript, JS	Programmēšanas valoda, kas darbojas lietotāja pusē (priekšgalsistēmā)
Blade	Laravel iebūvēto šablonu sistēma priekš skatu noformēšanas, kas ļauj dinamiski savienot PHP un HTML
jQuery	Javascript bibliotēka, kuras galvenais uzdevums ir Javascript un HTML elementu mijiedarbība
Bootstrap	Priekšgalsistēmas funkcionalitātes instrumentu kopa, rakstīta uz HTML/CSS/Javascript
JSON	Teksta datu formāts priekš apmaiņai, bazēts uz Javascript
Response	Servera atbilde, kas parasti tiek sūtīta no aizmugursistēmas uz priekšmugursistēmu
FTP	File Transfer Protocol, standartizēts protokols priekš failu apmaiņai tīklos
Artisan	Laravel iebūvētā komandrindu saskarne, kas palīdz darboties ar konsoles komandām
Composer	PHP pakotņu pārvaldnieks, kas tiek izmantots ar Laravel un pārvalda datu pakotnes (piemēram, bibliotēkas)
OpenSSL	SSL un TLS protokolu realizācijas bibliotēka

Tokenizer	PHP bibliotēka priekš datu aizsardzības (piemēram, ievadot paroli formā un sūtot to serverim)
Mbstring	Multi-Byte string, PHP bibliotēka, kas pārvalda simbolu virknes, kurās iekļautie simboli nav daļa no ASCII
PDO	PHP Data Objects, PHP bibliotēka darbam ar dažādu tipu datubāzēm
Crontab, Cron	UNIX sistēmu programma, kas ļauj palaist operācijas bez tiešās sadarbības ar lietotāju, automātiski pēc noteiktā grafika
SQL	Structured Query Language, programmēšanas valoda darbam ar attiecību datubāzēm
Eloquent ORM	Object-Relational Mapping, programmēšanas tehnika, iebūvēta iekš Laravel, kas ļauj ērti strādāt ar modeļiem, kas ir saistīti ar datubāzi
Redis	Datu glabāšanas sistēma tīmekļa vietnēm, kas glabā datus masīvu veidā bez attiecībām starp tiem un nodrošina ātru darbību arr datiem.
Redirect	Lietotāja novirzīšana un konkrētu saiti kā servera atbildes tips

Izstrādātās sistēmas iekšējā terminoloģija

Apzīmējums	**Skaidrojums**
Sistēma	Izstrādātā tīmekļa vietne, kas tiek aprakstīta šī darba ietvaros
Firma	Uzņēmums, pēc kura darbinieku un valdes pasūtījuma tika izstrādāta apskatamā Sistēma
Mediju plāns	Reklāmas kampaņu kopa, kuru definē izpildīšanas laiks dienās, klients un atbildīgā persona
Reklāmas kampaņa	Noteiktā tipa reklāmas risinājums, kuru definē izpildīšanas laiks, nepieciešamo parādīšanu/klikšķu skaits, partneru mājaslapas, kur kampaņa tiks parādīta, un pieejamo tipu reklāmkarogi
Richmedia	Reklāmas risinājums, kuram ir raksturīga viena reklāmkaroga parādīšanās virs mājaslapas satura
Raspberry	Reklāmas risinājums, kuram ir raksturīga viena vai vairako reklāmkarogu ielāde mājaslapas saturā atzīmētās vietās
Mobilfy	Reklāmas risinājums, kuram ir raksturīga viena reklāmkaroga parādīšanās virs mobīlās mājaslapas satura
Billboard	Reklāmas risinājums, kuram ir raksturīga viena vai vairako reklāmkarogu ielāde mobīlās mājaslapas saturā atzīmētās vietās
Reklāmkarogs	Bilde, video vai HTML fails, kas tiek ielādēts no Sistēmas servera un parādīts mājaslapas apmeklētājam ar iespēju pāriet uz reklāmas saiti

Impresija, parādīšanās	Viena reklāmkaroga parādīšanās mājaslapas lietotājam, kas tiek reģistrēta Sistēmas statistikā
Kliks, klikšķis	Mājaslapas apmeklētāja uzspiešana uz reklāmkaroga, kas jaunajā cilnē atver attiecīgo reklāmas saiti un reģistrē datus Sistēmas statistikā
Google Adwords	Google reklāmas sistēma un tās ietvaros piedāvāto instrumentu kopa

Ievads

Katrai idejai, kas ir realizējama ar programmatūras palīdzību, var atrast vispiemērotāko valodu un atbilstošo satvaru. Un pretēji - katra valoda un satvars vislabāk realizē kādu konkrētu problēmu kategoriju.

Darba ietvaros tika apskatīta viena no tādām situācijām. Bija jāizstrādā tīmekļa programma, kas nodrošinātu ērtu darbu ar tīmekļa reklāmu vairākām lietotāju kategorijām: firmas darbiniekiem, kas veido reklāmas risinājumus, publicē tos partneru mājaslapās, apskata statistiku un veic citas darbības, kā arī mājaslapu īpašniekiem un apmeklētājiem.

Darba izstrādes sākumā tika izvirzīta hipotēze, ka vispiemērotākais satvars šāda tipa programmai ir Laravel. Laravel ir bezmaksas satvars ar atvērtu pirmkodu, kas nodrošina programmas izveidi ar MVC arhitektūru un PHP valodas palīdzību. Tika izvēlēts izmantot Laravel versiju 5.2. Izstrādes procesā tika apskatīti arī citi satvari un citas valodas, un veikta salīdzinošā analīze. Rezultātā tika secināts, ka Laravel tiešām ir optimāls risinājums dotās problēmas risināšanai.

Laravel piedāvā plašu funkcionalitāti visos MVC arhitektūras līmeņos: modeļu klases ir piemērotas mijiedarbībai ar datubāzes datiem un datu attiecībām, skatos ir iespēja izmantot iebūvēto Blade templating, kas ļauj ērti un eleganti apvienot HTML un PHP, un kontrolleri nodrošina pilnvertīgu mijiedarbību starp modeļiem un skatiem.

Laravel nodrošina iespēju veidot stabilu un ērtu servera puses kodu. Klientiem piedāvājot standarta HTML/CSS/jQuery komplektu, kas var tikt papildināts ar jebkādām bibliotēkām pēc nepieciešamības.

1. **IZSTRĀDĀTĀS TĪMEKĻA VIETNES UZDEVUMI**

Izstrādātā Sistēma ir Firmas pasūtījums. Precīzas tehniskās specifikācijas priekš izstrādes netika veidotas un viss izstrādes process tika balstīts uz programmētāja un pasūtītāja komunikāciju. Šajā gadījumā pasūtītāja lomā darbojās visi Firmas darbinieki, nereti arī Firmas partneri. Programmētāja uzdevums bija apkopot visu vēlmes, uzbūvēt visiem derīgu risinājumu un atbilstoši tam izveidot tīmekļa vietni.

Pirms izstrādes uzsākšanas tika veikta potenciālo lietotāju grupas aptauja. Respondentu skaits bija 20 cilvēki, kas visi ir Firmas darbinieki ar dažādiem pienākumiem. Respondenti tika sadalīti grupās atbilstoši viņu lomām Firmā (šīs lietotāju grupas tiks plašāk apskatītas apakšnodaļā 1.1), un katrai grupai bija jāatbild uz sekojošiem jautājumiem:

1. Kādi ir jūsu ikdienas pienākumi darba procesā?
2. Kādi darbi ir jāveic regulāri?
3. Kuri darbi aizņem visvairāk laika?
4. Kādai informācijai ir jāseko?
5. Kurai informācijai un kādiem rīkiem ir jābūt pieejamiem Firmas partneriem un klientiem?
6. Kāda funkcionalitāte varētu palīdzēt jūsu darba uzlabošanā?
7. Vai jums ir pieredze darbā ar citām reklāmu sistēmām, piemēram, Google Adwords vai AdForm? Kā jūs varētu nokomentēt šo pieredzi? Kāda noderīga funkcionalitāte no citām reklāmu sistēmām jums ir zināma?

Rezultātā tika apkopotas visu lietotāju grupu pienākumi, vēlmes, kurus darbus varētu atvieglot vai automatizēt, kā arī nepieciešamā mijiedarbība ar citām reklāmas sistēmām vai citu sistēmu funkcionalitāte, kas būtu nepieciešamā arī šajā sistēmā. Darba sākumā nebija iespējas uzzināt Firmas partneru un klientu vēlmes, tāpēc viņu vajadzības tika apkopotas no tiem Firmas darbiniekiem, kas ikdienā sadarbojas ar partneriem un klientirm. Tika sastādīts standartizēts darba procesa scenārijs, kura izpilde ir jānodrošina sistēmai, kā arī lietotāju vajadzības tika apkopotas atbilstoši pienākumiem, kas ir apskatīts tabulās 1.1.1, 1.1.2 un 1.1.3.

Sistēmai ir jāpilda vairāki uzdevumi, kas tika vispārīgi apskatīti iepriekš. Tai jāsatur sevī gan Firmas darbiniekiem nepieciešamā un noderīgā funkcionalitāte, gan jāsniedz nepieciešamie dati Firmas partneriem, gan jāuzrunā mājaslapas apmeklētājus.

Standarta scenārijs, kad kāds no Firmas klientiem pasūta reklāmu (kas notiek ārpus Sistēmas) un tiek izmantota Sistēmas funkcionalitāte, ir sēkojošs:

1. Pārdošanas menedžeris veido mediju plānu, konkrētam klientam; mediju plāns sastāv no reklamas kampaņām, attiecīgi pārdošanas menedžeris izveido arī reklāmas kampaņu (vai vairākas, atkarībā no reklāmas risinājumiem, ko klients vēlas izmantot), atzīmējot nepieciešamo parādīšanu vai klikšķu skaitu, datumus, kad kampaņa darbosies, un partneru mājaslapas, kurās kampaņa darbosies;
2. Dizaineri veido attēlus vai citus reklāmkarogus (kas notiek ārpus Sistēmas), un pēc tam augšuplādē tos Sistēmas klientu failu katalogā, un saite uz attiecīgiem reklāmkarogiem tiek sūtīta klientam uz apstiprināšanu;
3. Kad reklāmkarogi tiek apstiprināti, atbalsta darbinieks augšupielādē tos pie dotājām reklāmas kampaņām, saliekot nepieciešamās specifikācijas (piemēram, saites, uz kurām jāiet pārlukprogrammai, kad mājaslapas apmeklētājs uzspiež uz reklāmkaroga) un pēc nepieciešamības izmanto Sistēmā pieejamos reklāmkarogu veidošanas šablonus (piemēram, skriptu, kas veido HTML failu ar dinamiski rotējošo kubu, skriptam pārsūtot attēlus priekš kuba malām);
4. Kad reklāmkarogi ir augšuplādēti un kampaņa ir gatava, pārdošanas menedžeris vai atbalsta darbinieks maina tās statusu uz aktīvs - no tā brīža kampaņas reklāmkarogi parādīsies izvēlētajās partneru mājaslapās kopā ar citiem reklāmkarogiem;
5. No savas puses mājaslapas īpašnieks (partneris), ienākot savā profilā Sistēmā, var nokopēt Javascript kodu priekš savas mājaslapas, un HTML kodu priekš konkrēta izmēra reklāmkarogiem, ja partneris savā mājas lapā vēlas izvietot Raspberry vai Billboard risinājumu, tad viņš ieliek tos mājaslapā (Javascript “global” kodu kopā ar citiem skriptiem, “izmēra” kodu — tur, kur attiecīgajam reklāmkarogam ir jāparādās);
6. Kad apmeklētājs ienāk attiecīgajā mājaslapā, aktivizējas Javascript kods, kas nosūta Sistēmas serverim izvēlētos risinājumus un no lapas nolasītos “izmēru” kodus; Serveris veic vairākas pārbaudes (piemēram, vai saņemtā lapas ID un adrese atbilst datubāzes ierakstam; vai konkrēta kampaņa vēl nav sasniegusi parādīšanu/klikšķu limitu uz doto brīdi; vai kampaņai ir vēlamā izmēra reklāmkarogs utt.) un, vai atbilstošā kampaņa ar atbilstošiem reklāmkarogiem ir atrasta, reklāmkaroga dati tiek sūtīti atpakaļ, līdz ar ko reklāmkarogs ielādējas lapā un tiek parādīts apmeklētājam, bet Sistēmas statistikā tiek pievienots ieraksts par parādīšanu no dotās kampaņas un dotās mājaslapas, no kā partnerim tiek pieskaitīta peļņa;

7. Ja apmeklētājs uzspiež uz reklāmkaroga, viņam atvēras jauna cilne ar attiecīgo saiti, savukārt Sistēmas statistikā tiek pievienots ieraksts par klikšķi no dotā reklāmkaroga un dotās mājaslapas, no kā partnerim tiek pieskaitīta peļņa;
8. Jebkurā brīdī atbalsta darbinieks vai pārdošanas menedžeris var apskatīties kampaņas vai media plāna statistiku un lejuplādēt datus Excel tabulas vai PDF formāta veidā;
9. Mājaslapas īpašnieks, ienākot savā profilā Sistēmā, var apskatīties savu peļņu un pieteikt atmaksu;
10. Pēc media plāna beigām atskaite par tā izpildi tiek sūtīta klientam.

Visas augstāk minētās lietotāju grupas dalās apakšgrupās, tāpēc zemāk ir detalizēti aprakstītas katras lietotāju grupas apakšgrupas, to vajadzības un vēlmes attiecīgi pret izstrādāto Sistēmu.

1.1. Lietoāju grupas

Ērtības nolūkos lietotāju pamatgrupas tālāk tiks sauktas attiecīgi par “Darbiniekiem”, “Partneriem” un “Apmeklētājiem”.

Sistēmas izstrādes gaitā katras grupas un apakšgrupas viedokļi tika ņemti vērā, apkopoti un analizēti. Veiktās analīzes rezultāti ir pieejami zemāk redzamajās tabulās.

1.1.1. tabulā ir apskatamās grupas “Darbinieki” apakšgrupas.

1.1.2. tabulā ir apskatamās grupas “Partneri” apakšgrupas.

1.1.3. tabulā ir apskatamās grupas “Apmeklētāji” apakšgrupas.

1.1.1 tabula

Grupas "Darbinieki" apakšgrupas

Apakšgrupas nosaukums	Vēlmes un vajadzības no Sistēmas	Pieejamās Sistēmas sadaļas
Admin (administrators)	Pārvaldīt sistēmas darbību, sekot līdzi veiktajām izmaiņām, komunicēt ar citiem darbiniekiem, tāpēc viņam ir nepieciešama pilna funkcionalitāte	Administratoram ir pieejamas visas sistēmas sadaļas, ieskaitot partneru sadaļu
Support (atbalsta darbinieks)	Apskatīties un pieņemt partneru pieteikumus, sūtīt izziņas partneriem un izvietot jaunumus, kā arī darboties ar reklāmas kampaņām — veidot, rediģēt un publicēt, pievienot kampaņu reklāmkarogus, veidot atskaites, darboties ar lietotājiem un mājaslapām	Dashboard (Instrumentu panēlis), Users (Lietotāji), Websites (Mājaslapas) un Media Plans (Media plāni). Media plānu sadaļā ir pieejama pilna funkcionalitāte — gan darbības ar plāniem un kampaņām, gan atskaišu lejuplādēšana
Sale (pārdošanas menedžeris)	Sekot līdzi reklāmas kampaņu izpildei, apskatīties un lejuplādēt atskaites, vērot kampaņu un reklāmas risinājumu statistiku, bet neko nevar rediģēt, dzēst vai pievienot — tikai lasīt.	Attiecīgi Media plānu, Atskaišu un Statistikas sadaļas, bet Media plānu sadaļā viņiem nav pieejamā pilnā funkcionalitāte — tikai kampaņu izpildes gaita, statistika un atskaites.
Designer (dizainers)	Reklāmas reklāmkarogu izvietošana speciālos katalogos. Izvietotie reklāmas risinājumi ar aprakstiem pēc tam var tikt sūtīti klientiem, partneriem, vai publiski izvietoti Sistēmas sākumlapā kā Firmas darbu piemēri.	Vienīgā pieejamā sadaļa — Banner Preview (Reklāmkarogu Priekšskatījums). Sadaļā ir pieejama trīs katalogu redaktēšana (relāmkarogu un aprakstu pievienošana un dzešana): klientu katalogs, partneru katalogs un publisks katalogs.
Tester (testētājs)	Apskatīties un testēt visu pieejamo sistēmas funkcionalitāti.	Visas sadaļas ir pieejamas, bet tikai testam. Piemēram, testētājs nevar publicēt reklāmas kampaņu (tikai izveidot melnrakstu).

