

**Šolski center Novo mesto
Srednja elektro šola in tehniška gimnazija
Šegova ulica 112
8000 Novo mesto**

Maturitetna seminarska naloga pri predmetu računalništvo

**APLIKACIJE IN INFORMACIJSKI SISTEMI - Najem
avtomobilov**

Predmet: Računalništvo

Avtor: Tim Rezelj, T4C

Mentor/mentorica: dr. Albert Zorko, univ. dipl. inž.

Novo mesto, april 2021

Povzetek

Z izdelavo seminarske naloge sem želel bralce podučiti o modernih metodah postavljanja spletnih aplikacij. Splet je v današnjem svetu pripomoček, brez katerega ne moremo živeti saj so od njega odvisni skoraj vsi procesi moderne družbe. Pri razvijanju spletnih aplikacij je na voljo veliko različnih tehnologij in pristopov, katere lahko uporabimo za izgradnjo naprednih aplikacij. Ob raziskovanju literature sem ugotovil, da je teh tehnologij ogromno, zato sem se lahko odločil samo za nekaj izmed njih, saj jih veliko deluje na isti način, razlikujejo se le v nekaterih lastnostih. Izbral sem tiste, ki se mi zdijo najbolj splošne in jih lahko uporabljamo tudi za druge aplikacije, saj orodja omogočajo veliko možnosti.

Ključne besede: spletna aplikacija, Yii2 ogrodje, podatkovna baza, Najem avtomobilov, model pogled krmilnik

Abstract

By creating this paper, I wanted to teach readers about modern methods of setting up web applications. In today's world, the Internet is a tool without which we cannot live, as almost all the processes of modern society depend on it. There are many different technologies and approaches available in developing web applications that can be used to build advanced applications. While researching the literature, I found that there is a lot of similar technologies, so I was only able to opt for a few of them, because many of them work in the same way, differing only in some properties. I chose the ones that I find most general and can also be used for other applications as the tools offer a lot of options.

Key keyword: web applications, Yii2 framework, database, Najem avtomobilov, model view controller

Kazalo vsebine

1	UVOD	1
1.1	Model Pogled Krmilnik (MVC)	1
1.2	YII2	2
1.3	PHP	2
1.4	HTML	3
1.5	CSS	3
1.6	JavaScript	3
1.7	Ajax	4
1.8	XAMPP	4
1.9	Apache spletni strežnik	5
1.10	MySQL	5
1.11	Visual Studio Code	5
2	Aplikacija Najem avtomobilov	6
2.1	Načrtovanje aplikacije	6
2.1.1	Podatkovna baza	7
2.2	Izvedba aplikacije	8
2.2.1	Začetna stran	8
2.2.2	Prikaz avtomobilov	9
2.2.3	Informacije o avtomobilu	11
2.2.4	Stran za registracijo	12
2.2.5	Stran za prijavo	13
2.2.6	Podstran za dodajanje avtomobilov	14
3	Zaključek	17
4	Zahvala	18
5	Bibliografija	19
6	Priloge	21

Kazalo slik

Slika 1: MVC diagram.....	2
Slika 2: Potek Ajax zahtevka	4
Slika 3: Apache logotip	5
Slika 4: Shema zgradbe spletne aplikacije.....	6
Slika 5: Diagram podatkovne baze	7
Slika 6: Začetna stran aplikacije	8
Slika 7: Podstran prikaz avtomobilov	9
Slika 8: Podatek o rezervaciji.....	9
Slika 9: Prikaz informacij o avtomobilu brez registracije.....	11
Slika 10: Prikaz informacij o avtomobilu z registracijo	11
Slika 11: Rezervacija brez sporočila za napačna datuma	12
Slika 12: Sporočilo ob napačni rezervaciji.....	12
Slika 13: Stran za registracijo	12
Slika 14: Stran za prijavo.....	13
Slika 15: Začetna stran po registraciji	13
Slika 16: Slika strani za dodajanje avtomobilov	14
Slika 17: Obrezovanje fotografije	15
Slika 18: Prikaz slike ko jo shranimo v bazo	15

Kazalo prilog

Priloga 1: index.php.....	21
Priloga 2: index.css	21
Priloga 3: IndexAsset.php.....	21
Priloga 4: Cars.php.....	21
Priloga 5: SiteController.php.....	21
Priloga 6: Index.js.....	21
Priloga 7: display_cars.php	21
Priloga 8: display_cars.css	21
Priloga 9: display_carsAsset.php.....	21
Priloga 10: car_info.php (datoteka z razredom)	21
Priloga 11: car_info.php (datoteka z HTML in PHP kodo).....	21
Priloga 12: car_info.css	21
Priloga 13: car_infoAsset.php.....	21
Priloga 14: BookingHistory.php	21
Priloga 15: signup.php.....	22
Priloga 16: SignupForm.php	22
Priloga 17: User.php.....	22
Priloga 18: SignupAsset.php	22
Priloga 19: signup.css	22
Priloga 20: signup.js	22
Priloga 21: login.php.....	22
Priloga 22: LoginAsset.php.....	22
Priloga 23: LoginForm.php	22
Priloga 24: login.css	22
Priloga 25: add_cars.php.....	22
Priloga 26: Add_new_car.php.....	22
Priloga 27: Add_cars.php	22
Priloga 28: TempPhotos.php	22
Priloga 29: add_cars.css	22
Priloga 30: croppie.css	23
Priloga 31: croppie.js.....	23
Priloga 32: add_cars.js.....	23

1 UVOD

Za svojo maturitetno seminarsko nalogo sem se odločil, da bom zgradil spletno aplikacijo za najemanje avtomobilov. V današnjih časih veliko podjetij seli svoje aplikacije na splet. Eden izmed glavnih razlogov je, da je aplikacija dosegljiva uporabnikom iz celega sveta. Pri izdelavi aplikacije sem uporabil tehnologije in pristope, ki so danes zelo popularni pri razvoju velikih in pomembnih projektov. Aplikacija uporabniku omogoča ogled, rezervacijo in ponujanje svojih avtomobilov za najem. Vgrajene ima varnostne mehanizme za kodiranje podatkov in varno shranjevanje le teh. Ko v brskalnik vnesemo spletni naslov nas aplikacija vedno preusmeri na začetno stran. Aplikacija vsebuje še štiri podstrani, ena od teh pa ima še dodatno podstran. Aplikacija bi imela lahko tudi praktično uporabo na trgu, saj za enkrat še nisem opazil, da bi na slovenskem trgu obstajala aplikacija za najem avtomobilov, kjer bi uporabniki avtomobile lahko najemali in svoje ponujali v najem drugim uporabnikom.

V prvem delu seminarske naloge opišem pristope in tehnologije, ki sem jih uporabil pri postavitvi spletne aplikacije. Opisana so tudi orodja, s katerimi sem aplikacijo gradil, vendar pa so to samo moja priljubljena orodja in bi lahko uporabil še veliko drugih. Za lažjo predstavo strukture aplikacije sem zraven priložil še diagram, ki prikazuje kako je aplikacija sestavljena. V nadaljevanju predstavim vse podstrani aplikacije, pri tem pa opišem tudi kako del tiste podstrani deluje v brskalniku in na strežniku. Pri opisu vsake podstrani so našteje datoteke, v katerih se nahaja programska koda, da si jo lahko bralec ogleda (koda pa se nahaja v prilogi).

