Angular JS aplikacija za raspoređivanje radnog vremena

Rajkovača, Ivan

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

Preddiplomski sveučilišni studij računarstva

ANGULAR JS APLIKACIJA ZA RASPOREĐIVANJE RADNOG VREMENA

Završni rad

Ivan Rajkovača

Osijek, 2020.

SADRŽAJ

1. UVOD 1
1.1. Zadatak završnog rada 1
2. RASPOREĐIVANJE RADNOG VREMENA RAČUNALNIM TEHNOLOGIJAMA 2
3. KORIŠTENI ALATI I RAZVOJNA OKRUŽENJA 6
3.1. Visual Studio Code 6
3.2. JavaScript 7
3.3. AngularJS 8
3.4. HTML 10
3.5. CSS 11
3.6. PHP 12
4. STRUKTURA I NAČIN RADA APLIKACIJE 13
4.1. Unos radnih zadataka 13
4.2. Unos zaposlenika 14
4.3. Generiranje tabličnog sučelja za raspoređivanje 16
4.4. Generiranje rasporeda za zaposlenike 17
5. ZAKLJUČAK 19
LITERATURA 20
SAŽETAK 21
ABSTRACT 22
ŽIVOTOPIS 23

1. UVOD

Tema ovog završnog rada je izrada AngularJS aplikacije za raspoređivanje radnog vremena jednog radnog dana. Cilj ove aplikacije je omogućiti unos zaposlenika jedne tvrtke, koji rade u više smjena. Također, potrebno je definirati radne zadatke te odrediti koji zaposlenik je u stanju obaviti koje radne zadatke. Kako bi aplikacija imala potpunu funkcionalnost potrebno je stvoriti tablično sučelje u kojem je moguće rasporediti zaposlenike po zadanim poslovima u zadanom vremenu. Aplikacija također mora uzimati u obzir radno vrijeme zaposlenika koji rade u više smjena, stoga određeni zaposlenik ne može obaviti sve radne zadatke, već one koje je sposoban obaviti i koji se obavljaju za vrijeme trajanja njegove smjene. Prva funkcija aplikacije je definiranje radnih zadataka, što se obavlja preko HTML oznaka za unos podataka u tekstualnom obliku. Druga funkcija je unos zaposlenika koji se obavlja na isti način. Na kraju se stvara tablično sučelje u kojem korisnik ručno raspoređuje koji zaposlenik obavlja koji zadatak. Kada je korisnik gotov s raspoređivanjem stvara se konačna tablica s radnim zadacima i dodijeljenim zaposlenicima. U sljedećem koraku se stvara raspored radnih zadataka po zaposlenicima koji svakom zaposleniku daje uvid u njegov radni dan, odnosno sve što zaposlenik treba obaviti kako bi se njegov radni dan smatrao uspješnim.

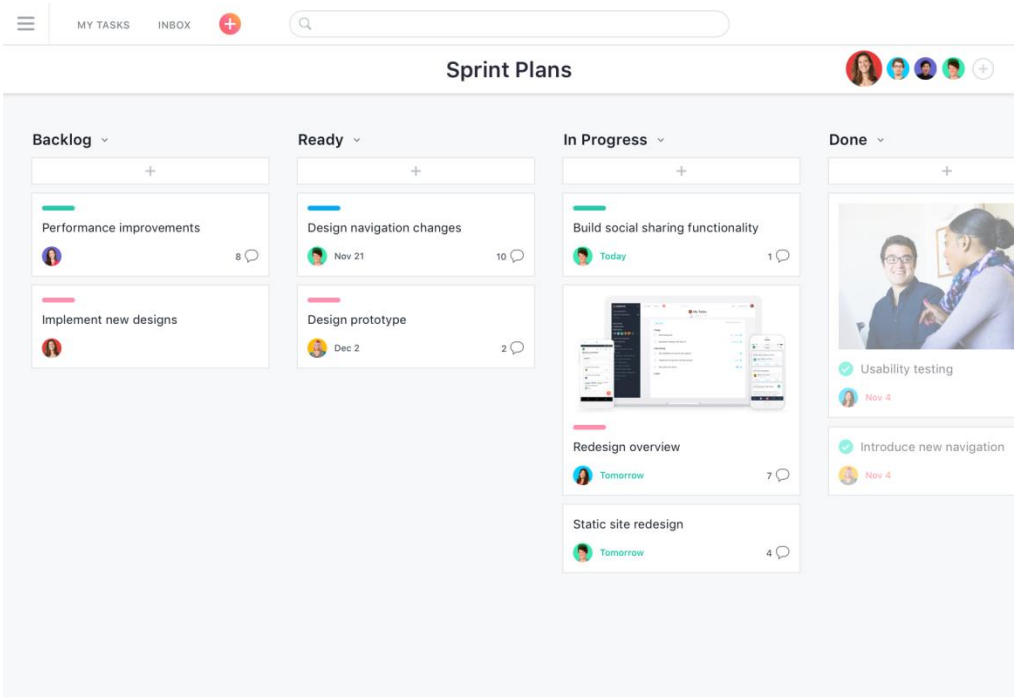
1.1. Zadatak završnog rada

Potrebno je izraditi web aplikaciju koristeći AngularJS skriptni jezik. Aplikacija omogućuje korisniku unos radnih zadataka i zaposlenika jedne tvrtke. Također, aplikacija ima mogućnost generiranja tabličnog sučelja pomoću kojeg korisnik dodjeljuje radne zadatke zaposlenicima unutar jednog radnog dana.

