Sustav pomoći pri kupnji računala

Szabó, Erik Valentin

Undergraduate thesis / Završni rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:914155

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-19**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij

Sustav pomoći pri kupnji računala

Završni rad

Erik Valentin Szabó

Osijek, 2017.

SADRŽAJ

1. UVOD ... 1
1.1. Zadatak završnog rada ... 1
1.2. Koncept određivanja razvojnog toka aplikacije ... 2
2. KONFIGURIRANJE RAČUNALA ... 3
2.1. Komponente osobnog računala bitne za odabir konfiguracije ... 4
2.1.1. Procesor ... 4
2.1.2. Matična ploča ... 5
2.1.3. Radna memorija ... 5
2.1.4. Grafička kartica ... 6
2.1.5. Napajanje ... 7
2.2. Uvjeti kompatibilnosti između komponenti ... 8
2.2.1. Matična ploča i procesor ... 8
2.2.2. Matična ploča i radna memorija ... 8
2.2.3. Matična ploča i kućište ... 8
2.2.4. Kućište i grafička kartica ... 8
2.2.5. Napajanje i potrošnja grafičke kartice i procesora ... 8
3. PROCES RAZVOJA APLIKACIJE „Sustav pomoći pri kupnji računala“ ... 9
3.1. Tehnologije korištene za izradu aplikacije ... 9
3.1.1. HTML i CSS ... 9
3.1.2. Javascript i jQuery ... 10
3.1.3. Bootstrap ... 10
3.1.4. MySQL ... 11
3.1.5. PHP ... 11
3.1.6. Laravel ... 11
3.2. Scrum ... 12
3.2.1. Scrum tim ... 13
3.2.2. Događaji u Scrumu ... 13
3.2.3. Artefakti u Scrumu ... 14
3.3. Proces izrade aplikacije ... 14
3.3.1. Sprint 1 ... 15
3.3.2. Sprint 2 ... 18
3.3.3. Sprint 3 ... 19
3.3.4. Sprint 4 ... 20
3.3.5. Sprint 5 ... 22

3.3.6. Sprint 6 23
3.3.7. Sprint 7 25
3.3.8. Sprint 8 25
3.3.9. Sprint 9 26
3.4. Osvrt na aplikaciju 26
4. ZAKLJUČAK 28
5. LITERATURA 29
SAŽETAK 30
ABSTRACT 31
ŽIVOTOPIS 32
PRILOZI 33

1. UVOD

Samostalno sastavljanje konfiguracije računala gledajući unazad par godina, postaje popularan izbor pri kupnji računala. Korisnik ima slobodu izbora komponenti po njegovim potrebama. Tema ovog rada je olakšati korisniku sastavljanje konfiguracije računala prema njegovima potrebama stvaranjem aplikacije pomoću koje korisnik može sastaviti kompatibilno računalo bez prethodnog znanja o sastavljaju konfiguracije računala. Komponente je moguće odabrati po određenim kategorijama računala ili sve komponente nevezane za kategoriju. Provjera kompatibilnosti postavljena je imajući na umu konstantan tehnološki napredak te se izvršava prije ispisa komponenti na način da se međusobno uspoređuju karakteristike pojedinih komponenti definirane uvjetima kompatibilnosti opisanih u radu. Stvaranjem aplikacije dobiva se radni okvir koji olakšava provjeru kompatibilnosti određenih komponenti i može poslužiti kao koristan alat koji skraćuje vrijeme potrebno za istraživanjem kompatibilnosti komponenti računala. Rad se sastoji od devet poglavlja. U prvom poglavlju nalazi se uvod, kratki opis zadatak završnog rada i koncept određivanja razvojnog toka aplikacije. Drugim poglavljem obuhvaćena je teorijska podloga konfiguriranja računala i upoznavanja pojedinih komponenti bitnih za odabir konfiguracije računala, te opis uvjeta kompatibilnosti komponenti. U trećem poglavlju navedene su i opisane tehnologije korištene pri izradi aplikacije, radni okvir Scrum, sam proces izrade aplikacije te osvrt na aplikaciju gdje su opisane prednosti i nedostatci aplikacije. Četvrto poglavlje objedinjuje zaključke.

1.1. Zadatak završnog rada

Zadatak ovog rada je izrada aplikacije pod nazivom „Sustav pomoći pri kupnji računala“. Aplikacija treba imati mogućnosti odabira komponenti i provjeru kompatibilnosti komponenti uz ispis cijene same komponente te pružanje informacije o pravilnom postavljanju pojedine komponente. Aplikacija provjerava kompatibilnost odabranih komponenti i adekvatno ishodu uvjetima kompatibilnosti daje informaciju jesu li odabrane komponente kompatibilne ili nisu, te razlog zbog čega komponente nisu kompatibilne. Uz mogućnost sastavljanja kompatibilne konfiguracije računala aplikacija pruža informaciju o pravilnom postavljanju pojedinih komponenti.

1.2. Koncept određivanja razvojnog toka aplikacije

Nakon istraživanja, s obzirom na današnju dostupnu količinu i raspon aplikacija uočeno je da je konceptna ideja stvaranja aplikacije pomoću koje korisnik odabirom komponenti sastavlja kompatibilno računalo već izrađena. Uz prilagodbe zahtjeva aplikacije i samog koncepta sastavljanja komponenti ostvarena je unikatnost projekta. Istraživanjem raznih internetskih stranica čije su funkcije sastavljanje računalnih konfiguracija uočena je potreba za bržim i slobodnijim rješenjem od tad već postojećih. Glavna funkcionalnost projekta je brza provjera odabiranih komponenti i ispis njihove kompatibilnosti. Uz daljnje istraživanje internetskih stranica vezanih uz komponente računala zaključena je polazišna točka razvoja same aplikacije, utvrđene su osnovne smjernice ispitivanja kompatibilnosti komponenti. Zatim je odabran Scrum radni okvir zbog jednostavnosti i prilagodljivosti projektu. Svojstvenost aplikacije leži u tome kako korisnik ima slobodu izbora iz spektra raznih komponenti, za razliku od aplikacije iste namjene. Slične aplikacije poput ove su ChooseMyPC.net, pcpartpicker.com, assemblio.hr, pugetsystems.com. U tablici (tablica 1.1) prikazan je kratki opis sličnih aplikacijskih rješenja.

Tablica 1.1. Aplikacije pomoću kojih korisnik sastavlja osobno računalo

Naziv/web lokacija	Opis	Prednosti	Nedostatci
ChooseMyPC.net	odabirom države i unosom dostupnog novca aplikacija automatski generira konfiguraciju računala	jednostavno za korištenje, mogućnost odabira države	nemogućnost odabira određenih komponenti, ne raznolikost konfiguracija
pcpartpicker.com	odabirom komponenti sasta se kompatibilno računalo	prilikom odabira komponenti ponuđene su samo kompatibilne komponente	-
assemblio.hr	odabirom kategorije i unosom dostupnom novca aplikacija automatski generira konfiguraciju računala	vrlo jednostavan za uporabu, informativan	ako se određena komponenta konfiguracije želi zamijeniti može doći do nekompatibilnosti
pugetsystems.com	aplikacija namijenjena sastavljanju serverskih konfiguracija	odlična korisnička podrška	sužen izbor konfiguracija

2. KONFIGURIRANJE RAČUNALA

U današnje doba tehničke razvijenosti računalo se smatra neophodnim uređajem u kućanstvima, uredima, postrojenjima, obrazovnim ustanovama, zdravstvu, vojsci, prijevozu putnika i stvari te u još mnogo raznovrsnih grana. Računalo je elektronički uređaj pomoću kojeg korisnik obavlja razne funkcionalnosti nad unesenim podatcima i/ili njihovim ispisom.

Prema potrebi obavljanja određenih funkcija računala mogu se generalizirati u kategorije:

- kućna i uredska računala – hardverska podloga je osnovna, računala su u mogućnosti izvoditi osnovne funkcije: pokretati aplikacije za obradu teksta, slika, za slušanje glazbe, pregledavati video zapise, Internet komunikacija,
- računala za igranje računalnih igara – računala su hardverski u mogućnosti obraditi velike količine grafičke interakcije, prikaza i podataka. Glavna namjena im je pokretanje zahtjevnih računalnih igara i održati stabilnost njihovog izvođenja,
- studijska ili *stream* računala – računala su u mogućnosti obraditi veliku količinu podataka u vrlo kratkom vremenskom periodu. Osnovni zahtjev na takvo računalo je hardverska podloga koja osigurava veliku brzinu obrade podataka i pruža mogućnost njihovog prijenosa u stvarnom vremenu sa što manjom latentnosti,
- poslužiteljska računala – računala koja se koriste za obradu ogromnih količina podataka u vrlo kratkom do neprimjetnom vremenu. Zahtjevi na takva računala su konzistentnost, performanse i sigurnost sustava,
- tablet ili laptop računala – prenosiva računala koja su u mogućnosti izvoditi sve osnovne funkcije.

Računalo sastoji se od skupa međusobno povezanih komponenti smještenih u kućištu računala, to su: procesor, matična ploča, radna memorija, grafička kartica, jedinica za pohranu podataka, napajanje te periferni uređaji, poput tipkovnice, miša i monitora.

2.1. Komponente osobnog računala bitne za odabir konfiguracije

Konfiguracija računala predstavlja funkcionalnu cjelinu njegovih sastavnih komponenti. Konfiguracija računala sastavlja se u odnosu na predviđen način njegove upotrebe imajući na umu međusobnu kompatibilnost komponenti kako bi se dobila funkcionalna cjelina. Glavne komponente računala su procesor, matična ploča, radna memorija, grafička kartica, jedinica za pohranu podataka i napajanje. U ovom potpoglavlju opisane su pojedine komponente i njihove karakteristike koje utječu na kompatibilnost s ostalim komponentama. Jedinica za pohranu podataka nije navedena jer su generalno svi čvrsti diskovi (eng. *Hard disk drive*) kompatibilni sa svim matičnim pločama.

TDP – (eng. *thermal design power*) prikazuje maksimalnu količinu topline proizvedenu od strane računalne komponente, mjeri se u vatima . Vat je mjerna jedinica za iskazivanje energije u jedinici vremena [J/s]. TDP je količina disipacije snage koja se pretvara u toplinu i služi kao pokazatelj nominalne vrijednosti pri izradi sustava hlađenja. U tekstu se TDP podrazumijeva kao potrošena snaga zbog konzistentnosti pojmova i jasnoće usporedbe komponenti. Prema [1] Procesor i grafička kartica su dvije komponente s najvećom potrošnjom električne energije računala, potrošnja ostalih komponenti svodi se na svega 30% ukupne potrošnje. Dakle, kako bi se računalo opskrbilo s dovoljnom količinom električne energije pri uvjetu usporedbe potrebne nazivne snage napajanja i TDP grafičke kartice i procesora koristio se faktor 1,3.