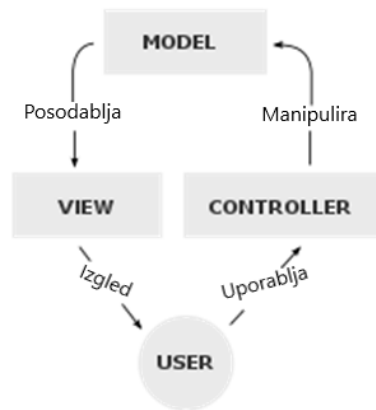
1.1 Model Pogled Krmilnik (MVC¹)

Model Pogled Krmilnik se prvič pojavi v Smalltalk-80 programskem okolju okoli leta 1980. Danes se ta način programiranja uporablja v programiranju namiznih grafičnih vmesnikov, zelo popularen pa je postal tudi v spletnih aplikacijah. MVC pristop danes uporabljajo skoraj vsi bolj popularni programski jeziki (predvsem jeziki za delovanje na strežniku) kot so JavaScript, Python, PHP, Java in še veliko drugih. (1)

Pogled komponenta aplikacije skrbi za prikazovanje informacij uporabnikom. V njih napišemo kodo za grafični vmesnik, v katero nato vstavljamo podatke, ki so shranjeni v Model komponenti. En Model lahko uporabljamo za več pogledov. Če se v enem od Modelov spremeni nek podatek in je od njega odvisnih več pogledov se bo v vseh ta podatek spremenil. Vsak pogled ima svoj krmilnik, ki skrbi za sprejemanje informacij, njihovo obdelavo in shranjevanje v Model komponente. Model tako vsebuje večino funkcionalnosti od shranjevanja podatkov do funkcij za njihovo obdelavo, krmilnik pa lahko ob zahtevi uporabnika (preko pogled komponente) te podatke spremeni in kliče funkcije, ki se nahajajo v Model komponenti. (2)

MVC ima torej tri komponente. Model komponenta skrbi za shranjevanje informacij in njihovo obdelavo. Pogled komponenta skrbi za grafični vmesnik med uporabnikom in Krmilnik komponento. Krmilnik komponenta pa povezuje Pogled in Model komponento, kjer sprejema, obdeluje in shranjuje informacije ter sproži prikazovanje pogledov. Vizualno reprezentacijo tega koncepta prikazuje Slika 1.

¹ ang. Model View Controller



Slika 1: MVC diagram (3)

1.2 Yii2

Yii2 ogrodje je odprtokodni projekt, ki je bil ustvarjen za hiter razvoj modernih spletnih aplikacij. Stoji na programskem jeziku PHP, ki ga veliko ljudi ne želi uporabljati, saj naj bi bil počasen in zastarel vendar je Yii2 ogrodje popolno nasprotje od teh trditev. Stoji okoli MVC pristopa, o katerem sem govoril že v Model Pogled Krmilnik (MVC) poglavju. Ogrodje ima vgrajene komponente za varno in napredno grajenje spletnih aplikacij. Primerno je za gradnjo ogromno različnih aplikacij od forumov, portalov do spletnih aplikacij itd. Ogrodje uporablja samo objektno programiranje, zato od programerjev zahteva nekoliko bolj napredno poznavanje programskih jezikov in konceptov. Vse kar potrebujemo za delovanje tega ogrodja je PHP 5.4+ in spletni strežnik. Eden izmed najboljših je Apache, katerega sem uporabil tudi sam, da sem omogočil dostop do spletne aplikacije iz celega sveta. (3)

1.3 PHP

PHP je odprtokodni programski jezik za ustvarjanje programov, ki se izvajajo na spletnem strežniku. Na internetu imajo ljudje do tega jezika različna mnenja, saj nekateri pravijo, da je prepočasen in neuporaben, spet drugi so njegovi zagovorniki in ga pogosto uporabljajo. Jezik uporabljajo tudi največja svetovna podjetja kot na primer Facebook, uporablja pa se tudi pri postavitvi WordPress strani (4). Programerjem omogoča izdelavo dinamičnih vsebin in komuniciranje z bazo podatkov. Uporablja se za izdelavo spletnih aplikacij in API² rešitev. Ob pisanju PHP programa lahko znotraj datoteke vključimo tudi HTML, CSS in JavaScript kodo, jezik pa nato brskalniku pošlje samo končno verzijo programske kode. (5)

² Application programming interface

1.4 HTML

Razvil ga je Tim Berners-Lee leta 1990. Uporablja se za programiranje spletnih strani današnjega spleta. Jezik predstavlja ogrodje spletne strani, katerega nato s pomočjo CSS-a uredimo in olepšamo, da spletna stran izgleda lepo. Uporablja ga čisto vsaka spletna stran, danes najbolj popularna pa je verzija HTML5, ki vsebuje največ možnosti za urejanje in drugih naprednih funkcionalnosti. (6)

Moja aplikacija ne vsebuje ločenih HTML datotek, saj je koda vgrajena v PHP datoteke, katero lahko PHP jezik dinamično spreminja glede na uporabnikove zadetke. Koda HTML sama po sebi ni težko razumljiva, zato je v seminarski nisem omenjal veliko lahko pa si jo ogledate v prilogi.

1.5 CSS

CSS (angl. Cascading Style Sheets) je jezik, ki se uporablja za olepšavo HTML in XML programske kode. Definira, kako naj bodo elementi postavljeni v dokumentu, kakšno barvo naj imajo, stil pisave, velikost pisave in okvirjev itd. Je eden izmed glavnih jezikov današnjega spleta in ga podpira skoraj vsak moderni brskalnik. Danes je najbolj popularna verzija CSS3, ki je postala nekakšen standard svetovnega spleta kot ga poznamo danes. (7)

Pri postavitvi svoje aplikacije sem uporabil veliko CSS, da sem elemente lahko lepo postavil kakor sem želel in jim dodal obliko ter barvo ozadja. Tekom seminarske naloge CSS ni opisan, saj menim, da v ozadju to nima posebne logike, saj skrbi samo za izgled spletne strani.

1.6 JavaScript

Ustvaril ga je Brendan Eich leta 1995. Namen je bil narediti HTML nekoliko bolj interaktiven in povezati brskalnik z programskim jezikom Java, ki bi deloval kot strežniška aplikacija (od tu tudi ime JavaScript). (8) JavaScript je do danes močno napredoval in se ga lahko uporablja tudi kot strežniško aplikacijo, za strojno učenje in veliko več. Najpogosteje pa se uporablja na brskalniku, kjer skrbi za interaktivnost spletne strani (ko uporabnik klikne na nek gumb naj se nekaj zgodi). Velikokrat lahko uporabljamo njena ogrodja kot so React, Angular in Vue, če pa želimo razvijati mobilno aplikacijo ostaja tudi React Native, vendar teh ogrodij v svoji seminarski nalogi nisem uporabil. (9)

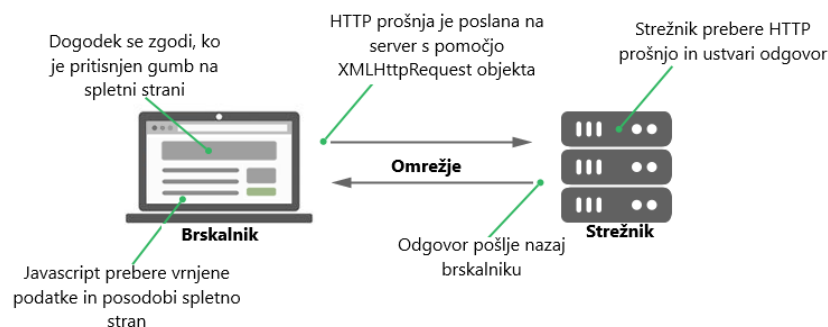
1.7 Ajax

Ajax je pristop s katerim dostopamo do podatkov in jih nalagamo v spletno stran, ne da bi pri tem ponovno nalagali celotno spletno stran, kar lahko vzame kar nekaj dragocenega časa. V času, ko brskalnik čaka na odziv spletnega strežnika lahko uporabnik počne druge stvari, ko pa podatki prispejo jih na spletni strani spremeni. Ta pristop je uporabniško izkušnjo izredno izboljšal, saj se lahko podatki na spletni strani ves čas posodablajo brez ponovnega nalaganja spletne strani in pri tem motnje uporabniške izkušnje. Ajax uporablja tri različne oblike, v katerih lahko strežnik pošilja podatke brskalniku:

- HTML
- XML
- JSON

(10)

To tehnologijo sem uporabil tudi v svoji spletni aplikaciji, kjer sem jo uporabil za pošiljanje in pridobivanje slik iz baze podatkov.