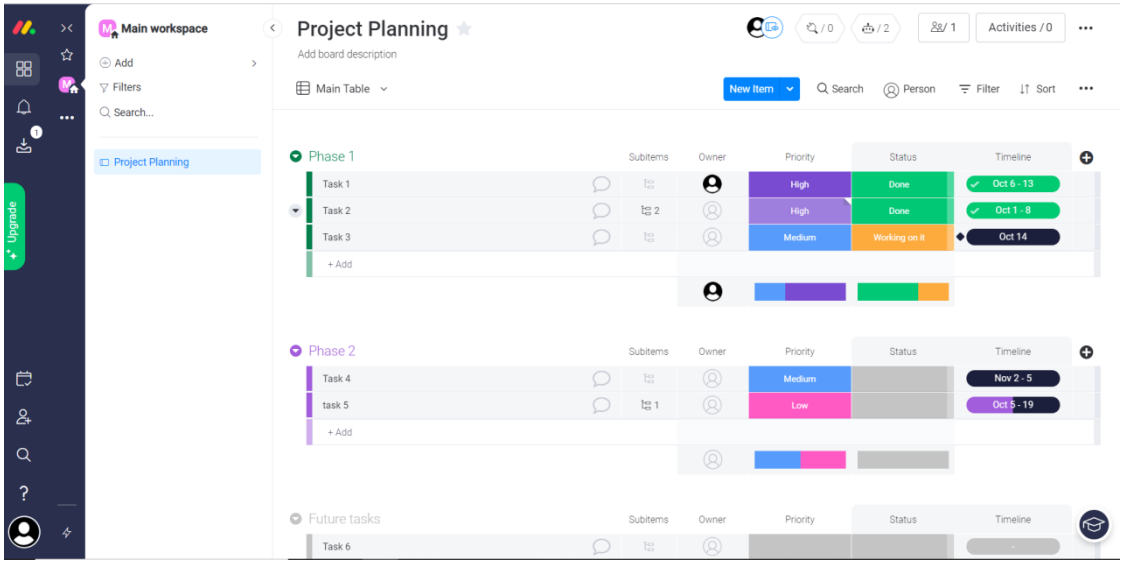
2. RASPOREĐIVANJE RADNOG VREMENA RAČUNALNIM TEHNOLOGIJAMA

U današnje vrijeme tehnologija je sve više prisutna u svim sferama života, a dobra organizacija radnog vremena je nužna za uspješno izvođenje bilo kojeg posla. Korištenjem računalnih tehnologija nastale su brojne aplikacije za raspoređivanje radnog vremena koje su besplatne i dostupne svima. AngularJS je popularno rješenje za izradu *single-page* aplikacija (SPA), a većina trenutno dostupnih aplikacija za raspoređivanje radnog vremena rade na istom principu. U takvim aplikacijama se stvara iluzija promjene stranica, odnosno sve promjene se odvijaju dinamički u pozadini, a korisnik se cijelo vrijeme nalazi na istoj stranici. Time se ostvaruje bolje korisničko iskustvo, jer korisnik ne čeka učitavanje novih stranica sa servera. Dobra aplikacija za raspoređivanje radnog vremena mora biti pouzdana, fleksibilna i učinkovita što je moguće ostvariti korištenjem AngularJS-a. Brojni stručnjaci predviđaju svijetlu budućnost AngularJS-a u području izrade aplikacija koje se bave raspoređivanjem radnog vremena. Posebno se cijeni brzina i fleksibilnost aplikacija izrađenih korištenjem AngularJS-a. Na internetu postoje brojne aplikacije za raspoređivanje radnog vremena, ali najpopularnije su Asana i Monday. Neke od mogućnosti tih aplikacija su praćenje napretka, postavljanje prioriteta i definiranje broja zaposlenika potrebnih za odrađivanje nekog zadatka. Aplikacijom Asana omogućeno je kvalitetno organiziranje radnog vremena za tvrtke u kojima zaposlenici rade u timovima. Aplikacijom je omogućeno organiziranje manjih, ali i puno većih timova ljudi. Također, korištenjem aplikacije se u svakom trenutku dobiva uvid u napredak koji je tim ostvario. Sučelje aplikacije Asana prikazano je slikom 2.1. [1] Aplikacija Monday slična je aplikaciji Asana. Monday aplikacijom je omogućeno planiranje rada i komuniciranje unutar timova. Pregled radnih zadatak prikazan je u obliku tabličnog sučelja. Izgled aplikacije Monday prikazan je na slici 2.2. [2] Popularnim izborom se također smatra aplikacija ProjectManager čije je sučelje prikazano slikom 2.3. [3] ProjectManager aplikacija nije besplatna, ali je korisnicima omogućeno besplatno probno razdoblje unutar kojeg su im dostupne sve mogućnosti aplikacije. ProjectManager aplikacijom se koriste brojni timovi koji rade za velike kompanije širom svijeta. Aplikacijom se prate radni zadaci prikazani unutar sučelja nalik kalendarima. Za sve one koji žele kvalitetno organizirati svoje obveze preporučuje se aplikacija OmniFocus prikazana slikom 2.4. [4] Aplikacija OmniFocus je moćan i fleksibilan alat za raspoređivanje zadataka unutar radnog vremena. Aplikacijom je omogućeno brzo i jednostavno dodavanje radnih zadataka u stvarnom vremenu. Korištenjem aplikacije OmniFocus produktivnost timova se uvelike povećava, što je dokazano brojnim istraživanjima. Također, tvrtka Microsoft je predstavila

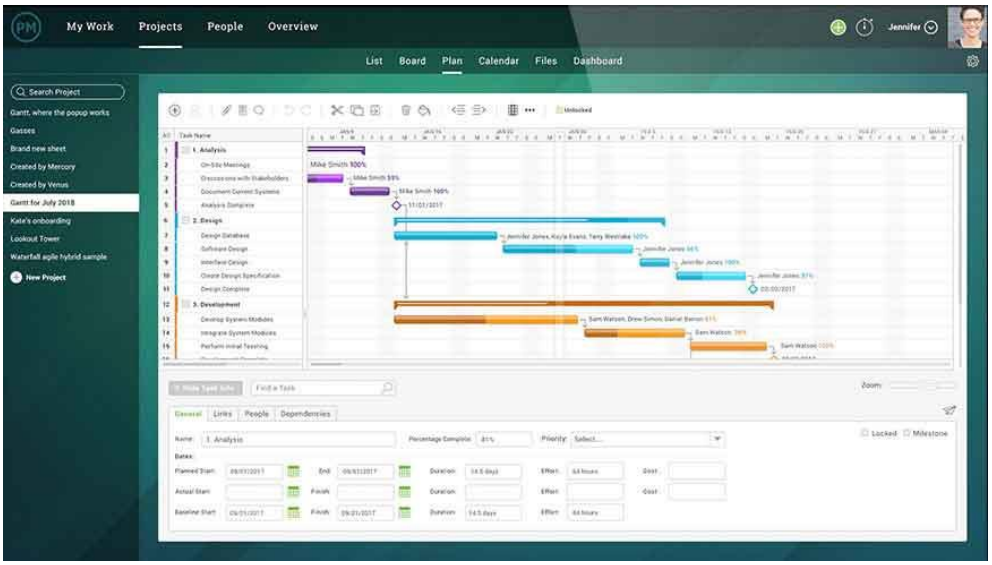
aplikaciju To Do koja pruža najmoderniji dizajn i jednostavno korisničko sučelje. Aplikacija To Do može se koristiti na pametnim telefonima, tabletima ili računalima. Aplikacija je javno dostupna od 2017. godine. Aplikacija To Do dokazano smanjuje stres i povećava produktivnost unutar timova koji rade za velike tvrtke. Izgled aplikacije To Do prikazan je na slici 2.5. [5]