1.1.2 tabula

Grupas "Partneri" apakšgrupas

Apakšgrupas nosaukums un apraksts	Vēlmes un vajadzības no Sistēmas	Pieejamās Sistēmas sadaļas
Partner (partneris): vienas vai vairāku mājaslapu īpašnieks, kura mājaslapās tiek izvietoti Firmas piedāvātie Javascript bloki, kas ļauj attiecīgās mājaslapās parādīt Sistēmā pieejamo reklāmu.	Saņemt Javascript kodus, kas ir nepieciešami reklāmas funkcionēšanai, apskatīt savu peļņu un izrakstīt rēķinu, kā arī sazināties ar Firmas darbiniekiem neskaidrību gadījumā.	Vienīgā sadaļa — privātais kabinets, kur ir pieejamas dažas profila konfigurācijas, saziņa ar Firmas darbiniekiem, rēķinu izrakstīšana, bilances statistikas pārskats un kodu bloku iegūšana priekš katras pie šī partnera reģistrētajām mājaslapām.
Client (klients): reklāmas pasūtītājs, priekš kura tiek izstrādāti risinājumi, kas pēc tam tiek izvietoti partneru mājaslapās un Firmas katalogos	Apskatīties savu pasūtīto reklāmas kampaņu izpildi un jaunos reklāmkarogus katalogā.	Klientu reklāmkarogu katalogs ir pieejams tikai apskatīšanai, un ir redzamas tikai ar šo klientu saistītās direktorijas. Publiskais katalogs ir pieejams apskatam pilnībā. Media plānu sadaļā var apskatīties piereģistrētās reklāmas kampaņas un to statistikas detaļas (kas arī var tikt lejuplādētas atskaites formā).

1.1.3 tabula

Grupas "Apmeklētāji" apakšgrupas

Apakšgrupas nosaukums	Vēlmes un vajadzības no Sistēmas	Pieejamās Sistēmas sadaļas
Partneru mājaslapu apmeklētāji	Neinteresējas par to, kā tieši mājaslapās tiek parādīta reklāma, bet ir vēlams, lai tā ielādētos ātri, nepiegružotu mājaslapu ielādi, netraucētu mājaslapas apskatīšanai, izskatītos interesanti un	Nekas no sistēmas iekšējām sadaļām nav pieejams, bet notiek mijiedarbība ar skriptiem, ko partneri izvieto savās mājaslapās. Kad skripts ielādē reklāmu, tā tiek parādīta apmeklētājam,

	uzrunoši, nekopētu personīgos datus, un lai pie reklāmkaroga piesaistītās hipersaites darbotos ātri un korekti.	un ja viņš uzspiež uz tās, tas tiek automātiski pievienots Sistēmas statistikas vākšanas rīka sarakstam un novīrzīts uz attiecīgo reklāmas saiti.
Potenciālie partneri	Vēlas piedāvāt sadarbību, no savas puses dodot vietu savās mājaslapās priekš reklāmas izvietošanas, un saņemot par to peļņu pēc atrunātiem noteikumiem.	Ir pieejams risinājumu katalogs un Sistēmas sākumlapa, no kuras var iesniegt sadarbības pieprasījumu.
Potenciālie klienti	Vēlas apskatīt Firmas piedāvātos reklāmas risinājumus — reklāmkarogu piemērus.	Ir pieejams risinājumu katalogs un publisks katalogs, kuros var redzēt jau esošus reklāmkarogus un variantus, kā tie var tikt izvietoti mājaslapās.

1.2. Firmas darbinieku sadaļa

Apkopojot lietotāju grupu vēlmes, sistēmā tika nolemts izveidot vairākas sadaļas, kur katrai sadaļai ir strikti noteikta funkcionalitāte un pieejamība dažādu lomu lietotājiem.

Tabulā 1.2.1 ir apskatamās visas lietotāju grupai "Darbinieki" pieejamās Sistēmas sadaļas un to funkcionalitāte.

1.2.1. tabula

Lietotāju grupai "Darbinieki" pieejamās Sistēmas sadaļas

Sadaļas nosaukums	Lietotāju lomas, kam ir pieejamā sadaļa	Sadaļas funkcionalitāte
Dashboard (Instrumentu panēlis)	Admin, Tester, Support	● potenciālo partneru pieteikumu apskatīšana un partneru kontu veidošana no pieteikumos piedāvātās informācijas; ● lietotāju iesūtīto kļūdu atskaišu apskatīšana; ● jaunumu rakstu veidošana un izvietošana (tiek parādīta partnera kabinetā); ● e-pastu sūtīšana partneriem (ļauj automātiski atsūtīt e-pastu visiem no saraksta izvēlētiem partneriem).

Media Plans (Media plāni)	Admin, Support, Sale, Client, Tester	● media plānu un reklāmas kampaņu meklēšana pēc nosaukuma, datuma, risinājuma, klienta (nav pieejams klientiem), atbildīgā pārdošanas speciālista; ● media plānu izveidošana (nav pieejama klientiem un pārdošanas speciālistiem) un rediģēšana; ● kampaņu izveidošana un rediģēšana; ● reklāmkarogu pievienošana un rediģēšana; ● media plāna atskaites lejuplāde; ● kampaņas statistika un attiecīgās atskaites (ne visi lauki un iestatījumi ir pieejami klientiem).
Banner Preview (Reklāmkarogu priekšskats)	Admin, Support, Designer, Sale, Tester	● klientu kataloga apskatīšana un redaktēšana (reklāmkarogu pievienošana un dzēšana ir pieejama tikai dizaineriem, testeriem un administratoriem); ● reklāmas risinājuma kataloga apskatīšana; ● publiskā kataloga apskatīšana un redaktēšana (reklāmkarogu pievienošana un dzēšana ir pieejama tikai dizaineriem, testeriem un administratoriem).
Users (Lietotāji)	Admin, Support, Tester	● visu sistēmas lietotāju apskatīšana, informācijas redaktēšana, dzēšana, jaunu lietotāju pievienošana; ● pāriešana uz partnera kabinetu (pieslēgšanās no partnera puses).
Settings (Iestatījumi)	Admin	● partneru sadaļas pieejamības pieslēgšana un atslēgšana; ● risinājumu peļņas ieskaites iestatījumi; ● kampaņu procentuālās izpildīšanas sadalīšana (cik % ir jāiegūst darba dienās, cik brīvdienās); ● kritiski zemas kampaņas izdošanas konstantes rediģēšana; ● Uznirstošo reklāmas risinājumu parādīšanās noilguma laika redaktēšana.
Statistics (Statistika)	Admin, Sale	● Kopējās risinājumu statistikas apskatīšana. Ir pieejami filtri pēc datuma un datu tipa (impresijas, kliki, parādīšanās utt.); ● Mājaslapu kopējās statistikas apskatīšana (izsaukšanu skaits, parādīšanās utt.).
Reports (Atskaites)	Admin, Sale, Tester, Support	● Adwords atskaišu veidošana un lejuplāde, izmantojot vairākus filtrus (klienta izvēli, datumus utt.); ● ir arī iespēja izveidot automātisko atskaišu

		veidošanu un sūtīšanu pēc noteiktā grafika.
Earnings (Peļņa)	Admin	● Kopējās peļņas apskatīšana pa risinājumiem; ● Kopējās peļņas apskatīšana pa partneriem.
Payments (Maksājumi)	Admin	● Rēķinu apskatīšana, meklēšana; ● Jaunu rēķinu un autorlīgumu pievienošana (izveidotie rēķini tiek automātiski reģistrēti trešās puses sistēmā).
Websites (Mājaslapas)	Admin, Support, Tester	● Mājaslapu meklēšana un apskatīšana; ● Tagu pievienošana un dzēšana; ● Mājaslapu pieslēgšana un atslēgšana no risinājumiem.
Errors (Kļūdas)	Admin, Tester	● Iespēja apskatīties pēdējas rindas no kļūdu žurnāla. Žurnālā tiek reģistrētas servera un datubāzes kļūdas, tīmekļa kļūdas, komandu izpildes kļūdas.

1.3. Firmas partneru sadaļa

Tabulā 1.3.1 ir apskatāmas visas lietotāju grupai "Partneri" pieejamās Sistēmas sadaļas un to funkcionalitāte.

1.3.1. tabula

Lietotāju grupai "Partneri" pieejamās Sistēmas sadaļas

Sadaļas nosaukums	Lietotāju lomas, kam ir pieejamā sadaļa	Sadaļas funkcionalitāte
Media Plans (Media plāni)	Client	● savu media plānu un reklāmas kampaņu meklēšana pēc nosaukuma, datuma, risinājuma, atbildīgā pārdošanas speciālista; ● media plāna atskaišu lejuplāde; ● kampaņas statistika un attiecīgās atskaites.
Richmedia	Partner	● mājaslapu statistika un peļņa siastībā ar risinājumu Richmedia.
Mobilfy	Partner	● mājaslapu statistika un peļņa siastībā ar risinājumu Mobilfy.
Billboard	Partner	● mājaslapu statistika un peļņa siastībā ar risinājumu Billboard.
Raspberry	Partner	● mājaslapu statistika un peļņa siastībā ar

		risinājumu Raspberry.
Mājaslapas	Partner	● partnera mājaslapu saraksts, kur var iegūt reklāmkarogu atspoguļošanai nepieciešamos Javascript un HTML blokus.
Bilance	Partner	● kopējo risinājumu peļņas pārskat;s ● rēķinu saraksts (datumi, statusi, dokumenti); ● iespēja pieteikt atmaksu (rēķins tiek automātiski aizsūtīts un reģistrēts trešās puses sistēmā);
Jaunumi	Partner	● jaunumu paziņojumu apskatīšana.

1.4. Nereģistrēto lietotāju mijiedarbība ar Sistēmu

Firmas darbiniekus un partnerus vieno tas, ka viņi ir reģistrēti Sistēmā. Savukārt nereģistrētos lietotājus, kas mijiedarbojas ar Sistēmu, var sadalīt divās grupās, kuras tiks aprakstītas kopā vienā apakšnodaļā, jo nevienai no tām nav paredzētas nekādas pieejamās Sistēmas sadaļas.

Tabulā 1.4.1 ir aprakstītas abām grupām pieejamās darbības.

1.4.1 tabula

Nereģistrēto lietotāju apraksts un mijiedarbība ar Sistēmu

Lietotāju grupa	Grupai pieejamās darbības ar Sistēmu
Sistēmas mājaslapas apmeklētāji	● iesniegt pieteikumu, lai kļūtu par Firmas partneri; ● nosūtīt atskaiti par kļūdu (apraksts + bilde); ● apskatīties publisko reklāmkarogu katalogu.
Partneru mājaslapu apmeklētāji	● apskatīt reklāmu, kas ielādējas no Sistēmas partneru mājaslapās; ● sekot reklāmkaroga saitei, kas piereģistrē apmeklētāju Sistēmas statistikā un pēc tam aizved uz attiecīgo reklāmas mājaslapu.

Var redzēt, ka nereģistrētie lietotāji pārsvarā vienkārši izmanto Sistēmas piedāvāto funkcionalitāti, nevis reāli mijiedarbojas ar to, jo viņi vai nu nolasa datus, vai nu sūta tos, un šie divi procesi neapvienojas ķēdē, kā tas notiek reģistrētiem lietotājiem.

2. SATVARA IZVĒLE

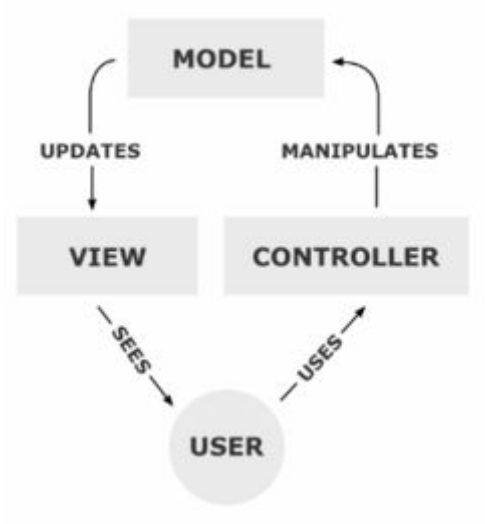
Sistēmas izstrādē tika izmantots Laravel 5.2, kas tajā brīdi bija jaunākā versija, tāpēc arī šī satvara iespējas darba gaitā tiks apskatītas balstoties uz šīs versijas dokumentāciju. [1] Jaunākās versijās tiek piedāvātas vēl plašākas iespējas un bagātāka funkcionalitāte, bet šī darba ietvaros pietiks arī ar to, ko piedāvāja versija 5.2, jo galvenās satvara īpašības ir palikušas tādas pašas, mainījušās sīkākas funkcijas un integrācijas nianses.

Satvara izvēles jomā priekšroka tika dota tieši Laravel satvaram, jo darba autorei jau ir bijusi liela pieredze gan ar darbu ar to, gan ar Laravel funkcionalitātes un dokumentācijas pētīšanu, jo iepriekš kursa darba ietvaros autore ir veikusi Laravel 4.2. un 5.2. versiju salīdzinājumu. Tāpēc šajā darbā tiks apskatīti arī citi satvari, un salīdzināti ar Laravel. Tiks apskatītas Laravel priekšrocības, lai pamatotu satvara izvēli.

2.1. MVC struktūras īpašības

MVC (Model-View-Controller, Modelis-Skats-Kontrolleris) ir tīmekļa programmas dalīšanas shēma, kur loģika, dati un lietotāja saskarne ir sadalīti trīs galvenajās komponentēs: modeli, skatu un kontrolleri, pateicoties kam katras komponentes izmaiņas var notikt neatkarīgi no citām [2]

MVC struktūrā atbildības sadalīšana notiek sekojoši: lietotājs redz skatu un mijiedarbojas ar to, skats padod lietotāja paveiktās darbības kontrollerim, kontrolleris liek mainīties attiecīgajam modelim, un atjaunotais modelis padod atbildi skatam, ko var redzēt attēlā 2.1.1 [2]. Tāda veida saistītā mijiedarbība ļauj maksimāli precīzi, uzticami un programmētājam ērti sadalīt programmas datus: MODELIS atjauno SKATU, SKATS parādās LIETOTĀJAM, LIETOTĀJS izmanto KONTROLLERI, KONTROLLERIS manipulē MODELI.



2.1.1 att. **MVC arhitektūras vizualizācija [2]**

2.2. Piedāvātās Laravel satvara iespējas

Laravel satvars piedāvā ne tikai visas MVC īpašības, bet arī daudz papildus funkcijas. Piemēram, ļoti daudz procesi, kas ir jāpilda sākot jaunu projektu, ar šo satvaru tiek automatizēti. Vietnes izveidošana, datubāzes konfigurēšana un citi procesi var tikt automātiski izpildīti ar vienas komandrindu saskarnes Artisan komandas palaišanu. Pirms sākt darbu ar Laravel un vietnes izstrādi, ir jāsakonfigurē nepieciešamās sistēmas uz datora. Ir nepieciešams:

- Lietotņu pakotņu pārvaldnieks Composer, jo lielākā daļa Laravel komandu tiek realizēta ar tā palīdzību
- PHP versijā no 5.5.9 līdz 7.1. ar dažiem paplašinājumiem:
    - OpenSSL
    - PDO
    - Mbstring
    - Tokenizer
- Direktoriju storage un bootstrap/cache publiskās rakstīšanas iespēja [1]

Tālāk satvara priekšrocības, kas bija noderīgas sistēmas izstrādes laikā, tiks apskatītas detalizēti.

2.2.1. Automatizēta vietnes pamata izveidošana

Kad ar Composer palīdzību tiek instalēts Laravel, jauna tīmekļa vietne var tikt automātiski izveidota ar komandu "laravel new", piemēram

```
laravel new blog
```

izveidojot www direktorijā mapi blog, kur jau būs sagatavoti galvenie vietnes darbam nepieciešamie modeļi, kontrolleri un skati.

Savukārt, komanda

```
php artisan make:auth
```

Pievienos izveidotajai vietnei klāt vēl dažus MVC elementus, kas ļaus reģistrēt jaunus lietotājus. Ja uz to brīdi jau ir izveidota datubāze un ir sakonfigurēta pieslēgšanās tai ar .env faila (kas tiks detalizēti apskatīts tālāk šajā sadaļā) vai database.php konfigurācijas faila palīdzību, tad arī attiecīgā datubāzē tiks izveidota tabula priekš lietotājiem un parolēm, kā arī tabula priekš paroles maiņas aizmiršanas gadījumā.