Slika 2: Potek Ajax zahtevka (11)

1.8 XAMPP

XAMPP je izredno popularen spletni strežnik, ki se uporablja za razvijanje in testiranje programov, preden jih naložimo na produkcijski strežnik. Program so na začetku razvili tako imenovani »Apache Prijatelji«, danes pa lahko kdorkoli pregleduje in spreminja kodo, saj je orodje postalo odprtokodni projekt. Uporablja Apache spletni strežnik vendar se uporablja za lokalni razvoj aplikacij (do aplikacije lahko dostopa samo računalnik, na katerem se to okolje izvaja) in ne služi kot strežnik za dostop do aplikacije preko interneta. Je zelo koristno orodje za razvoj in testiranje aplikacij, da tega ne počnemo na produkcijskem strežniku, kjer je aplikacija dostopna širnemu spletu. (11)

Pri razvoju svoje aplikacije sem uporabil popolnoma enak pristop. Najprej sem jo razvil na XAMPP okolju, nato pa sem aplikacijo prestavil na **Napaka! Vira sklicevanja ni bilo mogoče najti.** (kjer je sedaj dostopna vsakomur) in zakupil spletno domeno, preko katere je moja aplikacija sedaj dostopna.

1.9 Apache spletni strežnik

To je spletni strežnik, ki je zgrajen po vseh HTTP standardih. Projekt je napisan odprtokodno, zato ga lahko uporabi čisto vsak, tisti ki pa ga bolje razumejo pa lahko k projektu tudi pripomorejo in pomagajo skupnosti. Na trg je prvič vstopil leta 1995, najpopularnejši strežnik pa je postal leta 1996 in je tako ostalo vse do danes. Apache lahko uporabljamo tako za statične spletne aplikacije kot tudi za velike spletne aplikacije, ki imajo več 1000 uporabnikov in komunicirajo z več podatkovnimi bazami. (12)



Slika 3: Apache logotip (13)

1.10 MySQL

Najbolj popularen sistem za upravljanje z SQL podatkovnimi bazami. MySQL je razvilo Švicarsko podjetje MySQL AB leta 1994 danes pa ima to podjetje in s tem tudi MySQL v lasti podjetje Oracle. Uporablja se za veliko različnih spletnih aplikacij ter za druge sisteme, ki potrebujejo podatkovno bazo. V svoji aplikaciji sem to orodje uporabil za postavitev podatkovne baze, kamor se shranjujejo podatki uporabnikov. (13)

1.11 Visual Studio Code

VS Code je brezplačen odprtokodni urejevalnik besedil, ki ga je ustvaril Microsoft. Vsebuje veliko razširitev, ki jih lahko po želji naložimo in nam pripomorejo pri pisanju kode. Zaradi vsega tega in še mnogo drugih dodatkov je to eden izmed najboljših urejevalnikov. Uporabljam ga vsakodnevno in mi je prišel zelo prav tudi pri izdelavi te spletne aplikacije.

2 Aplikacija Najem avtomobilov

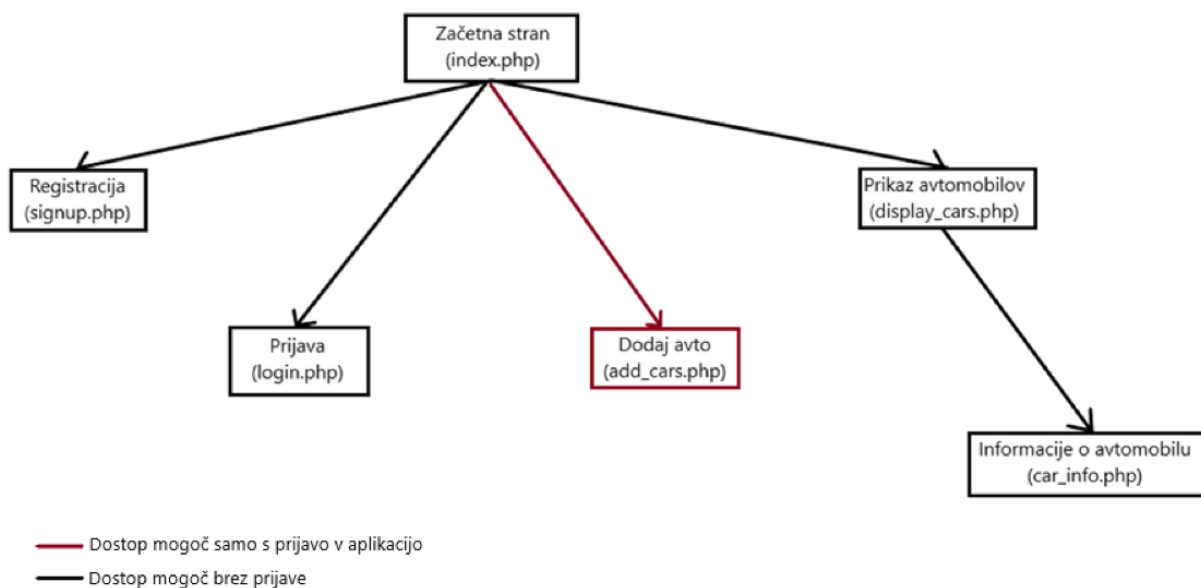
V do tem delu seminarske naloge je opisan postopek izdelave aplikacije od načrtovanja končnega izdelka. Opisuje zgradbo spletne strani, podatkovne baze, nato pa vsako podstran opiše podrobneje od njenega delovanja do njenega izgleda.

2.1 Načrtovanje aplikacije

Svojo spletno aplikacijo sem razdelil na več spletnih strani. V spletno aplikacijo se moramo registrirati, pri čemer moramo opraviti verifikacijo s pomočjo e-poštnega naslova. Po zaključeni registraciji se lahko v aplikacijo prijavimo, kar nam omogoči dostop do funkcij, ki so nam sicer ne dostopne. Z registracijo pridobimo možnost dodajanja svojih avtomobilov, ki bi jih radi oddajali v najem. Pri prikazu avtomobilov pa dobimo možnost rezervacije avtomobila za nek izbrani datum. Če se v aplikacijo ne prijavimo pa imamo dostop samo do nekaterih strani, do katerih imamo dostop tudi če smo v aplikacijo prijavljeni. Te podstrani so:

- Prikaz avtomobilov
- Informacije o avtomobilu
- Stran za registracijo
- Stran za prijavo

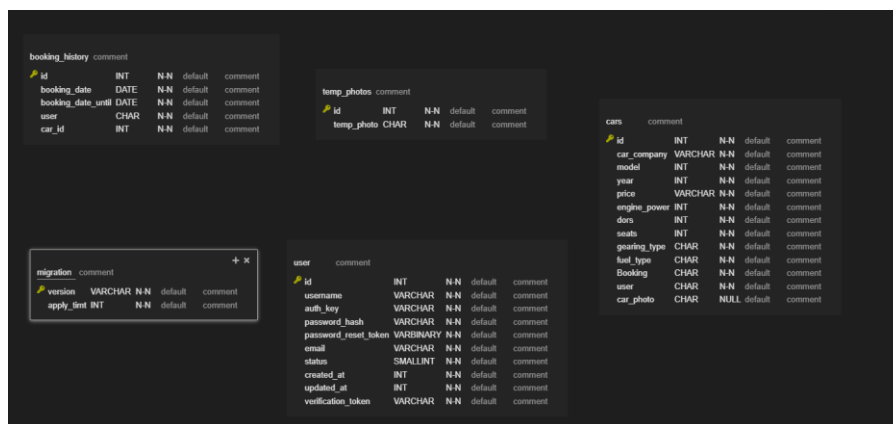
Zgradbo spletne strani predstavlja Slika 4, ki grafično prikazuje sestavo aplikacije.