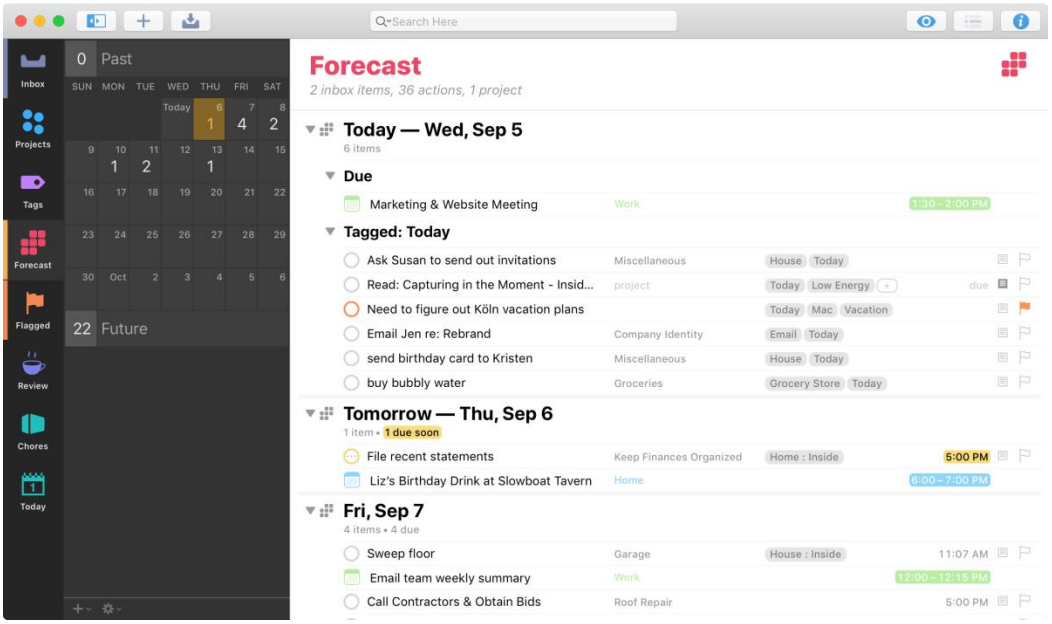
Sl. 2.1 *Sučelje web aplikacije Asana*



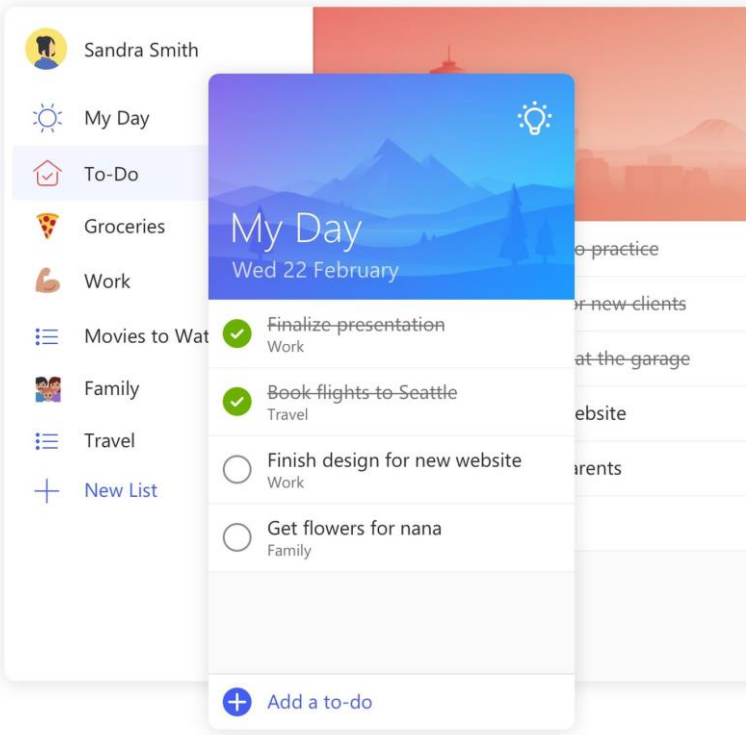
Sl. 2.2 *Sučelje web aplikacije Monday*



Sl. 2.3 *Sučelje web aplikacije ProjectManager*



Sl. 2.4 *Sučelje web aplikacije OmniFocus*



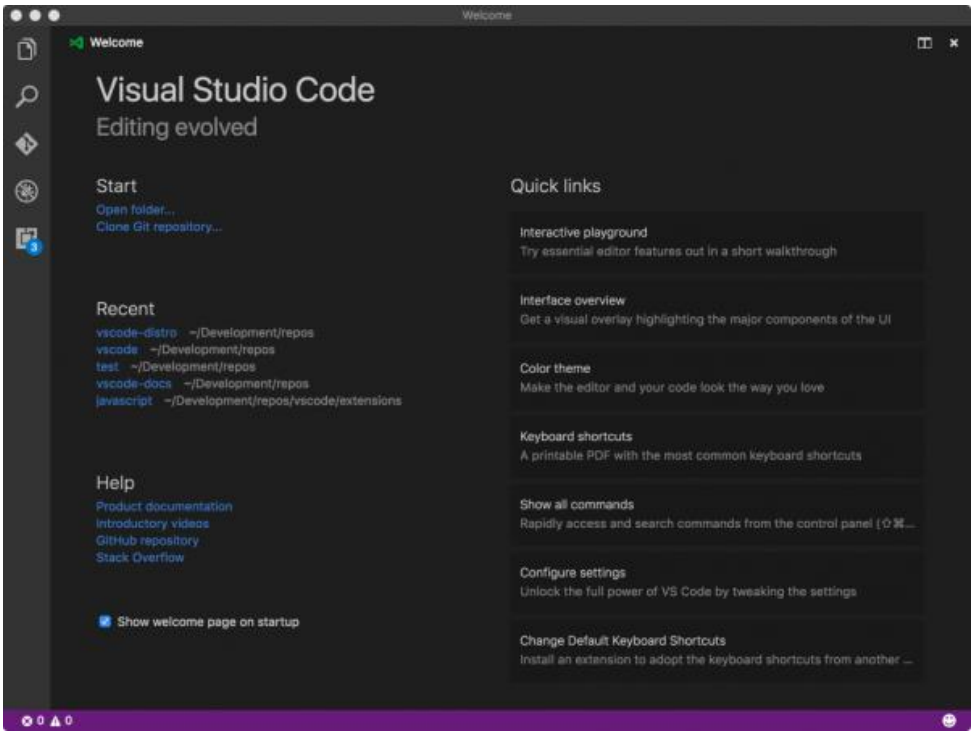
Sl. 2.5 *Sučelje web aplikacije OmniFocus*

3. KORIŠTENI ALATI I RAZVOJNA OKRUŽENJA

U ovom poglavlju opisani su alati korišteni za razvoj aplikacije, odnosno korišteni programski jezici koji su potrebni kako bi aplikacija imala potpunu funkcionalnost. Također, opisano je i razvojno okruženje korišteno u izradi aplikacije.