Rezultātā uzreiz ir sakārtotas visas mapes, app/Http/Controllers priekš kontrolleriem, app/Models priekš moduļiem un attiecīgi resources/views priekš skatiem, un ir sakonfigurētas visas mijiedarbības starp šīm mapēm. Sakonfigurējot virtuālo mitināšanu, uzreiz var redzēt vienti, kas sastāv no sākumlapas un reģistrācijas un seansa uzsākšanas blokiem, no kuriem arī ir iespējams pieteikties paroles atiestatīšanai. Protams, ka paroles atiestatīšanas funkcijas darbībai ir nepieciešams, lai būtu korekti sakonfigurēta epastu sūtīšana, jo jaunās paroles uzlikšana tiek nodrošināta, uz pieteikto epastu sūtot automātiski radīto unnikālo saiti.

2.2.2. .env fails

Uzreiz pēc jaunās Laravel vietnes izveidošanas projekta saknes direktorijā ir atradāms fails .env.example. Šajā gadījumā "env" ir saīsinājums no "environment" ("vide"), un tajā ir pieejams vides konfigurāciju šablons, kura daļa ir redzamā attēlā 2.2.2.1 [3]. Var redzēt, ka .env fails tiek izmantots vienotai konfigurāciju pārvaldei, ierakstot tajā datubāzes, epasta, vietnes adreses un citus iestatījumus.

```
40 lines (32 sloc)  651 Bytes
```

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=
APP_DEBUG=true
APP_URL=http://localhost

LOG_CHANNEL=stack

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=homestead
DB_USERNAME=homestead
DB_PASSWORD=secret

BROADCAST_DRIVER=log
CACHE_DRIVER=file
SESSION_DRIVER=file
SESSION_LIFETIME=120
QUEUE_DRIVER=sync

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_DRIVER=smtp
MAIL_HOST=smtp.mailtrap.io
MAIL_PORT=2525
MAIL_USERNAME=null
MAIL_PASSWORD=null
```

2.2.2.1. att. **.env.example daļa [3]**

Kad no faila nosaukuma ir noņemts ".example", un iekšā ir sarakstītas visas nepieciešamās konfigurācijas, tās kļūst pieejamās no jebkuras vietas projektā ar funkciju env() un getenv() palīdzību, pateicoties DotEnv PHP bibliotēkai, kas apvieno visus konfigurācijas failus kopā. Tādā veidā var nolasīt iestatījumus no .env faila, config direktorijā esošiem failiem, $_ENV un $_SERVER [4].

2.2.3. Skatu iespējas: Blade šabloni

Katra daļa no MVC struktūras ir pilnīgi atdalīta no citiem un darbojas neatkarīgi no citiem, kā jau bija minēts iepriekš. Tomēr visas šīs daļas var tikt brīvi savienotas savā starpā. Piemēram, skats var uzreiz izsaukt kontrollera vai modeļa darbību, kā tas ir redzams attēlā 2.2.3.1, kur klasiskais GET piieprasījums tiek īstenots ar jQuery palīdzību, bet saites vietā tiek izmantots uzreiz kontrollera metodes izsaukums. Kods ir ņemts no izstrādātās sistēmas.

```
    $.get('{{action("Admin\AdminMediaPlanController@getMediaPlan")}}',{id:em.current_edit_media_plan.id}).success(function(response) {
//$.getJSON('{{action("Admin\AdminMediaPlanController@getJSON")}}', {media_plan: id}).success(function(data) {
    $('#editMediaPlanModal').html(response)
});
```

2.2.3.1. att. **GET pieprasījuma īstenošana ar kontrollera palīdzību**

Kā arī:

- viena kontrollera metode var tikt izsaukta no cita kontrollera
- modeļa metode var tikt izsaukta no kontrollera
- modeļa metode var tikt izsaukta no cita modeļa

Bet modeļu un kontrolleru gadījumā šis process izskatās nedaudz citādāk, jo modeļi un kontrolleri ir klases, un to mijiedarbība tiek īstenota kā klašu mijiedarbība. Tas tiks sīkāk apskatīts nedaudz tālāk šajā sadaļā, bet pagaidām gribētos pieverst vairāk uzmanības tieši skatu funkcionalitātei.

Attēlā 2.1 ir redzamās dubultas figūriekavas, kas ir ieliktas apkārt kontrollera izsaukumam. Tas ir Laravel Blade šablons [5], kas ļauj brīvi savienot PHP un HTML/CSS/JS, skata faila nosaukumu rakstotveidā nosaukums.blade.php. Ar šo šablonu palīdzību modeļi, kontrolleri un PHP funkcijas var darboties uzreiz skatā jebkurā brīdī bez nepieciešamības ierakstīt skata kodā <?php> gabalus — tas viss tiek ģenerēts automātiski, izsaucot skatu.

Piemēram, attēlā 2.2.3.2 ir redzams skata koda apgabals, kur visi eksistējošie pieteikumi (kas ir objektu masīvs $applications) tiek atspoguļoti viens pēc otrā kā tabulas rindiņas, katrā rinfiņā parādot tekošā objekta laukus (message un created_at), kas tiek paveikts ar Blade @foreach funkcijas palīdzību un dubulto figūriekavu palīdzību, kas ļauj skatam uzreiz saprast, kur tiks ģenerēts PHP kods. Savukārt, ja nav pieejams neviens pieteikums, kas tiek pārbaudīts ar @if funkcijas palīdzību, tiek parādīts attiecīgs paziņojums.

```
<table class="striped highlight">
    <tr><th style="width: 60%"><a>APPLICATIONS</a></th><th style="width: 20%"">RECEIVED A
    @foreach($applications as $app)
        <tr><td><a>{{$app->message}} </a></td><td>{{$app->created_at}}</td><td><button cl
    @endforeach
    @if(count($applications)==0)
        <tr><td colspan="3">No applications to show yet</td></tr>
    @endif
</table>
```

2.2.3.2. att. **Blade koda piemērs no izstrādātās Sistēmas**

Savukārt, attēlā 2.2.3.3 ir redzams, kā izskatīsies rezultātā ģenerētais kods, izsaucot šo skatu. Uz attēla veidošanas brīdi bija pieejami cetri pieteikumi, kas tika atspoguļoti pēc kārtas, un attiecīgi paziņojums par to, ka pieteikumi nav pieejami, netika parādīts. Attēlā ir redzamā tabula ar četrām automātiski ģenerētām rindiņām, katra no kurām sastāv no pieteikuma teksta (kas netika parādīts attēlā, jo satur personīgos datus) un pieteikums saņemšanas datuma (kas ir redzams tabulas cilnēs).

```
▼<table class="striped highlight">
  ▼<tbody>
    ▼<tr>
      ▼<th style="width: 60%">
          <a>APPLICATIONS</a>
        </th>
        <th style="width: 20%" ">RECEIVED AT</th>
      ▼<th>
          <button class="btn-flat btn-flat waves-effect waves-light red btn white-text c
          ALL</button>
        </th>
      </tr>
    ▼<tr>
      ▶<td>…</td>
        <td>2017-11-14 14:13:37</td>
      ▶<td>…</td>
      </tr>
    ▼<tr>
      ▶<td>…</td>
        <td>2017-11-17 08:37:44</td>
      ▶<td>…</td>
      </tr>
    ▼<tr>
      ▶<td>…</td>
        <td>2018-04-27 16:09:35</td>
      ▶<td>…</td>
      </tr>
    ▶<tr>…</tr>
```

2.2.3.3. att. **Ar Blade palīdzību ģenerētais kods no izstrādātās Sistēmas**

2.2.4. Kontrolleru un Modeļu mijiedarbība

Vecākās Laravel versijās (piemēram, 4.2) visu kontrolleru un modeļu izsaukšana bija uzreiz pieejama jebkurā brīdī, bet jaunajās versijās (piemēram, 5.2), lai izmantotu no viena

kontrollera vai modeļa citu kontrolleru vai modeli, tie ir jādeklarē faila sākumā, kā ir redzams attēlā 2.2.4.1, kur ir deklarēti vairāki modeļi, divi kontrolleri un daži Laravel servisi.

Jebkuras klases ir deklarējamas pēc viena un tā paša principa. Tādā veidā deklarē ne tikai modeļus un citus kontrollerus, bet arī palīgus, bibliotēkas utt. Ir tikai pareizi jānosaka nosaukumvieta.

```php
<?php

namespace App\Http\Controllers\Admin;

use DB;
use StdClass;
use Auth;
use App\Models\User;
use App\Models\CoreClient;
use App\Models\Website;
use App\Models\WebsiteTag;
use App\Models\AdwordCampaign;
use App\Models\AdwordCampaignPeriod;
use App\Models\AdwordCampaignExists;
use App\Models\RichmediaCampaign;
use App\Models\RaspberryCampaign;
use App\Models\MobilfyCampaign;
use App\Models\BillboardCampaign;
use App\Models\InvolverCampaign;
use App\Models\OtherCampaign;
use App\Models\OtherPrCampaign;
use App\Models\PrCampaign;
use App\Models\MediaPlan;
use App\Http\Requests;
use Illuminate\Http\Request;
use Illuminate\Http\JsonResponse;
use App\Http\Controllers\Controller;
use App\Cacher;
use Cache;
use Exception;
```

2.2.4.1. att. **Nepieciešamo modeļu un kontrolleru deklarācija CampaignController sākumā no izstrādātās Sistēmas**

2.2.5. Konsoles komandas

Veidojot tīmekļa vietni ar Laravel, ir arī iespējams izmantot iebūvēto komandrindu saskarni Artisan. Tas piedāvā gan vairākas iebūvētas komandas, tādas, kā datubāzu tabulu migrācijas, datubāzu datu dēstīšanu un citas, kā arī ir iespēja rakstīt un pievienot klāt savas komandas, kas var būt palaižamās no konsoles gan manuāli, gan automatizēti ar Crontab palīdzību veidā “php artisan <komandas nosaukums>” [6].

Katra komanda ir PHP klase nosaukumvietā App\Console\Commands, kura ir reģistrēta Kernel.php failā. Attiecīgi, arī konsoles komanda var tikt izsaukta no modeļa vai kontrollera, kā jau tika minēts iepriekš.

Attēlā 2.2.5.1 ir redzamā konsoles komanda "inspire", kas tiek ģenerēta automātiski, veidojot jaunu Laravel vietni. Tajā ir redzams, pēc kāda principa ir veidota komanda — kā tiek deklarēts nosaukums, apraksts un pašas komandas kods (kas šajā gadījumā izvada konsolē vienu no PHP piedāvātām iedvesmojošām frāzēm).

```php
<?php

namespace App\Console\Commands;

use Illuminate\Console\Command;
use Illuminate\Foundation\Inspiring;

class Inspire extends Command
{
    /**
     * The name and signature of the console command.
     *
     * @var string
     */
    protected $signature = 'inspire';

    /**
     * The console command description.
     *
     * @var string
     */
    protected $description = 'Display an inspiring quote';

    /**
     * Execute the console command.
     *
     * @return mixed
     */
    public function handle()
    {
        $this->comment(PHP_EOL.Inspiring::quote().PHP_EOL);
    }
}
```

2.2.5.1. att. **Artisan konsoles komanda "inspire"**

2.2.6. Datubāze

Kā arī citi MVC satvari, plašāks salīdzinājums ar kuriem ir pieejams tālāk šajā darbā, Laravel piedāvā ciešu sadarbību ar vairāku veidu SQL datubāzēm.

Savienojums ar datubāzi (vai bāzēm, jo savienojumi var būt arī vairāki) tiek konfigurēts, kā arī gandrīz visi citi vietnes korektam darbam nepieciešamie iestatījumi, .env failā un database.php konfigurāciju failā. Attēlā 2.2.6.1 ir redamās noklusētas konfigurācijas, kas ir atrodamās database.php failā pēc automātiskās vietnes izveidošanas, programmētājam ir tikai jāieraksta iekšā visi dati.

```
'sqlsrv' => [
    'driver' => 'sqlsrv',
    'host' => env('DB_HOST', 'localhost'),
    'database' => env('DB_DATABASE', 'forge'),
    'username' => env('DB_USERNAME', 'forge'),
    'password' => env('DB_PASSWORD', ''),
    'charset' => 'utf8',
    'prefix' => '',
],
```

2.2.6.1. att. **Noklusētas database.php konfigurācijas**

Pēc konfigurēšanas savienojums ar datubāzi ie pieejams jebkurā koda vietā ar DB fasādes palīdzību (attiecīgi, klasēs, piemēram modeļos vai konsoles komandēs, tai jābūt sākotnēji deklarētai), un tas ļauj darboties gan ar "tīriem" SQL pieprasījumiem, gan noformēt tos programmētājam ērtākā un saprotamākā veidā, izmantojot Eloquent ORM [7].

Attēlā 2.2.6.2 ir redzams vienkāršs datubāzes pieprasījums ar Eloquent ORM izmantošanu, kas iegūst visus ierakstus no attiecīgas tabulas un atgriež to kā objektu masīvu, izmantojot noklusēto savienojumu.

```
public function getPossibleExpressions()
{
    return DB::table('targeting_expressions')->get();
}
```

2.2.6.2. att. **Eloquent pieprasījuma izmantošanas piemērs**

Savienojums ar datubāzi ir izmantojams ne tikai ar DB fasādes palīdzību, to noklusēti izmanto arī visi modeļi. Katrs modelis pēc noklusējuma ir savienots ar noklusētās datubāzes tabulu, kuras nosaukums ir modeļa klases nosaukums "čūskas reģistrā" (Snake case) angļu valodas daudzskaitlī. Attiecīgi, modelis RegisetredUser noklusēti nolasīs datus no tabulas "registered_users". Šis parametrs ir maināms, kā arī pastāv citi parametri, piemēram, kolonnu redzamība, nolasot datus, vai "mīkstās dzēšanas" opcija, kad dzēšot ierakstu tas nepazūd no tabulas, tam tikai tiek piešķirts "dzēšanas" laiks, līdz ar ko tas vairs neparādas, nolasot ierakstus.

Atsēvišķi ir jāpievērš uzmanība tam, kā vairāki modeļi var tikt savienoti savā starpā ar datubāzes palīdzību. Attēlā 2.2.6.3 ir redzams izstrādātās sistēmas koda fragments, kas

atspoguļo vienu no modeļiem. Ir redzamā gan augstāk minētā tabulas maiņa, gan saikne ar citu modeli, gan redzamo lauku konfigurācija.

```php
<?php
namespace App\Models;

use App\Models\RaspberryCampaign;

class ActiveRaspberryCampaign extends RaspberryCampaign {

    protected $visible = ['id', 'updated_at'];
    protected $table = 'raspberry_campaigns';