Slika 4: Shema zgradbe spletne aplikacije

2.1.1 Podatkovna baza

Pri spletni aplikaciji sem za podatkovno bazo uporabil MySQL sistem za upravljanje s podatkovnimi bazami. Ob začetku izdelovanja aplikacije še nisem vedel veliko o podatkovnih bazah, zato sem z vsem začenjal od začetka in nisem vedel nič o ER diagramu in podobnih stvareh, zato tudi v bazi nimam relacijskih povezav, vendar šele sedaj vidim da bi bilo veliko bolje, če bi jih imel, saj bi bilo programiranje in shranjevanje podatkov veliko lažje. Slika 5 predstavlja diagram vseh tabel in njihovih stolpcev. Tabela *migration* in *user* sta tabeli, ki nam jih poda Yii2 ogrodje. V tabeli *migration* se shranjujejo verzije migracij (migracija je spreminjanje podatkovne baze z predhodno napisanimi ukazi) in čas ob katerem je bila le ta opravljena. Tabela *user* pa nam služi za shranjevanje podatkov o uporabnikih. Ostale entitete sem v podatkovno bazo dodal jaz in jih moji programi uporabljajo za shranjevanje ostalih podatkov.

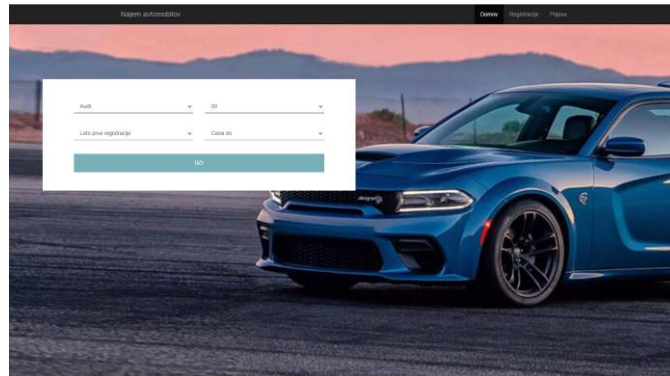


Slika 5: Diagram podatkovne baze

2.2 Izvedba aplikacije

2.2.1 Začetna stran

Ko v spletni brskalnik vnesemo naslov www.timrezelj.tk pristanemo na začetni spletni strani moje spletne strani. Kot lahko vidimo imamo v belem okvirju obrazec za izbiranje kriterijev za prikaz avtomobilov, ki so na voljo za najem. V tem obrazcu imamo možnost izbire štirih kriterijev: znamko avtomobilov, model, prvo leto registracije in ceno do katere želimo, da se avtomobili prikažejo. Ko pritisnemo gumb iskanje nas spletna stran preusmeri na Prikaz avtomobilov, kjer pa imamo obrazec v katerem lahko rezultate še dodatno filtriramo.



Slika 6: Začetna stran aplikacije

Datoteke uporabljene na strani:

- *index.php*
- *index.css*
- *IndexAsset.php*
- *Cars.php*
- *SiteController.php* (funkcija *actionIndex*)
- *Index.js*

Uporaba datotek

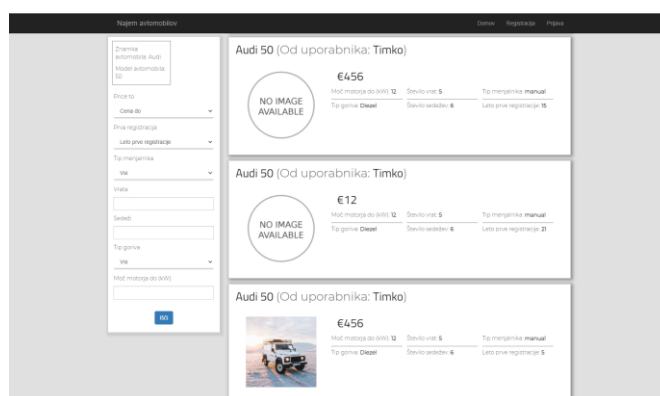
Datoteka *index.php* je namenjena izgledu spletne strani, tukaj notri se nahaja tudi HTML spletne strani, s pomočjo jezika PHP pa notri dodajamo podatke in jih sprejemamo (ko uporabnik pritisne na gumb), končno HTML kodo pa še vizualno urejamo z *index.css*. Za boljše delovanje obrazca in da se vanj samodejno posodablajo podatki se uporablja programska koda, v *index.js*, ki zazna spremembo enega polja in ob tem spremeni še drugega. Da lahko ti dve datoteki dodamo v HTML uporabimo dodatno datoteko, *IndexAsset.php*, ki je izpeljan iz razreda *AssetBundle* (14), ki ga ogrodje Yii uporablja za upravljanje z CSS in JS datotekami. Znotraj razreda lahko najdemo dodani datoteki CSS in JS.

Upravljanje s podatki

Za podatke in upravljanje z njimi skrbita datoteki *Cars.php* in *SiteController.php*. *Cars.php* je model, v katerega shranjujemo podatke. Razred *Cars* je razširjen iz razreda *ActiveRecord* (15), ki se uporablja za dostopanje do podatkovne baze. V njem so definirana imena spremenljivk za podatke in njihovi podatkovni tipi, ki jih je možno shraniti v njega in posledično v podatkovno bazo. Funkcija v datoteki *SiteController.php* (v tej datoteki je koda, ki predstavlja kontroler opisan v Model Pogled Krmilnik (MVC)) pa skrbi za nalaganje *Cars* modela in prikazovanja vsebine v datoteki *index.php*.

2.2.2 Prikaz avtomobilov

Do te podstrani pridemo z pritiskom na gumb iskanje na začetni strani, ki pobere podatke iz tamkajšnjega obrazca in jih uporabi za prikazovanje rezultatov na tej podstrani. Zadetki, ki ustrezajo kriterijem njihove aplikacije prikaže, ostale pa izpusti (to nam stori že podatkovna baza in s tem ne obremenjujemo spletnega strežnika). Pri vsakemu avtomobilu prikažemo osnovne informacije, kot so: moč motorja, število vrat, tip menjalnika, vrsto goriva, število sedežev, leto prve registracije, ceno dnevnega najema, znamko ter model avtomobila. Če smo v aplikacijo prijavljeni pa se nam dodatno pojavi še podatek, ali je avtomobil že rezerviran ali ne (prikazuje Slika 8).



Slika 7: Podstran prikaz avtomobilov



Slika 8: Podatek o rezervaciji

Datoteke, ki jih stran uporablja:

- *Cars.php*
- *display_cars.php*
- *display_cars.css*
- *display_carsAsset.php*
- *SiteController.php* (podstran uporablja samo funkcijo *actionDisplay_cars*)

Delovanje podstrani na strežniku

Slika 7 prikazuje tudi meni na levi strani (za izgradnjo le tega sem uporabil razred *ActiveForm* (15), ki je vgrajen v Yii ogrodje in se uporablja za programiranje obrazcev ter pošiljanje podatkov kontrolerju), v katerem lahko izbiramo dodatne kriterije za filtriranje izdelkov. Poleg kriterijev, ki smo jih že izbrali predhodno imamo tukaj na voljo še izbiro tipa menjalnika, število vrat, število sedežev, tip goriva in moči motorja (omejitev, do katere moči v kW aplikacija prikaže zadetke). Ko pritisnemo gumb iskanje se sproži funkcija *actionDisplay_cars* (ki se nahaja v datoteki *SiteController*), ki sproži ponovno nalaganje spletnega mesta *display_cars*, kateri podamo model *Cars* (*return \$this->render('display_cars',['model' => \$model]);*). Tako se ponovno izvede datoteka *display_cars.php*, kjer s pomočjo modela dostopamo do podatkovne baze, katera nam vrne vse avtomobile, ki ustrezajo podanemu kriteriju (

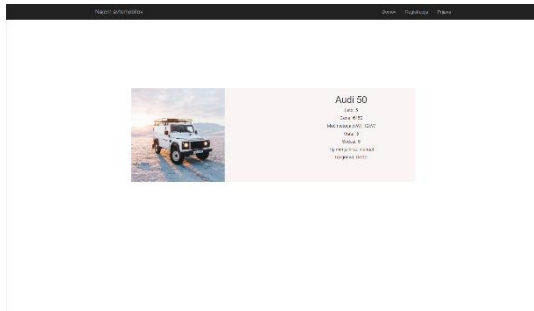
```
Cars::find()->where($all_filtering)->andWhere($price_filtering)->andWhere($engine_filtering)->all();
ali
Cars::find()->where($all_filtering)->andWhere($price_filtering)->all();
```

Odvisno, ali je bil podan kriterij za moč motorja ali ne).