3.1. Visual Studio Code

Visual Studio Code je razvojno okruženje kojeg je stvorila tvrtka Microsoft. Sučelje razvojnog okruženja prikazano je slikom 3.1. Radi se o okruženju s izvornim kodom koji je javno dostupan (engl.*Open-source*). Koristi se za razvoj web stranica, aplikacija i ostalih usluga, a podržava i uklanjanje pogrešaka, refaktoriranje koda te brojne druge mogućnosti. Okruženje omogućuje dodavanje brojnih proširenja koja poboljšavaju funkcionalnost na skoro svakom nivou te pružaju dodatnu podršku sistema za upravljanje izvornim kodom. Visual Studio Code podržava programske jezike C, C++, C++/CLI, C# i F#, a podršku za ostale programske jezike potrebno je zasebno instalirati. Razvojno okruženje je najavljeno 29.4.2015. , a postalo je javno dostupno 18.11.2015. kada je izvorni kod postavljen na GitHub. [6]



Sl. 3.1 *Visual Studio Code*

3.2. JavaScript

JavaScript je skriptni programski jezik namijenjen razvoju interaktivnih HTML stranica. Primjer JavaScript koda prikazan je slikom 3.2. Naziv JavaScript nije vezan s nazivom programskog jezika Java. Izvorno se JavaScript trebao zvati LiveScript, ali kako bi se potaklo korisnike na korištenje novog jezika, nazvan je slično jeziku Java, koji se smatra vrlo uspješnim programskim jezikom. Iako se radi o različitim jezicima, JavaScript je vrlo sličan jezicima Java i C++ od kojih nasljeđuje sintaksu i brojne ugrađene naredbe (engl. *statements*). JavaScript se smatra jednim od najpopularnijih rješenja za brojne programere zbog jednostavne sintakse i lakog upravljanja memorijom te podrške za brojne preglednike. Jezik je najviše korišten u razvoju web stranica, ali postoje primjeri uspješne upotrebe ovog jezika u okruženjima koji ne zahtijevaju korištenje preglednika (engl. *browser*). JavaScript se pokreće na klijentskoj strani što omogućuje promjenu izgleda i ponašanja web stranica kada se pojavi neki od događaja (engl. *event*). Varijable se deklariraju ključnim riječima *var* i *let*, a konstante ključnom riječi *const*. Varijable koje su dostupne u cijelom programu nazvane su globalne varijable, a varijable koje su dostupne samo unutar funkcije lokalne varijable. Ključna riječ za deklariranje funkcije je *function*. Prilikom deklariranja funkcije nije potrebno navesti popis argumenata. [7]

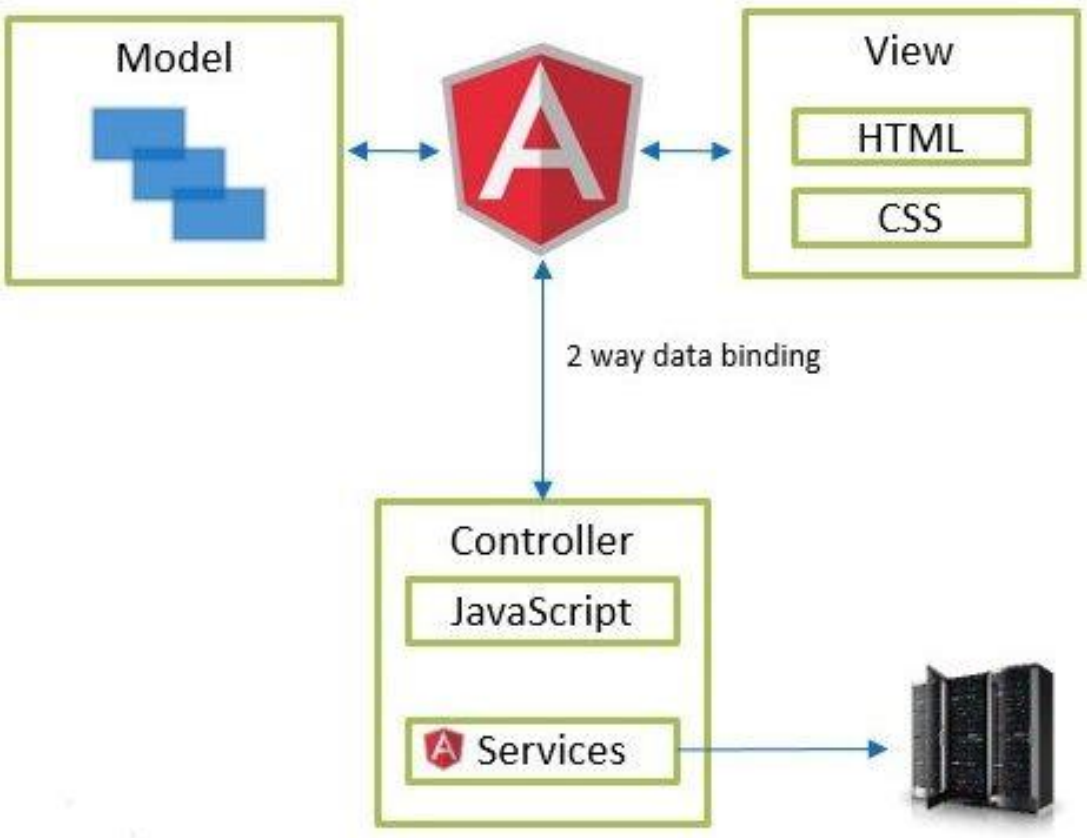
```
JS script.js
//OVDJE POČINJE KOD ZA ZAPOSLENIKE
$scope.employees = [];

$scope.flname = "";
$scope.oib = "";
$scope.shift = "";
$scope.employeeCounter = 0;

function Employee(flname, oib, shift, tasksDoing) {
  this.flname = flname;
  this.oib = oib;
  this.shift = shift;
  this.tasksDoing = tasksDoing;
}

$scope.saveEmployee = function () {
  if (checkEmployeeInputs()) {
    $scope.employees[$scope.employeeCounter] = new Employee(
      $scope.flname,
      $scope.oib,
      $scope.shift,
      getCheckedTasks()
    );
  } else {
    alert("Netočan unos!");
    resetEmployeeInputs();
    return;
  }
  $scope.employeeCounter++;
  resetEmployeeInputs();
};

$scope.logEmployees = function () {
  console.log($scope.employees[$scope.employeeCounter - 1]);
};

var resetEmployeeInputs = function () {
```

Sl. 3.2 *Isječak JavaScript koda korištenog u aplikaciji*

3.3. AngularJS

AngularJS je programski okvir (engl. *framework*) korišten za razvoj web aplikacija. U najvećoj mjeri ga održava Google i zajednica programera koji nailaze na brojne izazove prilikom razvoja *single-page* aplikacija (SPA), odnosno aplikacija čiji se kompletan sadržaj nalazi na jednoj stranici. Glavna zadaća ovog *framework*-a je pojednostaviti razvoj i testiranje takvih aplikacija. AngularJS radi na principu model-pogled-upravljač (engl. *Model-View-Controller*) kao što je prikazano na slici 3.3. Kako bi se koristio AngularJS potrebno ga je uključiti u izvorni HTML dokument tako što se unutar *script* oznake navede veza na verziju AngularJS-a koju se planira koristiti. AngularJS proširuje klasični HTML s dodatnim atributima omogućujući interakciju s korisnikom. Glavni dio interakcije s korisnikom obavlja se dvosmjernim povezivanjem podataka što omogućuje uzajamno povezivanje modela i pogleda. Dvosmjerno povezivanje omogućuje da svaka promjena na modelu uzrokuje

promjenu na pogledu i obrnuto. Ovakvim načinom rada se u velikoj mjeri rasterećuje server što čini osnovu svake *single-page* aplikacije. Prilikom takvog načina rada pogled se stvara u čistom HTML-u prema podacima koji se nalaze unutar dosega (engl. *scope*). AngularJS je originalno razvio Miško Havery 2009. godine. Od tada su postale dostupne brojne verzije AngularJS-a pri čemu je svaka nova verzija donosila niz novih mogućnosti, što je uvelike olakšalo razvoj i testiranje single-page aplikacija. Iako je AngularJS vrlo uspješan *framework* brojni stručnjaci kritiziraju brzinu izvođenja skripti napisanih u AngularJS-u koja je znatno sporija kada se unutar dosega nalazi velik broj podataka, odnosno više od 2000. [8]



