

**Univerzitet Singidunum**

**Tehnički fakultet**

**Modelovanje softvera za upravljanje obnovljivim izvorima energije u kućnom sistemu**

**- Projektni rad -**

Predmet: **Metodologija razvoja softvera**

|  |  |  |
| --- | --- | --- |
| **Profesor**: | **Student**: | **Poeni** |
| prof. dr Angelina Njeguš | Marko Dojkić 2018/201682 | 30 |

Beograd, 2020. godine

Sadržaj

[1. Uvod 3](#_Toc58517854)

[2. Metodologija razvoja softvera 3](#_Toc58517855)

[3. Analiza korisničkih zahteva 4](#_Toc58517856)

[4. Analiza aplikacije 6](#_Toc58517857)

[4.1. Arhitektura sistema 6](#_Toc58517858)

[4.2. Analiza slučajeva korišćenja 7](#_Toc58517859)

[5. Projektovanje aplikacije 8](#_Toc58517860)

[6. Projektovanje sloja prezentacije 10](#_Toc58517861)

[7. Zaključak 11](#_Toc58517862)

[Literatura 11](#_Toc58517863)

[Popis akronima 11](#_Toc58517864)

[Prilog 12](#_Toc58517865)

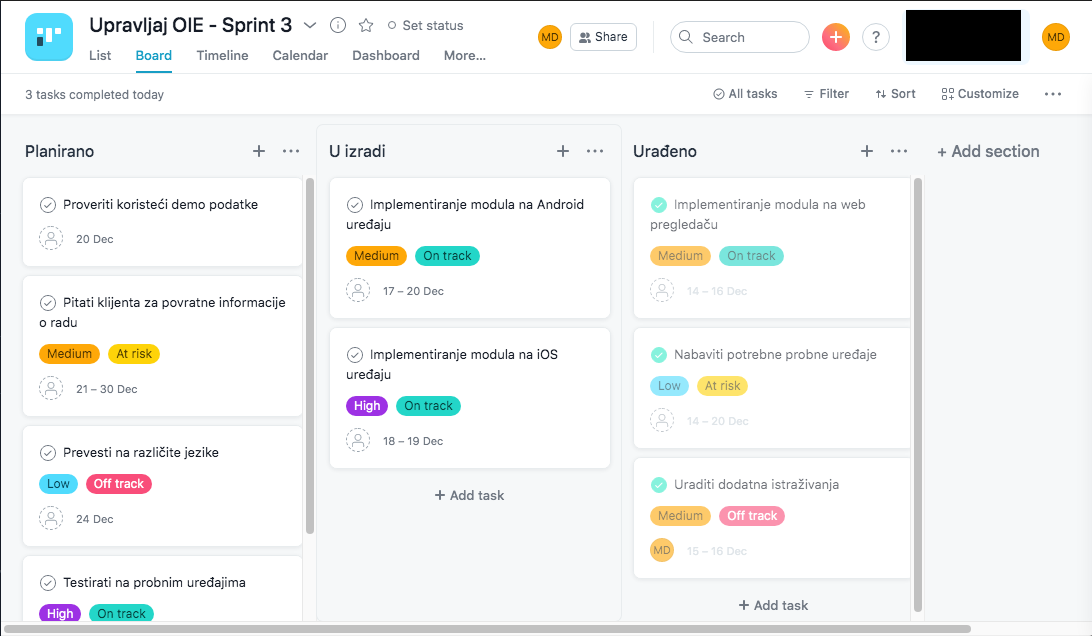
# Uvod

Kako је emisija štetnih gasova sve veća i raspoloživost neobnovljivih resursa (gasa, nafte, uglja) sve manja postoji potreba za prelazak na druge vrste energije. Naučnici stalno rade na unapređivanju postojećih i nalaženju novih alternativnih izvora energije. Ova aplikacija bi služila kao interfejs za upravljanje sistemima tih izvora. Proizvođači kontrolera obično prave svoje aplikacije za upravljanje, ali su one ograničene samo na te uređaje i taj sistem.

# Metodologija razvoja softvera

U ovom projektu će biti korišćena Scrum metodologija. Ona predstavlja agilan pristup za upravljanje razvojem softvera, što omogućava podelu projekta na manje celi i konstantan prolazak kroz sve faze razvoja za svaku celinu. Posebna karakteristika ove metodologije je postojanje „Sprint”-ova, tj. iteracija kreiranja određenih delova softvera za unapred definisani vremenski period (tipično 1-4 nedelje). Oni omogućavaju da korisnik ima uvid u izgled aplikacije dok se ona razvija i pruži povratne informacije o njegovom utisku. Ovaj projekat se sastoji od sledećih iteracija:

1. Kreiranje UI dizajna I njegova implementacija za web pregledač, Android i iOS uređaje (3-5 nedelja)
2. Implementacija modula za logovanje I registraciju (1 nedelja)
3. Implementacija modula „Pregled stanja sistema” (1-2 nedelje)
4. Implementacija modula „Promena parametara sistema” (1-3 nedelje)
5. Testiranje sistema (1 nedelja)



Slika 1: Prikaz plana izrade trećeg sprint-a

Tim za izradu ovog projekta ima sledeće uloge:

* **Rukovodilac projekta**: Osoba koja je odgovorna za praćenje celokupnog projekta i komunikaciju sa klijentom (tj. naručiocem projekta).
* **UI/UX dizajneri**: Osobe zadužene za kreiranje celokupnog grafičkog korisničkog interfejsa aplikacije (Web + Android + iOS).
* **Web programeri**: Osobe zadužene za programiranje modula web pregledač.
* **Android/iOS programeri**: Osobe zadužene za programiranje modula za Android/iOS uređaje.
* **Testeri**: Treća lica koji su angažovana na testiranju softvera.