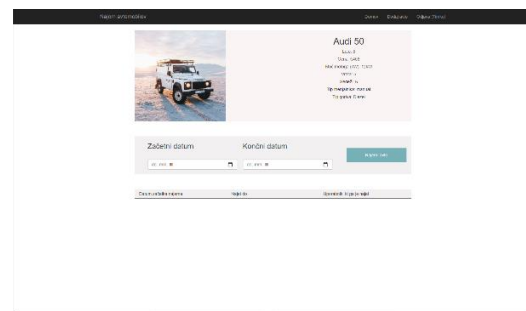
2.2.3 Informacije o avtomobilu

Za ogled podrobnejših informacij o avtomobilu in rezervaciji le tega se na strani *display_cars* postavimo na avtomobil, katerega si želimo ogledati in kliknemo nanj. To nas preusmeri na spletno stran *car_info*, kjer se nam stran prikaže različno na glede na to ali smo v aplikacijo prijavljeni ali ne.

Če v aplikacijo nismo prijavljeni se prikaže samo slika in podatki o avtomobilu. Aplikacija nam ne omogoči ogleda preteklih rezervacij avtomobila in nam ne omogoči rezervacije le tega.



Slika 10: Prikaz informacij o avtomobilu brez



Slika 9: Prikaz informacij o avtomobilu z registracijo

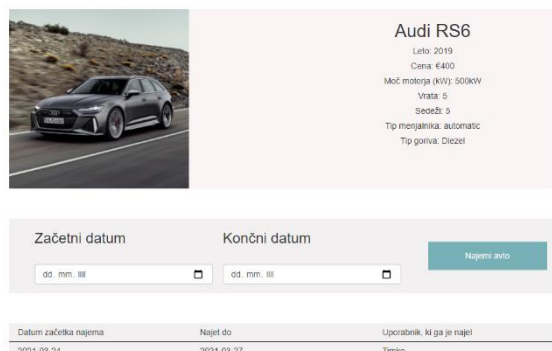
Datoteke uporabljene na strani:

- *car_info.php* (datoteka z razredom)
- *car_info.php* (datoteka z HTML in PHP kodo)
- *car_info.css*
- *car_infoAsset.php*
- *SiteController.php* (funkcija *actionCar_info*)
- *BookingHistory.php*

Delovanje programa za rezervacijo avtomobila

Če se uporabnik odloči, da mu avtomobil ustreza, si ga lahko rezervira za določen datum. V polju mora izbrati začetni in končni datum rezervacije. Ko si uporabnik izbere datuma, ki mu ustrezata pritisne gumb rezerviraj avto, ki sproži funkcijo imenovano *actionCar_info*, ki se nahaja v *SiteController* datoteki. Ko se funkcija prične izvajati se najprej ustvari objekt razreda *car_info*, nato pa se vanj naložijo podatki, ki smo jih pridobili iz obrazca. V nadaljevanju funkcija v podatkovni bazi preveri, ali obstaja kakšen zapis, pri katerem je končni ali začetni datum večji od današnjega. Če noben takšen zapis ne obstaja (podatkovna baza nam ne vrne nobenega rezultata) in je izbrani začetni datum enak ali večji današnjemu potem lahko rezervacijo shranimo v podatkovno bazo, v nasprotnem primeru pri ponovnem nalaganju spletnega mesta izpišemo sporočilo napake. V primeru, da nam podatkovna baza vrne zapise o rezervaciji avtomobilov, ki imajo začetni ali končni datum večji kakor je današnji, potem preverimo ali se mogoče kateri izmed vrnjenih datumov prekriva z datumi, ki jih je izbral uporabnik. Če se datumi prekrivajo potem se tudi tukaj pri ponovnem nalaganju spletne strani prikaže sporočilo o napaki, če pa se datumi ne prekrivajo se podatki s pomočjo *car_info*

modela shranijo v podatkovno bazo uporabnika pa preusmeri nazaj na stran **Napaka! Vira sklicevanja ni bilo mogoče najti.**, kjer se ponovno izpišejo avtomobili.



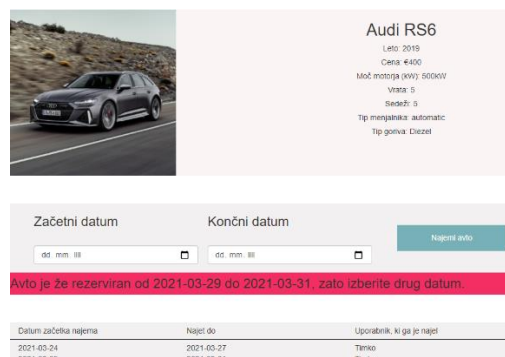
Audi RS6
 Leto: 2019
 Cena: 6400
 Moč motorja (kW): 500kW
 Vrata: 5
 Sedel: 5
 Tip menjalnika: avtomatič
 Tip goriva: Dizel

Začetni datum: dd. mm. III
 Končni datum: dd. mm. III

Najemi avto

Datum začetka najema	Najet do	Uporabnik, ki ga je najel
2021-03-24	2021-03-27	Temko

Slika 11: Rezervacija brez sporočila za napačna datuma



Audi RS6
 Leto: 2019
 Cena: 6400
 Moč motorja (kW): 500kW
 Vrata: 5
 Sedel: 5
 Tip menjalnika: avtomatič
 Tip goriva: Dizel

Začetni datum: dd. mm. III
 Končni datum: dd. mm. III

Najemi avto

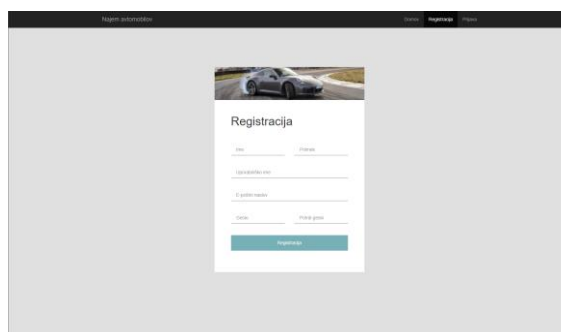
Avto je že rezerviran od 2021-03-29 do 2021-03-31, zato izberite drug datum.

Datum začetka najema	Najet do	Uporabnik, ki ga je najel
2021-03-24	2021-03-27	Temko
2021-03-29	2021-03-31	Temko

Slika 12: Sporočilo ob napačni rezervaciji

2.2.4 Stran za registracijo

Če želimo pridobiti preostale podstrani in njihove funkcionalnosti se moramo v aplikacijo registrirati. Za registracijo moramo vnesti svoj ime, priimek, uporabniško ime (s katerim se bomo kasneje na strani za prijavo lahko v aplikacijo prijavili), aktiven e-poštni naslov (aplikacija ima vgrajeno funkcijo za pošiljanje e-sporočila za verifikacijo le tega), geslo in polje za ponovni vpis gesla, s katerim preverimo če sta gesli enaki.