Sl. 3.3 *Model-pogled-upravljač*

3.4. HTML

HTML je kratica za HyperText Markup Language, što prevedeno znači prezentacijski jezik za kreiranje internet stranica. Strogo gledano HTML nije programski jezik, nego jednostavan jezik za postavljanje sadržaja u dokument internet stranica. Danas se HTML uglavnom koristi za kreiranje "kostura" web stranice, dok se oblikovanje web stranice radi preko CSS-a. Prikaz HTML dokumenata omogućuje web preglednik. HTML dokument sastoji se od oznaka (engl. *tags*) koje definiraju kako će web preglednik nešto prikazati. Oznake unutar HTML dokumenta uglavnom se otvaraju korištenjem početne oznake koja izgleda kao <naziv_oznake>, a zatvaraju završnom oblika </naziv_oznake> (npr. <body>Neki sadržaj web stranice</body>). Sve oznake koje se mogu koristiti prikazane su na slici 3.4. HTML stranice su statične. Kako bi web stranica bila dinamična, koriste se razne druge tehnologije kao što su PHP, ASP i JavaScript. Iako je HTML osnova svake web stranice, u većini slučajeva čini manje od 10 posto ukupnog koda.[9]

- <!-- -->
- <!DOCTYPE>
- <a>
- <applet>
- <area>
- <audio>
- <b>
- <base>
- <big>
- <blockquote>
- <body>
-
- <button>
-
- <code>
- <del>
-
-
- <embed>
- <font>
- <form>
- <frame>
- <frameset>
- <h1>
- <h2>
- <h3>
- <h4>
- <h5>
- <h6>
- <head>
- <hr>
- <html>
- <i>
- <iframe>
- <img>
- <ins>
-
- <map>
- <meta>
- <noframes>
- <noscript>
- <object>
-
- <p>
- <param>
- <pre>
- <q>
- <s>
- <samp>
- <script>
- <small>
- <video>
- <title>
-
- <strike>
-
- <style>
- <sub>
- <sup>
- <table>
- <tbody>
- <tfoot>
- <td>
- <thead>
- <th>
- <tr>
- <u>
- <var>
- <wbr>

Sl. 3.4 *Popis oznaka u HTML-u*

3.5. CSS

CSS je kratica za Cascading Style Sheets. Smatra se stilskim jezikom koji daje upute pregledniku kako prikazati HTML elemente. CSS je jednostavan mehanizam za dodavanje stilova: fontova, boja razmaka između paragrafa, uređivanje tablica. Stilski obrasci sačinjeni su od stilskih pravila. Svako pravilo se sastoji od selektora i deklaracije. Deklaracija se sastoji od svojstva i vrijednosti što je prikazano slikom 3.5. Svojim dolaskom CSS je izazvao pravu revoluciju na internetu i to zahvaljujući nizu prednosti koje ima pred tabličnim *layoutom* (korištenje tablica za formiranje stranice). Korištenjem CSS-a postalo je moguće odvojiti prezentaciju sadržaja i dizajn od same strukture dokumenta. HTML kod postaje pregledniji i manji što znači da ga je puno lakše kontrolirati, a također je moguće jednostavnom primjenom parametara promijeniti izgled stranice. [10]

Sl. 3.5 *Primjer CSS koda*

3.6. PHP

PHP: Hypertext Preprocessor je skriptni jezik koji se izvršava na poslužitelju (eng. server), a glavna mu je namjena dinamičko stvaranje web stranica. Primjer jednostavnog PHP koda prikazan je slikom 3.6. PHP je nastao 1994. godine kao osobni projekt Rasmus Lerdorfa, a kasnije se u njegov razvoj uključio veliki broj programera koji su doprinijeli razvoju jezika. Omogućuje programeru brzo razvijanje programa koristeći tehnike proceduralnog i objektno-orijentiranog programiranja. U PHP-u moguće je korištenje mnogih postojećih biblioteka koje su uključene u osnovnu instalaciju ili se mogu instalirati unutar PHP okruženja. Lakoća dodavanja biblioteka u okruženje je jedna od glavnih prednosti i ono što PHP čini popularnim.[11]

```php
<?php
$str_isset = "";
$bol_isset = isset($str_isset);

If ($bol_isset){
   echo "The variable is set";
}
else {
    echo "The variable is not set";
}
?>

```

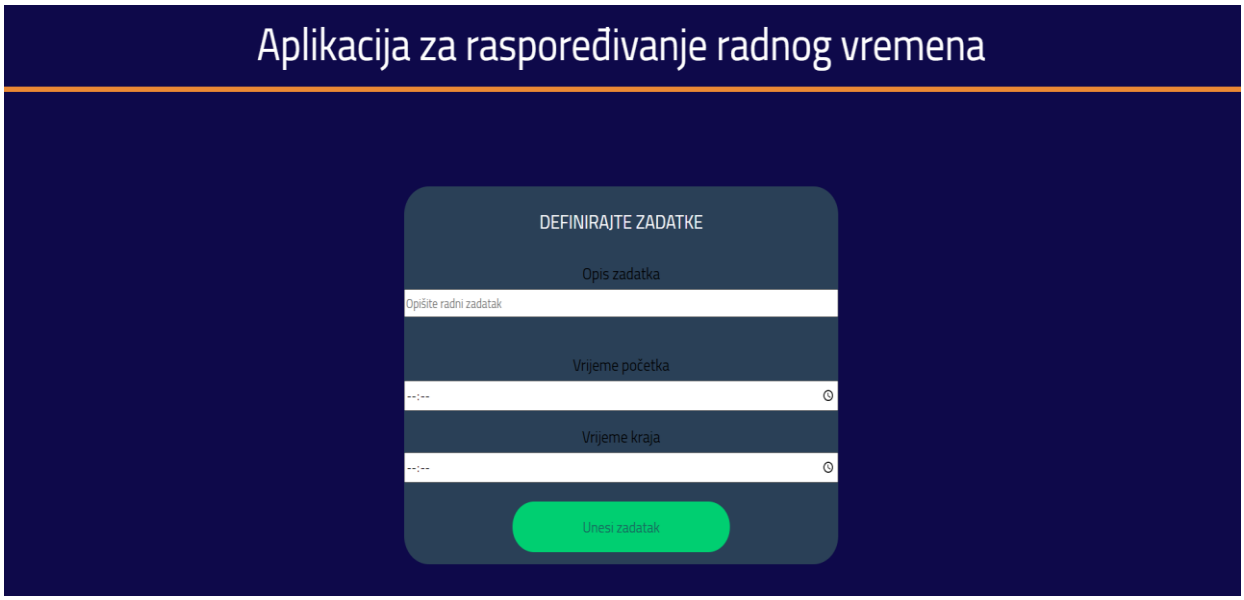
Sl. 3.6 *Jednostavan PHP kod*

4. STRUKTURA I NAČIN RADA APLIKACIJE

U ovom poglavlju opisan je način na koji je ostvareno programsko rješenje aplikacije. Definirani su svi koraci potrebni za funkcioniranje aplikacije i sve što se očekuje od korisnika aplikacije.