# Analiza korisničkih zahteva

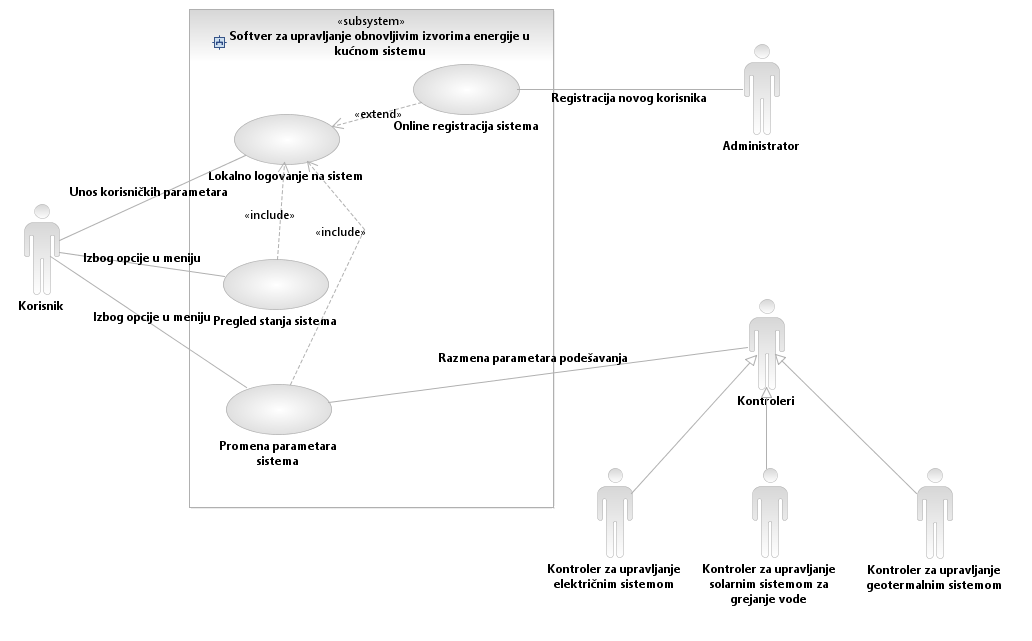
Korisnički zahtevi od kojih zavisi aplikacija:

* Vrsta sistema OIE kojih korisnik ima instalirane u kući (PV, eolski (energija vetra – vetrogeneratori), geotermalni,...),
* Da li je kuća dizajnirana kao pametna i ima utičnice opremljene uređajima za komunikaciju sa WLAN ruterom,
* Da li korisnik želi da aplikacija bude dostupna na internetu i/ili komunicira sa mobilnim uređajem,
* Lokacija i jezik korisnika (pr. ako je korisnik iz Evrope njegov naizmenični napon je 220V i frekvencija 60Hz, ali ako je iz SAD-a onda je 110V/50Hz, željeni jezik aplikacije,...)

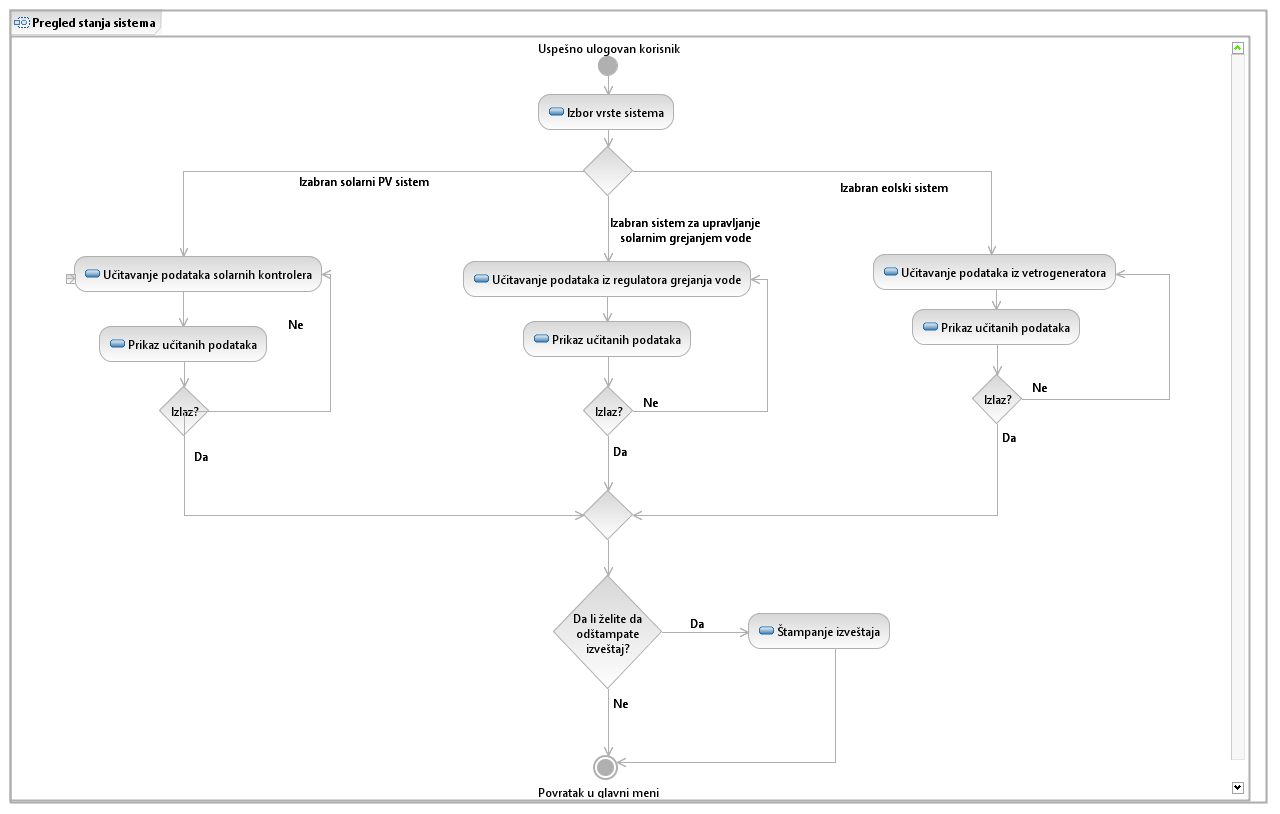
Korisnički zahtevi od kojih ne zavisi aplikacija:

* Veličina sistema kojeg korisinik koristi (pr. snaga PV/eolskog/geotermalnog sistema u KW),
* Kompletnost i način dizajna sistema (pr. da li je PV sistem dizajniran kao off-grid ili on-grid, da li generiše samo jednosmernu struju ili je pretvara u naizmeničnu, da li koristi akumulatore,...),
* Način projektovanja postojeće LAN mreže korisnika

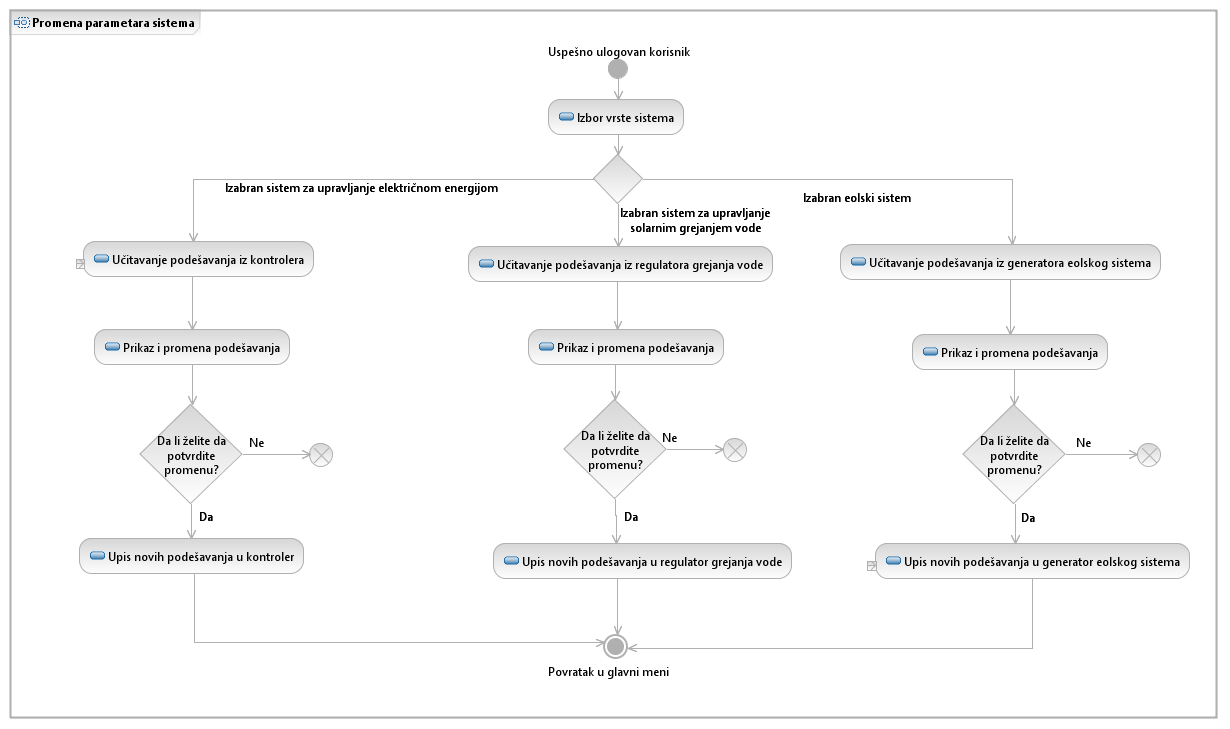
Biće detaljnije obrađeni use case-ovi: ,,*Pregled stanja sistema”* i ,,*Promena parametara sistema*”.



Slika 2:Use case dijagram



Slika 3:Dijagram aktivnosti za use case ,,Pregled stanja sistema"



Slika 4:Dijagram aktivnosti za use case ,,Promena parametara sistema”

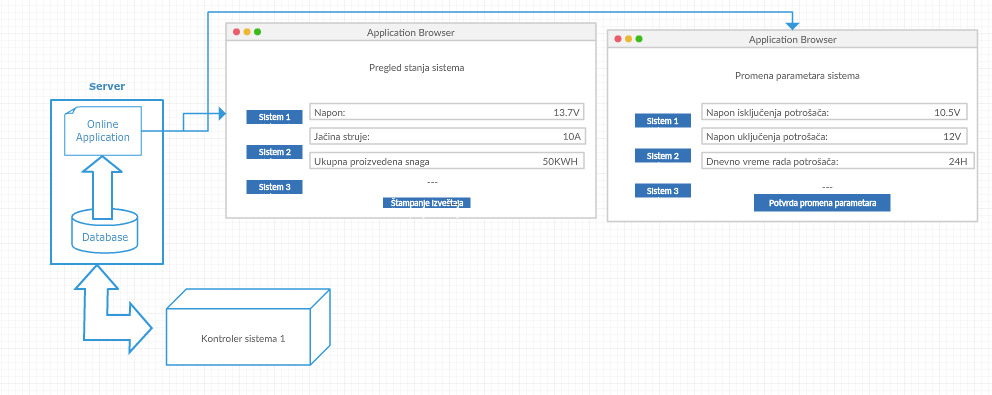
# Analiza aplikacije

## Arhitektura sistema

Ova aplikacija koristi MVC arhitekturu. Ona je prvobitno formulisana krajem '70 godina od strane Norveškog informatičara *Trygve Reenskaug*-a kao deo ,,*Smalltalk*" programskog jezika. Danas se MVC koristi u velikom broju modernih Web i GUI framework-a kao što su CodeIgniter, Ruby on Rails, Apple Cocoa, Apache Struts, i mnogi drugi. MVC deli aplikacije na tri različite komponente i definiše njihove međusobne interakcije:

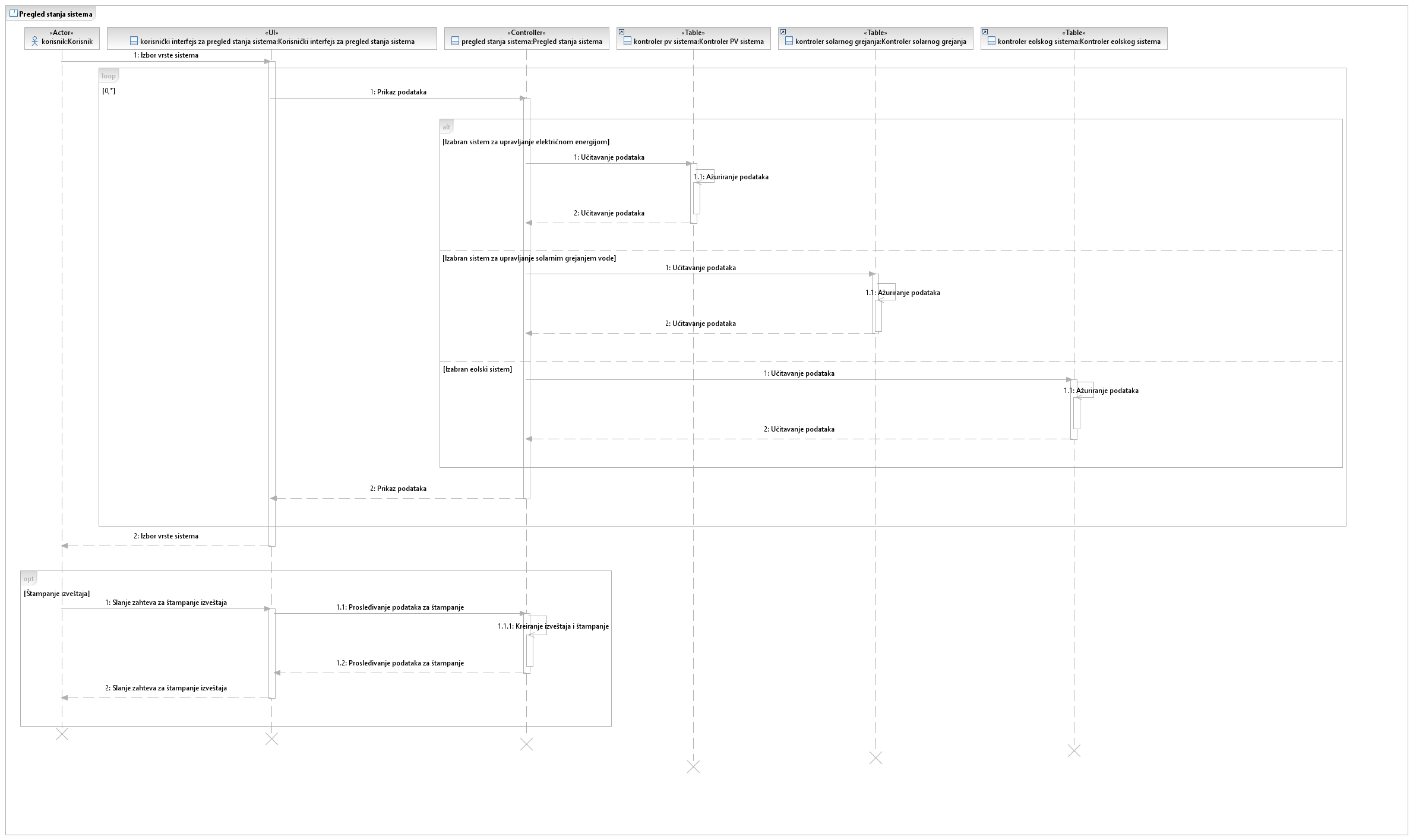
* **Model** koji je zadužen za procesiranje događaja koje mu kontroler prosledi i vraćanje obrađenih podataka kontroleru,
* **Pogled (View)** koji korisniku prezentuje podatke primljene od kontrolera i prima korisničke zahteve koje prosleđuje kontroleru,
* **Kontroler (Controller)** koji služi kao posrednik izmeću pogleda i modela. On preuzima korisničke zahteve od pogleda, manipuliše njima i prosleđuje ih modelu koji vraća određene podatke koji se šalju pogledu na prikaz.

Glavne prednosti MVC arhitekture su ponovno korišćenje koda i podela sistema na međusobno nezavisne celine, a mane te da je previše kompleksan za implementaciju kod razvija manjih aplikacija i njegovo korišćenje u tim slučajevima dovodi do pogoršanja kako dizajna tako performansi same aplikacije.