    public function websites()
    {
        return $this->belongsToMany('App\Models\ActiveWebsite',
            strtolower($this->campaignType).'_campaign_website',
            strtolower($this->campaignType).'_campaign_id', 'website_id');
    }
}
```

2.2.6.3. att. **Modeļa piemērs ar vairākām datubāzes konfigurācijām**

2.2.7. FTP serveris

Veidojot tīmekļa vietni ar Laravel, ir iespējams izmantot PHP Filesystem pakotni, kas nodrošina failu glabāšanu un mijiedarbību ar tiem lokālajā vai tālvadības diskā. Failu sistēma attiecīgi ir konfigurējama filesystem.php failā, kur ir pieejami noklusētie iestatījumi, kurus var mainīt uz jebkuriem nepieciešamiem.

Failu sistēma ļauj ātri augšuplādēt, lejuplādēt, nolasīt un rediģēt failus un direktorijas izvēlētajā glabāšanas sistēmā (kas šīs konkrētās Sistēmas gadījumā bija attālināts FTP serveris), izmantojot Laravel iebūvēto Storage fasādi.[8]

Mijiedarbība ar tīmekļa reklāmu ir cieši saistīta ar visuāliem failiem (bildēm, HTML failiem utt.), tāpēc ātra un ērtai failu apmaiņai izstrādes laikā tika piešķirta liela nozīme.

3. SISTĒMAS STRUKTŪRA

Sākot izstrādi, uzreiz tika nolemts izmantot MVC struktūru. Ņemot vērā Sistēmas īpašības (kas jau tika aprakstītas Sistēmas uzdevumos — darbības ar vairāku tipu failiem, plašas darbības ar datubāzi, epastu sūtīšana utt.), šajā situācijā tai ir vairākas priekšrocības. Galvenie iemesli, kāpēc tika izvēlēta tieši šī struktūra, ir sekojoši:

- koda apgabali var būt atkārtoti-izmantojami, kas nav iespējams, piemēram, RAD arhitektūras gadījumā [9]
- atbilde (response) var būt ne tikai HTML, bet arī JSON, fails utt. [9]
- lietotāja darbība var uzreiz izsaukt darbības ar modeļiem, kontrolleriem un citiem skatiem, tie nav speciāli jāapvieno šim nolūkam [9]
- skati var ģenerēties dinamiski, savienojot HTML un no kontrollera vai modeļa padotus datus, kas ir ideāls variants lapām, kas pielagojas katram lietotājam individuāli
- sadalot kodu trīs galvenos loģiskos blokos, kas var brīvi mijiedarboties savā starpā, tajā pašā laikā nesajaucoties kopā, krietni samazinās koda sarežģītība [10]

Izstrādātas sistēmas gadījumā MVC struktūras nianses tika apkopotas un sadalītas dažās daļās. Attiecīgi, skati skatu mapē un kontrolleri kontrolleru mapē tika sadalīti apakšmapēs balstoties uz Sistēmai nepieciešamās struktūras. Tika izdalītas četras galvenās sadaļas:

- Admin
- Partner
- Serve
- viss pārējais (Public utt.)

Tālāk tiks apskatīta katra sadaļa, t.i. attiecīgo kontrolleru, skatu un modeļu kopas, kas lasīšanas ērtībai tiks saukta par moduļiem.

3.1. Administratora modulis

Administratora modulī tiek apkopoti visi kontrolleri un skati, ko izmanto Firmas darbinieki un klienti. Attiecīgi, kontrolleri tiek glabāti app/Controllers/Http/Admin apakšdirektorijā ar attiecīgo nosaukumvietu. Savukārt skati tiek apkopoti

resources/views/admin apakšdirektorijā. Bet nekādi īpaši modeļi administratoram nav domāti, jo modeļi šajā sistēmā ir tieši tas, kas saista visus moduļus savā starpā.

Tālāk tiks apskatīti visi administratora moduļa kontrolleri un to funkcionalitāte, kā arī tiks sniegti pamatojumi sistēmas struktūrai.

AdwordController tika apvienotas visas ar Adwords API saistītās funkcijas, lai mijiedarbība ar API būtu centralizēta un ērta. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas iebūvēto Google Adwords sadaļu, t.i., sadaļas sākumlapas skatu un visos nepieciešamos daļējus skatus
- nolasīt Sistēmā pieejamos klientus un kampaņas, izmantojot Adwords API, un padot tos skatiem
- ģenerēt un dzēst atskaišu automātiskās ģenerēšanas grafiku, kas pēc tam tiks izmantots konsoles komandā, lai sūtītu atskaites bez nepieciešamības tās katru reizi pieprasīt manuāli, jo ir vairākas situācijas, kad atskaite tiek pieprasīta pēc viena un tā paša principa reizi kādā laika posmā (piemēram, atskaite par Google meklēšanām par pagājušo mēnesi)
- ģenerēt un lejuplādēt dažādu veidu Adword atskaites Excel tabulas formātā, jo Adwords API uzreiz piedāvā tādu iespēju, padodot nepieciešamo filtru vērtības

CampaignController apvieno visas funkcijas, saistītas ar reklāmas kampaņām, lai mijiedarbība ar kampaņu modeļiem būtu ātra un centralizēta, un lai loģika būtu sakārtota. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Media Plans (“mediju plāni”) sadaļā par kampaņām atbildīgos daļējus skatus. CampaignController un MediaPlanController nodrošina vienas sadaļas darbību, bet to funkcionalitāte ir pārāk plaša un loģiski atšķirīga, lai to visu likt iekš vienas klases
- pievienot, rediģēt un dzēst reklāmas kampaņas, darbojoties ar kampaņu modeļiem
- nolasīt un izmantot dažādu risinājumu iestatījumus ar kampaņu modeļu palīdzību, un padot tos skatam, lai tie attiecīgi pielāgotos
- pārbaudīt, vai visi kampaņas lauki ir aizpildīti korekti, kad notiek mēģinājums saglabāt kampaņu; tas ir nepieciešams ne tikai korektai datubāzes aizpildei, bet arī korektam kampaņas darbam nākotnē
- atjaunot kampaņas statistiku, izmantojot modeļus, lai netiktu pārsniegti izpildes limiti
- nolasīt kampaņas statistiku, jo visas darbības ar kampaņu datiem notiek centralizēti iekš šī kontrollera

- apstrādāt kampaņas statistiku, jo daži dati tiek glabāti lietotājam neērtos formātos (piemēram, datumi ir YYYY-MM-DD, kas nav ērti uztverams), formatēt datus, paaugstinot lasāmību, un padot skatam, lai tas ģenerētu tabulas un grafikus
- ģenerēt un lejuplādēt atskaites par kampaņu un tās statistiku PDF vai Excel formātos pēc lietotāja izvēles, izmantojot attiecīgas bibliotēkas
- nolasīt, apstrādāt un padot daļējam skatam kampaņai piederošo reklāmkarogu un mājaslapu statistiku: tas notiek šajā kontrollerī, nevis reklāmkarogu (Creative) un mājaslapu (Website) kontrolleros, jo šī statistika ir kampaņas apakšsadaļas daļa, bet attiecīgie kontrolleri ir vairāk domāti darbībai ar attiecīgām apakšsadaļām un skatiem
- atjaunot kampaņas statusu atbilstoši statistikai vai mediju plāna statusam: pēc saņemtā pieprasījuma vai pēc datu apstrādes kampaņa var tikt apstādināta vai pabeigta

CreativeController atbild par visām funkcijām, kas ir saistītas ar Media Plans sadaļas reklāmkarogu apakšsadaļu. Tas darbojas ar attiecīgām klasēm, lai nodrošinātu centralizētu un sakārtotu darbu ar failu ielādi, nolasīšanu, rediģēšanu. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Media Plans sadaļā par reklāmkarogiem atbildīgos daļējus skatus, padodot tiem datus no reklāmkarogu modeļiem un no FTP servera
- pievienot, rediģēt un dzēst kampaņas reklāmkarogus, izmantojot attiecīgos modeļus, kas definē nepieciešamos laukus
- mijiedarbojoties ar FTP serveri, augšuplādēt reklāmkarogu failus: FTP servera fasāde Storage tiek izmantojamā tikai šeit un iekš FileCatalogController, lai nenotiek nekādas liekās klašu deklarācijas
- veikt nepieciešamās failu modifikācijas: izgūt failus no ZIP arhīviem, pievienot nepieciešamās saites uz citiem Sistēmas failiem, ģenerēt HTML failus atbilstoši izvēlētam šablonam un risinājumam (piemēram, Richmedia risinājumam ir šablons "kubs", kas nolasa no skata četras padotas bildes un ielādē tās automātiski ģenerētajā HTML/JavaScript failā, kur šīs bildes tiek izmantotas kā trīsdimensionālā rotējošā kuba malas)
- formatēt failus atbilstoši izvēlētam reklāmas risinājumam: atkarībā no risinājuma tipa (modeļa), failiem tiek automātiski pierakstīta reaģēšana uz ekrāna orientācijas maiņu, ekrāna izmēru utt.
- formatēt direktorijas FTP serverī: dzēst tukšas, veidot jaunas; izmantojot Storage fasādi, var viegli nolasīt, vai direktorija eksistē, vai tajā ir failu utt., un atbilstoši šai

informācijai veidot jaunās direktorijas, dzēst tukšas, augšuplādēt failus

- pārbaudīt, vai reklāmkaroga lauki ir aizpildīti korekti, izmantojot reklāmkarogu modeļus; tas nodrošina korektu reklāmkaroga parādīšanos un statistikas uzskaitīšanu
- parādīt reklāmkarogu priekšskatījumu, gan iekš pašas Sistēmas, gan "live" režīmā partneru mājaslapā, izmantojot sīkdatnes un reklāmkarogu modeļus; šī funkcionalitāte ir cieši saistīta ar ServeController, kas tiks apskatīts tālāk šajā nodaļā

DashboardController atbild par "Dashboard" sadaļu, kur ir viens skats un vairāki modeļi. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Dashboard ("instrumentu panēlis") sadaļu, iepriekš nolasot datus no modeļiem un padodot to skatam
- pārbaudīt pieteikuma datu korektību: šī funkcija ir saistīta ar publiski pieejamo pieteikumu sūtīšanas formu, tomēr šeit tiek pārbaudīts, vai sistēmā vēl nav lietotājs ar šādu epastu, vai nav jau pievienotas šādas mājaslapas
- izveidot jaunu lietotāju no pieteikuma, ja visi iepriekš minētie dati ir korekti: šī funkcija mijiedarbojas ar UserController, sūtot formatētus pieteikuma datus tam
- dzēst pieteikumus, izmantojot datubāzes fasādi DB
- apskatīt un dzēst atskaites par kļūdām, izmantojot DB fasādi, jo tām ir informatīva nozīme, un vairs nekas ar tām nav darāms
- sūtīt jaunumus vai informatīvos epastus, izmantojot fasādi Mail un User modeli

EarningsController atbild par "Earnings" sadaļu, kurai ir informatīva nozīme, un nekādas darbības, izņemot datu nolasīšanu, tur nav domātas. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Earnings ("peļņa") sadaļu, padodot tai datus no datubāzes, izmantojot DB fasādi

ErrorController atbild par administratoram pieejamo "Erorrs" sadaļu, kurai ir informatīva nozīme, un nekādas darbības, izņemot datu nolasīšanu pieprasītajos apjomos, tur nav domātas. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Errors ("kļūdas") sadaļu, padodot tai datus, nolasītus no servera slēpta kļūdu žurnāla, izmantojot funkciju env(), lai atvērtu žurnāla failu
- izvadīt pieprasījumā definēto rindu skaitu no kļūdu žurnāla, jo sākotnēji tiek nolasītas pēdējas 20 rindiņas

ExcelController ir vienīgā funkcija, kas tiek izsaukta no Media Plan sadaļas, bet tika atdalīta atsēvišķā kontrollerī, jo tās darbībai ir nepieciešamas vairākas bibliotēkas, fasādes un

modeļi, kas vairs nekur šajā sadaļā netiek izmantoti

- ģenerēt un lejuplādēt Excel mediju plāna atskaiti — pēc atskaites tipa lejuplādējot uzreiz Excel tabulas veidā vai arī ZIP arhīvā ar paroli, nolasot datus no datubāzes ar DB fasādi, un mediju plāna datus no atitecīgā modeļa, kurā ir definētas arī visas saiknes ar kampaņu modeļiem (savukārt kampaņu modeļiem ir definētas saiknes ar reklāmkarogu modeļiem, tāpēc nolasot mediju plāna modeli tiek uzreiz nolasīta visa nepieciešamā informācija), un veidojot failu ar PHPExcel bibliotēkas palīdzību

FileCatalogController atbild par "File Catalog" sadaļu, un tajā ir centralizēta visa funkcionalitāte, kas ir nepieciešamā failu katalogu darbībai. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas File Catalog ("failu katalogs") sadaļu, t.i. sadaļas pamatskatu un nepieciešamos daļējos skatus pēc pieprasījuma
- nolasīt informāciju par pieejamiem klientiem un direktorijām FTP serverī ar Storage fasādes palīdzību
- pievienot, rediģēt un dzēst reklāmkarogu failus, izmantojot Storage un DB fasādes
- rediģēt failu katalogus — mainīt parādīšanas secību, pievienot aprakstus; informācija tiek uzreiz atjaunota datubāzē

MediaPlanController apvieno sevī visu ar mediju plāniem saistīto funkcionalitāti un atbild par Media Plan sadaļas pamatu. Tas cieši mijiedarbojas ar citiem kontrolleriem (CampaignController, ExcelController, CreativeController) un nodrošina sekojošas funkcijas:

- parādīt Sistēmas Media Plans sadaļu, t.i. sadaļas pamatskatu un daļējos skatus pēc pieprasījuma (tos, par kuriem neatbild citi iepriekš minētie kontrolleri), no mediju plāna modeļa uzreiz nolasot informāciju par mediju plāniem, to kampaņām, kampaņu aktivitātes periodiem un reklāmkarogiem
- pievienot, rediģēt un dzēst mediju plānus, izmantojot attiecīgo modeli
- meklēt jau esošus mediju plānus pēc ID, nosaukuma, reklāmas risinājumiem, datumiem pēc pieprasījuma, jo dāžreiz ir jāatrod vecās kampaņas
- pārbaudīt, vai visi jaunā mediju plana lauki ir ievadīti korekti, izmantojot modeļa iebūvēto validāciju un kļūdu gadījumos padodot ziņas ar JsonResponse fasādes palīdzību

PaymentController atbild par Payments sadaļu, vienīgo vietu administratora modulī, kur notiek maksājumu nolasīšana, rediģēšana un sūtīšana trešai pusei vai epastā. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Payments (“maksājumi”) sadaļu, t.i. pamatskatu un rediģēšanas daļējo skatu pēc pieprasījuma
- pievienot un rediģēt maksājumus, izmantojot maksājumu modeli
- augšuplādēt maksājumus trešās puses sistēmā, izmantojot curl() funkciju
- pievienot un rediģēt autorlīgumus ar failu pielikumiem, izmantojot DB fasādi un env() funkcionalitāti, lai saglabātu un dzēstu failus paslēptās servera direktorijās
- sūtīt autorlīgumus epastā, izmantojot Mail fasādi
- pārbaudīt, vai visi maksājuma vai autorlīguma lauki ir aizpildīti korekti, nolasot no datubāzes datus par izvēlētā lietotāja ienākumiem

ProfileController atbild par paroles maiņu aktīvajam lietotājam, un tika izdalīts atsēvišķi, jo tam jābūt uzreiz pieejamam no visām sistēmas sadaļām. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas daļējo skatu ar iespēju mainīt paroli
- pārbaudīt veco un jauno paroli un atjaunot, ja viss ir ievadīts korekti, izmantojot bcrypt() funkciju

RetargetingController atbild par Retargeting sadaļu, kur notiek darbības ar lietotāju interešu grupām, kas pēc tam var tikt pieslēgtas pie reklāmas kampaņām un reklāmkarogiem. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Retargeting Groups (“interešu grupas”) sadaļu, kur ir iespējams darboties ar mājaslapu apmēklētāju interešu grupām — savākt tās un vēlāk savākto lietotāju unikālos ID izmantot, lai radītu vai nerādītu konkrētu reklāmu (piemēram, parādīt sieviešu produktu reklāmu cilvēkiem, kas ir apmeklējuši sieviešu mājaslapas)
- pievienot vai dzēst interešu grupu, izmantojot attiecīgo modeli
- nolasīt, cik daudz lietotāji šobrīd ir reģistrēti izvēlētajā grupā, izmantojot Redis fasādi
- izveidot vai dzēst interešu grupu loģisko kombināciju, kas veidos vairāklīmeņu pieprasījumu datubāzē, lai definētu, kuri lietotāju pieder izvēlētai grupu kombinācijai

SettingsController atbild par dažiem sistēmas iestatījumiem — skaitliskām vērtībām, kuras nevajadzētu ierakstīt pašā kodā, bet jābūt iespējai jebkurā brīdī rediģēt, tāpēc tiem tikai izveidota speciālā sadaļa un attiecīga tabula datubāzē. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Settings (“iestatījumi”) sadaļu, nolasot no datubāzes esošos iestatījumus ar DB fasādi
- saglabāt no skata saņemtos iestatījumus atpakaļ datubāzē

StatisticsController atbild par “Statistics” sadaļu, kurai ir informatīvā nozīme, tātad tur notiek tikai datu nolasīšana no datubāzes. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Statistics (“statistika”) sadaļu, nolasot statistikas datus no datubāzes, izmantojot sākotnēji definētos laika apgabalus un datu tipus
- nolasīt un apstrādāt statistikas datus
- formatēt nolasītos datus priekš dažādu veidu grafikiem, tabulām un apkopojumiem, un padot formatētos datus skatam
- pēc pieprasījuma nolasīt datus par citu periodu, risinājumu, vai cita veida datus

TagController atbild par “Tags” sadaļu, kur notiek darbs ar mājaslapu tagiem, kas pēc tam var tikt pievienoti pie mājaslapas Websites sadaļā. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Tags (“tagi”) sadaļu, nolasot jau eksistējošos tagus ar attiecīgā modeļa palīdzību
- pievienot vai dzēst tagu
- nolasīt, kurām mājaslapām šobrīd ir piešķirts izvēlētais tags, nolasot datus ar DB fasādes palīdzību

UserController atbild par visu ar lietotājiem saistīto funkcionalitāti. Tas tiek izsaukts gan no Dashboard sadaļas, gan no Payments sadaļas, kā arī pilnīgi nodrošina Users sadaļas darbību. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Users (“lietotāji”) sadaļu, nolasot lietotāju vārdus un epastus no attiecīgā modeļa
- nolasīt plašāku informāciju par izvēlēto lietotāju, saņemot no skata padoto ID un padodot to modelim
- rediģēt vai dzēst esošo lietotāju ar modeļa palīdzību
- pievienot jauno lietotāju
- pārbaudīt, vai dati ir ievadīti korekti, izmantojot modelī iebūvēto validāciju
- uzģenerēt paroli jaunam lietotājam un atsūtīt to epastā ar bcrypt() funkcijas un Mail fasādes palīdzību
- virtuāli ienākt Sistēmā no partnera konta, izmantojot no skata padoto ID

WebsiteController ir centralizēta mājaslapu datu pārvalde. Mājaslapas tiek pievienotas pie partnera, kas ir definēts attiecīgajos modeļos. Šis kontrollers nodrošina sekojošas funkcijas:

- parādīt Sistēmas Websites (“mājaslapas”) sadaļu, nolasot no mājaslapu modeļa un

piesaistītā lietotāja modeļa datus par mājaslapām un atbildīgiem partneriem

- pievienot, rediģēt un dzēst mājaslapas ar mājaslapu un tagu saistīto modeļu palīdzību
- pieslēgt vai atslēgt mājaslapu no reklāmas risinājuma, uzreiz atjaunojot datus datubāzē
- mainīt mājaslapas iestatījumus (kādas pārbaudes veikt, ielādējot reklāmu utt.)

Savukārt skati administratora modulī ir apkopoti apakšdirektorijās atbilstoši Sistēmas sadaļām, jo tur ir ne tikai patstāvīgie skati, kas atspoguļo visu sadaļu, bet arī tā saucamie daļēji skati, kas ielādējas kādu īpašo funkciju izpildes ietvaros un papildina jau esošo skatu.

Tabulā 3.1.1 ir apskatāmi visi tie skati un to funkcionalitāte.

3.1.1. tabula

Administratora moduļa skati

Sadaļas nosaukums	Skatu funkcionalitāte
Adwords	● izvadīt klientu, kampaņu, datumu, kampaņu tipu filtrus ● apkopot izvēlētos filtrus un padot tos kontrollerim, lai lejuplādētu atskaiti vai izveidotu atskaišu ģenerēšanas grafiku
Creative Preview (File Catalog)	● parādīt failu katalogus un tajos pieejamos failus, kā arī saites uz failu katalogiem un to apakšdirektorijām ● piedāvāt formu jauno failu pievienošanai vai esošo failu rediģēšanai, un to aprakstiem ● sūtīt pieprasījumu uz serveri, lai dzēstu failu vai direktoriju ● mainīt klienta reklāmkarogu secību
Dashboard	● parādīt partneru pieteikumus ● pieprasīt lietotāja izveidošanas formu ar pieteikuma datiem un atsūtīt tos serverim, lai izveidotu jauno lietotāju ● dzēst pieteikumu ● parādīt kampaņu izpildes procentuālo sadalīšanu ● uzrakstīt epastu, izvēlēties partnerus un atsūtīt viņiem ievadīto epastu ● pievienot vai dzēst jauno ziņu, kas parādīsies partneriem ● apskatīt un dzēst kļūdu paziņojumus
Earnings	● izvadīt datus par peļņu ● meklēt maksājumus pēc partnera, summas, datuma vai statusa datiem
Errors	● izvadīt pēdējas rindiņas no servera kļūdu žurnāla, ● izvadīt citu rindiņu skaitu pēc pieprasījuma
Media Plans	● izvadīt aktīvos mediju plānus atkarībā no datumu, statusu, risinājumu, klientu iestatījumiem filtros ● sūtīt pieprasījumu uz serveri kādas konkrētas kampaņas meklēšanai ● pieprasīt un parādīt formu jaunā mediju plāna un kampaņu izveidei, vai eksistējošā mediju plāna un tā kampaņu rediģēšanai ● pieprasīt un mediju plāna atskaites ģenerēšanu un lejuplādēšanu

	● pieprasīt un parādīt formu reklāmas kampaņas reklāmkarogu pievienošanai, rediģēšanai un dzēšanai ● pieprasīt un parādīt kampaņas statistiku ● lejuplādēt kampaņas statistiku PDF vai Excel faila formātā, izmantojot kampaņas, reklāmkarogu un mājaslapu lauku filtrus
Payments	● parādīt pieejamos maksājumus un informāciju par tiem ● pieprasīt un parādīt formu jaunā maksājuma izveidošanai vai eksistējošā maksājuma rediģēšanai ● pārbaudīt ievadītos datus un sūtīt to serverim, ja tie ir korekti
Targeting Groups	● parādīt pieejamās interešu grupas un grupu kombinācijas ● pieprasīt un parādīt izvēlētās interešu grupas individuālo kodu ● pieprasīt un parādīt, cik lietotāji šobrīd ir reģistrēti izvēlētajā grupā ● pieprasīt un parādīt formu jaunās interešu grupas vai grupu kombinācijas izveidei, un atsūtīt serverim ievadītos datus