4.1. Unos radnih zadataka

Unos radnih zadataka sastoji se od tri djela, odnosno tri različita polja za unos podataka što je prikazano na slici 4.1. Prvo što se očekuje od korisnika prilikom definiranja radnog zadatka je da korisnik navede opis radnog zadatka. Opis radnog zadatka obavlja se unutar prvog elementa za unos podataka, što znači da korisnik radni zadatak opisuje upisivanjem teksta. Za opis radnog zadatka nema posebnih ograničenja, odnosno uvjeta, jer se korisniku daje potpuna sloboda imenovanja radnog zadatka. Sljedeće što se očekuje od korisnika je definiranje vremena početka i vremena kraja imenovanog radnog zadatka. Unos vremena obavlja se u jednostavnom 24-satnom formatu s obzirom na to da aplikacija raspoređuje radno vrijeme unutar jednog radnog dana. Kako bi se radni zadatak smatrao potpuno definiranim od korisnika se očekuje potvrda unosa pritiskom na gumb „Unesi zadatak“. Nakon što je korisnik pritisnuo gumb aktivira se petlja događaja (engl. *Event loop*). Petlja događaja je beskonačna petlja koja klik miša ili pritisak tipke na tipkovnici prepoznaje kao događaj (engl. *event*). Aktiviranjem događaja poziva se funkcija koja obrađuje taj događaj, a petlja se nastavlja izvršavati sve dok se ne aktivira neki novi događaj. Petlja prepoznaje pritisak gumba i poziva funkciju za provjeru unosa. Funkcija za provjeru unosa provjerava je li korisnik popunio sve elemente za unos podataka. Također, petlja provjerava je li vrijeme početka manje od vremena kraja, odnosno ne može se dogoditi slučaj u kojem naprimjer neki posao počinje u 14:00, a završava se u 13:00. Ako je korisnik obavio uspješan unos, odnosno funkcija za provjeru unosa vrati *boolean* tip podatka čija je vrijednost u tom slučaju *true*, unos se smatra uspješnim i radni zadatak se sprema u bazu podataka. Rad s bazom podatka obavlja se pomoću programskog jezika PHP. Svaki radni zadatak se unutar baze podataka sprema kao objekt koji ima tri atributa koji predstavljaju vrijednosti unesene u tri elementa za unos podataka. Glavni razlog korištenja baze podataka je olakšano testiranje rada aplikacije. U slučaju kada se baza podataka ne bi koristila, tada bi se prilikom svakog testiranja aplikacije moralo ponovo unositi svaki radni zadatak.

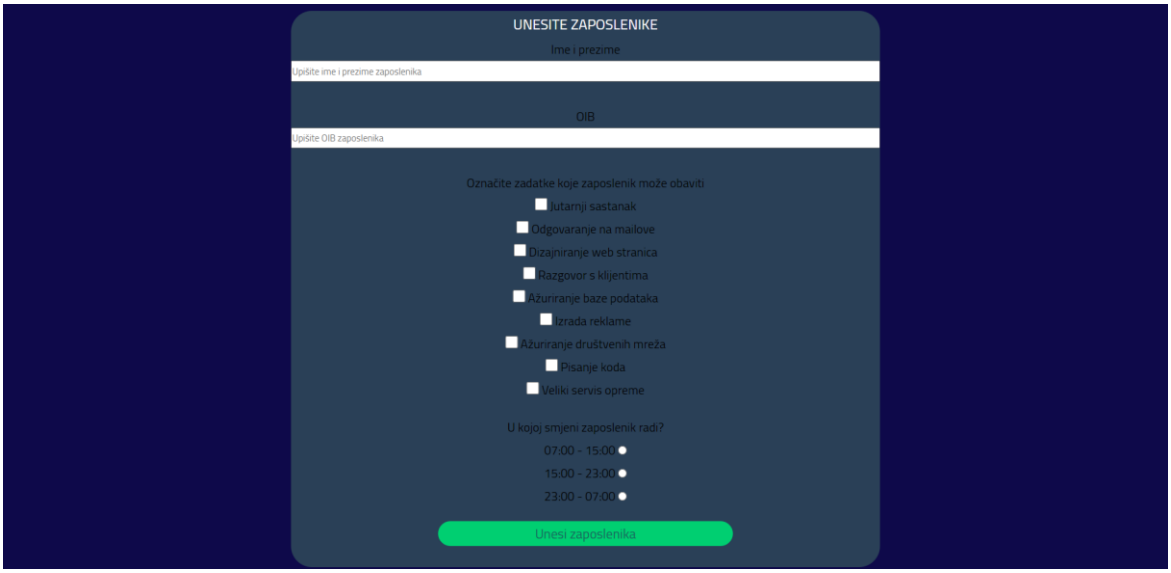


Sl. 4.1 *Unos radnih zadataka*

4.2. Unos zaposlenika

Unos zaposlenika obavlja se na sličan način kao unos radnih zadataka. Unos zaposlenika prikazan je na slici 4.2. Prvo je potrebno upisati ime i prezime zaposlenika što se obavlja upisivanjem teksta unutar prvog elementa za unos podataka. Sljedeće što je potrebno definirati je OIB (osobni identifikacijski broj) zaposlenika, koji se također obavlja upisivanjem teksta, odnosno brojeva. OIB je potrebno definirati zbog rada s bazom podataka koja zahtjeva identifikacijsku oznaku za svakog korisnika, što čini OIB idealnim izborom s obzirom na to da je OIB jedinstven broj, odnosno dva zaposlenika ne mogu imati jednaki OIB. Korištenjem OIB-a izbjegava se pojavljivanje duplikata i svakom zaposleniku se dodjeljuje jedinstven broj koji predstavlja tog zaposlenika. Kako bi zaposlenik bio potpuno definiran potrebno je odrediti koje radne zadatke je zaposlenik sposoban obaviti. Radni zadaci su predstavljeni korištenjem *checkbox* oznake koja je dio HTML jezika, a omogućuje korisniku odabir više ponuđenih izbora. Korištenjem *checkbox* unosa omogućen je višestruk odabir, odnosno korisnik može odabrati više poslova koje je zaposlenik u stanju obaviti. Svakim unosom radnog zadatka generira se jedan *checkbox* što je jedna od glavnih mogućnosti AngularJS-a. Radni zadaci dohvaćaju se iz baze podataka i postaju dostupni unutar dosega (engl.*scope*). Generiranje svakog *checkbox*-a obavlja se pomoću *ng-repeat* direktive. *Ng-repeat* direktiva funkcionira slično *foreach* petlji koja je dostupna u nekim programskim jezicima. Direktiva prolazi kroz radne zadatke dohvaćene iz