2.1.1. Procesor

Slika 2.1 Procesor

Procesor (eng. *central processing unit*, CPU) glavni je dio računala koji, vođen zadanim programskim naredbama, izvodi osnovne radnje nad podacima i upravlja tokom podataka među dijelovima računala, što su ujedno glavna funkcije procesora. Procesori se razliku po proizvođačima, generacijama, takovima rada i po priključku na matičnu ploču (eng. *socket*). Najčešći proizvođači su tvrtke AMD i Intel. U tablici (tablica 2.1.) nalazi se informativni prikaz nekoliko trenutno najčešće korištenih procesora. Taktovi rada označuju koliko naredbi je moguće obraditi u jedinici vremena, mjere se u gigahercima (GHz). Priključak na matičnu ploču je utor procesora pomoću kojeg se ugrađuje na kompatibilnu matičnu ploču. Performanse procesora određene su radnim taktom, brojem jezgri i veličinom pričuvne memorije. AMD procesori zahtijevaju jače napajanje u smislu opskrbe

električne energije jer imaju tendenciju velike potrošnje, dok su Intel procesori optimizirani potrošači.

Tablica 2.1. Nekoliko najčešće korištenih procesora

AMD	FX X4 4320	FX X6 6300	FX X4 7600	FX X8 8350	Ryzen 5 1500
Intel	i3 7100	i3 7350K	i5 4460K	i5 7400	i7 6700K

2.1.2. Matična ploča

Slika 2.2. Matična ploča

Matična ploča je centralni dio sklopovlja u računalu. Na nju se priključuju ostale komponente računala: procesor, radna memorija, grafička kartica, jedinica za pohranu podataka, optički čitač, te se na nju napajanjem dovodi električna energija. Matične ploče razlikuju se po priključku procesora, dimenziji i ugrađenim tehnologijama. Najčešće dimenzije matičnih ploča su mATX i ATX što je ujedno bitna karakteristika pri kompatibilnosti matične ploče i kućišta računala. Matične ploče podržavaju jedan tip i nekoliko frekvencija radne memorije. U tablici (tablica 2.2.) prikazano je nekoliko najčešće korištenih matičnih ploča.

Tablica 1.2. Nekoliko najčešće korištenih matičnih ploča

ASROCK 970 Extreme R2.0	ASUS A88XM-A	ASUS B250M	ASUS Z170	GIGABYTE GA-AX370 Gaming k7	ASROCK Z270 Gaming Pro

2.1.3. Radna memorija

Slika 2.3. Radna memorija

Memorija s nasumičnim pristupom, tj. radna memorija (eng. *random access memory*, RAM) oblik je primarne memorije računala koja omogućuje upisivanje i čitanje podataka u koju se upisuju aktivni programi, te informacije potrebne za rad računala. Prilikom prekida napajanja podatci iz radne memorije se brišu, za razliku od ROM-a (eng. *read-only memory*, memorija samo za očitavanje) u koju su trajno zapisani podatci i ne brišu se prilikom prekida napajanja. Kapacitet

radne memorije ukazuje na maksimalnu količinu memorijskog prostora raspoloživog za upis i čitanje podataka i programa. Radna memorija kategorizira se po tipu, tj. tehnologiji izrade. Trenutno najzastupljeniji tipovi radnih memorija su DDR3 i DDR4, prikaz nekoliko najčešće korištenih radnih memorija nalazi se u tablici (tablica 2.3.). Razlikuju se po brzini, odnosno frekvencijskom opsegu. DDR3 radne memorije troše 15 W više od DDR4 radnih memorija, što nije toliko bitno kod standardnih računala nego se ta razlika osjetno primjećuje kod poslužiteljskih računala.

Tablica 2.3. Nekoliko poznatijih modela radne memorije

G.Skill Ripjaws 4GB DDR3 1333MHz	G.Skill Ripjaws X serija 8GB DDR3 1600MHz	Kingston Savage 8GB DDR3 2400MHz	G.Skill Ripjaws V serija 8GB DDR4 2400MHz	Kingston HyperX Predator 16GB DDR4 3000MHz

2.1.4. Grafička kartica

Slika 2.4. Grafička kartica

Grafička kartica je komponenta računala koja služi za prikazivanje slike na zaslonu monitora. Tehnički gledano grafička kartica je podsustav koji se sastoji se od grafičkog procesora (GPU – engl. *Graphic procesing unit*), video memorije (VRAM – engl. *Video random access memory*), izlaznih priključaka na koje se spaja monitor i priključka pomoću kojeg se postavlja u utor na matičnoj ploči uglavnom PCI-E-a (eng. PCI Express). Najčešći proizvođači grafičkih kartica su ATI i nVidia. Informativni prikaz nekih od poznatijih grafičkih kartica vidljiv je u tablici (tablica 2.4.). Karakteristike grafičke kartice su grafički procesor, radni takt grafičkog procesora, kapacitet memorije, količina propusnosti podatkovne sabirnice, mogućnost spajanja s drugom grafičkom karticom i potrošnja električne energije. Model grafičkog procesora određuje njegov radni takt odnosno koliko podataka može obraditi u jedinici vremena. Bitne karakteristike pri sklapanju računala su fizička duljina grafičke kartice i njezin TDP.

Tablica 2.4. Neke od poznatijih grafičkih kartica

Asus Strix GTX 1050Ti 4GB DDR5	Asus GeForce GTX 1060 6GB DDR 5	Asus GeForce GTX 970 4GB DDR 5	AMD Radeon R9 370 4 GB DDR 5	AMD Radeon RX 580 8GB DDR5

2.1.5. Napajanje

Slika 2.5. Napajanje

Napajanje je hardverski dio računala koji mu osigurava napon i struju. Napajanje osigurava da svaki dio računala dobije određenu količinu energije koja mu je potrebna, s obzirom da svi dijelovi računala ne troše istu količinu električne energije. Glavni zadatak napajanja je pretvaranje izmjeničnog napona od 220 V u istosmjerni 3,3 V, 5 V i 12 V što je u skladu s naponskim zahtjevima hardvera u računalu. Ono također ima i vlastito hlađenje. Glavna karakteristika napajanja je nazivna snaga. Mjerna jedinica snage je vat (W). Napajanja se razlikuju po nazivnoj snazi i razredima učinkovitosti. Napajanje pretvara izmjeničnu struju u istosmjernu koja se šalje u sve dijelove računala, ali prilikom pretvorbe izmjenične struje u istosmjernu napajanje potroši nešto energije u tom procesu pretvorbe. Dakle, učinkovitost napajanja se svodi na to koliko struje potroši prilikom pretvorbe. Razredi učinkovitosti napajanja prikazani su na slici (slika 2.6.).

	Opterećenje	80 +	Bronca	Srebro	Zlato	Platina	Titanij
Učinkovitost	10%	-	-	-	-	-	90,00%
	20%	80,00%	81,00%	85,00%	88,00%	90,00%	94,00%
	50%	80,00%	85,00%	89,00%	92,00%	94,00%	96,00%
	100%	80,00%	81,00%	85,00%	88,00%	91,00%	91,00%

Slika 2.6. Razredi učinkovitosti napajanja

Najveći potrošači su grafička kartica i procesor. Prikaz nekih od trenutno najčešće korištenih napajanja nalazi se u tablici (tablica 2.5.).

Tablica 2.5. Neka od najčešće korištenih napajanja

Coolermaster G550M 550W	Coolermaster GX 650W	Corsair VS550 550W	EVGA 600B 600W	Thermaltake SFX 600W

2.2. Uvjeti kompatibilnosti između komponenti

Kako bi računalo funkcioniralo kao cjelina sve komponente međusobno moraju biti kompatibilne. Prema [1] slijedi opis kompatibilnosti među komponentama:

2.2.1. Matična ploča i procesor

Odabirom matične ploče određuje se njen *socket* i maksimalni dozvoljen TDP procesora. *Socket* procesora i *socket* matične ploče moraju biti isti kako bi te dvije komponente bilo kompatibilne, uz dodatni uvjet da TDP procesora mora biti manji od maksimalnog dozvoljenog TDP-a procesora matične ploče.

2.2.2. Matična ploča i radna memorija

Odabirom matične ploče određuje se podržani tip radne memorije i njenih frekvencija. Frekvencija radne memorije treba odgovarat podržanoj frekvenciji radne memorije na matičnoj ploči, ako se frekvencije ne podudaraju radna memorija će funkcionirati ali sa smanjenom performansom. Dostupni utori radne memorije na matičnoj ploči određuju maksimalnu proširivost radne memorije.

2.2.3. Matična ploča i kućište

Odabirom matične ploče određuje se njen format. Format matične ploče i format koji kućište podržava trebaju biti jednaki, s tim da ako odaberemo kućište čiji je podržani format matične ploče ATX a format same matične mATX ugradnja će biti moguća ali u obrnutom slučaju to nije moguće zbog fizičke veličine matične ploče.

2.2.4. Kućište i grafička kartica

Duljina grafičke kartice mora biti manja od propisane duljine grafičke kartice kućišta u suprotnom ugradnja grafičke kartice u kućište fizički neće biti moguća.

2.2.5. Napajanje i potrošnja grafičke kartice i procesora

Napajanje mora biti u mogućnosti opskrbljivati dovoljno električne energije kako bi sve komponente u računalu funkcionirale. Opskrbljiva li napajanje dovoljno električne energije provjerava se formulom (2-1).

$$\textit{Snaga napajana} > 1{,}3 * (\textit{potrošnja grafičke kartice} + \textit{potrošnja procesora}) \qquad (2\text{-}1)$$

3. PROCES RAZVOJA APLIKACIJE „Sustav pomoći pri kupnji računala“

Razvoj aplikacije odvija se u nekoliko koraka. Potreba koja pokazuje uvođenje ili izradu programskog rješenja neke problematike ili zadatka ujedno je i zahtjev za izradu aplikacije. U zahtjevima izrade aplikacije točno se definiraju glavna funkcionalnost i mogućnosti aplikacije. Nakon definiranja zahtjeva provodi se planiranje izrade i izvedbe aplikacije. U koraku planiranja definiraju se tehnologije potrebne za izrazu aplikacije te potrebna hardverska podrška. Utvrđivanjem zahtjeva na aplikaciju odabire se optimalni proces razvoja aplikacije. Aplikacija je izrađena unutar *Scrum* okvira, okvira za fleksibilno izrađivanje kompleksnih aplikacijskih rješenja. Nakon izrade aplikacije, izvodi se testiranje njene funkcionalnosti. Pri izradi projekta zahtjevi se mijenjaju i upisuju u poseban dokument naziva *Product Backlog,* čiji je izgled definiran *Srcum* okvirom.