Registracija

Ime: _____ Priimek: _____

E-pošta: _____

Geslo: _____ Potrdi geslo: _____

Registriraj se

Slika 13: Stran za registracijo

Datoteke uporabljene na strani:

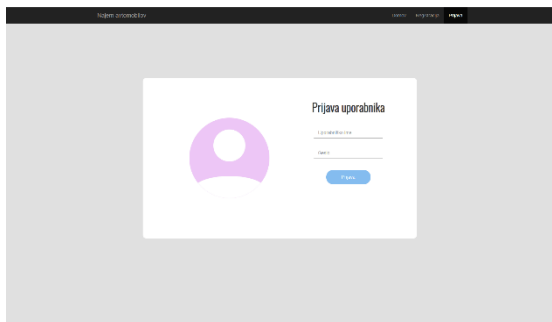
- `signup.php`
- `SignupForm.php`
- `SignupAsset.php`
- `SiteController.php` (funkcija `actionSignup`)
- `signup.css`
- `signup.js`
- `User.php`

Delovanje programa za registracijo

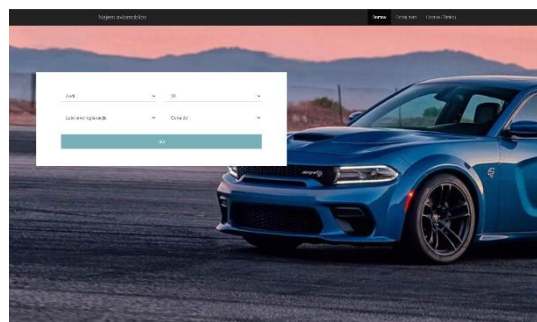
Ko v obrazcu pritisnemo gumb registriraj se, se na spletnem strežniku izvede funkcija *actionSignup* (je del krmilnika). Najprej se ustvari objekt razreda *SignupForm* (16), v katerega naložimo podatke iz obrazca. S pomočjo metode *signup* (16) (je del *SignupForm* (16) razreda) se začne verifikacija in shranjevanje podatkov v bazo. Metoda najprej izvede funkcijo *validate*, ki pripada *Model* (17) objektu (*SignupForm* model je izpeljan iz *Model* (17) objekta), katera preveri ali se pridobljeni podatki ujemajo z pogoji definiranimi v funkciji *rules* (16) (v *SignupForm* (16) razredu). Če metoda ugotovi, da so podatki pravilni se lahko izvajanje *signup* (16) funkcije nadaljuje, v nasprotnem primeru pa vrne vrednost *null*, kar povzroči ponovno nalaganje strani za registracijo, podatki pa se ne shranijo v podatkovno bazo. Če je bila verifikacija uspešna se ustvari nov objekt tipa *User* (18), v katerega iz trenutnega razreda (*SignupForm*) naložimo podatke prejete iz obrazca in pri tem uporabimo funkcije razreda *User* (18): *setPassword* (18) (generira kodirano besedilo iz teksta, ki ga podamo in ga shrani v *password* spremenljivko ki se nahaja v *User* modelu), *generateAuthKey* (18) (generira ključ, katerega nam ogrodje shrani v piškotek, če ob prijavi kliknemo zapomni si me, da se ne rabimo prijaviti vsakič ko vstopimo na spletno stran) in *generateEmailVerificationToken* (18) (generira ključ, s katerim aplikacija preveri verodostojnost vnesenega e-naslova). Metoda *signup* (16) vrne *true*, če je uporabnik uspešno shranjen v podatkovno bazo in če je e-sporočilo za verifikacijo uspešno poslano (sporočilo pošlje s pomočjo *sendEmail* metode v *SignupForm* (16)), po tem pa se v *actionSignup* izvede ponovno nalaganje strani za registracijo, le da je v tem primeru uporabnik shranjen v bazo podatkov, na e-pošti pa ga čaka e-sporočilo za verifikacijo e-poštnega naslova.

2.2.5 Stran za prijavo

Po uspešni registraciji in potrditvi e-naslova se lahko v aplikacijo za najem avtomobilov prijavimo na strani za prijavo. Slika 14 nam prikazuje, da moramo za prijavo vnesti uporabniško ime in geslo (Stran za registracijo je ta podatka predhodno posredovala podatkovni bazi). Ko vnesemo pravilno uporabniško ime in geslo ter pritisnemo na gumb prijava nas aplikacija preusmeri na začetno stran (Začetna stran jo podrobno opiše), pri tem pa nam omogoči dostop do vseh podstrani, ki so bile predhodno onemogočene, ker nismo bili prijavljeni. V menijski vrstici se nam namesto gumbov registracija in prijava pojavi gumb za odjavo v oklepaju pa je uporabnikovo uporabniško ime.



Slika 14: Stran za prijavo



Slika 15: Začetna stran po registraciji

Datoteke uporabljene na strani:

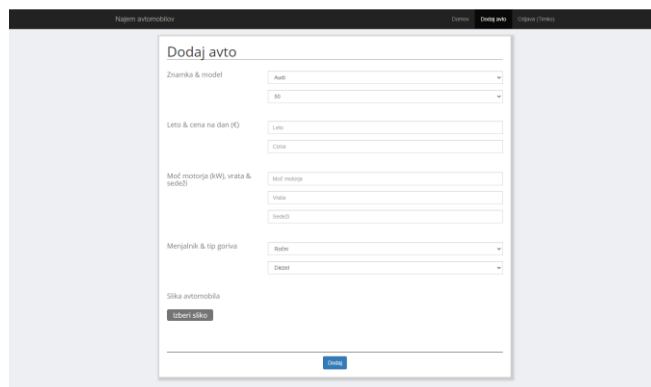
- `login.php`
- `LoginAsset.php` (koda v tej datoteki strani doda css)
- `SiteController.php` (funkcija `actionLogin`)
- `LoginForm.php` (model za shranjevanje in verifikacijo podatkov)
- `login.css`

Delovanje programske kode za prijavo v aplikacijo

Ob pritisku na gumb prijava se izvede funkcija `actionLogin`, ki se kot pri vseh predhodnih straneh nahaja v datoteki `SiteController.php`. Funkcija najprej preveri ali je morda uporabnik že prijavljen in ga v tem primeru preusmeri nazaj na začetno stran. Drugače pa funkcija ustvari nov objekt razreda `LoginForm` (19), nato pa v ta objekt naloži podatke iz obrazca in izvede funkcijo `login` (19) (ki se nahaja v `LoginForm` (19) razredu), ki uporabnika prijavi v aplikacijo za 24 ur, nato pa ga avtomatično izpiše (uporabnik se lahko tudi sam izpiše). Če nalaganje podatkov v model ali funkcija za prijavo nista izvedena uspešno potem nas aplikacija vrne na prijavno stran kjer se lahko ponovno prijavimo, v primeru uspešne prijave pa nas aplikacija napoti na začetno stran, kjer imamo dostop do vseh delov aplikacije.

2.2.6 Podstran za dodajanje avtomobilov

Ko smo se v aplikacijo uspešno registrirali in prijavili imamo na voljo dostop do nove spletne strani za dodajanje avtomobilov (v vrstici za dostop do podstrani na vrhu spletne strani), katere si bodo lahko kasneje drugi uporabniki ogledali in po možnosti tudi rezervirali. Kot lahko vidimo na spodnji fotografiji moramo za dodajanje novega avtomobila dodati kar nekaj podatkov o avtomobilu. Izbrati moramo znamko iz seznama in enega izmed modelov, ki pripadajo tej znamki. Vnesti moramo tudi podatke o letu prve registracije, ceni avtomobila na dan, moči motorja v kW, vratih in sedežih. Vrsta menjalnika in tip goriva sta že vnaprej izbrana, vendar ju lahko uporabnik po potrebi spremeni. Kot zadnjo možnost pa lahko uporabnik priloži še sliko avtomobila, kar je zelo uporabno, saj uporabniki želijo videti, kako avtomobil izgleda preden ga najamejo.