Settings	● parādīt esošos iestatījumus ● mainīt un sūtīt serverim partnera profila pieejamības statusu (izslēgts vai ieslēgts), Raspberry risinājuma cenu skaitīšanas metodi, kampaņu izpildes procentuālo sadalīšanu, kritiski zemo radītāju vērtības, Richmedia un Mobilfy risinājumu parādīšanas biežuma laiku
Stats	● izvadīt statistiskos datus par tekošo mēnesi vai par citu laiku, vai cita tipa datus (parādīšanās, klikšķus utt.) pēc pieprasījuma
Tags	● parādīt pieejamos tagus un pie tiem pieslēgto mājaslapu skaitu ● pieprasīt un parādīt formu jaunā taga izveidei ● sūtīt pieprasījumu dzēst esošo tagu
Users	● parādīt esošo lietotāju sarakstu ● meklēt lietotāju pēc visiem pieejamiem laukiem ● pieprasīt un parādīt formu jaunā lietotāja pievienošanai vai esošā lietotāja rediģēšanai ● sūtīt ievadīto lietotāja informāciju serverim uz pārbaudi ● sūtīt pieprasījumu dzēst lietotāju ● administratoram — simulēt ienākšanu izvēlētā partnera kontā
Websites	● parādīt esošās mājaslapas ● filtrēt mājaslapas pēc pieslēgtiem risinājumiem ● meklēt mājaslapas pēc partnera epasta, adreses vai ID ● rediģēt mājaslapu tagus ● sūtīt pieprasījumu pieslēgt vai atslēgt reklāmas risinājumu izvēlētai mājaslapai ● sūtīt pieprasījumu pieslēgt vai atslēgt mājaslapas adreses pārbaudi, ielādējot reklāmu ● pāriet uz izvēlēto mājaslapu

3.2. Partnera modulis

Partnera modulī tiek apkopoti visi kontrolleri un skati, ko izmanto Firmas partneri. Attiecīgi, kontrolleri tiek glabāti app/Controllers/Http/Partner apakšdirektorijā ar attiecīgo nosaukumvietu. Katram risinājumam atbilstošās sadaļas funkcionalitāte ir centralizēta atsēvišķā kontrollerī, lai maksimāli paātrinātu datu ielādi.

Savukārt skati tiek apkopoti resources/views/partner apakšdirektorijā. Nekādi īpaši modeļi partnerim nav domāti.

Tālāk tiks apskatīti visi partnera moduļa kontrolleri un to funkcionalitāte.

BillboardController apkopo visu funkcionalitāti, kas ir saistīta ar Billboard risinājuma statistikas un peļņas parādīšanu partnerim, un nodrošina sekojošas funkcijas:

- parādīt Sistēmas Partnera konta sadaļu Billboard, nolasot un apkopojot attiecīgā risinājuma statistiku un saistītos ienākumus, izmantojot atbilstošo modeli
- pēc pieprasījuma nolasīt datus par citu laika periodu

PartnerController apkopo visu funkcionalitāti, kas ir saistīta ar partnera profilu, kopējo peļņu un piekļuvi kontam. Tas centralizē partnera globālo informāciju un nodrošina sekojošas funkcijas:

- parādīt Sistēmas Partnera konta sākumlapu, nolasot aktīvā lietotāja datus ar Auth fasādes palīdzību
- nolasīt un apstrādāt partnera ienākumu un maksājumu datus, un saņemto bilanci padot visiem partnera skatiem
- pēc pieprasījuma, kas ir pieejams no visiem skatiem, atsūtīt epastu Firmas tehniskā atbalsta darbiniekam, izmantojot Mail fasādi
- parādīt aizvietojošu skatu gadījumā, ja šobrīd Partnera sadaļa ir izslēgta, nolasot attiecīgo iestatījumu no datubāzes (“tehniskie darbi”)

MobilfyController apkopo visu funkcionalitāti, kas ir saistīta ar Mobilfy risinājuma statistikas un peļņas parādīšanu partnerim, un nodrošina sekojošas funkcijas:

- parādīt Sistēmas Partnera konta sadaļu Mobilfy, nolasot un apkopojot attiecīgā risinājuma statistiku un saistītos ienākumus, izmantojot atbilstošo modeli
- pēc pieprasījuma nolasīt datus par citu laika periodu
- NewsController nodrošina sekojošas funkcijas:
- parādīt Sistēmas Partnera konta sadaļu “Jaunumi”, nolasot datus ar DB fasādes

palīdzību

- reģistrēt datubāzē, kad partneris pēdējo reizi bija ienācis sadaļā

PaymentsController apkopo visu ar partnera rēķiniem saistīto funkcionalitāti. Tas centralizē pieejamo rēķinu datus, kā arī peļņas un izmaksu datus, un nodrošina sekojošas funkcijas:

- parādīt Sistēmas Partnera konta sadaļu "Rēķini", nolasot un apkopojot informāciju par partnera ienākumiem un pieejamiem rēķiniem un to statusiem
- ja partnera bilance to atļauj, pēc pieprasījuma ģenerēt atmaksas pieteikumu, atsūtīt to epastā un augšuplādēt trešās puses sistēmā
- ProfileController nodrošina sekojošas funkcijas:
- parādīt Sistēmas daļējo skatu ar iespēju mainīt paroli un reklāmas vizuālās izdalīšanas iespējas iestatījumu (šī funkcija ir sīkāk aprakstīta tālāk šajā nodaļā)
- pārbaudīt veco un jauno paroli un atjaunot, ja viss ir ievadīts korekti, izmantojot bcrypt() funkciju

RaspberryController apkopo visu funkcionalitāti, kas ir saistīta ar Billboard risinājuma statistikas un peļņas parādīšanu partnerim, un nodrošina sekojošas funkcijas:

- parādīt Sistēmas Partnera konta sadaļu Billboard, nolasot un apkopojot attiecīgā risinājuma statistiku un saistītos ienākumus, izmantojot atbilstošo modeli
- pēc pieprasījuma nolasīt datus par citu laika periodu

RichmediaController apkopo visu funkcionalitāti, kas ir saistīta ar Billboard risinājuma statistikas un peļņas parādīšanu partnerim, un nodrošina sekojošas funkcijas:

- parādīt Sistēmas Partnera konta sadaļu Billboard, nolasot un apkopojot attiecīgā risinājuma statistiku un saistītos ienākumus, izmantojot atbilstošo modeli
- pēc pieprasījuma nolasīt datus par citu laika periodu

WebsitesController apkopo visu informāciju par partnera mājaslapām, pieejamiem risinājumiem un mājaslapu individuāliem kodiem reklāmas parādīšanai, un nodrošina sekojošas funkcijas:

- parādīt Sistēmas Partnera konta sadaļu "Mājaslapas", nolasot un apkopojot mājaslapu datus ar attiecīgā modeļa palīdzību
- pēc pieprasījuma ģenerēt Javascript kodus priekš izvēlētās mājaslapas

Tāpat kā administratora moduļa gadījumā, partnera moduļa skati ir sadalīti apakšdirektorijās atbilstoši sadaļām, lai daļējo un pamatskatu mijiedarbība būtu vieglāka, sakārtotāka un programmētājam saprotamāka.

Tabulā 3.2.1 ir apskatāmi visi tie skati un to funkcionalitāte.

3.2.1. tabula

Partnera moduļa skati

Sadaļas nosaukums	Skatu funkcionalitāte
Billboard	● parādīt Billboard risinājuma statistiku par tekošo mēnesi, grupējot to pēc mājaslapām un reklāmas kampaņām ● pēc pieprasījuma parādīt statistiku par citu izvēlēto periodu ● izsaukt saiti ar risinājuma izskata demonstrāciju
Mobilfy	● parādīt Billboard risinājuma statistiku par tekošo mēnesi, grupējot to pēc mājaslapām un reklāmas kampaņām ● pēc pieprasījuma parādīt statistiku par citu izvēlēto periodu ● izsaukt saiti ar risinājuma izskata demonstrāciju
News / Jaunumi	● parādīt pieejamos jaunumus
Payments / Maksājumi	● parādīt tekošo bilanci un rēķinus ● pieprasīt un parādīt formu atmaksas pieteikumam ● aizpildīt un atsūtīt atmaksas pieteikumu serverim uz pārbaudi
Raspberry	● parādīt Billboard risinājuma statistiku par tekošo mēnesi, grupējot to pēc mājaslapām un reklāmas kampaņām ● pēc pieprasījuma parādīt statistiku par citu izvēlēto periodu ● izsaukt saiti ar risinājuma izskata demonstrāciju
Richmedia	● parādīt Billboard risinājuma statistiku par tekošo mēnesi, grupējot to pēc mājaslapām un reklāmas kampaņām ● pēc pieprasījuma parādīt statistiku par citu izvēlēto periodu ● izsaukt saiti ar risinājuma izskata demonstrāciju
Websites / Mājaslapas	● parādīt pieejamās mājaslapas un instrukciju, kā izvietot reklāmas iegūšanas kodus mājaslapā ● pieprasīt un parādīt mājaslapas kodus priekš reklāmas ielādēšanas ● parādīt izmēra HTML kodus priekš Raspberry un Billboard

3.3. Kalpošanas modulis

Kalpošanas (“serve”) modulī tiek apkopoti kontrolleri, publiskie faili un skati, kas ir nepieciešami sistēmas tehniskai darbībai (statistikas reģistrēšanai, atskaišu ģenerēšanai utt), bet nav saistīti ar datu atspoguļošanu, t.i. nekādā veidā nedarbojas vizuāli, tikai pilda aizmugursistēmas funkcijas.

Kalpošanas modulī ietilpst viens galvenais kontrolleris, kura funkcionalitāte ir aprakstīta un pamatota zemāk, un ir jāpiemin, ka šajā sarakstā partnera mājaslapas apmeklētājs ērtākai lasīšanai ir saukts par “lietotāju”:

1. reģistrēt Sistēmas statistikā (izmantojot Redis fasādes pipeline() funkciju, un vēlāk šie dati tiks ierakstīti datubāzē) jaunos klikšķus un parādīšanās, pirms tam veicot nepieciešamās pārbaudes:
    a. vai reklāma netika parādīta priekšskata režīmā
    b. vai datu tips ir pareizs
    c. vat mājaslapa ar tādu ID ir atrodamā datubāzē
    d. ja datu tips ir klikšķis — atsūtīt atpakaļ attiecīgo pāriešanas saiti
2. atsēvišķi pēc nepieciešamības reģistrēt unikālos klikšķus un parādīšanās,nolasot datus no Redis un pārbaudot, vai lietotājs ar šādu ID jau ir reģistrēts šīs kampaņas statistikā
3. reģistrēt Sistēmas statistikā konkrētā reklāmas risinājuma izsaukšanu un servera atbildi:
    a. vai tika atrasta reklāmas kampaņa, kuras reklāmkarogu varētu parādīt mājaslapas apmeklētājam
    b. ja nē, tad ierakstīt pieejamo kampaņu nederīguma iemeslu
4. reģistrēt Sistēmas statistikā lietotāja pārlukprogrammas nosaukumu
5. pēc pieprasījuma no partnera mājaslapas nolasīt un atsūtīt lietotāja pusei Richmedia vai Raspberry risinājumā pieejamos datus,
    a. nolasīt tos ar Cache fasādes palīdzību un uzreiz pārbaudīt, vai šis lietotājs atbilst pieejamo kampaņu interešu grupām, ja tādas ir izvēlētas
    b. visas pārējas pārbaudes šiem risinājumiem notiek lietotāja pusē
6. pēc pieprasījuma nolasīt un atsūtīt lietotāja pusei Mobilfy vai Billboard risinājumā

pieejamos datus, veicot nepieciešamās pārbaudes:

a. vai kampaņa ir pieslēgta pie šīs mājaslapas,
b. vai ir vēlamā izmēra reklāmkarogi,
c. vai netika vēl sasniegts parādījumu/klikšķu limits šim lietotājam un kampaņai,
d. vai lietotājs atbilst kampaņas interešu grupai

7. palaist pieprasīto risinājumu (vai konkrētu kampaņu vai reklāmkarogu) priekšskatīšanai partnera mājaslapā, izmantojot sīkdatnes; šajā gadījumā visas pārbaudes tiek ignorētas un ir pieņemts, ka izvēlētais reklāmkarogs atbilst visiem nosacījumiem
8. simulēt reklāmas risinājuma izsaukumu un izvadīt saņemtos datus

Par "publiskiem failiem" šī darba ietvaros tiek saukti JavaScript un CSS faili, kas atrodas public direktorijā. Sistēmas tehniskās darbības jomā šiem skriptu failiem ir liela nozīme, jo tieši tie nodrošina reklāmkarogu korekto parādīšanos lietotājam (partnera mājaslapas apmeklētājam). Tabulā 3.3.1 ir apskatamā šo failu funkcionalitāte.

3.3.1. tabula

Kalpošanas Javascript un CSS faili

Faila nosaukums	Faila tips	Faila funkcionalitāte
billboard	JavaScript	● nolasīt reklāmkaroga datus un padot tos galvenai bibliotēkai ● nolasīt lietotāja darbības ar reklāmkarogu un sūtīt tos bibliotēkai
billboard	CSS	● pārvaldīt reklāmkaroga fonu, parādīšanas laika skaitītāja pozīciju un aizvēršanas pogas parādīšanu
billboard_lib	JavaScript	● kontrolēt un pārvaldīt reklāmkarogu parādīšanos (izmēru, pozīciju atbilstoši ierīces orientācijai utt.) ● saņemt signālus no citiem failiem un sūtīt attiecīgos ziņojumus galvenai bibliotēkai
lib	JavaScript	● pārvaldīt Richmedia un Raspberry risinājumu parādīšanos (izmēru, pozīciju utt.) ● nolasīt ziņojumus no citiem failiem un rīkoties atbilstoši tiem: ○ ielādēt reklāmkarogu ○ aizvērt reklāmkarogu ○ nosūtīt serverim paziņojumu par parādīšanu ○ nosūtīt serverim paziņojumu par klikšķi ○ atvērt jauno cilni ar reklāmkarogam

		piesaistīto saiti
mobilfy	CSS	● pārvaldīt reklāmkaroga fonu, parādīšanas laika skaitītāja pozīciju un aizvēršanas pogas parādīšanu
mobilfy	JavaScript	● nolasīt reklāmkaroga datus un padot tos galvenai bibliotēkai ● nolasīt lietotāja darbības ar reklāmkarogu un sūtīt tos bibliotēkai
mobilfy_lib	JavaScript	● kontrolēt un pārvaldīt reklāmkarogu parādīšanos (izmēru, pozīciju atbilstoši ierīces orientācijai utt.) ● saņemt signālus no citiem failiem un sūtīt attiecīgos ziņojumus galvenai bibliotēkai
listeners	JavaScript	● nolasīt lietotāja darbības ar interaktīviem reklāmkarogiem un sūtīt tos galvenai bibliotēkai
eye_of_mordor	JavaScript	● nolasīt mājaslapā izsauktos Sistēmas reklāmas risinājumus ● parādīt rīkjoslu ar pieejamiem risinājumiem un pēc klikšķa izdalīt tos ar krāsu (pieejams tikai Sistēmā aktīvajam administratoram)

Kā arī pie šī moduļa attiecās viens skats, kas, kā jau bija minēts iepriekš, neko neatspoguļo vizuāli, bet tikai satur JavaScript un PHP kodu. Šis skats tiek padots kā servera atbilde, un tajā notiek visas nepieciešamās pārbaudes un izmaiņas. Skata funkcionalitāte ir sekojoša:

1. Nolasīt pieejamo kampaņu datus — pieejamos reklāmkarogus, cik reizes šīs kampaņas reklāmkarogi tika parādīti šim lietotājam, un cil vēl ir šobrīd jāsasniedz
2. Ja šis lietotājs pirms tam nav saskaries ar Sistēmas reklāmu, viņa pārlukprogrammā tiek izveidots attiecīgs ieraksts ar kampaņu datiem; ja šis lietotājs nav jauns Sistēmā, kampaņu dati tiek nolasīti no viņa pārlukprogrammas, un tiek modificēti atbilstoši pēdējām izmaiņām — beigušās kampaņas tiek dzēstas, jaunās tiek pieievienotas utt.
3. Pārbaudīt mājaslapu — vai tā ir pieslēgta izsauktām reklāmas risinājumam, un vai padotais ID atbilst reālai adresei
4. Pārbaudīt kampaņas —
    a. vai kampaņa ir pieslēgta pie šīs mājaslapas,
    b. vai ir vēlamā izmēra reklāmkarogi,
    c. vai netika vēl sasniegts parādījumu/klikšķu limits šim lietotājam un kampaņai,
    d. vai lietotājs atbilst kampaņas interešu grupai

5. Ja visas pārbaudes ir veiksmīgi nokārtotas, izvēlētās kampaņas un izvēlētā reklāmkaroga dati tiek atsūtīti Sistēmas JavaScript bibliotēkai, kas
    a. ielādē reklāmkarogu mājaslapā
    b. reģistrē parādīšanu Sistēmas statistikā
    c. palielina parādīšanu skaitu lietotāja pārlukprogrammā, lai nākamreiz salīdzinātu kampaņas limitus ar to
    d. ja lietotājs uzklikšķina uz reklāmkaroga, reģistrē attiecīgos datus par klikšķi

3.4. Publiskais modulis

Publiskajā modulī ietilpst viens kontrollers un daži skati, kas nodrošina publiski pieejamo funkcionalitāti. Tā, šī publiskā kontrollera funkcijas ir:

1. atspoguļot Sistēmas sākumlapu
2. apstrādāt un saglabāt potenciālo partneru pieteikumus
3. atspoguļot publiski pieejamo reklāmkarogu katalogu
4. ļaut vai neļaut partnerim ienāklt sistēmā

Savukārt tālāk ir apskatāmi publiski pieejamie skati un to funkcionalitāte:

- Banner Preview funkcija ir parādīt pieprasītā klienta reklāmkarogus (opcionāli — no konkrētās kampaņas, par izvēlēto laika posmu) no klientu failu kataloga
- Clients Examples funkcija ir parādīt visu katalogā pieejamo klientu sarakstu ar logotipiem, un pēc klikšķa jaunajā logā pieprasīt un parādīt konkrētā klienta reklāmkarogus
- Doge ir slēptais joks (tā saucamais “easter egg”), jo nekur sistēmā nav norādīta saite uz to, tas vienkārši eksistē, un tas parāda līdojošo suni uz kosmosa fona, kas rej pēc klikšķa
- Error Report ir pieejams no sākumlapas, un tas piedāvā formu priekš kļūdas atskaites sūtīšanas, kur var ievadīt vārdu, epastu, problēmas aprakstu un kategoriju, pievienot bildi un atsūtīt šo informāciju serverim
- Landing Page ir Sistēmas sākumlapa ar vispārīgo informāciju par Sistēmu, Firmas adresi, iespēju ienākt Sistēmā, saiti uz kļūdas paziņojuma formu un partnera pieteikuma formu, kuru aizpildot un iesniedzot var atsūtīt serverim, ja dati ir ievadīti korekti un visi nepieciešamie lauki ir aizpildīti

- Partner Disabled ir paziņojums par to, ka šobrīd partnera sadaļa nav pieejamā tehnisko darbu dēļ, ar saiti atpakaļ uz sākumlapu

3.5. Konsoles komandas

Sistēmā ir izstrādātas vairākas Artisan komandas, kas veic tehniskās darbības vairākkārt Firmas darbinieku un partneru pusē. Tika pieņemts lēmums šiem nolūkiem izmantot tieši konsoles komandas, jo tās ir viegli palaižamās no komandrindas, savukārt komandrindu komandu izpildi var konfigurēt jebkurā serverī un ielikt to sistemātisko izpildi pēc noteiktā grafika, piemēram, ar Cron palīdzību.

Tabulā 3.5.1 ir apskatāmas šīs komandas un to funkcionalitāte.

3.5.1. tabula

Artisan komandas

Komandas nosaukums	Izpildes biežums	Komandas funkcionalitāte
CheckMediaPlans	Reizi dienā	● pārbaudīt, kuri mediju plāni ir beigušies un, ja tiem ir definēts trešās puses ID, izveidot rēķinu par šo mediju plānu trešās puses sistēmā
GenerateReports	Reizi dienā	● pārbaudīt, vai uz izvēlēto datumu ir plānotas kādas atskaites ● ģenerēt un atsūtīt uz definētiem epastiem plānotās atskaites
Stager	Pēc pieprasījuma	● nolasīt no Sistēmas Bitbucket repozitorija vecās JavaScript bibliotēkas versijas ● pēc izvēles lejuplādēt vecāko versiju un aizvietot jaunāko ar to
Payments	Reizi dienā	● nolasīt neapmaksāto maksājumu statusus no trešās puses sistēmas ● atjaunot maksājumu statusus atbilstoši trešās puses sistēmas datiem
Adwords	Reizi dienā	● nolasīt no Firmas Google Adwords konta pieejamās reklāmas kampaņas, klientus un izmaiņas salīdzinājumā ar iepriekšējo dienu, un atjaunot datus datubāzē atbilstoši nolasītajām izmaiņām
Balance	Reizi dienā	● nolasīt partnera ienākumus dienas laikā un apkopot tos speciālajā tabulā datubāzē
Stats	Reizi 10 minūtēs	● nolasīt no Redis aktīvo kampaņu statistiku un atjaunot statistiku datubāzē atbilstoši nolasītiem datiem

Websites	Reizi dienā	• pārbaudīt, cik sen no katras Sistēmā pieejamās mājaslapas tika reģistrēta aktivitāte, un ja pēdējo divu nedēļu laikā no tās nav bijusi neviena parādīšanās, atslēgt attiecīgo reklāmas risinājumu no šīs mājaslapas

Ir labi redzams, ka konsoles komandu galvenā funkcija šajā sistēmā ir regulāro darbu automatizācija, jo citādāk šie darbi būtu jāveic grāmatvežiem un atbalsta darbiniekiem.

4. IZSTRĀDES PROCESS SALĪDZINĀJUMĀ AR CITIEM POPULĀRIEM SATVARIEM

Darba izstrādes gaitā tika veikts salīdzinājums ar citiem populāriem satvariem, lai noskaidrotu, vai tiešām Laravel ir viespiemērotākais satvars konkrēti šiem nolūkiem, jo eksistē arī citi PHP MVC satvari, kuriem arī ir noteiktā popularitāte. Jau izstrādātā koda uzlabošanas procesā rādījās interese, vai šis kods varētu būt ātrāks, ērtāk lasāms un minimalistiskāks cita satvara izmantošanas gadījumā. Lai analīze būtu godīga, tika nolemts paņemt vēl trīs satvarus, uz kuriem potenciāli varētu būt izstrādāta Sistēma. Tāpēc tika paņemti PHP valodai piemēroti MVC satvari — Symfony, CodeIgniter un Yii 2. Izvēle tika pamatota uz šo satvaru popularitātes saraksta [11], kur arī var redzēt, ka Laravel ir vispopulārākais PHP MVC satvars.

MVC struktūras izvēle jau tika plaši pamatota 2. un 3. nodaļā, bet tieši PHP valodas izvēlei ir sēkojošs pamatojums:

1. ir iespējamā ātra izstrāde
2. kodu var viegli izmantot atkārtoti
3. labi piemērots MVC struktūrai
4. aizsargā vietnes datus
5. vispopulārākā valoda aizmugursistēmas izstrādei [11]

Tika izdalīti galventie pieturpunkti, uz kuriem jāskatās, analizējot satvaru, t.i., tās nianses, kuras ir vissvarīgākas izstrādātai sistēmai, kas ir redzams no 3. nodaļas:

1. Datubāze: tabulu izveide, aizpildīšana un rediģēšana no satvara puses vai no komandrindas, datu nolasīšanas un apstrādes iespējas
2. Modeļi: to saistība ar datubāzi, konfigurēšanas ērtība, modeļu savstarpēja saistīšana
3. Kontrolleri: to mijiedarbība, koda atkārtotā izmantošana
4. Skati: PHP un HTML mijiedarbība, dinamiskās veidošanas iespējas
5. Failu krātuve un citas funkcijas: kādas ir pieejamās fasādes, kādas bibliotēkas ir iespējams pieslēgt un cik ērti
6. Konsoles komandas: to veidošana, mijiedarbība, darbināšana serverī

Tālāk atbilstoši šiem kritērijiem tiks apskatīti trīs satvari, kas sekoja Laravel satvaram popularitātes sarakstā.

4.1. Symfony

Symfony daudzās vietās izskatās līdzīgi Laravel, jo var teikt, ka Laravel tika izstrādāts uz Symfony pamata. Tomēr tam nav pieejamā tik plaša dokumentācija, kā Laravel, tā izskatās vairāk pēc apmācības. Tālāk tiks apskatītas un salīdzinātas izstrādātai sistēmai galvenās nianses.

4.1.1 Datubāze

Symfony arī izmanto Composer, tāpēc darbā ar datubāzēm arī pastāv iespēja izmantot migrācijas un veidot datubāzes tabulas attālināti no servera [12]. Tomēr, datu aizpildīšana un citu pieprasījumu izpildīšana notiek ne tik ērti, kā Laravel, kas ir redzams attēlā 4.1.1.1., kur Product tipa modelis tika uzģenerēts, pēc tam aizpildīts, un saglabāts datubāzē (divos posmos). Galā ir nolasīts izveidotā ieraksta ID.