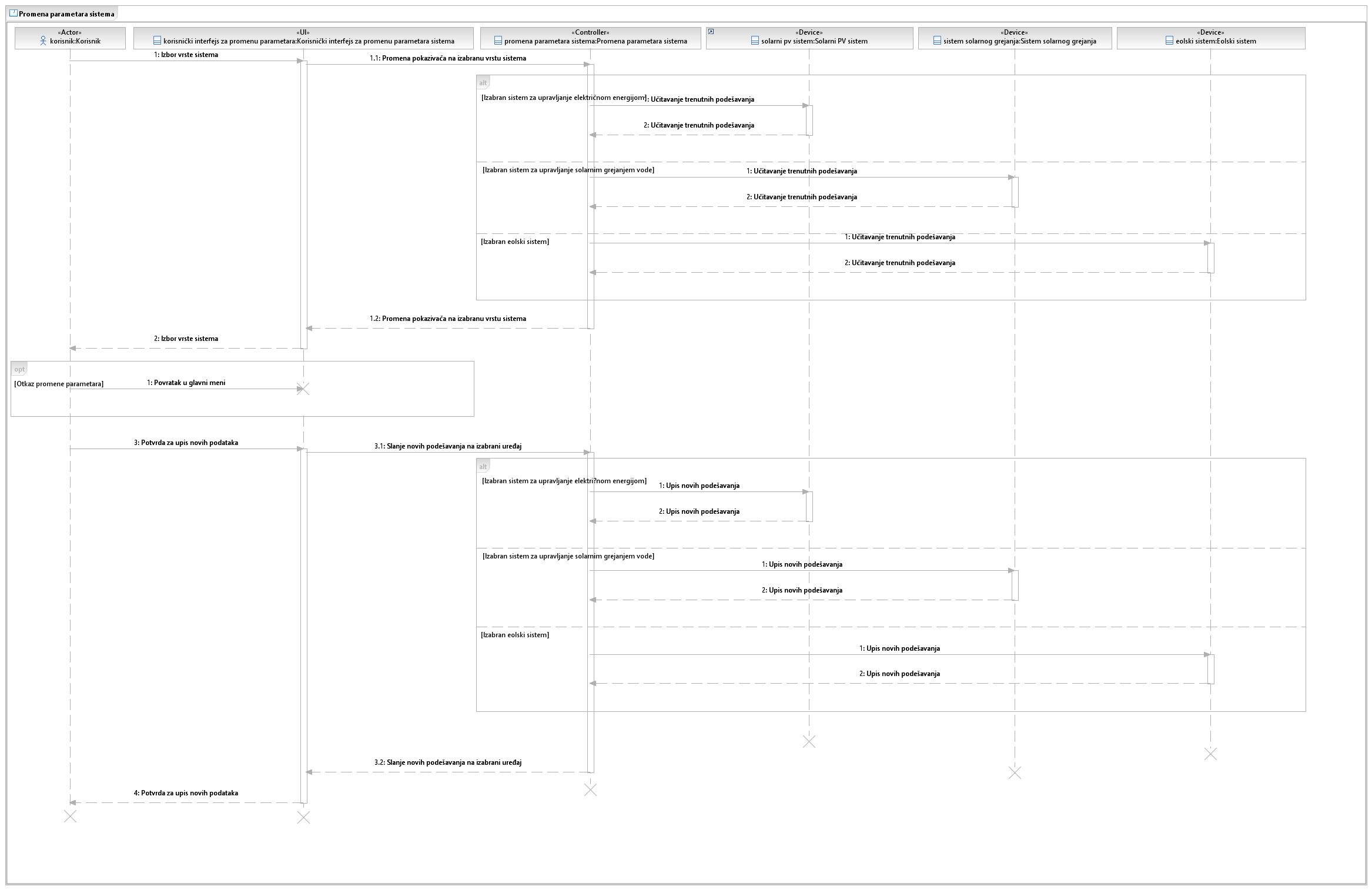


Slika 5:Arhitektura sistema

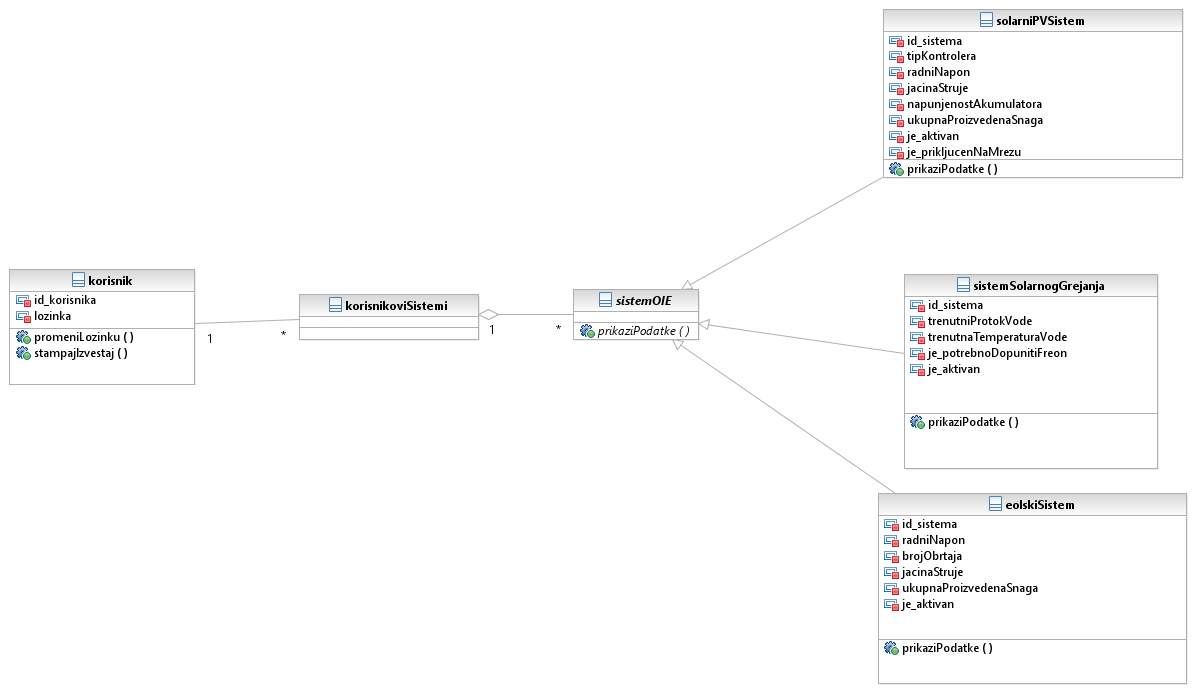
## Analiza slučajeva korišćenja



Slika 6:Dijagram sekvenci za use case ,,Pregled stanja sistema”