3.1. Tehnologije korištene za izradu aplikacije

Aplikacija „Sustavi pomoći pri kupnji računala“ se može podijeliti u dva glavna logička dijela, to su prikazno sučelje (eng. *Frontend)* i poslužiteljsko sučelje (eng. *Backend)*. *Frontend* dio zaslužan je za prikazivanje sadržaja aplikacije i njegovu interakciju s korisnikom, odnosno sve što je vidljivo na zaslonu monitora. Prema [2] slijedi kratki presjek korištenih tehnologija pri izradi aplikacije. *Frontend* dio izrađen je pomoću HTML-a i CSS-a, Javascript-a, jQuery-a te Bootstrap radnog okvira.

Backend dio zadužen je za upravljanje varijablama, izvršavanje funkcija i metoda poput upita nad bazom podataka, spremanje i ispis dohvaćenih podataka, provjera kompatibilnosti komponenti na osnovu usporedbe sadržaja varijabli u koje su prethodno spremljeni podatci iz baze podataka. *Backend* je onaj programski dio koji se odvija u pozadini i korisnik aplikacije nema saznanja o njemu. Poslužiteljski dio odrađen je, tj. pokrenut, lokalno pomoću skupa alata virtualnog servera XAMPP (eng. *crossplatform Apache MySQL perl/php/python*). Glavna srž *backend* programskog dijela aplikacije izvedena je pomoću PHP programskog jezika, rad nad bazom podataka ostvaren je pomoću MySQL-a. U kasnijim fazama razvoja aplikacije, cijeli koncept se prebacuje u skladu s PHP radnim okvirom, pod nazivom Laravel.

3.1.1. HTML i CSS

HTML je skraćenica od *HyperText Markup Language.* Pojam "*Hypertext Markup*" upućuje na jezik za označavanje, te mogućnost međusobnog povezivanja dokumenata poveznicama (eng.

Hyperlink). Označavanje se vrši korištenjem oznaka (eng. *tag*) kojima se stvaraju, povezuju i strukturiraju elementi HTML dokumenta. Oznake upućuju Internet preglednik na način kako će prikazati tekst koji slijedi nakon oznake. HTML datoteka mora imati ekstenziju .htm ili .html, te može biti kreirana korištenjem bilo kojeg alata za uređivanje teksta.

CSS odnosno kaskadni stilovi (eng. *Cascading Style Sheets*), jednostavan su mehanizam za dodavanje stilova u HTML datoteku. Mogućnosti formatiranja HTML-a poprilično su ograničene, zato su ove dvije tehnologije usko povezane i često dolaze u zajedničkom kontekstu. Dizajn izgleda stranice u HTML-u ograničen je na tablice, kontrolu fontova te nekoliko stilova teksta poput podebljavanja i podvlačenja. Pomoću CSS-a definiraju se pravila u stilskom obrascu koji određuje kako će biti prikazan kod pisan u HTML datoteci. Korištenjem CSS obrasca mogu se kontrolirati bilo koji segmenti web stranice.

3.1.2. Javascript i jQuery

Javascript je skriptni programski jezik, koji se izvršava u web pregledniku na strani korisnika. Priključenjem Javascript-a u web stranicu omogućuje se određivanje načina ponašanja stranice odnosno uvodi se dinamičnost web stranice. Jedan je od najpopularnijih skriptnih jezika kojeg podržavaju svi poznati web preglednici (Chrome, Firefox, Safari, Opera). jQuery je Javascript biblioteka koja omogućuje bržu i lakšu izradu dinamičnosti web stranice. Veća efikasnost pisanja nad običnim Javascript, jedna linija koda pisana u jQuery-ju može biti jednaka 20 linija koda u Javascriptu. Dinamičnosti karakteristične za jQuery:

- dodavanje efekata tekstu (nestajanje, ulaz/izlaz teksta, nestajanje teksta),
- upravljanje nad DOM-om (eng. *Docuemnt object model*),
- animirani prikaz slika,
- upozorenja o neispravno upisanim podatcima HTML forme.

3.1.3. Bootstrap

Bootstrap je besplatni programski okvir. Opisan u [3] kao takav koristi se kombinacijom HTML-a, CSS-a i Javascript-a, razvijen s ciljem olakšavanja razvoja web formi, naprednih web komponenti te osigurava responzivni izgled web stranice. Sastoji se od skupa razvijenih CSS i Javaskript alata i biblioteka. Olakšava integriranje raznih vrsta komponenti (html forme, tipke, promjena teksta) i omogućuje izradu predložaka izgleda stranice, te rad s tipografijom, umetanje različitih font ikonica unutar teksta. Zašto koristiti Bootstrap:

- olakšava i ubrzava izradu aplikacija,
- omogućuje respozivni dizajn mobilnih aplikacija,
- identično se prikazuje u različitim preglednicima,
- jednostavan za uporabu i besplatan,
- omogućuje uređivanje izgleda po individualnoj mjeri.

3.1.4. MySQL

MySQL besplatan je sustav za upravljanje bazom podataka (eng. *DBMS – database managment system*). Vrlo je stabilan, zadovoljavajućih performansi, jednostavan za upotrebu i ima veliku podršku programskih jezika (PHP, Java, Python). Radi na principu pisanja upita (eng. *query*) uporabom engleskog jezika u svakodnevnoj uporabi. Korišten je u sklopu XAMPP koji mu omogućuje korištenje grafičkog sučelja pod nazivom *phpmyadmin*. Jednostavnost grafičke izrade baze podataka olakšava i ubrzava proces same izrade baze podataka. Pri razvoju aplikacije MySQL je korišten u integraciji s PHP programskim jezikom.

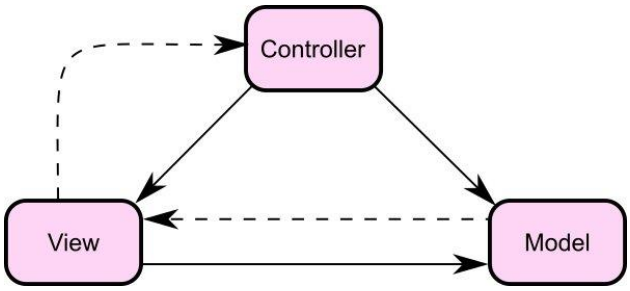
3.1.5. PHP

PHP (eng. *PHP*: *Hypertext Procesor*) popularan je besplatni skriptni jezik namijenjen razvoju web aplikacija. Lako se implementira u HTML i u toj simbiozi se kreira dinamična web stranica. Za razliku od Javascripta, PHP se izvršava na poslužitelju i kao rezultat šalje HTML kod u korisnikov preglednik. PHP je zamišljen kao skriptni jezik, što znači da ima mogućnost prikupljanja podataka iz obrazaca, dinamično generiranje sadržaja stranice te slanje i primanje kolačića (eng. *cookies*). PHP nije ograničen samo na generiranje HTML-a već se pomoću njega mogu generirati slike, PDF i Flash datoteke koje se mogu ali ne moraju odmah prikazati već se spremaju na poslužitelju. Najznačajnija mogućnost PHP-a je podrška za različite baze podataka. Bitno je spomenuti i podršku PHP-a za komuniciranje s većinom Internet servisa poput LDAP-a, IMAP-a, POP3-a, HTTP-a, te mogućnost otvaranja bilo kojeg mrežnog *socketa* što omogućuje komunikaciju bilo kojim protokolom.

3.1.6. Laravel

Prema [4] Laravel je besplatni PHP programski okvir. Vrlo je stabilan i pogodan za početnike i napredne programere zbog svoje učinkovitosti toka razvoja aplikacija. Zasniva se na konceptu pogleda, modela i upravljača (eng. *Model – View – Controller*, MVC), prikazan slikom (slika 3.1.). MVC je obrazac softverske arhitekture. Koristi se u softverskom inženjeringu za odvajanje

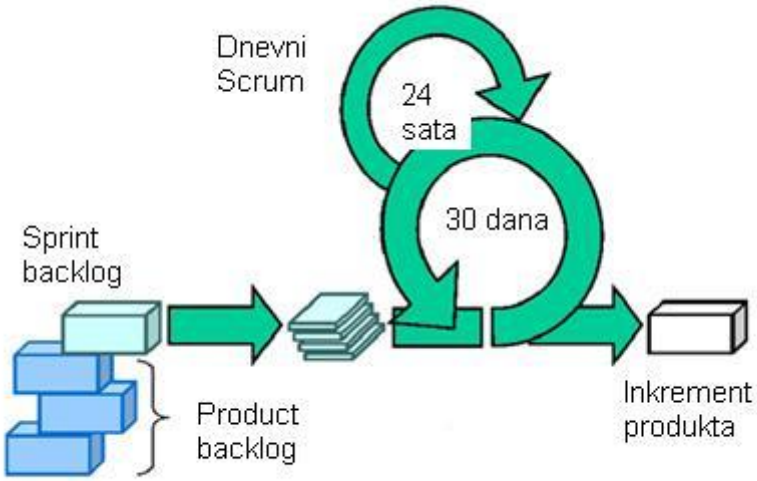
pojedinih dijelova aplikacije u komponente ovisno o njihovoj namjeni. Model se sastoji od podataka, poslovnih pravila, logike, i funkcija ugrađenih u programsku logiku. Pogled (eng *View*) je bilo kakav prikaz podataka kao što je obrazac, tablica ili dijagram. Moguć je prikaz podataka kroz više različitih pogleda. Upravljač (eng *Controller*) prihvaća ulazne naputke (eng *input*) i pretvara ih u naloge modelu ili pogledu. Ovakva arhitektura olakšava nezavisan razvoj, testiranje i održavanje određene aplikacije.



Slika 3.1. MVC koncept

3.2. Scrum

Scrum je procesni radni okvir koji se koristi za upravljanje složenim razvojem aplikacija. *Scrum* nije metodologija ili tehnika razvoja softvera , nego je okvir unutar kojega se mogu koristiti razni procesi i tehnike (npr. ekstremno programiranje). *Scrum* se sastoji od *Scrum* timova, njihovih pridruženih uloga, događaja, artefakata i pravila.