```php
// you can fetch the EntityManager via $this->getDoctrine()
// or you can add an argument to your action: index(EntityManagerInte
$entityManager = $this->getDoctrine()->getManager();

$product = new Product();
$product->setName('Keyboard');
$product->setPrice(1999);
$product->setDescription('Ergonomic and stylish!');

// tell Doctrine you want to (eventually) save the Product (no querie
$entityManager->persist($product);

// actually executes the queries (i.e. the INSERT query)
$entityManager->flush();

return new Response('Saved new product with id '.$product->getId());
```

4.1.1.1. att. **Darbības ar datubāzi, izmantojot modeļus [12]**

Var redzēt, ka Symfony piedāvā visas nepieciešamās funkcijas darbam ar datubāzi, tomēr ātra un bieža mazo pieprasījumu izpildīšana, kas pastāvīgi notiek uzstrādātā sistēmā, nebūtu tik viegla.

4.1.2. Modeļi

Mijiedarba ar datubāzes klasēm jau tika nedaudz apskatīta iepriekš. Symfony nepiedāvā klasiskos modeļus, to vietā ir tā saucāmie Objekti (Entity), kas ir gandrīz tas pats. Tomēr,

dokumentācijā nav nekas atrodams par objektu savstarpējo savienošanu ar datubāzes atslēgu palīdzību no faila — tikai no komandrindas, kas krietni apgrūtinātu datubāzes izmaiņu pārnešanu no lokāla servera uz attālināto[12]. Kā arī, kā jau bija redzams attēlā 4.1.1.1, lauku aizpildīšanai objektā ir nepieciešamās set() funkcijas, kas atšķirās katram laukam (savukārt Laravel veido objektus, kuru lauki ir visi uzreiz sasniedzami pēc vienotā principa), kas arī apgrūtinātu darbu, jo dinamiskai objektu aizpildīšanai būtu attiecīgi dinamiski jāveido funkcijas.

4.1.3. Kontrolleri

Kontrolleru izveide un darbības ir ļoti līdzīgas Laravel. Tāpat tās ir klases, kuru sākumā ir jādefinē visas citas klases, ar kurām notiks sadarbība, katrai klasei ir sava nosaukumvieta. Symfiny kontrolleri atbalsta vairāku tipu servera atbildes, piemēram, Response, Redirect (vairāku tipu, kas redzams attēlā 4.1.3.1, kur ir parādīti dažādu redirect tipa piemēri — uz saiti, uz maršrutu, uz maršrutu ar padotiem datiem) vai skatu datus.

```php
// redirects to the "homepage" route
return $this->redirectToRoute('homepage');

// redirectToRoute is a shortcut for:
// return new RedirectResponse($this->generateUrl('homepage'));

// does a permanent - 301 redirect
return $this->redirectToRoute('homepage', array(), 301);

// redirect to a route with parameters
return $this->redirectToRoute('app_lucky_number', array('max' => 10));

// redirects externally
return $this->redirect('http://symfony.com/doc');
```

4.1.3.1. att. **Redirect tipa atbildes [13]**

Arī Symfony kontrolleri var saņemt datus no skatiem un apstrādāt tos ar Request objektu palīdzību [13], tāpat kā Laravel. Var teikt, ka kontrolleru piedāvātā funkcionalitāte tiem ir vienādi plaša.

4.1.4. Skati

Skati var tikt padoti no kontrollera ar uzreiz dinamiski saņemtiem individuāliem datiem, kas jau tika apskatīts iepriekšējā punktā. Symfony arī piedāvā šablonu valodu priekš skatu ērtākai sastādīšanai un PHP savienošanai ar HTML [14], kas ir ļoti līdzīga Laravel Blade,

tomēr ne tik minimalistiska un nedaudz citādāk noformēta, kas ir redzams attēlā 4.1.4.1., kur ir parādīta HTML saraksta ģenerēšana, katru punktu dinamiski ielādējot ar PHP palīdzību, un tukša saraksta gadījumā parādot paziņojumu. Tomēr no dokumetācijas ir redzams [14], ka visa funkcionalitāte, kas ir pieejama Blade šabloniem un Laravel skatiem kopumā, Symfony ir arī pieejama.