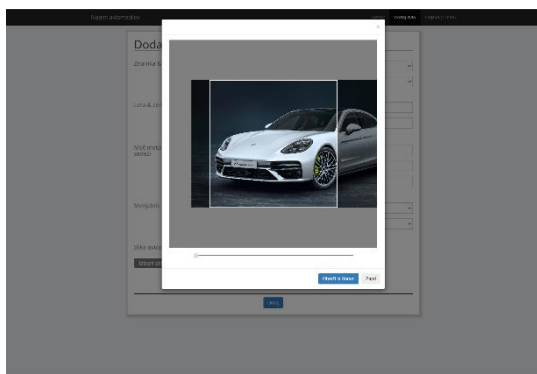
Slika 16: Slika strani za dodajanje avtomobilov

Datoteke uporabljene na strani:

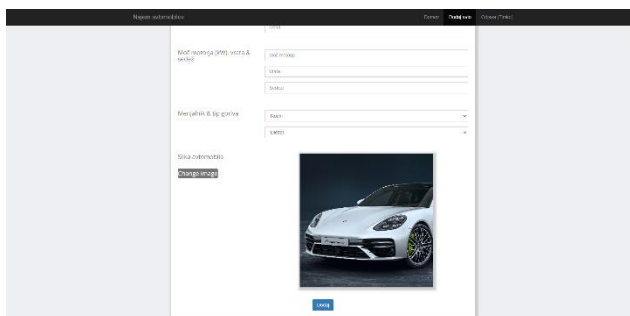
- *add_cars.php* (pogled z HTML in PHP kodo)
- *Add_new_car.php* (model, v katerega shranjujemo podatke in jih obdelujemo)
- *Add_cars.php* (datoteka v kateri so definirane CSS in JS datoteke)
- *SiteController.php* (funkcija *actionAdd_cars*)
- *add_cars.php*
- *croppie.css* (datoteka za izgled, ki pripada Croppie (20) orodju)
- *croppie.js* (datoteka z JS kodo, ki pripada Croppie (20) orodju)
- *add_cars.js*
- *TempPhotos.php*

Pošiljanje in shranjevanje slike na strežnik

Če želimo, da se pri prikazu avtomobila pokaže tudi njegova fotografija jo moramo pri izpolnjevanju podatkov o avtomobilu podati aplikaciji. To storimo tako, da pritismo na gumb izberi fotografijo, kar nam odpre meni na računalniku, kjer izberemo zeleno fotografijo. Po izbrani fotografiji se nam v aplikaciji pokaže orodje za obrezovanje fotografije. Orodje je zgrajeno s pomočjo Croppie (20) orodja za obrezovanje fotografij v JavaScriptu, pri programiranju te rešitve pa mi je zelo prav prišel članek Crop Image with Croppie (21), kjer je avtor izdeloval enako programsko rešitev kakor jaz. Ko uporabnik postavi sliko tako kot mu je všeč prisne na gumb obreži in shrani. To sproži JavaScript funkcijo, ki se nahaja v *add_cars.php*, katera sliko shrani v objekt razreda *FormData*, in ga s pomočjo **Napaka! Vira sklicevanja ni bilo mogoče najti.** tehnologije pošlje na spletni strežnik (to stori funkcija *ajaxFormPost*, ki se nahaja v *add_cars.php*). Na spletnem strežniku prejete podatke obdela funkcija *actionAdd_cars* (nahaja se v *SiteControler.php* datoteki), ki sliko shrani v datoteko, kjer so shranjene začasne slike (to stori s pomočjo funkcij *generateRandomString* in *savingToTemporary*, ki pripadata razredu *Add_new_car*), brskalniku pa vrne obrezano sliko (vrnemo jo s pomočjo **Napaka! Vira sklicevanja ni bilo mogoče najti.** tehnologije), da jo lahko le ta prikaže, to nam prikazuje Slika 18.



Slika 17: Obrezovanje fotografije



Slika 18: Prikaz slike ko jo shranimo v bazo

Shranjevanje podatkov v podatkovno bazo

Ko smo izpolnili celoten obrazec z pravilnimi podatki in izbrali fotografijo našega avtomobila lahko vse podatke pošljemo na strežnik, da jih shrani v podatkovno bazo. To storimo z pritiskom na gumb pošlji, ki sproži funkcijo *actionAdd_cars*, v kateri ustvari objekt razreda *Add_new_car* v katerega naloži podatke iz obrazca. Podatke nato shrani v podatkovno bazo, sliko avtomobila pa prestavi iz datoteke za začasne fotografije v datoteko s končnimi fotografijami. To stori s pomočjo funkcije *saveLinkToPhoto*, ki se nahaja v razredu *Add_new_car*, ki sliko prestavi iz ene datoteke v drugo, ime slike pa shrani v podatkovno bazo. Ko funkcija vse podatke uspešno shrani v bazo nam ponovno naloži spletno stran za dodajanje avtomobilov, obrazec pa je prazen in vanj lahko vnesemo nove podatke.

3 Zaključek

Ob izdelavi tega projekta in seminarske naloge sem se naučil veliko novega in spoznal, kako izgleda načrtovanje, planiranje in izdelava aplikacije od začetka do konca. Izvedba večjih projektov definitivno ni lahka naloga in občudujem programerje, ki jim to uspe z odliko, hkrati pa si tudi sam želim, da bi bil nekega dne sposoben voditi tako velike projekte za druga podjetja ali po možnosti tudi za svoje. Prvič sem se podrobneje poglobil v raziskovanje literature o spletnem programiranju in spoznal, da obstaja ogromno virov, katere bi lahko že predhodno uporabil pri učenju in si olajšal delo, saj so knjige in obširnejše spletne strani izredno lepo strukturirane in se iz njih ni težko učiti.

S to seminarsko nalogo sem vam želel predstaviti delovanje nekaterih aktualnih konceptov in tehnologij, ki se danes uporabljajo v spletnem programiranju. Poskušal sem razmišljati kot programer, koncepte in delovanje kode pa želel predstaviti čim bolj enostavno. Z končnim izdelkom sem zelo zadovoljen, saj mi je aplikacijo uspelo postaviti tako kot sem jo načrtoval. Seveda ima aplikacija še veliko prostora za izboljšave. Lahko bi ji dodali stran za kontaktiranje stran za predstavitev aplikacije, stran za posodabljanje profila in stran za brisanje in spreminjanje informacij avtomobila. Vendar sem z končnim produktom vseeno zelo zadovoljen, saj sem se ogromno naučil in spoznal veliko novega.

4 Zahvala

Seminarsko nalogo mi je uspelo uspešno narediti zaradi spodbude in pomoči določenih ljudi. Največja zahvala gre mojemu očetu in materi, ki me že od nekdaj spremljata na moji poti in me pri tem podpirata in spodbujata. Rad bi se zahvalil tudi mojemu mentorju, ki mi je pomagal z nasveti o izdelavi seminarske naloge in mi podal veliko znanja o računalništvu skozi vsa štiri leta obiskovanja gimnazije, ter svojim sošolcem, ki so mi priskočili na pomoč z uporabnimi nasveti kako izboljšati spletno stran in aplikacijo. Velika zahvala pa gre tudi podjetju Mikrografija in njihovi ekipi programerjev, ki so mi omogočili opravljati študentsko delo med poletnimi počitnicami, poleg tega pa mi podali ogromno znanja o programiranju v gospodarstvu in me podučili o tehnologijah, ki sem jih uporabil v tej aplikaciji.