Slika 3.2. Životni ciklus Scum proizvoda

3.2.1. Scrum tim

Scrum tim sastoji se od vlasnika proizvoda (eng. *Product Owner*), razvojnog tima (eng. *Development team*) i *Scrum mastera*. *Scrum* isporučuje proizvode iterativno i inkrementalno, maksimirajući moguće povratne informacije.

Vlasnik proizvoda je odgovoran za maksimizaciju vrijednosti proizvoda i rada razvojnog tima. Načina na koji se postiže maksimizacija vrijednosti vrlo je različit između raznih organizacija, timova i ljudi. Vlasnik proizvoda je jedini odgovoran za upravljanje Product Backlogom. Što uključuje jasno objašnjavanje razvojnom timu vizije, ciljeve i stavke na Product Backlogu. Razvojni tim radi prema naputcima Vlasnika stoga i cijela organizacija mora poštivati njegove odluke.

Razvojni tim sastoji se od profesionalaca koji rade konkretan posao isporučujući inkrement proizvoda na kraju svakog Sprinta. Samo članovi razvojnog tima stvaraju inkrement proizvoda. *Scrum master* je odgovoran da se *Scrum* razumije i koristi. *Scrum* masteri to postižu na način da osiguravaju da se *Scrum* timovi pridržavaju teorije, prakse i pravila *Scruma*.

3.2.2. Događaji u Scrumu

Scrum koristi propisane događaje radi uspostave pravilnosti i minimizacije potrebe za sastancima koji nisu definirani *Scrumom*. *Scrum* koristi vremenski ograničene događaje na način da svaki vremenski događaj ima određeno maksimalno trajanje. Na taj način se osigurava da se dovoljno vremena koristi za planiranje bez uzaludnog trošenja vremena.

Srce *Scruma* je Sprint, vremenski ograničeni period od jednog mjeseca ili manje tijekom kojega se proizvede „završen“, upotrebljiv i potencijalno isporučiv inkrement proizvoda. Sprintovi su jednakog trajanja tijekom cijelog razvoja proizvoda. Novi Sprint započinje neposredno nakon što završi prethodni. Sprint se sastoji od sastanka za planiranje Sprinta, dnevnog *Scruma*, posla razvoja, revizije Sprinta i retrospektive Sprinta. Svaki Sprint se može smatrati projektom čiji horizont ne prelazi mjesec dana. Poput projekata Sprint se koristi da se obavi neki posao. Svaki Sprint ima definiciju što će se obaviti, na koji način i koji će odrediti izradu, posao i konačni proizvod.

Dnevni *Scrum* je 15-minutni, vremenski ograničen događaj, koji služi da razvojni tim uskladi aktivnosti i donese plan za sljedeća 24 sata. To se čini kontrolom rada od prethodnog dnevnog *Scrum* sastanka i procjenom posla koji bi mogao biti odrađen prije slijedećeg sastanka. Dnevni *Scrum* se održava uvijek na istom mjestu svaki dan da bi se smanjila kompleksnost. *Scrum master*

forsira pravilo da samo razvojni tim sudjeluje na dnevnom *Scrum* sastanku i da sastanak traje 15 minuta.

3.2.3. Artefakti u Scrumu

Product backlog je sortirana lista svega što će možda biti potrebno za proizvod i jedini izvor zahtjeva za bilo kakvim promjenama koje se rade na proizvodu. Vlasnik proizvoda je odgovoran za *Product backlog*, uključujući njegov sadržaj, raspoloživost i sortiranje. *Product backlog* nikad nije konačan. U početku sadrži samo one zahtjeve koji su inicijalno poznati i razumljiviji. *Product* backlog evoluira kako evoluira proizvod i okolina na kojoj će se primjenjivati. *Product backlog* sadrži listu svih mogućnosti, funkcionalnosti, zahtjeva, unapređenja i popravaka koja zajedno čine promjene koje će se primijeniti nad proizvodom u budućnosti. Obično je sortiran prema vrijednosti, riziku, prioritetu i nužnosti. Stavke na vrhu *backloga* su dio trenutnih razvojnih aktivnosti.

Sprint *backlog* je skup stavki s *Product Backloga* koje su odabrane za Sprint plus plan realizacije Inkrementa i realizacije Cilja Sprinta. Sprint *Backlog* je procjena razvojnog tima koje funkcionalnosti će biti u sljedećem Inkrementu i posao koji je potreban za realizaciju tog Inkrementa. Sprint *backlog* je plan sa dovoljno detalja da bi se na Dnevnom *scrumu* mogle razumjeti aktualne promjene. Razvojni tim mijenja Sprint *backlog* tijekom Sprinta, te se na njega dodaju zadaci tijekom Sprinta . Ti zadaci se događaju kada Razvojni tim, radeći prema planu, nauči nešto više o poslu koji je potreban da se zadovolji cilj Sprinta. Kako se pojavi potreba za novim poslom, Razvojni tim ga dodaje na Sprint *backlog*. Samo razvojni tim može mijenjati Sprint *backlog* tijekom Sprinta. Sprint *backlog* je vidljiva slika posla u realnom vremenu kojeg Razvojni tim namjerava obaviti tijekom Sprinta i koji pripada isključivo Razvojnom timu.

3.3. Proces izrade aplikacije

Razvija se aplikacija pod nazivom „Sustav pomoći pri kupnji računala“. Sam naziv daje naputak da se radi o nekakvim računalnim konfiguracijama. Pomoću aplikacije korisnik sastavlja kompatibilno računalo u vrlo kratkom vremenu. Neovisno o korisnikovom poznavanju računala treba imat mogućnost odabira komponenti specifičnih odabranoj kategoriji računala. Ispisati njihove cijene kako bi korisnik dobio predodžbu cijene komponenti. Korisnik treba imat mogućnost prikaza svih komponenti kako bi dobio uvid koje komponente su dostupne i sužavanje izbora odabirom kategorija računala. Potrebno je kreirati bazu podataka koja sadrži tablice naziva komponenti, odrediti koje karakteristike i svojstva su potrebna za usporedbu komponenti i

informacijski važne za točno definiranje komponenti pri ispisu (proizvođač, model, kapacitet, radni takt, cijena i sl.). U sljedećim potpoglavljima opisivati će se razvojni tijek aplikacije, svaki *sprint* jedan je korak u toku razvoja aplikacije. Na početku svakog potpoglavlja nalazi se kratki pregled sadržaja dokumenta *Sprint Backlog*. Sadržaj dokument sastoji se od zahtjeve koji se obrađuju u zadanom *sprintu*, te opisnih komentara. Na kraju potpoglavlja nalazi se kratki pregled *sprinta* odnosno što se točno napravilo u *sprint*-u, koliko je vremena bilo potrebno za njegovu izradu i jeli sprint uspješno odrađen u predviđenom vremenskom roku. Zahtjevi koji se zapisuju u zasebni dokument *Product Backlog*, ukazuju na daljnji tok razvoja aplikacije, no nisu nužno sljedeći korak u izradi aplikacije. Detaljni prikaz zahtjeva razvojnog toka, upisanih u *Product Backlog* prikazan je u prilogu (prilog 3.1.).

3.3.1. Sprint 1

SPRINT BACKLOG

> Potrebno je kreirati bazu podataka koja sadrži tablice naziva komponenti, odrediti koje karakteristike i svojstva su potrebna za usporedbu komponenti i informacijski važne za točno definiranje komponenti pri ispisu (proizvođač, model, kapacitet, radni takt, cijena i sl.). Napisati uvjete određivanja kompatibilnosti s nazivima napisanim u bazi podataka. Prethodno je definiran cilj sprinta. Vrijeme dostupno za realizaciju je tjedan dana.

Glavne komponente računala potrebne za usporedbu su: procesor, matična ploča, radna memorija, grafička kartica, napajanje i kućište (eng. *cpu, motherboard, ram memory, graphics card, power supply unit, case*). Koriste se engleski izrazi u bazi podataka i većini programskog koda zbog konzistencije i jednostavnosti pisanih izraza, dodani u prilogu.

Određuju se karakteristike i svojstva pojedinih komponenti u zasebne tablice, imajući na umu svojstva potrebna za određivanje kompatibilnosti komponenti i karakteristike potrebne za odrađivanje komponenti Skica baze podataka prikazana je tablicom (tablica 3.1.) gdje je svaki stupac zasebna tablica. U prilogu nalaze se na engleskom jeziku, skica baze podataka (prilog 3.2.) i tablica s prikazom tipova podataka baze (prilog 3.3).

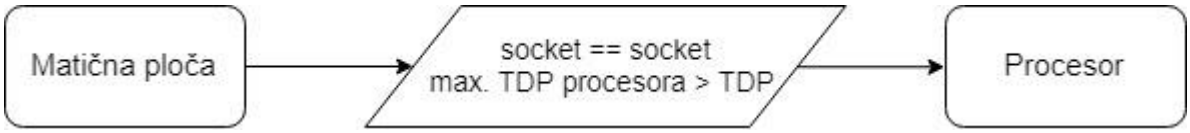
Tablica 3.1. Skica baze podataka i njenih tablica

Procesor	**Radna memorija**	**Matična ploča**	**Grafika**	**Napajanje**	**Kućište**
Identifikacijs ki broj	Identifikacijs ki broj	Identifikacijs ki broj	Identifikacijs ki broj	Identifikacijs ki broj	Identifikacijs ki broj
kategorija	kategorija	kategorija	kategorija	kategorija	kategorija
proizvođač	proizvođač	proizvođač	proizvođač	proizvođač	Proizvođač
model	model	model	model	model	Model
Priključak	kapacitet	Priključak	Kapacitet memorije	Snaga	Format matične ploče
Radni takt	Tip memorije	TDP procesora	duljina	cijena	Duljina grafičke
Broj jezgri	Frekvencija	format	TDP	Detalji	Cijena
Pričuvna memorija	Cijena	Tip radne memorije	Cijena		detalji
TDP	Detalji	Frekvencija memorije	detalji		
Cijena		Cijena			
detalji		Detalji			

Uvjeti kompatibilnosti:

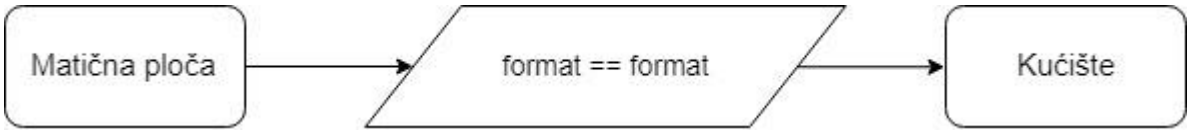
Prethodno opisani u poglavlju 2.2., uvjeti kompatibilnosti među komponentama su pojednostavljeni i prilagođeni imajući na umu sveukupnu performansu računala.