```
<ul>
    {% for user in users if user.active %}
        <li>{{ user.username }}</li>
    {% else %}
        <li>No users found</li>
    {% endfor %}
</ul>
```

4.1.4.1. att. **Skata ģenerēšana ar Twig šablonu valodu [14]**

4.1.5. Failu krātuve, funkcijas, bibliotēkas

Tur, kur Laravel piedāvā fasādes un bibliotēkas, Symfony piedāvā tā saucamos Komplektus (Bundles), kuriem ir tāda pati nozīme — pievienot projektam papildus funkcijas, kas ne vienmēr un ne katrā projektā ir vajadzīgas, tāpēc nav jēgas tas pievienot visur automātiski.

Ir pieejami vairāki komplekti dažādiem nolūkiem — labākai šablonu darbībai, jauno klašu veidošanai utt [15]. Darbībām ar FTP serveri ir pieejams viens no tādiem komplektiem, kas tomēr ļauj nolasīt datus, bet nekas nav aprakstīts par failu ģenerēšanu, augšuplādēšanu un citām darbībām ar failu servera funkcionalitāti [16]. Izskatās, ka pastāvīgs darbs ar failiem ar Symfony būtu krietni apgrūtināts.

4.1.6. Komandas

Symfony neizmanto Artisan, kas ļauj Laravel satvaram viegli pievienot un izmantot ne-iebūvētas komandas. Tomēr, tam ir pieejamās gan iebūvētas komandas, gan komandu veidošanas iespēja [17]. Komanda ir tāda pati klase, kā Laravel, kurai jābūt deklarētai un kurā jābūt deklarētām visām citām klasēm, ko tā izmanto, un kā ir redzams attēlā 4.1.6.1 (kur tiek parādīta konsoles komandas darbināšana un simbolu virknes parādīšana), izmantot var gan datubāzi, gan objektus, gan visu citu satvara funkcionalitāti.

```php
protected function execute(InputInterface $input, OutputInterface $output)
{
    // ...

    $this->userManager->create($input->getArgument('username'));

    $output->writeln('User successfully generated!');
}
```

4.1.6.1. att. **Objekta un datubāzes izmantošana no konsoles komandas [17]**

4.2. CodeIgniter

CodeIgniter sver tikai 2MB un strādā bez konfliktiem uz gandrīz visām PHP versijām, kā arī tas ir viegli instalējams uz jebkuriem serveriem un mitināšanas servisiem, jo tam nav nekas jāinstalē papildus, ir nepieciešams tikai pats PHP. Uzreiz pēc instalēšanas tas nepiedāvā tik plašu funkcionalitāti, kā Laravel un Symfony, bet var viegli pieslēgt papildus bibliotēkas un funkcionalitāti pēc nepieciešamības [10].

Tālāk tiks apskatītas un salīdzinātas izstrādātai sistēmai galvenās nianses.

4.2.1 Datubāze

CodeIgniter piedāvā darbības ar datubāzi vairākos variantos — gan izmantojot iebūvētās funkcijas (tādas, kā insert()), gan uzreiz izmantojot SQL pieprasījumus [18]. Dati no datubāzes var būt nolasīti gan kā masīvi, gan kā objekti, kas ir redzams attēlā 4.2.1.1., kur ir redzams klasisks SQL pieprasījums, un saņemtie datu pēc tam tiek izvadīti ciklā kā objektu masīvs.

```php
$query = $this->db->query('SELECT name, title, email FROM my_table');

foreach ($query->result() as $row)
{
        echo $row->title;
        echo $row->name;
        echo $row->email;
}

echo 'Total Results: ' . $query->num_rows();
```

4.2.1.1. att. **Datu nolasīšana no datubāzes [18]**

No dokumentācijas ir redzams, ka CodeIgniter piedāvā visu nepieciešamo datubāzes funkcionalitāti — ērtas darbības ar datiem, plašas konfigurācijas un vairākas iebūvētas funkcijas [18], kas padara darbu daudz ērtāku.

4.2.2. Modeļi

Modeļu izmantošana nav obligāta izmantojot CodeIgniter, tas var pilnīgi darboties tikai ar kontrolleru palīdzību, tāpēc arī dokumentacijā modeļiem nav pievērsts īpaši daudz uzmanības [19]. Dokumentācijā ir redzams, ka modeļu uzbūve ir līdzīga Laravel modeļa uzbūvei, ir ciešā saikne ar datubāzi un modeļu savstarpēja saistīšana, tomēr kopējā modeļu funkcionalitāte nav tik plaša, kā Laravel, kā arī modeļu izmantošana no citām vietām projektā nav tik ērta, kas ir redzams attēlā 4.2.2.1., kur ir parādīta modeļa ielādēšana no kontrollera.

Your models will typically be loaded and called from within your controller methods. To load a model you will use the following method:

```
$this->load->model('model_name');
```

4.2.2.1. att. **Modeļa ielādēšana [18]**

Var redzēt, ka CodeIgniter nav tik specificēti orientēts uz MVC modeli, tā ir neobligāta funkcionalitāte.

4.2.3. Kontrolleri

Kontrolleri ir vienīgā obligāta MVC sastāvdaļa, izmantojot CodeIgniter. Kā ir teikts dokumentācijā [20], tie ir orientēti uz HTTP pieprasījumu apstrādi. Tie saņem informāciju no lietotāja puses, apstrādā to, un padod atbildi atpakaļ, tomēr nav piedāvāta atbildes formātu izvēle, kā tas bija iepriekšejiem diviem satvariem. Attiecīgi, dažādu veidu atbilžu, statusu, saišu un failu saņemšana no servera būtu apgrūtināta, izmantojot šo satvaru.

Kā arī kontrolleru definēšana no projekta maršrutiem nav tik ērta, kā ir redzams attēlā 4.2.3.1., kur no maršrutu saraksta tiek definēts noklusētais kontrollers. Šajā gadījumā būtu strikti jāseko līdzi gan masīva elementu nosaukšanai, gan klašu nosaukumu maiņai, ja tāda notiks.

```
$route['default_controller'] = 'blog';
```

4.2.3.1. att. **Kontrollera maršrutizācija [20]**

Ir redzams, ka CodeIgniter nav domāts tik plašu uzdevumu risināšanai. Nepieciešamās funkcionalitātes realizācija ir iespējamā, bet būtu krietni apgrūtināta.

4.2.4. Skati

Skati arī nav obligāti, būvējot projektu ar CodeIgniter. Kā arī citiem apskatītiem satvariem, skats var būt ielādēts ar kontrollera izpildīto komandu, kā arī tam var tikt padoti dati, kas ļauj veidot dinamiskās skatu lapas, kā ir redzams attēlā 4.2.4.1., kur dati tiek padoti skatam kā asociēta masīva lauki.

```
$data = array(
        'title' => 'My Title',
        'heading' => 'My Heading',
        'message' => 'My Message'
);

$this->load->view('blogview', $data);
```

4.2.4.1. att. **Datu padošana skatam no kontrollera [21]**

Tomēr, CodeIgniter skati nepiedāvā nekādu šablonu valodu, kas nozīmē, ka visi PHP dati, kas tiek padoti skatam, būtu jāraksta iekš HTML lapas standartā veidā, izmantojot <?php> apgabalus, kas uzreiz padara reālu PHP un HTML dinamisku savienošanu, kas ir nepieciešamā izstrādātai sistēmai, ļoti neērtu un dažās vietās neiespējamu.

4.2.5. Failu krātuve, funkcijas, bibliotēkas

CodeIgniter nepiedāvā tik plašu iebūvēto funkciju izvēli, tomēr tam ir iespēja ielādēt jebkuru bibliotēku. Tas nav darāms ar deklarāciju kontrollera sākumā, kā tas bija iepriekš apskatītiem satvariem, bet izmantojot speciālo iebūvēto funkciju, kura ir redzamā attēlā 4.2.5.1 un kurā ir parādīta FTP bibliotēkas ielādēšana un konfigurēšana projekta [22].

```
$this->load->library('ftp');

$config['hostname'] = 'ftp.example.com';
$config['username'] = 'your-username';
$config['password'] = 'your-password';
$config['debug']        = TRUE;

$this->ftp->connect($config);

$this->ftp->upload('/local/path/to/myfile.html', '/public_html/myfile.html', 'ascii', 0775);

$this->ftp->close();
```

4.2.5.1. att. **Pieslēgšanās FTP serverim un datu augšuplādēšana [22]**

Arī no attēla 4.2.5.1. ir redzams, ka CodeIgniter FTP bibliotēka piedāvā ne tikai datu nolasīšaqnu no servera, bet arī augšuplādēšanu, kas ir realizēta līdzīgi Laravel failu ielādēšanai, un no attiecīgas dokumentācijas sadaļas [22] ir saprotams, ka ar darbībām ar failiem izmantojot šo satvaru līdzīgiem nolūkiem problēmas nerastos.

4.2.6. Komandas

Atšķirībā no citiem iepriekš apskatītiem satvariem, CodeIgniter nepiedāvā speciālo komandrindu funkcionalitāti. Tomēr kontrolleru funkciju palaišana ir iespējamā arī no komandrindas [23]. Tā kā kontrolleri ir cenrālā CodeIgniter funkcionalitātes daļa, visas satvara un bibliotēku funkcijas no tādas "komandas" būtu pieejamas, tomēr tas krietni bojātu projekta loģisko struktūru un kārtību, kas ir redzams attēlā 4.2.6.1., kur no komandrindas tiek izsaukta klases funkcionalitāte.

```
$ cd /path/to/project;
$ php index.php tools message
```

4.2.6.1. att. **Kontrollera funkcionalitātes darbināšana no konsoles [23]**

Ir jāpiemin, ka arī komandu noformēšana konsolē nebūtu tik ērta un saprotama, un varētu sabojaties kontrollera funkcionalitātes vai pat nosaukuma maiņas dēļ. Tomēr, precīzi sekojot izveidoto komandu sarakstam, nepieciešamās darbības varētu tikt izpildītas.

4.3. Yii 2

Yii ir ātrāks par citiem iepriekš apskatītiem satvariem. Tas nāk uzreiz komplektā ar jQuery un AJAX funkcionalitāti, kas ir labi ātrai nelielo datu gabalu apmaiņai. Tā piedāvātais kods ir labi atkārtoti izmantojams. Vispārīgi, šis satvars ir vairāk saistīts ar priekšgalsistēmas izstrādi un ir ērts tiem, kam ir lielāka pieredze tieši ar to [10].

Tālāk tiks apskatītas un salīdzinātas izstrādātai sistēmai galvenās nianses.

4.3.1 Datubāze

Yii piedāvā plašu ar datubāzēm saistītu funkcionalitāti, kas ir detalizēti aprakstīta attiecīgā dokumentācijas nodaļā [24]. Attēlā 4.3.1.1. ir redzams klasisks datubāzes

pieprasījums, kas gan filtrē, gan nolasa datus, kā arī uzreiz ir pieejamā grupēšanas, limitēšanas un saistīšanas funkcionalitāte.

```
$rows = (new \yii\db\Query())
    ->select(['id', 'email'])
    ->from('user')
    ->where(['last_name' => 'Smith'])
    ->limit(10)
    ->all();
```

4.3.1.1. att. **Klasisks datubāzes pieprasījums ar Yii funkcionalitāti [24]**

Yii piedāvātās darbības ar datubāzi ir līdzīgas tam, ko piedāvā Laravel iebūvētais Eloquent ORM, kas liek domāt, ka visa izstrādātai sistēmai nepieciešamā datubāzu funkcionalitāte arī būti ērti nodrošinājamā.

4.3.2. Modeļi

Atšķirībā no CodeIgniter un tapat kā Laravel un Symfony, Yii piedāvā plašu aprakstu darbībām ar modeļiem, jo tas ir pilnīgi orientēts uz MVC arhitektūru, kas arī ir pieminēts dokumentācijā [25].

Tā kā Yii ir vairāk orientēts uz priekšgalsistēmas struktūru, modeļu atribūti ir pieejami gan kā objektu lauki, gan kā masīva elementi, kas ir raksturīgs JavaScript valodai. Kopējā sintakse darbā ir līdzīga Laravel, kas ir redzams attēlā 4.3.2.1., kur modelis tiek aizpildīts ar datiem, kas tika nolasīti no skata formas.

```
$model = new \app\models\ContactForm;
$data = \Yii::$app->request->post('ContactForm', []);
$model->name = isset($data['name']) ? $data['name'] : null;
$model->email = isset($data['email']) ? $data['email'] : null;
$model->subject = isset($data['subject']) ? $data['subject'] : null;
$model->body = isset($data['body']) ? $data['body'] : null;
```

4.3.2.1. att. **Modeļu aizpildīšana ar datiem [25]**

Yii piedāvā ne tik plašu modeļu funkcionalitātes izvēli, kā Laravel. Modeļi nav tik viegli savienojami savā starpā, un pašu modeļu definēšana nav tik ērta un daudzfunkcionāla, tomēr galvenās funkcijas, tādas, kā datu nolasīšana un rakstīšana, ir stabīli un ātri pieejamas.

4.3.3. Kontrolleri

Kontrolleru funkcionalitāte un sintakse Yii izskatās gandrīz tāpat kā Laravel. Kontrolleri ir klases, kuru sākumā tiek definētas citas klases, ar kurām notiks mijiedarbība. Kā

arī dokumentācijā sniegtos piemēros, viens no kuriem ir redzams attēlā 4.3.3.1. [26] ir redzams, ka Yii kontrolleri var padot visu nepieciešamom veidu atbildes no servera puses — kļūdas, ciparus, statusus, skatus utt.

```php
public function actionView($id)
{
    $model = Post::findOne($id);
    if ($model === null) {
        throw new NotFoundHttpException;
    }

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

4.3.3.1. att. **Atbildes no kontrollera [26]**

Arī ir redzams, ka no kontrollera ir uzreiz pieejamā sadarbība ar modeļiem, un saņemto modeļu datu padošana skatam, tapat kā Laravel.

4.3.4. Skati

Yii nepiedāvā speciālos šablonus, lai savienotu PHP un HTML, un attiecīgi skatos tie ir jāapvieno manuāli, izmantojot klasisko PHP sintaksi [27]. Skatiem var tikt padoti dati no kontrollera, skatos var tikt izsauktās jebkuras funkcijas vai modeļi, tomēr izstrādes ziņā skatu taisīšana ir apgrūtināta, jo nekāds risinājums PHP un HTML funkcionalitātes savienošanai nav pieejams.

Attēlā 4.3.4.1. var redzēt, kā izskatās Yii satvarā izstrādātais skats kods, kas veido HTML formu.

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

<p>Please fill out the following fields to login:</p>

<?php $form = ActiveForm::begin(); ?>
    <?= $form->field($model, 'username') ?>
    <?= $form->field($model, 'password')->passwordInput() ?>
    <?= Html::submitButton('Login') ?>
<?php ActiveForm::end(); ?>
```

4.3.4.1. att. **Skats ar savienotu HTML un PHP [27]**

Tomēr ir jāatzīmē, ka Yii skatos ir pieejamā gan datu padošana vairākiem skatiem uzreiz, gan koda gabalu šabloni (layouts), kas var tikt automātiski pievieonti, gan iebūvēto darbību pievienošana, no kā var secināt, ka Sistēmai nepieciešamā funkcionalitāte būtu tomēr sasniedzama.

4.3.5. Failu krātuve, funkcijas, bibliotēkas

Yii dokumentācijā nav pieejamā sadaļa par darbu ar failiem, kā arī FTP ir pieminēts tikai viena vietā, kur ir aprakstīts pieslēgšanas iespējamība [28].

Kontrolleri var saņemt datus no skatiem, tajā skaitā — failus, kas tiek augšuplādēti no skatos pieejamām formām, tomēr īpašas darbības ar failiem nekur netiek aprakstītas, kas liek domāt, ka failu augšuplādēšana un nolasīšana būtu jānodrošina ar pašā PHP pieejamām funkcijām un palīgbibliotēkām (Helpers), jo tomēr vismaz ir pieejams FileHelper palīgs, bet tas tik un tā krietni apgrūtinātu sistēmas izstrādi.

4.3.6. Komandas

Yii piedāvā gan vairākas iebūvētas konsoles komandas, gan iespēju veidot papildus komandas pēc nepieciešamības [29].

Kā ir redzams attēlā 4.3.6.1., kur komanda izvada dažādu tipu datus, konsoles komanda ir tāda pati klase, kā visas citas, kurai attiecīgi ir pieejamā visa satvara un projekta funkcionalitāte.