Tim Rezelj

5 Bibliografija

1. Wikipedia. [Elektronski] 9. 3 2021. [Navedeno: 9. 3 2021.] <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>. ISBN 123456789.
2. Luputan. *luputan.org*. [Elektronski] 8. 7 2012. [Navedeno: 9. 3 2021.] <http://www.luputan.org/pub/papers/POSA-MVC.pdf>. ISBN 123456789.
3. Wikipedia. Model–view–controller. *Wikipedia*. [Elektronski] Wikipedia, 8. april 2021. [Navedeno: 13. april 2021.] <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>.
4. Tutorialspoint. Yii Tutorial . *Tutorialspoint*. [Elektronski] tutorialspoint, 2021. [Navedeno: 14. marec 2021.] <https://www.tutorialspoint.com/yii/index.htm>. ISBN 123456789.
5. w3schools. PHP Introduction. *w3schools*. [Elektronski] W3Schools, 1999-2021. [Navedeno: 22. marec 2021.] https://www.w3schools.com/php/php_intro.asp.
6. eNSA. Uvod v PHP. *eNSA*. [Elektronski] eNSA, 2014-2021. [Navedeno: 22. marec 2021.] <https://nsa-splet.si/php/uvod/php-uvod-01.php>.
7. Hope, Computer. HTML. *Computer Hope*. [Elektronski] Computer Hope, 1998-2021. [Navedeno: 22. marec 2021.] <https://www.computerhope.com/jargon/h/html.htm>.
8. Mozilla. CSS: Cascading Style Sheets. *MDN Web Docs*. [Elektronski] Mozilla, 2005-2021. [Navedeno: 22. marec 2021.] <https://developer.mozilla.org/en-US/docs/Web/CSS>.
9. Rauschmayer, Axel. *JavaScript for impatient programmers*. s.l. : Samostojni založnik, 2021. ISBN 978-1-09-121009-7.
10. Reactor, Hack. What is JavaScript Used For? *Hack Reactor*. [Elektronski] Galvanize, 18. oktober 2018. [Navedeno: 22. marec 2021.] <https://www.hackreactor.com/blog/what-is-javascript-used-for>.
11. Duckett, John. *JAVASCRIPT & JQUERY*. Indianapolis : John Wiley & Sons, inc., 2014. ISBN: 978-1-118-53164-8.
12. TutorialRepublic. JavaScript Ajax. *TutorialRepublic*. [Elektronski] TutorialRepublic, 2021. [Navedeno: 7. april 2021.] <https://www.tutorialrepublic.com/javascript-tutorial/javascript-ajax.php>.
13. XAMPP Tutorial. *Javatpoint*. [Elektronski] JavaTpoint, 2011-2018. [Navedeno: 16. marec 2021.] <https://www.javatpoint.com/xampp>.
14. Foundation, Apache Software. About Apache. *Apache*. [Elektronski] The Apache Software Foundation, 1997-2020. [Navedeno: 14. marec 2021.] https://httpd.apache.org/ABOUT_APACHE.html. ISBN.
15. Apache HTTP Server. *Wikipedia*. [Elektronski] Wikipedia, 3. april 2021. [Navedeno: 13. april 2021.] https://en.wikipedia.org/wiki/Apache_HTTP_Server.

16. Herawan. What is MySQL: MySQL Explained For Beginners. *Hostinger*. [Elektronski] Hostinger, 4. marec 2020. [Navedeno: 16. marec 2021.] <https://www.hostinger.com/tutorials/what-is-mysql>.
17. Yii. Class yii\web\AssetBundle. *yiiframework*. [Elektronski] Yii, 2008-2021. [Navedeno: 17. 3 2021.] <https://www.yiiframework.com/doc/api/2.0/yii-web-assetbundle>.
18. —. Active Record. *yiiframework*. [Elektronski] Yii, 2008-2021. [Navedeno: 17. 3 2021.] <https://www.yiiframework.com/doc/guide/2.0/en/db-active-record>.
19. —. Class frontend\models\SignupForm. *yiiframework*. [Elektronski] Yii, 2008-2021. [Navedeno: 19. marec 2021.] <https://www.yiiframework.com/extension/yiisoft/yii2-app-advanced/doc/api/2.0/frontend-models-signupform>.
20. —. Class yii\base\Model. *yiiframework*. [Elektronski] Yii, 2008-2021. [Navedeno: 19. marec 2021.] <https://www.yiiframework.com/doc/api/2.0/yii-base-model>.
21. —. Class yii\web\User. *yiiframework*. [Elektronski] Yii, 2008-2021. [Navedeno: 19. marec 2021.] <https://www.yiiframework.com/doc/api/2.0/yii-web-user>.
22. —. Class common\models\LoginForm. *yiiframework*. [Elektronski] Yii, 2008-2021. [Navedeno: 21. marec 2021.] <https://www.yiiframework.com/extension/yiisoft/yii2-app-advanced/doc/api/2.0/common-models-loginform>.
23. Smith, Dustin. Croppie - A Javascript Image Cropper. *GitHub*. [Elektronski] 16. junij 2020. [Navedeno: 21. marec 2021.] <https://github.com/Foliotek/Croppie>.
24. Hassan, Sifat. Crop Image with Croppie. *Medium*. [Elektronski] Analytics Vidhya, 29. junij 2020. [Navedeno: 21. marec 2021.] <https://medium.com/analytics-vidhya/crop-image-with-croppie-26146a4b933>.
25. Yii. yiiframework. *Class yii\widgets\ActiveForm*. [Elektronski] Yii, 2008-2021. [Navedeno: 18. marec 2021.] <https://www.yiiframework.com/doc/api/2.0/yii-widgets-activeform>.

6 Priloge

Priloga 1: index.php

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/views/site/index.php>

Priloga 2: index.css

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/css/index.css>

Priloga 3: IndexAsset.php

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/assets/IndexAsset.php>

Priloga 4: Cars.php

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/models/Cars.php>

Priloga 5: SiteController.php

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/controllers/SiteController.php>

Priloga 6: Index.js

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/js/index.js>

Priloga 7: display_cars.php

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/views/site/display_cars.php

Priloga 8: display_cars.css

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/css/display_cars.css

Priloga 9: display_carsAsset.php

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/assets/display_carsAsset.php

Priloga 10: car_info.php (datoteka z razredom)

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/models/car_info.php

Priloga 11: car_info.php (datoteka z HTML in PHP kodo)

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/views/site/car_info.php

Priloga 12: car_info.css

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/css/car_info.css

Priloga 13: car_infoAsset.php

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/assets/car_infoAsset.php

Priloga 14: BookingHistory.php

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/models/BookingHistory.php>

Priloga 15: *signup.php*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/views/site/signup.php>

Priloga 16: *SignupForm.php*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/models/SignupForm.php>

Priloga 17: *User.php*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/common/models/User.php>

Priloga 18: *SignupAsset.php*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/assets/SignupAsset.php>

Priloga 19: *signup.css*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/css/signup.css>

Priloga 20: *signup.js*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/js/signup.js>

Priloga 21: *login.php*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/views/site/login.php>

Priloga 22: *LoginAsset.php*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/assets/LoginAsset.php>

Priloga 23: *LoginForm.php*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/common/models/LoginForm.php>

Priloga 24: *login.css*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/css/login.css>

Priloga 25: *add_cars.php*

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/views/site/add_cars.php

Priloga 26: *Add_new_car.php*

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/models/Add_new_car.php

Priloga 27: *Add_cars.php*

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/assets/Add_cars.php

Priloga 28: *TempPhotos.php*

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/models/TempPhotos.php>

Priloga 29: *add_cars.css*

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/css/add_cars.css

Priloga 30: croppie.css

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/css/croppie.css>

Priloga 31: croppie.js

<https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/js/croppie.js>

Priloga 32: add_cars.js

https://github.com/Timcek/Najem-avtomobilov/blob/master/frontend/web/js/add_cars.js

Priloga 33: PDF seminarske naloge

Priloga 34: Power Point seminarske naloge

<https://github.com/Timcek/Najem-avtomobilov/blob/master/seminarska/Aplikacije%20in%20informacijski%20sistemi%20%E2%80%93%20Najem%20avtomobilov.odp>