Kako bi matična ploča i procesor bili kompatibilni, priključak procesora i matične ploče mora biti jednak, te TDP procesora mora biti manji od maksimalno dozvoljenog TDP-a matične ploče. Skica uvjeta prikazana slikom (slika 3.3.).



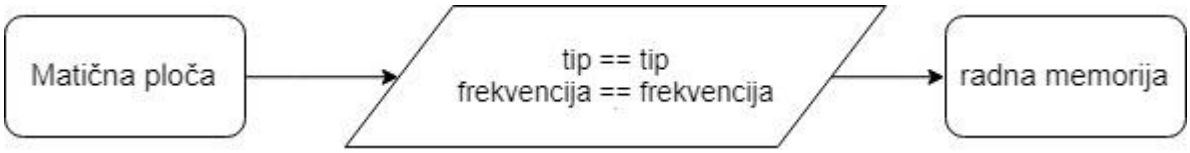
Slika 3.3. Uvjet kompatibilnosti matične ploče i procesora

Kako bi matična ploča i kućište bili kompatibilni, format matične ploče i kućište treba biti jednak. Prikaz skice slikom (slika 3.4.).



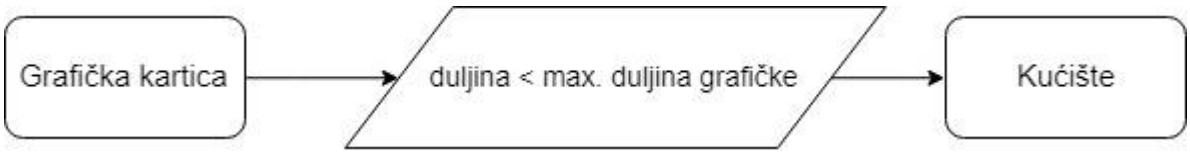
Slika 3.4. Uvjet kompatibilnosti matične ploče i kućišta

Kako bi matična ploča i radna memorija bili kompatibilni tip radne memorije mora odgovarati podržanom tipu radne memorije matične ploče, također i frekvencije radne memorije moraju biti jedna od podržanih frekvencija radne memorije matične ploče. Uvjet prikazan slikom (slika 3.5.).



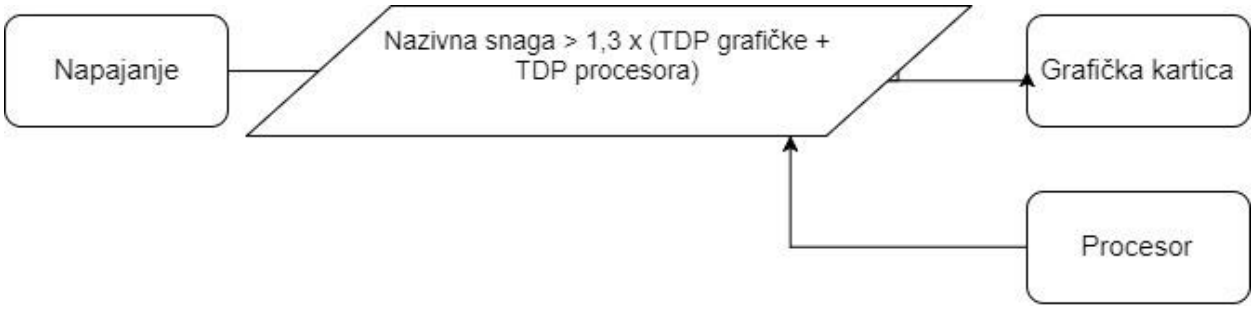
Slika 3.5. Uvjet kompatibilnosti matične ploče i radne memorije

Kako bi grafička kartica bila kompatibilna s kućište dulja grafičke kartice treba biti manja od podržane duljine grafičke kartice kućišta. Uvjet prikazan slikom (slika 3.6.).



Slika 3.6. Uvjet kompatibilnosti grafičke kartice i kućišta

Kako bi napajanje opskrbljivalo dovoljno električne energije potrebne za ispravan rad računala snaga koju napajanje daje mora biti veća od umnoška faktora 1,3 i zbroja TDP-a procesora i grafičke kartice. Uvjet prikazan slikom (slika 3.7.).



Slika 3.7. Uvjet kompatibilnosti napajanja potrebne opskrbe snage za normalan rad računala

PREGLED SPRINTA

Sprint je uspješno odrađen u predviđenom vremenu. Izrađena je baza podataka s adekvatnim nazivima tablica i njezinih stupaca kao svojstva i karakteristike za ispis i provjeru kompatibilnosti

komponenti. Baza podataka stvorena je pomoću ugrađenog alata phpmyadmin u sklopu XAMPP alatnog paketa. Uočena je potrebna daljnja razrada kod usporedbe podržanih frekvencija matične ploče i frekvencije odabrane radne memorije. Ova stavka upisuje se u *Product Backlog*.

3.3.2. Sprint 2

SPRINT BACKLOG

> Potrebno je kreirati naslovnu stranicu koja će u sebi imati sve osnovne elemente za prikaz informacija na web stranici. Kreirati vizualni identitet aplikacije i odabrati paletu boja koja će biti korištena pri izradi aplikacije. Potrebno je korisnika informirati općenito o sastavljanju konfiguracija i najbitnijim stvarima koje mora imati na uvidu, te opisati generalne kategorije računala kao pomoći pri odabiru. Cilj je prethodno definiran. Vrijeme dostupno za realizaciju je 2 tjedna.

Proces izrade:

Kreira se HTML web stranica u koju se implementira Bootstrap programski okvir i jQuery biblioteke kao osnova razvoja *frontend* dijela aplikacije. Stranica sadrži osnovne HTML elemente, ugrađene jQuery biblioteke, te prilagođenu verziju CSS-a kako bi se ostvarila mogućnost uređivanja izgleda stranice prema propisanim potrebama. Prikaz na slici (sl. 3.8.) Također dodana je prilagođena jQuery skripta za prilagođenu funkcionalnost.

```
1  <!DOCTYPE html>
2  <html lang="hr">
3    <head>
4      <meta charset="utf-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1">
7      <!-- The above 3 meta tags *must* come first in the head; any other head content must come *after* these tags -->
8      <title>Sustav Pomoći Pri Kupnji Računala</title>
9      <!-- Bootstrap -->
10     <link rel="stylesheet" type="text/css" href="css/bootstrap.css" >
11     <link href="css/style.css" rel="stylesheet" type="text/css"  media="all">
12     <link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/font-awesome/4.5.0/css/font-awesome.min.css"/>
13     </head>
14       <body>
15     <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
16     <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
17     <script src="js/bootstrap.min.js"></script>
18       <script src="js/custom.js"></script>
19   </body>
20 </html>
```

Slika 3.8. Kostur početne stranice na početku razvoja aplikacije

Kad je kostur stranice sastavljen formira se uvod o sastavljanju konfiguracije računala, te opis generalne kategorizacije računala.

PREGLED SPRINTA

Sprint je realiziran uspješno u određenom vremenu. Kreiran je vizualni identitet stranice, informacije o sastavljanju konfiguracija su jednostavne i razumljive. Stranica je responzivna, što znači da se može pregledavat na uređajima raznih veličina zaslona bez smetnji u izgledu stranice, izgled stranice prilagođuje se raznih veličinama zaslona. Dodane su tri tipke koje trenutno nisu u funkciji, no to ulazi u *Product Backlog* radi daljnjeg planiranja. Pokazala se potreba za izradom stranice (vizualni dio) koja će sadržavati glavnu funkcionalnost web aplikacije odabir komponenti i izrada skripte (poslužiteljski dio) za spajanje na bazu podataka i dohvaćanje podataka.

3.3.3. Sprint 3

SPRINT BACKLOG

> Potrebno je kreirati skriptu pomoću koje će se aplikacija povezati s bazom podataka na serveru i ujedno omogućiti ispis potrebnih podataka iz nje. Vrijeme dostupno za realizaciju je tjedan dana.

Na slici (sl. 3.9.) prikazan je izvorni kod *connection.php* skripte. Od 3. do 6. linije koda obavlja se definiranje varijabli potrebnih za spajanje na bazu. 'DB_HOST' – označava ime servera na koji se spaja, u ovom slučaju to je *localhost* jer je poslužitelj lokalno pokrenut. 'DB_USER' – označava ime korisničkog računa s kojim se spaja na bazu podataka, u ovom slučaju to je *masteradmin* pod imenom root. 'DB_PASS' – označava zaporku korisničkog računa s kojom se spaja. 'DB_NAME' – ime baze na koju se spaja. Nakon definiranja varijabli, varijabli $conn dodjeljujemo vezu unosom prethodno definiranih parametara veze. Ako se pojavi greška prilikom spajanja program se završava s ispisanom porukom „conncetion failed“ i opis greške koja je nastala. Nadalje izvršava se MySQL upit (SELECT * FROM cpu), što znači da se odaberu svi podatci iz tablice „cpu“ – procesor, te se spremaju u varijablu $cpu. Na taj način dohvaćaju se podatci iz baze i spremaju u varijable prikladnog naziva.

```php
<?php

    defined('DB_HOST') or define('DB_HOST', 'localhost');
    defined('DB_USER') or define('DB_USER', 'root');
    defined('DB_PASS') or define('DB_PASS', '');
    defined('DB_NAME') or define('DB_NAME', 'pc_components');

    $conn = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);

    if($conn->connect_error){

        die("connection failed " . $conn->connect_error());
    }

    $cpu = $conn->query ("SELECT * FROM cpu") or die($conn->error);
    $graphics = $conn->query ("SELECT * FROM graphics ") or die($conn->error);
    $motherboard = $conn->query ("SELECT * FROM motherboard") or die($conn->error);
    $pc_case = $conn->query ("SELECT * FROM pc_case") or die($conn->error);
    $psu = $conn->query ("SELECT * FROM psu") or die($conn->error);
    $ram = $conn->query ("SELECT * FROM ram") or die($conn->error);

?>
```

Slika 3.9. Izvorni kod connect.php skripte

Napisana skripta poziva se u stranici za sastavljanje pri samom vrhu prije <html> oznaka. Prije <html> oznaka obavljaju se provjere zahtjeva i odgovora HTTP protokola, zato je bitno na to mjesto unijeti zahtjev za *connection.php* skriptu, prikaz na slici (sl. 3.10.):

```php
<?php require_once("connection.php");?>
<!DOCTYPE html>
<html lang="hr">
  <head>
```

Slika 3.10. Isječak programskog koda koji prikazuje naredbu koja zahtijeva connection.php skriptu

PREGLED SPRINTA

Uspješno odrađen sprint u predviđenom vremenu. Uspješno spajanje na bazu podataka i probni ispis podataka. Potreba za stranicom koja prikazuje podatke iz baze podataka u nekoj formi i daje korisniku mogućnost odabira, upis u *Product Backlog* kao prioritet nad ostalim upisima.