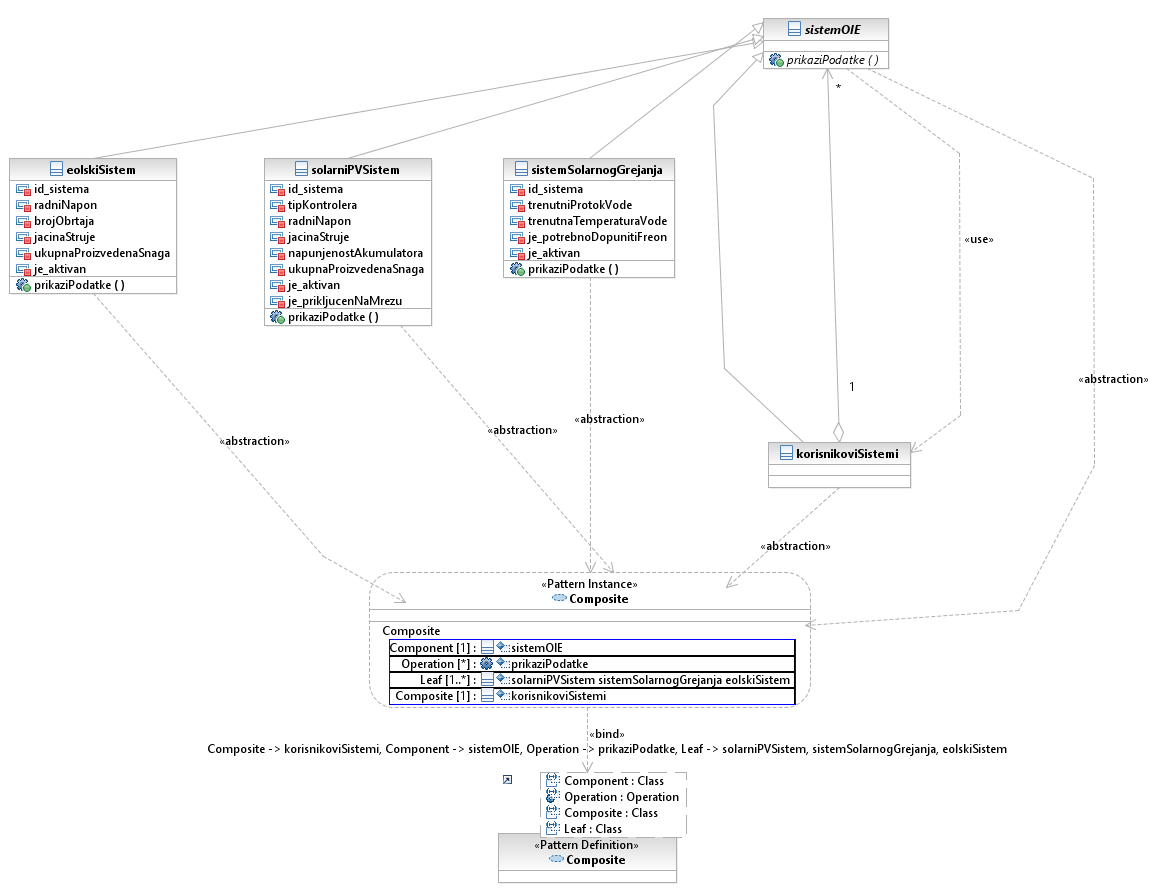


Slika 7:Dijagram sekvenci za use case ,,Promena parametara sistema”