```php
namespace app\commands;

use yii\console\Controller;

class HelloController extends Controller
{
    public $message;

    public function options($actionID)
    {
        return ['message'];
    }

    public function optionAliases()
    {
        return ['m' => 'message'];
    }

    public function actionIndex()
    {
        echo $this->message . "\n";
    }
}
```

4.3.6.1. att. **Konsoles komandas piemērs [29]**

Yii konsoles komandas var strādāt ar jebkura tipa datiem un saņemt vērtības no komandrindas ievada, attiecīgi visa sistēmai nepieciešamā funkcionalitāte konsoles komandu jomā būtu nodrošināta.

4.4. Salīdzinājuma rezultāti

Tabulā 4.4.1 ir apskatāmi visi izvirzītie kritēriji visiem apskatītiem satvariem, un katrs ir novērtēts no 1 līdz 3, kur

3 — pilnīgi piemērots attiecīgās funkcionalitātes nodrošināšanai

2 — daļēji piemērots, izstrādes procesā radītos noteiktās problēmas

1 — nav piemērots šī tipa funkcionalitātes nodrošināšanai, izstrādes procesā radītos vairākas problēmas

4.4.1. tabula

Satvaru salīdzinājuma rezultāti

	Laravel	Symfony	CodeIgniter	Yii 2
Datubāze	3	2	3	3
Modeļi	3	1	1	2
Kontrolleri	3	3	1	3
Skati	3	3	1	2
Funkcijas	3	1	3	1
Komandas	3	3	2	3
KOPĀ	18	13	11	14

No paveiktā salīdzinājuma ir redzams, ka Laravel izvēle šīs sistēmas gadījumā ir pilnīgi pareiza, jo tikai tas piedāvā tik plašu dokumentāciju un koda piemērus (ne tikai oficiālā lapā, bet arī citur Internetā). Arī var pieminēt, ka pateicoties savai popularitātei, vienmēr bija iespējams uzreiz saņemt atbildi uz jautājumiem, uzdodot tos attiecīgos forumos.

No visiem apskatītiem satvariem Laravel piedāvā visplašāko un vispiemērotāko funkcionalitāti priekš izstrādātās sistēmas tipa.

Rezultāti

Pirms izstrādes un darba rakstīšanas uzsākšanas tika veikta potenciālo lietotāju aptauja, ar kuras palīdzību tika noformulētas visu lietotāju grupu vēlmes un vajadzības, kuras arī ir apkopotas šī darba 1. nodaļā. Visu Sistēmai nepieciešamo funkcionalitāti var sadalīt sekojošās galvenās apakšgrupās:

- nepārtrauktā mijiedarbība ar datubāzi — datu nolasīšana, atjaunošana, rakstīšana un dzēšana
- dažādu vietnes sadaļu pieejamība no dažādām vietām — gan no citām sadaļām, gan arī no citām mājaslapām
- plaša mijiedarbība ar failiem, tajā skaitā failu ģenerēšana, kas attiecās gan un HTML failiem, gan uz ZIP arhīviem, kā arī Excel un PDF failiem
- statistikas datu vākšana nepārtraukti un reālajā laikā

Kad visi sie dati tika apkopoti, uzreiz palika saprotams, ka daudzas sistēmas sastāvdaļas šķērsosies savā starpā, tāpēc, lai izvairītos no koda haotiskās kopēšanas, kods bija jāveido tā, lai to varētu izmantot atkārtoti un no dažādām vietām.

No sistēmas moduļu apskatīšanas 3. nodaļā ir ļoti labi redzams, ka šāda veida sistēmai ir vislabāk piemērota MVC struktūra, jo tieši tā ļauj precīzi sadalīt projekta arhitektūru un loģiku. Kā jau bija aprakstīts 2. nodaļā, tieši MVC struktūra nodrošina, lai katrs projekta modulis darbotos tikai ar tam nepieciešamiem datiem, neielādētu liekās klases un uzreiz mijiedarbotos ar nepieciešamām tabulām un saiknēm datubāzē. 3. nodaļas sākumā ir aprakstītas MVC struktūras priekšrocības, un izstrādes procesa apraksta gaitā var redzēt, ka tie visi tika izmantoti izstrādātā sistēmā:

- koda apgabali ir atkārtoti-izmantojami: ar citu struktūru būtu ļoti daudz kopētie koda gabali
- ir sastopamā vairāku tipu servera atbilde: HTML, JSON, faili, skati utt.
- lietotāja darbība var uzreiz izsaukt darbības ar modeļiem, kontrolleriem un citiem skatiem; gandrīz nekur skatos netika izmantotas tiešās saites HTML formātā, tās bija ģenerētas ar action() funkcijas palīdzību
- skati ģenerējas dinamiski, saņemot tieši šim lietotājam pieejamos datus, kas tiek uzreiz organizēti modelī vai kontrollerī

- viss kods ir precīzi sadalīts trīs galvenos blokos, līdz ar ko gan samazinās koda sarežģītība, gan paaugstinās koda lasamība un izmantojamība talākajā sistēmas attīstībā

Kā arī no 3. nodaļas ir labi redzams, ka izstrādātā sistēma ļoti plaši izmanto tieši Laravel piedāvāto funkcionalitāti:

- fasādes
- bibliotēkas
- modeļu savstarpējo saišu deklarēšanu
- iebūvētās funkcijas

Savukārt 4. nodaļā veiktā salīdzinošā analīze parāda, ka Laravel ir vispiemērotākais satvars priekš šīs sistēmas (un tas liek domāt, kā arī priekš citām šāda veida sistēmām), un tas pilnīgi pamato izvēli, kaut arī sākot izstrādi šim satvaram uzreiz tika dota priekšroka.

No veiktās salīdzināšanas ir redzams, ka tas nodrošina gan visērtāko mijiedarbību ar datubāzi, gan piedāvā visplašāko iebūvētās funkcionalitātes izvēli, kā arī nodrošina trešo puši bibliotēku un API vieglu un ērtu pieslēgšanu. Var arī pieminēt, ka Laravel funkcionalitāte ir ļoti labi savienojamā ar Linux vidi, uz kuras darbojas gan lokālā izstrādes vide, gan gala serveris.

Secinājumi

Kā jau bija minēts darba sākumā, sistēmas izstrādes process un projekta arhitektūra tika definēti ļoti brīvi, tāpēc visa izstrāde notika tikai un vienīgi balstoties uz potenciālo lietotāju vajadzībām. Dažās vietās tas radīja neērtības koda strukturēšanas procesā, dāzreiz arī sistēmas struktūra sanāca ne tik precīzi un loģiski sadalīta, kā tam vajadzētu būt, un nebija drosmes to mainīt.

Sākot bakalaura darbu izstrādi, visas nepieciešamās sistēmas funkcijas tika apkopotas atkārtoti, vēlreiz analizētas, un sadalītas uz vecām, kurām bija nepieciešami uzlabojumi, un jaunām, kas vēl bija jāpievieno.

Darba veidošana lika arī atkārtoti iziet caurī visai tai sistēmas daļai, kas uz to brīdi jau bija gatava. Tas ļoti palīdzēja strukturēt esošo funkcionalitāti un saprast, kādi uzlabojumi ir jāveic, kā jāpārveido projekta loģika un arhitektūra. Un tajā pašā laikā jaunās funkcionalitātes pielikšana uzreiz arī bija sakārtota un loģiski strukturēta tā, lai jaunās funkcijas maksimāli pareizi paplašinātu jau esošo funkcionalitāti.

Uzsākot darbu, priekšroka tika dota Laravel satvaram jau esošās pieredzes dēļ, bet izstrādes gaitā rādījās šaubas, vai satvars tika izvēlēts pareizi. Par MVC struktūru, savukārt, šaubu nebija (un sistēmas strukturizācija darba rakstīšanas ietvaros liecina par šī lemuma pareizību), tāpēc tika nolemts veikt salīdzinošo analīzi ar citiem populāriem MVC satvariem un uz sistēmas struktūras piemēra apskatīties, kādi bija citi izstrādes varianti.

Salīdzinājuma gaitā kļuva saprotams, ka Laravel tomēr ir vispiemērotākais satvars priekš šāda veida sistēmas, jo tas pilnībā izmanto piedāvāto funkcionalitāti, un ar piedāvāto funkcionalitāti tam pilnīgi pietiek, kas jau tika aprakstīts darba rezultātos.

Izstrādātais darbs ļāva ne tikai izstrādāt labi strukturētu, ātru un uzticamu tīmekļa vietni ar plašu daudzveidīgo funkcionalitāti, bet arī uzlabot teorētiskās zināšanas gan par Laravel, gan par citiem satvariem. Kā arī tika iegūta svarīga pieredze oficiālo dokumentu rakstīšanā, kas iekļauj sevī lasītājam ērtu un saprotamu noformēšanu, datu padošanu un strukturēšanu, teksta stilizāciju.

Var droši teikt, ka paveiktā darba gaitā tika sasniegti visi izvirzītie mērķi.

Pateicības

Sēkojošo cilvēku secība sarakstā nav svarīga, jo visiem esmu pateicīga vienādi:

- darba vadītājai — Darjai Solodovņikovai, kas arī man vadīja kursa darbu: bez viņas iedomīgas iesaistības, dzīvās intereses un ātrās izpalīdzēšanas jautājumu gadījumos šie darbi nebūtu iespējami;
- darba vietas valdei — Modrim Šķēlam un Agrim Magonam, kas de facto pasūtīja šajā darbā aprakstītās sistēmas izstrādi: bez viņu augstiem mērķiem un spilgtām idejām šis darbs nebūtu iespējams;
- darba kolēģiem — Kristapam Tarandam, Kristīnai Meļņikovai, Markam Gerasimovam, Mārcim Krūmiņam, Kristiānai Vuškānei, Viktorijai Jeļeņinai, Agnesei Peterei, Līvai Kesenfeldei, Reinim Ziebergam, Janai Pauliņai, Ivo Seilis, Didzim Lindem, Gustavam Zvīnim, Jāzepam Lindem, Edmundam Kanapeckim, Evijai Krūzei, Jurijam Judkinam, Lolitai Lapiņai, Sandim Mūrniekam un Annijai Rabšai: bez viņu izpalīdzības, pacietības un savlaicīgām atsauksmēm šis darbs nebūtu iespējams;
- maniem vecākiem — Pāvelam Gončarovam un Olgai Gončarovai: bez viņu pārliecinātības manī un manos spēkos šis darbs nebūtu iespējams;
- maniem draugiem — Mihailam Ļimonovam, Kirillam Kiseļovam, Markam Bogdanovam, Vadimam Ņikišinam, Aleksandram Ļebedevam, Pāvelam Zencovam, Vladimiram Brovinam: bez viņu iedvesmojošiem piemēriem un bez viņu atbalsta šis darbs nebūtu iespējams.

Izmantotā literatūra un avoti

[1] Laravel 5.2 [Tiešsaite] Pieejams: laravel.com/docs/5.2/ [2018. g. 12. maijs]

[2] Model-view-controller [Tiešsaite] Pieejams: en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller [2018 g. 10. maijs]

[3] Laravel GitHub, *.env.example* [Tiešsaite] Pieejams: github.com/laravel/laravel/blob/master/.env.example [2018. g. 18. maijs]

[4] Vance Lucas, *PHP dotenv* [Tiešsaite] Pieejams: github.com/vlucas/phpdotenv [2018. g. 18. maijs]

[5] Laravel 5.2 documentation, *Blade Templates* [Tiešsaite] Pieejams: laravel.com/docs/5.2/blade [2018. g. 12. maijs]

[6] Laravel 5.2 documentation, *Artisan Console* [Tiešsaite] Pieejams: laravel.com/docs/5.2/artisan [2018. g. 15. maijs]

[7] Laravel 5.2 documentation, *Database: Getting Started* [Tiešsaite] Pieejams: laravel.com/docs/5.2/database [2018. g. 15. maijs]

[8] Laravel 5.2 documentation, *Filesystem* [Tiešsaite] Pieejams: laravel.com/docs/5.2/filesystem [2018. g. 15. maijs]

[9] Inamullah Khan, *What Is MVC and Why Do We Use MVC?* [Tiešsaite] Pieejams: www.c-sharpcorner.com/article/what-is-mvc-and-why-we-use-mvc [2018. g. 10. maijs]

[10] Jesse Anderson, *Why Use MVC?* [Tiešsaite] Pieejams: www.slideshare.net/JesseAnderson/mvc-5697864 [2018. g. 10. maijs]

[11] Anna Monus, *10 PHP Frameworks For Developers – Best of* [Tiešsaite] Pieejams: www.hongkiat.com/blog/best-php-frameworks [2018. g. 26. maijs]

[12] Symfony Documentation, *Doctrine* [Tiešsaite] Pieejams: symfony.com/doc/current/doctrine.html [2018. g. 26. maijs]

[13] Symfony Documentation, *Controller*[Tiešsaite] Pieejams: symfony.com/doc/current/controller.html [2018. g. 26. maijs]

[14] Symfony Documentation, *Creating and Using Templates* [Tiešsaite] Pieejams: symfony.com/doc/current/templating.html [2018. g. 26. maijs]

[15] Symfony Documentation, *Symfony Bundles* [Tiešsaite] Pieejams: symfony.com/doc/bundles [2018. g. 26. maijs]

[16] Symfony Documentation, *Stream Loader* [Tiešsaite] Pieejams: symfony.com/doc/current/bundles/LiipImagineBundle/data-loader/stream.htm [2018. g. 26. maijs]

[17] Symfony Documentation, *Console Commands* [Tiešsaite] Pieejams: symfony.com/doc/current/console.html [2018. g. 26. maijs]

[18] CodeIgniter Documentation, *Database Quick Start* [Tiešsaite] Pieejams: www.codeigniter.com/user_guide/database [2018. g. 26. maijs]

[19] CodeIgniter Documentation, *Models* [Tiešsaite] Pieejams: www.codeigniter.com/user_guide/general/models.html [2018. g. 26. maijs]

[20] CodeIgniter Documentation, *Controllers* [Tiešsaite] Pieejams: www.codeigniter.com/user_guide/general/controllers.html [2018. g. 26. maijs]

[21] CodeIgniter Documentation, *Views* [Tiešsaite] Pieejams: www.codeigniter.com/user_guide/general/views.html [2018. g. 26. maijs]

[22] CodeIgniter Documentation, *FTP Class* [Tiešsaite] Pieejams: www.codeigniter.com/user_guide/libraries/ftp.html [2018. g. 26. maijs]

[23] CodeIgniter Documentation, *Running via the CLI* [Tiešsaite] Pieejams: www.codeigniter.com/user_guide/general/cli.html [2018. g. 26. maijs]

[24] The Definitive Guide to Yii 2.0, *Query Builder* [Tiešsaite] Pieejams: www.yiiframework.com/doc/guide/2.0/en/db-query-builder [2018. g. 26. maijs]

[25] The Definitive Guide to Yii 2.0, *Models* [Tiešsaite] Pieejams: www.yiiframework.com/doc/guide/2.0/en/structure-models [2018. g. 26. maijs]

[26] The Definitive Guide to Yii 2.0, *Controllers* [Tiešsaite] Pieejams: www.yiiframework.com/doc/guide/2.0/en/structure-controllers [2018. g. 26. maijs]

[27] The Definitive Guide to Yii 2.0, *Views* [Tiešsaite] Pieejams: www.yiiframework.com/doc/guide/2.0/en/structure-views [2018. g. 26. maijs]

[28] The Definitive Guide to Yii 2.0, *Shared Hosting Environment* [Tiešsaite] Pieejams: www.yiiframework.com/doc/guide/2.0/en/tutorial-shared-hosting [2018. g. 26. maijs]

[29] The Definitive Guide to Yii 2.0, *Console applications* [Tiešsaite] Pieejams: www.yiiframework.com/doc/guide/2.0/en/tutorial-console [2018. g. 26. maijs]

Bakalaura darbs „Reklāmas sistēmas izstrāde” izstrādāts LU Datorikas fakultātē.

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

Autors: Anastasija Gončarova

Rekomendēju/nerekomendēju darbu aizstāvēšanai

Vadītāja: Dr.dat. Asociētā profesore Darja Solodovņikova 27.05.2018.

Recenzents: profesors Māris Vītiņš

Dekāna pilnvarotā persona: vecākā metodiķe Ārija Sproģe _____

Darbs aizstāvēts bakalaura gala pārbaudījuma komisijas sēdē __.06.2018

_____. prot. Nr. _____.

Komisijas sekretārs: _____ _____