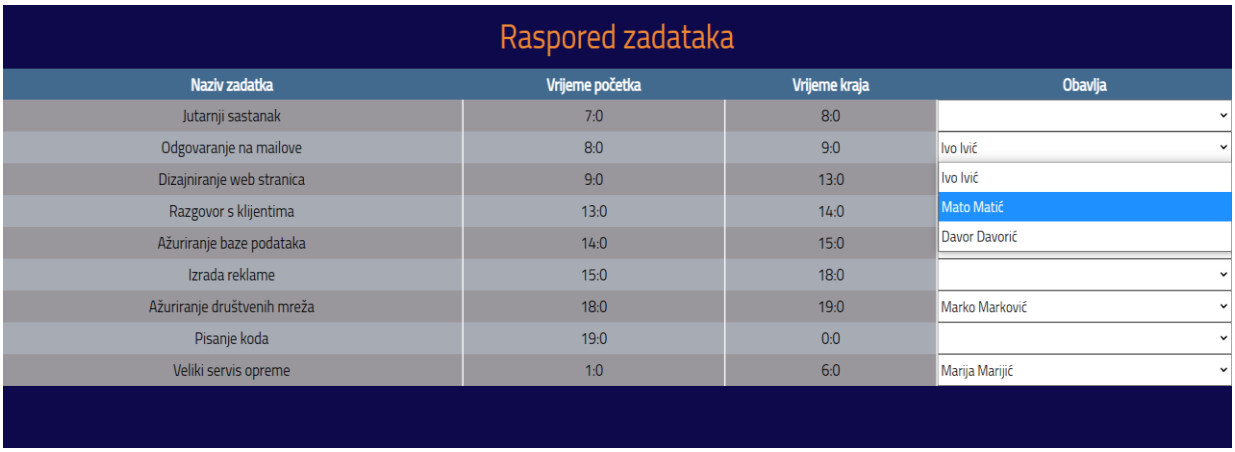
baze podataka i za svaki radni zadatak generira po jedan *checkbox*. Posljednje što se očekuje od korisnika je određivanje smjene u kojoj zaposlenik radi. Za primjer ovog završnog rada može se pretpostaviti da zaposlenici rade u tri smjene. Prva smjena tada počinje u 07:00, a završava se u 15:00. Druga smjena počinje u 15:00, a završava se u 23:00. Treća smjena počinje u 23:00, a završava se u 07:00 idućeg dana. Određivanje smjene u kojoj zaposlenik radi je važno jer neki poslovi počinju i završavaju van opsega smjene u kojoj radi određeni zaposlenik, što znači da taj zaposlenik ne može obaviti taj radni zadatak. Slično kao i prilikom unosa radnih zadataka ovdje također korisnik potvrđuje svoj unos pritiskom gumba „Unesi zaposlenika“. Nakon što je korisnik pritisnuo gumb aktivira događaj i poziva se funkcija za provjeru unosa. U ovom slučaju funkcija provjerava postoje li elementi za unos podataka koje korisnik nije popunio, odnosno ima li onih koji su ostali prazni. Također, funkcija provjera je li OIB broj i ima li 11 znamenki. Ako funkcija vrati vrijednost *true*, unos zaposlenika se smatra uspješnim i sprema se u bazu podataka u obliku objekta sa četiri atributa. Prvi predstavlja ime i prezime zaposlenika, drugi OIB, treći polje zadataka koje zaposlenik može obaviti i četvrti predstavlja smjenu u kojoj zaposlenik radi, odnosno radno vrijeme.



Sl. 4.2 *Unos zaposlenika*

4.3. Generiranje tabličnog sučelja za raspoređivanje

Jedan od zadataka završnog rada je omogućiti korisniku ručno raspoređivanje zaposlenika po zadanim radnim zadacima. Kako bi se raspoređivanje omogućilo potrebno je stvoriti tablično sučelje koje se sastoji od četiri stupca što je prikazano slikom 4.3. Prva tri stupca prikazuju podatke o radnim zadacima, odnosno naziv zadatka te predviđeno vrijeme početka i kraja. U zadnjem stupcu se pojavljuje HTML *select* element kojim se stvara lista opcija koje korisnik može birati. Unutar *select* elementa korisniku je omogućeno biranje zaposlenika koji obavlja radni zadatak u istom retku. Zaposlenici su prikazani HTML *option* elementom unutar *select* elementa koji omogućava odabir samo jedne opcije, odnosno jednog zaposlenika po poslu. Samo tablično sučelje generirano je pomoću AngularJS *ng-repeat* direktive. Generiranje se odvija trenutno, odnosno svaki uspješan unos radnog zadatka pokreće stvaranje novog retka u tablici. Ovo je omogućeno korištenjem *ng-repeat* direktive koja prolazi kroz svaki radni zadatak spremljen u bazi podataka i generira novi redak tablice. U zadnjem stupcu se opcije za odabir također generiraju pomoću *ng-repeat* direktive. Međutim, u zadnjem stupcu logika samog stvaranja opcije je puno složenija. Kako bi se zaposlenik pojavio kao opcija za odabir u nekom retku aplikacija mora u obzir uzimati niz uvjeta. Prvi uvjet je da zaposlenik može obaviti radni zadatak u danom retku. Drugi uvjet je da predviđeno vrijeme obavljanja radnog zadatka ulazi u radno vrijeme određenog zaposlenika. Također, svaku opciju potrebno je smjestiti u AngularJS model što bi značilo da se mora pratiti vrijeme unutar kojeg je zaposlenik zauzet. Iz tog razloga aplikacija pazi da ako je zaposlenik zauzet na nekom radnom zadatku u određeno vrijeme, tada on ne može biti dostupan za obavljanje nekog drugog zadatka za to vrijeme. S druge strane ako zaposlenik obavi posao u trajanju od 2 sata, to znači da preostaje još 6 sati njegovog radnog vremena unutar kojeg mu je moguće dodijeliti neki drugi radni zadatak. Također, moguć je slučaj u kojem niti jedan zaposlenik ne može obaviti određeni radni zadatak. U tom slučaju u tom retku se ne pojavljuje niti jedna opcija za odabir zaposlenika što korisnika upozorava da je potrebno naći zaposlenika koji je sposoban obaviti taj zadatak. Ovom logikom je korisnik rasterećen s obzirom na to da ne mora paziti na niz problema koji se mogu pojaviti prilikom odabira, već aplikacija sve to radi u pozadini. Od korisnika se još očekuje potvrda odabira zaposlenika po radnim zadacima, a to se obavlja pritiskom gumba „Pohrani odabir“. Nakon toga se tablica sprema i pojavljuje se unutar aplikacije kao gotova tablica prikazana slikom 4.4, u kojoj nije moguće ništa mijenjati jer ona tada predstavlja gotovi raspored radnih zadataka po zaposlenicima u tom radnom danu.