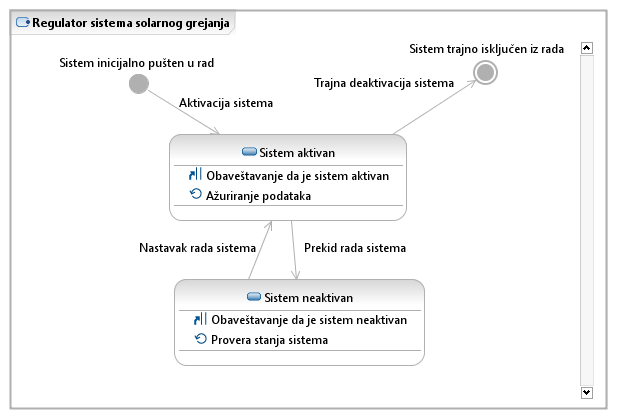


Slika 8:Logički dijagram klasa

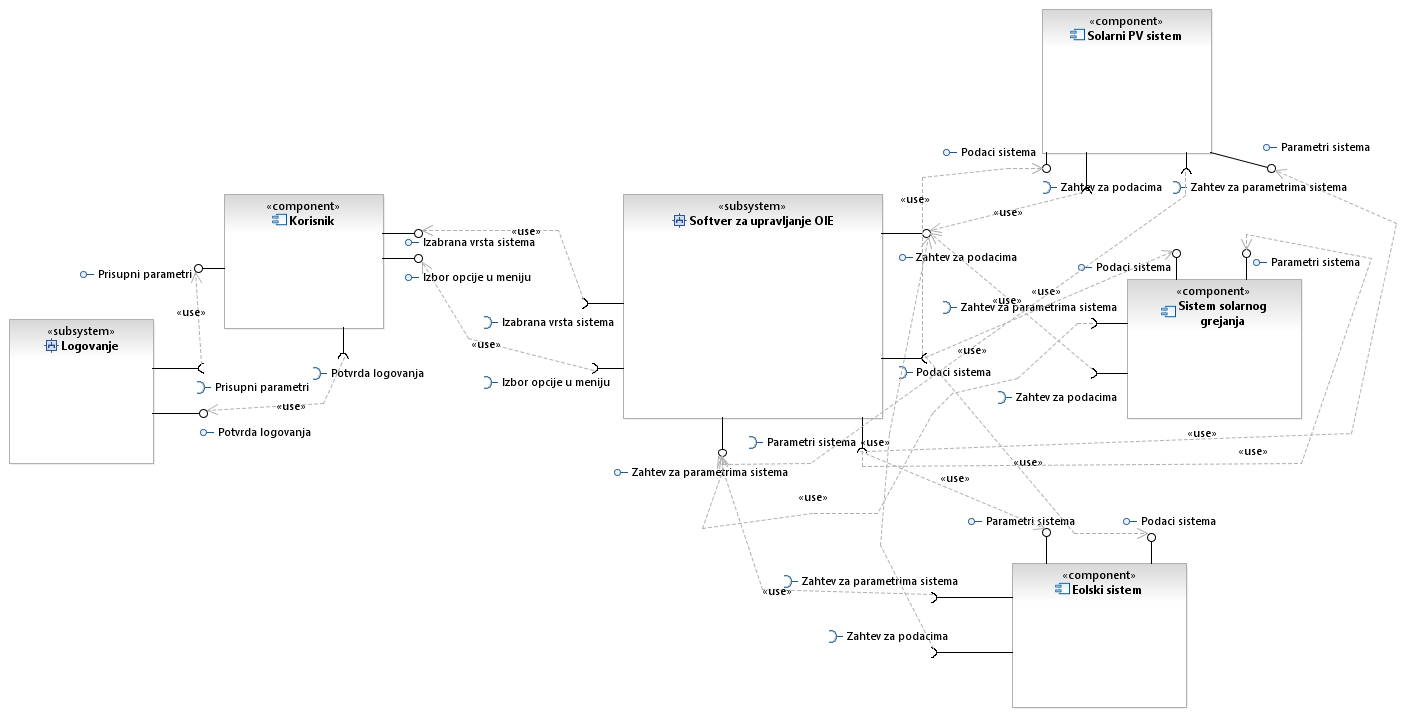
# Projektovanje aplikacije



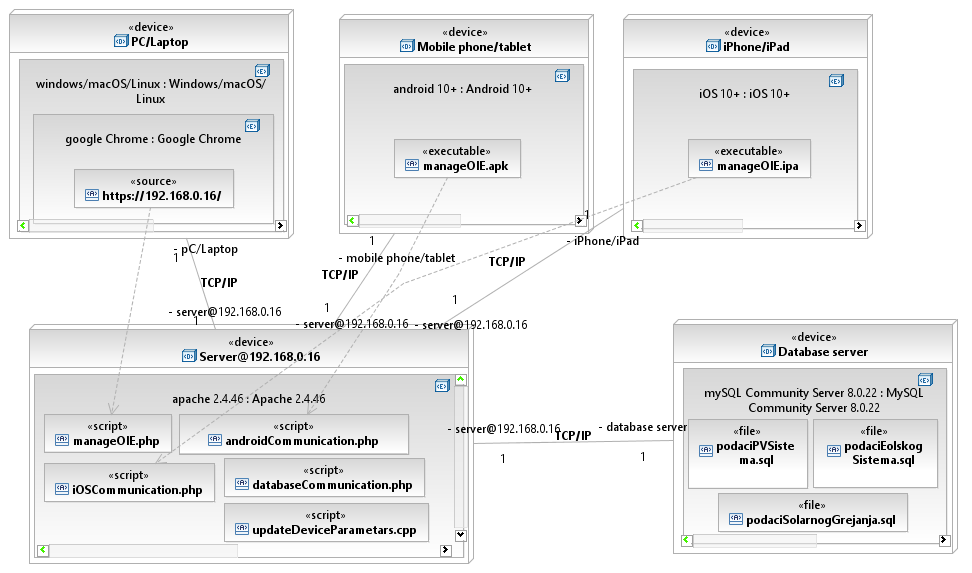
Slika 9: Dizajn patern



Slika 10: Dijagram stanja

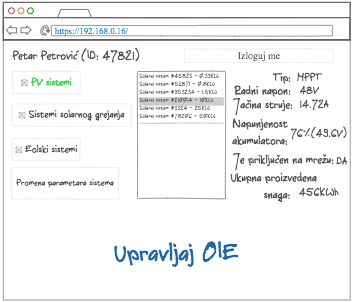


Slika 11:Dijagram komponenti



Slika 12:Dijagram uvođenja

# Projektovanje sloja prezentacije



Slika 13: Prikaz pregleda sistema u pregledaču



*Slike 14-16 redom s leva na desno: Logovanje na android uređaju,*

*Promena parametara sistema na android uređaju, Registracija na iOS uređaju*

# Zaključak

Aplikacija je, zbog male kompleksnosti problematike koje ona rešava, jednostavana. Za dalji razvoj trebalo bi da se utvrdi kolika je mogućnost proizvodnje i prelaska na univerzalan hardver koji bi ova aplikacija koristila.

Prednost ove aplikacije bi bila mogućnost da ona bude univerzalna i korisiti univerzalne kontrolere (pr. Arduino) za sve vrste sistema, a mana je što bi zahtevala stalno održavanje i unapređivanje.