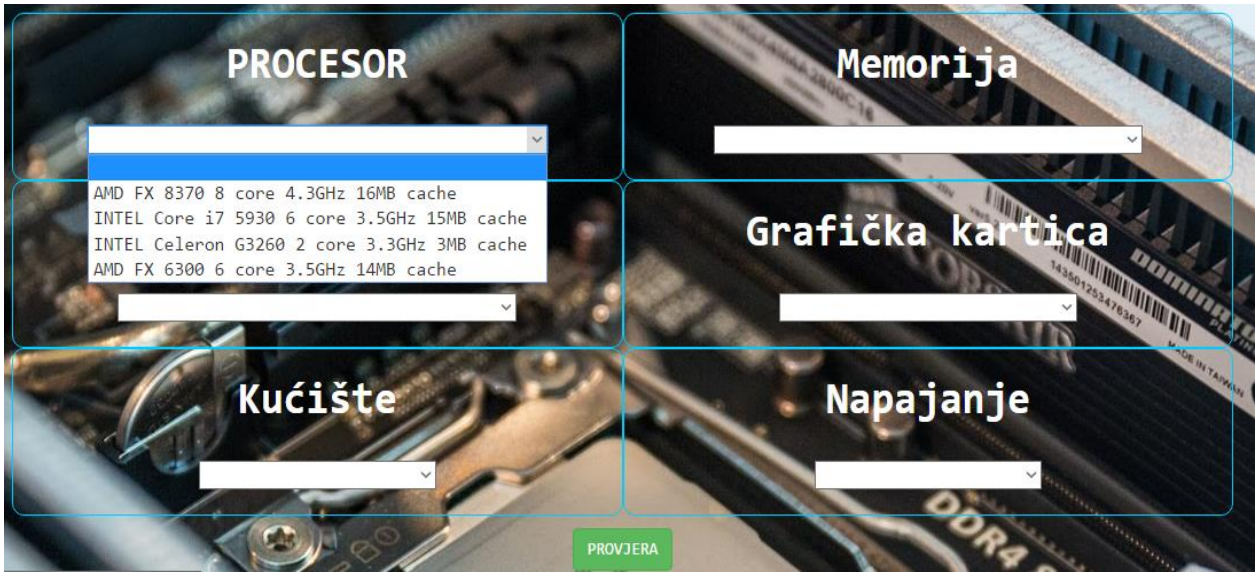
3.3.4. Sprint 4

SPRINT BACKLOG

Potrebno je izraditi korisničko sučelje odabira komponenti, uz prethodno napisanu skriptu za dohvaćanje podataka iz baze. Odabrati formu u koju se ispisuju podatci s mogućnošću slanja odabranih podataka na drugu skriptu radi daljnje orade. Odabrati prikladan način ispisa pojedinih komponenti unutar forma. Vrijeme dostupno za realizaciju sprinta četiri dana.

Pri kreiranju korisničkog sučelja odabira komponenti korišten je *select* element čija funkcija je odabrati određenu komponentu iz liste ponuđenih (podatci iz baze podataka). Prikaz sučelja na slici (sl. 3.11.). Definiran je način ispisa pojedinih komponenti:

- Procesor – proizvođač, model, broj jezgri, radni takt, veličina pričuvne memorije
- Matična ploča – proizvođač, model, *socket*, format, podržan tip radne memorije
- Radna memorija – proizvođač, kapacitet, tip memorije, radni takt
- Grafička kartica – proizvođač, model, kapacitet memorije
- Kućište računala – proizvođač, model, format podržane matične ploče
- Napajanje – proizvođač, model, nazivna snaga



Slika 3.11. Prikaz korisničkog sučelja za odabir komponenti

Nakon odabira svih komponenti, korisnik klikom miša na tipku PROVJERA na dnu korisničkog sučelja prosljeđuje odabrane podatke na drugu skriptu koja provjerava kompatibilnost odabranih uređaja.

PREGLED SPRINTA

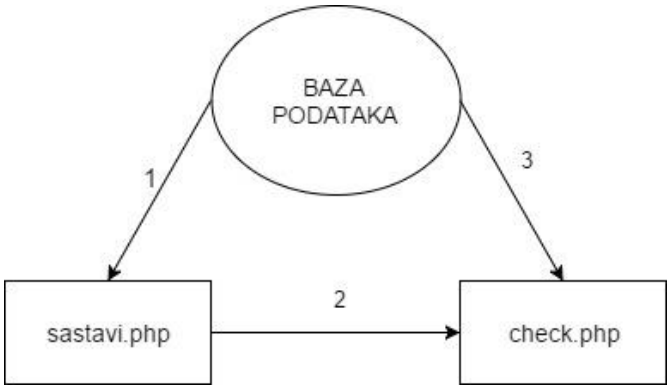
Sprint je uspješno odrađen u zadanom vremena. Korisničkim sučeljem dobiva se predodžba vizualnog napretka aplikacije. Uočena je potreba skripte koja će obavljati funkciju provjere kompatibilnosti komponenti računala, te nedostatak tipke koja poništava sve trenutne odabire. Stavke se upisuju u *Product Backlog* s tim da skripta za provjeru kompatibilnosti ima prioritet nad izradom tipke za poništavanje trenutnog odabira komponenti.

3.3.5. Sprint 5

SPRINT BACKLOG

Potreba skripte koja će provjeravati kompatibilnost među odabranim komponentama računala. Obaviti testiranje skripte za provjeru kompatibilnosti komponenti. Predviđeno vrijeme izrade svedeno je na tjedan dana.

U tri koraka opisana je logika i tijek izvođenja aplikacije, prikazano slikom (slika 3.12.)



Slika 3.12. Dijagram toka izvođenja aplikacije

1. Prilikom otvaranja stranice sastavi.php dohvaćaju se podatci iz baze podataka i smještaju se u *select* html element prikladno za svaku komponentu, gdje svaki *select* element sadrži unikatno ime za razlikovanje sadržaja podataka pojedinih *select* elemenata. Odabirom određene komponente određuje se njegov identifikacijski broj (id). Identifikacijski broj služi za unikatno određivanje određene komponente
2. Nakon odabira svih komponenti klikom miša na tipku „PROVJERA“ šalju se podatci (identifikacijski brojevi) na skriptu za provjeru kompatibilnosti check.php
3. Primljeni podatci (identifikacijski brojevi) s prethodne stranice (sastavi.php) spremaju se u varijable adekvatnih imena. Postavljaju se uvjeti za ispitivanje postojanost varijabli, te se unutar tih uvjeta programiraju uvjeti za ispitivanje kompatibilnosti i ispis odgovarajuće poruke o kompatibilnosti ili poruke o ne odabiru određenih komponenti.

PREGLED SPRINTA

Uspješno odrađen sprint u predviđenom vremenu. Funkcionalnost skripte ja zadovoljavajuća, uspješan dohvat podataka iz baze na osnovu odabranog identifikacijskog broja komponenti i njihova provjera kroz uvjete kompatibilnosti. Potreba za optimizacijom tijeka izvođenja aplikacije i skripte za provjeru kompatibilnosti komponenti zahtijeva uvođenje PHP programskog okvira, odnosno Laravel-a. U *Product Backlog* upisuje je se potreba za optimizacijom tijeka izvođenja

aplikacije pomoću Laravel-a, te potreba stranice za detaljni prikaz odabranih komponenti i rezultata provjere kompatibilnosti, s tim da optimizacija aplikacije ima najveći prioritet.

3.3.6. Sprint 6

SPRINT BACKLOG:

Potreba optimizacije aplikacije uvođenjem Laravel-a. Vrijeme dostupno za realizaciju sprinta je 10 dana.

Optimizacija se provodi tri koraka:

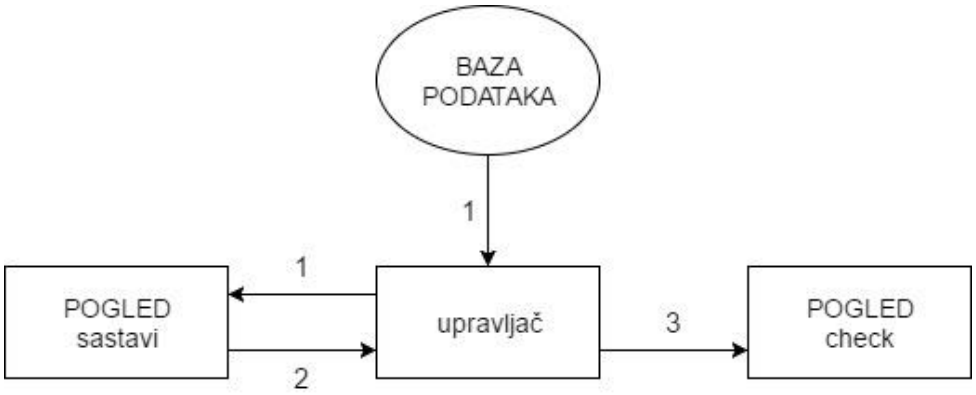
1. Implementacija Laravel na lokalni poslužitelj. Laravel se preuzima sa službenih stranica, instalacija se izvršava u naredbenoj liniji. Naredbena linija se koristi za pokretanje lokalnog poslužitelja, kreiranje Laravel projekta, upravljača, modela.
2. Planiranje procesa implementacije izvornog koda. Prilagođuje se tijek izvođenja aplikacije specifikacijama radnog okvira. Moguće pojave nekompatibilnosti izvornog koda, priprema se njihovo rješenje.
3. Izvedba planiranog dijela procesa.

Opis pojmova bitnih pri optimizaciji:

- Put (eng. *route*) – pravilo koje određuje na koji način se izvodi tok aplikacije
- Pogledi (eng. *view*) – web stranica koja služi za prikaz podataka
- Upravljač (eng. *controller*) – radi s bazom podataka, varijablama. Prihvaća ulazne naputke i pretvara ih u naloge modelu ili pogledu.