Sl. 4.3 *Dodjeljivanje radnih zadataka zaposlenicima*

Sl. 4.4 *Konačan raspored zadataka*

4.4. Generiranje rasporeda za zaposlenike

Posljednja funkcija aplikacije je stvaranje rasporeda za zaposlenike. Raspored za zaposlenike je potrebno napraviti kako bi svaki zaposlenik dobio uvid u svoj radni dan, odnosno kako bi zaposlenik znao što kada treba obaviti. Raspored je zamišljen u obliku tabličnog sučelja u kojem svaki stupac predstavlja interval od jednog sata. Svaki radni zadatak koji se obavlja je prikazan različitom bojom kako bi razlike između poslova bile jasno vidljive. Radi lakšeg stvaranja tablice, tablica je fiksne veličine, odnosno ima veličinu definiranu u pikselima, ne u postotcima.

Za potrebe testiranja aplikacije odabrana je širina tablice od 1250 piksela. Tablica ima 25 stupaca, gdje prvi stupac predstavlja ime i prezime zaposlenika, a ostalih 24 stupca predstavljaju 24 sata. Kako je tablica širine 1250 piksela, dobiveno je 50 piksela po stupcu, odnosno po jednom satu. Izgled rasporeda prikazan je slikom 4.5. Za stvaranje rasporeda u ovom obliku u najvećoj mjeri je korišten CSS kojim je omogućeno mijenjanje veličina redaka i stupaca te dodjeljivanje nove boje svakom radnom zadatku. Ovakav oblik tablice, odnosno rasporeda daje jasan i precizan uvid što svaki zaposlenik treba obaviti i u kojem vremenu.

Sl. 4.5 *Raspored za zaposlenike*

5. ZAKLJUČAK

Tema ovog završnog rada je bila izraditi web aplikaciju za raspoređivanje radnog vremena unutar jedne tvrtke. Osim izrade samog rasporeda, cilj aplikacije je također bio primijeniti znanja stečena na Fakultetu iz raznih kolegija, ali i nova znanja stečena učenjem AngularJS-a. AngularJS je savršen izbor za izradu aplikacija čiji se kompletan sadržaj nalazi na jednoj stranici, odnosno većina promjena se odvija dinamički. U ovom radu najveća moć AngularJS-a pokazana je u dinamičkom stvaranju tabličnih sučelja i dinamičkoj provjeri iskorištenog radnog vremena određenog zaposlenika. Sama aplikacija je relativno jednostavna za korištenje, a od korisnika aplikacije se očekuje pravilan unos podataka o radnim zadacima i zaposlenicima unutar tvrtke. Također, aplikacija nije ograničena samo na izradu rasporeda radnih zadataka unutar tvrtke, već je potpuno fleksibilna i može se koristiti za brojne druge svrhe. Primjer korištenja ove aplikacije van neke tvrtke bi bila izrada rasporeda unutar jedne škole ili fakulteta čime bi svaki profesor, odnosno učitelj dobio jasan raspored nastave za jedan radni dan. Također, aplikaciju bi mogao iskoristiti bilo tko kome je potrebno da organizira svoj dan i učini ga maksimalno produktivnim. Osim AngularJS-a stečena znanja primijenjena su i na ostalim jezicima, poput HTML-a, CSS-a i PHP-a. Za stvaranje aplikacije tim jezicima zaslužno je razvojno okruženje Visual Studio Code koje je potpuno besplatno, dostupno svima i jednostavno za korištenje. Za učenje novih tehnologija kao najbolji resurs pokazao se YouTube i popularna web forum za programere StackOverflow.

LITERATURA

[1] Asana, Asana (zadnji pristup 12.10.2020.)

https://app.asana.com/

[2] Monday, monday (zadnji pristup 12.10.2020.)

https://monday.com/

[3] ProjectManager, ProjectManager (zadnji pristup 12.10.2020.)

https://www.projectmanager.com/

[4] OmniFocus, Omnigroup (zadnji pristup 12.10.2020.)

https://www.omnigroup.com/omnifocus/

[5] To Do, Microsoft (zadnji pristup 12.10.2020.)

https://to-do.microsoft.com/tasks/

[6] Visual Studio Code, Visual Studio Code (zadnji pristup 20.9.2020.)

https://code.visualstudio.com/

[7] Javascript, Developer Mozilla (zadnji pristup 20.9.2020.)

https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript

[8] AngularJS, AngularJS (zadnji pristup 10.7.2020.)

https://angularjs.org/

[9] HTML, HTML.COM (zadnji pristup 12.8.2020.)

https://html.com/

[10] CSS, Developer Mozilla (zadnji pristup 15.7.2020.)

https://developer.mozilla.org/en-US/docs/Web/CSS

[11] PHP, NetAkademija (zadnji pristup 10.7.2020.)

http://www.netakademija.hr/sto-je-php/

SAŽETAK

Ovaj završni rad nastoji riješiti problem raspoređivanja radnog vremena unutar jedne tvrtke za vrijeme jednog radnog dana. Na početku rada odrađen je kratki uvod u kojem se navode svi korišteni alati i razvojna okruženja potrebna za izradu aplikacije te kratki opis svake od korištenih tehnologija. U glavnom dijelu rada obrađena je struktura i načina rada aplikacije uz priložene slike dizajna i logike rada aplikacije. Također, u glavnom dijelu je opisan način na koji se u programskom okruženju ostvario rad aplikacije. U krajnjem dijelu rada dan je kratki osvrt na rad i primjenu stečenih znanja.

Ključne riječi:

AngularJS, CSS, HTML, JavaScript, PHP, Tablica

ABSTRACT

AngularJS working time scheduling application

The project deals with the development of a web application, which generates a working schedule. The main goal of the project is to show how AngularJS is used and to apply the acquired knowledge in programming and basic web development. In the theoretical part the used tools and technologies are elaborated on. The application design and logic are explained in the development section.

Keywords:

AngularJS, CSS, HTML, JavaScript, PHP, Table

ŽIVOTOPIS

Ivan Rajkovača je rođen 07. 04. 1999. godine u mjestu Slavonski Brod, Hrvatska. Ime njegovog oca je Nedeljko, a ime majke Vesna. Pohađao je osnovnu školu "Ivan Goran Kovačić" u Slavonskom Brodu. Nakon završene opće gimnazije u Slavonskom Brodu, upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Na fakultetu pokazuje prvi interes za svijet programiranja gdje su mu predstavljene brojne tehnologije korištene u razvoju web i mobilnih aplikacije. Posjeduje dobro znanje HTML-a, CSS-a i JavaScripta, a upoznat je i s programskim jezicima C, C# i C++.

Ivan Rajkovača

_____