# ****Literatura****

1. Njeguš, A. (2017) Poslovni informacioni sistemi. Univerzitet Singidunum, Beograd.
2. Mihajlović I. (2012) MVC arhitektura Dostupno na: https://mint.rs/blog/tech/mvc-arhitektura/ [pregledano: 10.12.2020]
3. Energija sunca. Dostupno na: https://www.solarnipaneli.org/energija-sunca/ [pregledano: 10.12.2020]
4. Energija vetra. Dostupno na: https://www.solarnipaneli.org/energija-vetra/ [pregledano: 10.12.2020]
5. Obnovljivi izvori energije. U Wikipediji. Dostupno na: https://sr.wikipedia.org/wiki/Обновљиви\_извори\_енергије [pregledano: 10.12.2020]

# ****Popis akronima****

|  |  |  |
| --- | --- | --- |
| Akronim | Naziv na engleskom | Naziv na srpskom |
| OIE | Renewable energy sources | Obnovljivi izvori energije |
| PV | Photovoltaic | Fotonaponski |
| WLAN | Wireless local area network | Bežična lokalna mreža |
| KW | Kilowatt | Kilovat |
| LAN | Local area network | Lokalna mreža |
| Hz | Hertz | Herc |
| SAD | United States of America | Sjedinjene američke države |
| MVC | Model-View-Controller | Model-Pogled-Upravljač |
| GUI | Graphical user interface | Grafički korisnički interfejs |

# ****Prilog****

1. Deo programskog koda za klasu „solarniPVSistem”

**private** **int** id\_sistema, brojObrtaja, radniNapon;

**private** **double** jacinaStruje, ukupnaProizvedenaSnaga;

**private** **boolean** je\_aktivan;

**public** eolskiSistem(**int** id\_sistema, **int** brojObrtaja, **int** radniNapon,

**double** jacinaStruje, **double** ukupnaProizvedenaSnaga,

**boolean** je\_aktivan) {

**super**();

**this**.id\_sistema = id\_sistema;

**this**.brojObrtaja = brojObrtaja;

**this**.radniNapon = radniNapon;

**this**.jacinaStruje = jacinaStruje;

**this**.ukupnaProizvedenaSnaga = ukupnaProizvedenaSnaga;

**this**.je\_aktivan = je\_aktivan;

}

….

**public** **void** prikaziPodatke() {

System.*out*.println("---------------------");

System.*out*.println("Eolski sistem #" + **this**.getId\_sistema());

System.*out*.println("Radni napon: " + **this**.getRadniNapon() + "V");

System.*out*.println("Jacina struje: " + **this**.getJacinaStruje() + "A");

System.*out*.println("Broj obrtaja: " + **this**.getBrojObrtaja() + " rpm");

System.*out*.println("Ukupna proizvodnja: " +

**this**.getUkupnaProizvedenaSnaga()/1000 + " kWh");

System.*out*.println("Status sistema: " +

(**this**.getJe\_aktivan() ? "AKTIVAN" : "NEAKTIVAN"));

System.*out*.println("---------------------");

}

1. Programski kod glavne klase („Klijent“)

**public** **static** **void** main(String[] args) {

solarniPVSistem solar\_1kw = **new** solarniPVSistem(234,"PWM", 48, 0.0, 100, 862200.46, **false**, **false**);

solarniPVSistem solar\_2kw = **new** solarniPVSistem(321, "PWM", 24, 83.34, 89, 1124400.37, **true**, **false**);

solarniPVSistem solar\_5kw = **new** solarniPVSistem(417, "MPPT", 48, 104.16, 68, 447123.567, **true**, **true**);

korisnikoviSistemi solarniSistemi = **new** korisnikoviSistemi();

solarniSistemi.addChild(solar\_1kw);

solarniSistemi.addChild(solar\_2kw);

solarniSistemi.addChild(solar\_5kw);

sistemSolarnogGrejanja solarHeating\_2

= **new** sistemSolarnogGrejanja(774, 7.6, 28.7, **false**, **true**);

sistemSolarnogGrejanja solarHeating\_4

= **new** sistemSolarnogGrejanja(953, 14.12, 35.7, **true**, **true**);

korisnikoviSistemi sistemiSolarnogGrejanja

= **new** korisnikoviSistemi();

sistemiSolarnogGrejanja.addChild(solarHeating\_2);

sistemiSolarnogGrejanja.addChild(solarHeating\_4);

eolskiSistem wind\_15kw

= **new** eolskiSistem(104,15,96,156.25,75115.0,**true**);

eolskiSistem wind\_30kw

= **new** eolskiSistem(108,0,96,0.0,30000.0,**false**);

korisnikoviSistemi eolskiSistemi = **new** korisnikoviSistemi();

eolskiSistemi.addChild(wind\_15kw);

eolskiSistemi.addChild(wind\_30kw);

korisnikoviSistemi sistemiOIE = **new** korisnikoviSistemi();

sistemiOIE.addChild(solarniSistemi);

sistemiOIE.addChild(sistemiSolarnogGrejanja);

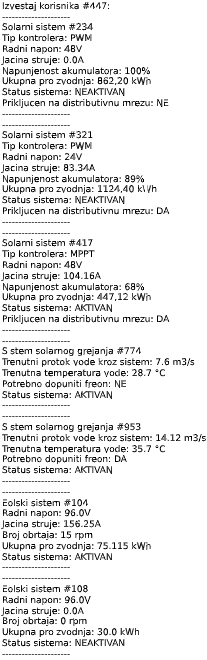
sistemiOIE.addChild(eolskiSistemi);

System.*out*.println("Izvestaj korisnika #447:");

sistemiOIE.prikaziPodatke();

}

}



Slika 17: Prikaz konzolnog izlaza nakon pokretanja softvera