Kreira se novi Laravel projekt u direktoriju sustav_pomoci. Kreira se nacrtna datoteka (eng. *layout*) koja sadrži osnovni kod za prikazivanje stranice i njen stil ali bez sadržaja, moguća ponovna upotreba za razne stranice bez potrebe kopiranja ili ponovnog pisanja koda zasebno za svaku stranicu. Kreiraju se pogledi , te se primjenjuju izmjene uočene kao prikladniji način ispisa sadržaja stranica. Konfiguriraju se putevi unutar projekta. Stvara se upravljač koji služi kao sredstvo komunikacije među korisničkim sučeljem i bazom podataka. Upravljač se konfigurira tako da ima mogućnost provjere kompatibilnosti među odabranim komponentama. Većina izvornog koda se može prilagoditi programskom okviru. Prilagođuju se odstupanja logičkog modela i logike dosadašnjeg koncepta izvođenja aplikacije.

Prilagođen tijek i logika izvođenja aplikacije prikazana slikom (slikom3.13.) i kratki opis:



Slika 3.13. Prilagođen tijek i logika izvođenja aplikacije unutar programskog okvira Laravel

1. Putem na 27. liniji koda (slika 3.14.) pokrenut je upravljač koji u *indeks()* funkciji dohvaća podatke iz baze podataka te ih sprema u varijable prikladnih imena, koji su bitni za ispis forme za odabir komponenti. Nakon izvršavanja funkcija vraća pogled „sastavi“ s pripadajućim varijablama.
2. Na pogledu „sastavi“ korisnik odabire komponente na taj način postavljajući identifikacijski broj istih. Klikom na tipku „PROVJERA“ šalju se informacije o identifikacijskim brojevima komponenti na upravljač.
3. Put kojim se slanje podataka s pogleda „sastavi“ na upravljač izvršava naredbom na 28. liniji koda (slika 3.14.). Unutar *show()* funkcije upravljača odvija se proces usporedbe komponenti i generiranje prikladnog rezultata. Upravljač nakon izvršavanja svih naredbi *show()* funkcije vraća pogled „check“ s varijablama u koje su spremljeni rezultati provjere i identifikacijskim brojevima.

```
14 ▼ Route::get('welcome', function () {
15       return view('welcome');
16   });
17
18 ▼ Route::get('/', function () {
19       return view('naslovna');
20   });
21
22 ▼ Route::get('upute', function () {
23       return view('upute');
24   });
25
26
27   Route::get('provjera','ProvjeraController@index');
28   Route::get('provjera/check','ProvjeraController@show');
```

Slika 3.14. Prikaz puteva aplikacije

PREGLED SPRINTA

Uspješno odrađen sprint u predviđenom vremenu. Optimizacija provedena uspješno, uz napomenu izostanka korištenja „Model“ dijela MVC koncepta što predstavlja potencijalne komplikacije i/ili nekompatibilnosti u daljnjem razvoju aplikacije. Napomena se upisuje u *Product Backlog*.

3.3.7. Sprint 7

SPRINT BACKLOG

> Potreba kreiranja stranice odnosno pogleda za detaljni prikaz odabranih komponenti i rezultata provjere kompatibilnosti među komponentama. Raspoloživo vrijeme izrade 4 dana.

Kreiran je pogled „check“. Korištenjem nacrtne datoteke (emg. *layout*) skraćuje se vrijeme potrebno za izradu sučelja za detaljni ispis odabranih komponenti. Odabrane komponente određene su identifikacijskim brojem koji se koristi za ispis komponenti definiran u četvrtom *sprint*-u. Korisnik mora odabrati sve komponente, inače se ispisuje poruka koja naznačuje da određena komponenta nije odabrana. Prilikom odabira svih komponenti izvršava se provjera kompatibilnosti, te ovisno o rezultatu ispisuje se određena ne kompatibilnost među komponentama ili poruka koja naznačuje uspješnu kompatibilnost. U prilogu se nalazi prikaz ispisa rezultata provjere kompatibilnosti među komponentama (prilog 3.4. – 3.6.).

PREGLED SPIRNTA

Sprint je uspješno odrađen u predviđenom vremenu. Kreirano je sučelje s ispisom poruka pogreške, ne odabiranje određene komponente i/ili nekompatibilnosti.

3.3.8. Sprint 8

SPRINT BACKLOG

> Potreba za kreiranjem funkcionalnošću tipki na početnoj stranici i na stranici za sastavljanje konfiguracije. Vrijeme predviđeno za izvođenje 1 dan.

Tipkama se daje prikladan naziv po kategoriji računala kojoj pripada. Pridodan naziv sprema se u varijabli *$*category. Odabiranjem kategorije računala postavlja se uvjet po kojem se dohvaćaju podatci iz baze podataka.

PREGLED SPRINTA

Sprint je uspješno odrađen u predviđenom vremenu.

3.3.9. Sprint 9

SPRINT BACKLOG

Potreba stranice koja ću pružati informacije o pravilnom sastavljanju komponenti računala u kućište kako bi računalo bilo spremno za korištenje. Vrijeme dostupno za izradu 5 dana.

Stranica sadrži mjere opreza kojih se korisnik mora pridržavati zbog osobne sigurnosti, sigurnosti okoline i kako ne bi došlo do oštećenja komponenti. Upute su interaktivni dijelovi koji sadrže kratki opis određene komponente, informaciju pravilnog postavlja komponente korak po korak i video materijal koji sažima tekstualni dio uputa.

PREGLED SPRINTA

Sprint je uspješno odrađen u predviđenom vremenu. Krake i jasne upute postavljanja komponenti sastavljene su tako da ih korisnik može obaviti bez dodatnih prethodnih znanja postavljanja komponenti.

3.4. Osvrt na aplikaciju

Aplikacija Sustav pomoći pri kupnji računala klasificira se u kategoriju web aplikacija. Web aplikacijama potreban je poslužitelj za izvršavanje funkcionalnosti. Cilj aplikacije bio je pružati korisniku mogućnost sastavljanja kompatibilne konfiguracije računala s ili bez prethodnog znanja o sastavljanju konfiguracije računala. Tehnologije korištene pri izradi aplikacije su besplatne. Aplikacija je korištena i testirana na osobnom računalu koje je imalo ulogu poslužitelja i klijenta. Provedena su testiranja tijekom i nakon izrade aplikacije.

PREDNOSTI:

Najveći aspekt prednosti aplikacije leži u njenoj jednostavnosti i jednoznačnosti. Vizualni dizajn aplikacije čini informacije brzo dostupnim i lako čitljivim. Dizajn aplikacije je potpuno responzivan, što znači da će jednako dobro biti prikazani na uređajima različitih dimenzija zaslona. Aplikacija se može u potpunosti koristiti bez obzira na korisnikovo prethodno znanje. Korišten je aktualni PHP radni okvir, Laravel, što aplikaciju čini sukladnu propisanim standardima developmenta i sigurnom od neovlaštenih, zlonamjernih upada s sustava. Aplikacija ne sadrži osjetljive podatke koji bi mogli biti zlouporabljeni.

NEDOSTATCI:

Nedostatak aplikacije leži u nedostatku korisnih mogućnosti unutar aplikacije. Izbačene su poslužitelj i prijenosna kategorija računala. Korisnik aplikacije ne dobiva detaljne upute savjeta

pri kupnji računala već je ograničen na opis i odabir kategorije računala, što ne mora biti nedostatak ali može biti ovisno o samom korisniku. Nedostatak kreiranja PDF datoteke za ispis sastavljene konfiguracije. Velika odabirna lista pri velikom broju unesenih podataka u bazi podataka.

MOGUĆE NADOGRADNJE:

Glavna funkcionalnost aplikacije bazirana je na dohvatu, usporedbi i ispisu podataka iz baze podataka što ju čini idealnom podlogom za radni okvir. Funkcionalni dio aplikacije može se definirati kao zaseban programski kod koji se može komercijalizirati. Neke od mogućih nadogradnji: mogućnost kreiranja PDF datoteka za ispis sastavljene konfiguracije, izrada interaktivnog vodiča pri kupnji s naglaskom na savjete koji bi korisnika dublje uvelo u svijet računala, mogućnost kupovine sastavljene konfiguracije. Daljnji razvojni tok aplikacije može u potpunosti promijeniti osnovni princip rada stranice i njeni naziv.

4. ZAKLJUČAK

Aplikacija „Sustav pomoći pri kupnji računala“ radila se unutar Scrum radnog okvira zbog jednostavnosti i prikladnosti ovom projektu, provođenjem cijelog projekta kroz tzv. sprintove. Sprint je korak u izradi aplikacije, traje određeno vrijeme i unutar njega potrebno je izvršiti definirane zahtjeve koji su zapisani u *Spint Backlog*-u, dokumentu koji sadrži sve napisane zahtjeve definirane za određeni sprint. Prije početka izrade definirale su se temeljne informacije o zahtjevima i mogućnostima aplikacije koje su upisane u *Product Backlog*. Odrađivanjem sprintova uočile su se dodatne potrebe koje prethodno nisu bile vidljive. Nakon završetka svakog sprinta uočene potrebe upisane su u *Product Backlog*. Prilikom izrade aplikacije korištena su dva programska okvira, Bootstrap za *frontend* i Laravel za *backend*. *Frontend* dio obuhvaća tehnologije poput HTM-a i CSS-a, Javascripta i jQueryja te je zaslužan za vizualni dio aplikacije. *Backend* osim Laravel-a obuhvaća i XAMPP program za lokalno postavljanje poslužitelja na računalu. Obrađuje zahtjeve na bazu podataka, osigurava serversku podršku sustava te obrađuje razne zahtjeve među upravljačem i pogledima. Vrijeme potrebno za izradu aplikacije znatno se može smanjiti s povećanjem broja . Aplikacija može se koristiti kao brza provjere kompatibilnosti komponenti računala i uputa za pravilno postavljanje komponenti. U daljnjem razvoju aplikacije predviđeno je dodavanje raznih mogućnosti poput kupovine sastavljane konfiguracije računala, rekonstrukcija naslovne stranice zbog pružanja savjeta prilikom kupovine računala, detaljni opis dodataka računalu kao što su ugradnja vodenog hlađenja, ugradnji više grafičkih kartica u računalo. Predviđena je i promjena naziva cijele aplikacije. Funkcionalni dio aplikacije može se koristiti kao zasebno kod, te kao takav poslužiti kao radni okvir provjeri kompatibilnosti komponenti pri ugradnji u web shopove trgovina koje se bave prodajom i sastavljanjem računala. Trenutno nije odabran daljnji razvoj aplikacije, no popis dodataka je stvoren.

5. LITERATURA

[1.] M. Murray, How to build your PC [online], PCMag, 2015., dostupno na: https://www.pcmag.com/article2/0,2817,2485172,00.asp
[Datum zadnje posjete stranici: travanj 2017.]

[2.] L. Sveskis, The Ultimate Web Developer How To Guide [online], Udemy, 2015., dostupno na: https://www.udemy.com/learn-web-development-complete-step-by-step-guide-to-success
[Datum zadnje posjete stranici: svibanj 2017.]

[3.] M. Otto, Bootstrap [online], Bootstrap, July 2016., dostupno na: http://getbootstrap.com
[Datum zadnje posjete stranici: travanj 2017.]

[4.] T. Otwell, Laravel [online], Laravel 5.4. i Laracast, siječanj 2017., dostupno na: https://laravel.com/
[Datum zadnje posjete stranici: lipanj 2017.]

[5.] K. S. i J. S., Schwaber i Sutherland, The Scrum guide [online, e-knjiga], 2013. dostupno na: http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf
[Datum zadnje posjete stranici: lipanj 2017.]

SAŽETAK

Razvila se aplikacija pomoću koje se prikazuje kompatibilnost odabranih komponenti računala, te upute pravilnog postavljanja komponenti unutar kućišta računala. Rad sadrži teorijsku podlogu komponenti računala i njihove kompatibilnosti, kratki opis korištenih tehnologija, opis *Scrum* okvira po kojem je izrađena aplikacija, opis cijelog postupka izrade s izvornim kodovima pojedinih koraka izrade i opis osnovnih računalnih komponenti bitnih za konfiguriranje osobnog računala. Glavna funkcionalnost bazirana je na dohvatu podataka iz baze, odabir određenih podataka, njihova usporedba i ispis. Moguća je nadogradnja odnosno poboljšanje aplikacije. Pogledi na daljnji razvoj aplikacije opisani su u osvrtu na aplikaciju. Uz poglede na budućnost opisane su prednosti, nedostatci i vrijeme potrebno za izradu aplikacije. Sa zaključkom aplikacija je prividna kraju.

Ključne riječi: samogradnja računala, Bootstrap, Laravel, radni okvir, web aplikacija

ABSTRACT

Title: PC purchasing assistant and configurator

The application was developed with a purpose of showing compatibilities between chosen pc components with a tutorial how to correctly install the components inside the pc case. The project contains the following things: theory concept of pc components and their compatibilities, short description of the building technologies, description of the Scrum framework which is used in the development process, description of the whole development through sprints with added source code contained in the attachment and a description of the basic pc building components. The main functionality is based on getting data from the database, their comparison and show the results. The application can be upgraded with the development of features described in this document. Pros and cons, the time needed to complete the project are described beside the future work. The project was finished with a conclusion.

Keywords: build your own compatible pc, Bootstrap, Laravel, framework, web application

ŽIVOTOPIS

Erik Valentin Szabó rođen je 19. studenog 1993. u Bonyhád u Republici Mađarskoj. Nakon završetka srednje škole, 2012. upisao je Elektrotehnički Fakultat u Osijeku, današnji Fakultet elektrotehike, računarstva i informacijskih tehnologija. Tijekom druge godine studija odlučio je posvetiti se razvijanju programske podrške. Motivacija mu se nalazi u želji za savladavanjem znanja i tehnika potrebih za izradu video igara. Praktični dio nastave kolegija „Stručna praksa i pojekt" odrađivao je u tvrtki Siemens pod mentorstvom Hrvoje Ovničevića. Prisutstvovao je raznim otvorenim predavanjima o tehnologijama razvoja aplikacija u Osijeku. Krenuo je na Software Startup akademiju, ali zbog nedostatka ambicije i raznih životnih problema ideju je pretvorio u projekt završnog rada.

PRILOZI

Prilog 3.1. Detaljni prikaz sadržaja dokumenta Product Backlog

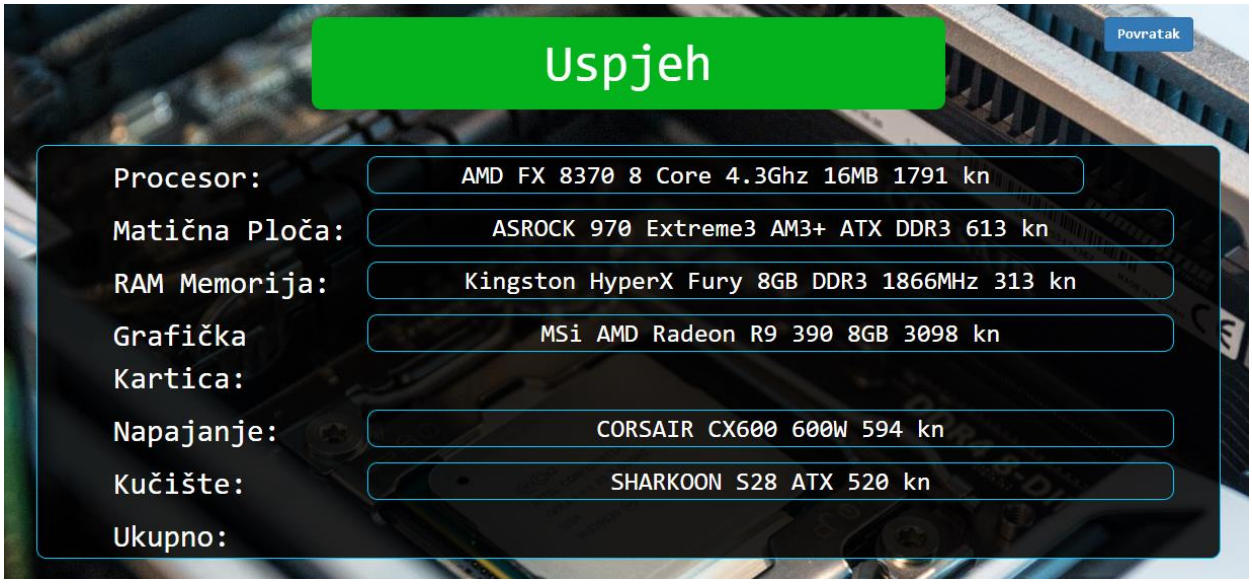
9.	5 dana	-
8.	1 dan	-
7.	4 dana	Kreirano je sučelje s ispisom poruka pogreške, ne odabiranje određene komponente i/ili nekompatibilnosti.
6.	10 dana	Napomena: Izostanka korištenja „Model“ dijela MVC koncepta što predstavlja potencijalne komplikacije i/ili nekompatibilnosti u daljnjem razvoju aplikacije.
5.	1 tjedan	Potreba za optimizacijom tijeka izvođenja aplikacije i skripte za provjeru kompatibilnosti komponenti zahtijeva uvođenje PHP radnog okvira. Potreba stranice za detaljni prikaz odabranih komponenti i rezultata provjere kompatibilnosti
4.	4 dana	Uočena je potreba skripte koja će obavljati funkciju provjere kompatibilnosti komponenti računala, te nedostatak tipke koja poništava sve trenutne odabire.
3.	1 tjedan	Potreba za stranicom koja prikazuje podatke iz baze podataka u nekoj formi i daje korisniku mogućnost odabira komponenti
2.	2 tjedna	Pokazala se potreba za izradom stranice (vizualni dio) koja će sadržavati glavnu funkcionalnost web aplikacije odabir komponenti i izrada skripte (serverski dio) za spajanje na bazu podataka i dohvaćanje podataka. Napraviti funkcionalnost za tipke na početnoj stranici kako bi postavile kategoriju odabira komponenti.
1.	1 tjedan	Potreba za mogućnošću usporedbe frekvencije odabrane radne memorije iz liste ponuđenih podržanih frekvencija radne memorije matične ploče
SPRINT	VRIJEME TRAJANJA	Razvija se aplikacija pod nazivom „Sustav pomoći pri kupnji računala“. Sam naziv daje naputak da se radi o nekakvim konfiguracijama. Pomoću aplikacije korisnik sastavlja kompatibilno računalo u vrlo kratkom vremenu. Neovisno o korisnikovom poznavanju računala treba imat mogućnost odabira komponenti specifičnih odabranoj kategoriji računala. Ispisati njihove cijene kako bi korisnik dobio predodžbu cijene pojedinih i ukupno svih komponenti. Korisnik treba imat mogućnost prikaza svih komponenti kako bi dobio uvid koje komponente su dostupne i sužavanje izbora odabirom kategorija računala. Potrebno je kreirati bazu podataka koja sadrži tablice naziva komponenti, odrediti koje karakteristike i svojstva su potrebna za usporedbu komponenti i informacijski važne za točno definiranje komponenti pri ispisu (proizvođač, model, kapacitet, radni takt, cijena i sl.).
		PRODUCT BACKLOG

Prilog 3.2. Skica baze podataka s engleskim izrazima

cpu	**ram**	**motherboard**	**graphics**	**psu**	**pc_case**
id	id	id	id	id	id
category	category	category	category	category	category
manufacturer	manufacturer	manufacturer	manufacturer	manufacturer	manufacture r
model	model	model	gpu	model	model
socket	capacity	socket	memory_cap city	power	motherboar d_format
frequency	memory_type	cpu_tdp	length	price	gpu_length
core	frequency	format	tdp	details	price
cache	price	memory_type	price		details
tdp	details	mem._freq.	details		
price		price			
details		details			

Prilog 3.3. Prikaz tipova podataka baze

cpu	**ram**	**motherboard**	**graphics**	**psu**	**pc_case**
smallint	smallint	smallint	smallint	smallint	smallint
tinytext	tinytext	tinytext	tinytext	tinytext	tinytext
tinytext	tinytext	tinytext	tinytext	tinytext	tinytext
varchar(20)	varchar(20)	varchar(20)	varchar(20)	varchar(20)	varchar(20)
tinytext	tinyint	tinytext	tinyint	smallint	tinytext
decimal(2,1)	tinytext	smaillint	decimal(3,1)	smallint	decimal(3,1)
tinytext	smallint	tinytext	smallint	tinytext	smallint
tinytext	smallint	tinytext	smallint		tinytext
tinytext	tinytext	tinytext	tinytext		
smallint		smallint			
tinytext		tinytext			



Prilog 3.4. Prikaz kompatibilne sastavljene konfiguracije računala

Prilog 3.5.. Prikaz nekompatibilne sastavljene konfiguracije računala

Prilog 3.6. Prikaz kompatibilnosti sastavljene konfiguracije računala gdje nisu odabrane komponente