

**SVEUČILIŠTE U RIJECI  
POMORSKI FAKULTET U RIJECI**

**ALEN BARAĆ**

**RAZVOJ DINAMIČKIH WEB APLIKACIJA**

**DIPLOMSKI RAD**

Rijeka, 2014.

**SVEUČILIŠTE U RIJECI**  
**POMORSKI FAKULTET U RIJECI**

**RAZVOJ DINAMIČKIH WEB APLIKACIJA**  
**DEVELOPMENT OF DYNAMIC WEB APPLICATIONS**

**DIPLOMSKI RAD**

Kolegij: Operacijski sustavi

Mentor: dr.sc.Božidar Kovačić

Student: Alen Barać

Studijski smjer: elektroničke i informatičke tehnologije u pomorstvu

JMBAG:1311987360029

Rijeka, listopad 2014.

Student/studentica: Alen Barać

Studijski program: elektroničke i informatičke tehnologije u pomorstvu

JMBAG: 1311987360029

## **IZJAVA**

Kojom izjavljujem da sam diplomski rad s naslovom RAZVOJ DINAMIČKIH WEB APLIKACIJA izradio/la samostalno pod mentorstvom prof.dr.sc./izv.prof.dr. sc./doc.dr.sc Božidara Kovačića .

U radu sam primijenio/la metodologiju znanstvenoistraživačkog rada i koristio/la literaturu koja je navedena na kraju diplomskog rada. Tuđe spoznaje, stavove, zaključke, teorije i zakonitosti koje sam izravno ili parafrazirajući naveo/la u diplomskom radu na uobičajen, standardan način citirao/la sam i povezo/la s fusnotama i korištenim bibliografskim jedinicama. Rad je pisan u duhu hrvatskoga jezika.

Suglasan/na sam s objavom diplomskog rada na službenim stranicama.

Student/studentica

Ime i prezime studenta/studentice

<b>1. UVOD</b> .....	<b>1</b>
<b>2. PRIKAZ DINAMIČKOG SADRŽAJA NA WEB – U</b> .....	<b>3</b>
2.1. DINAMIČKE WEB STRANICE .....	3
2.2. WEB APLIKACIJE .....	5
<b>2.3. FRONT-END I BACK-END RAZVOJ</b> .....	<b>6</b>
2.3.1. Front-end razvoj .....	6
2.3.2. Back-end razvoj .....	7
<b>3. FUNKCIONALNOST KAO MOTIVACIJA</b> .....	<b>8</b>
3.1. MOTIVACIJA .....	8
3.2. Temeljne funkcionalnosti web aplikacije za registar pomoraca .....	9
<b>4. RAZVOJNO OKRUŽENJE WEB APLIKACIJE</b> .....	<b>13</b>
4.1. APACHE WEB-SERVER .....	14
4.2. HTML .....	14
4.3. CASCADING STYLE SHEETS(CSS) .....	17
4.4. PHP PROGRAMSKI JEZIK .....	18
4.5. BAZA PODATAKA MYSQL .....	20
4.6. BOOTSTRAP 3.0 .....	21
<b>5. PREGLED PROGRAMSKIH PAKETA I ALATA PRI IZRADI WEB APLIKACIJE</b> .....	<b>24</b>
5.1. WAMP SERVER .....	24
5.2. PHPMYADMIN .....	26
5.3. NETBEANS IDE .....	28
<b>6. ARHITEKTURA WEB APLIKACIJE</b> .....	<b>30</b>
<b>7. IMPLEMENTACIJA</b> .....	<b>31</b>
7.1. ODABIR RADNE PLATFORME .....	31
7.2. BAZA PODATAKA MYSQL .....	32
7.2.1. Modeliranje entiteta i veza .....	33
7.2.2. Relacije i atributi web aplikacije za registar pomoraca .....	34
7.2.3. ER dijagram .....	43

<b>7.3. KORISNIČKO SUČELJE .....</b>	<b>45</b>
<b>7.4. APLIKACIJSKA LOGIKA .....</b>	<b>47</b>
7.4.1. ORGANIZACIJA KODA .....	52
7.4.2. OPIS I NAMJENE DATOTEKA APLIKACIJE .....	54
<b>8. RANJIVOST PHP APLIKACIJA I NJIHOVA ZAŠTITA.....</b>	<b>58</b>
8.1. SQL INJECTION .....	58
8.2. NAČIN ZAŠTITE .....	59
8.2.1. Izdvajanje podataka .....	59
8.2.2. Pripremljeni upiti .....	63
<b>9. DODATNE FUNKCIONALNOSTI WEB APLIKACIJE ZA REGISTAR POMORACA .....</b>	<b>67</b>
<b>10. ZAKLJUČAK.....</b>	<b>69</b>
<b>LITERATURA : .....</b>	<b>71</b>
<b>SKRAĆENICE : .....</b>	<b>73</b>
<b>POPIS PRILOGA : .....</b>	<b>74</b>
<b>PRILOG-KORIŠTENJE APLIKACIJE : .....</b>	<b>75</b>

# 1. UVOD

U posljednjih petnaestak godina događa se veliki napredak u razvoju usluga koje su dostupne korisnicima putem WWW-a (engl. *World Wide Web*, skraćeno *Web*). Raspon tih usluga je vrlo velik, od nekomercijalnih, poput pružanja svakodnevnih informacija vezanih uz vrijeme, promet, kulturna događanja, sport, usluga elektroničke pošte itd., pa do komercijalnih poput velikih *online* trgovina (knjižara, trgovina sa kućanskim potrepštinama, potrošačkom elektronikom...), *online* izdanja novina i časopisa, i, kod nas sve popularnijih, usluga bankovnog poslovanja preko interneta. Pravu revoluciju u posljednjih 10 godina napravili su takozvani “pametni telefoni”(engl. *smartphone*) koji su ubrzali već ionako “zahuktalu” industriju komercijalnih i poslovnih aplikacija koje uvelike postaju dio svakodnevnog načina života, a što je još važnije u velikoj mjeri olakšavaju i unaprijeđuju svakodnevni radni proces. U jednu od takvih poslovnih aplikacija spadaju i CMS-ovi(engl. *Content management system*), programska sučelja koja omogućuju korisnicima lakše i učinkovitije upravljanje podacima.

Cilj ovog diplomskog rada jest izrada, razvoj i implementacija dinamičke Web aplikacije za registar pomoraca gdje će se zadovoljiti spomenuti zahtjevi. Zahtjevi olakšavanja i unapređenja svakodnevnog radnog procesa, zahtjevi sigurnosti očuvanja podataka, te smjernice dodatnih mogućnosti razvoja aplikacijske platforme ovisno o potražnji i potrebi korisnika. Osnovni model funkcionalnosti prikazan je kroz primjer pomorske agencije, koja ima svoje zahtjeve i standarde pri upravljanju podacima i dokumentaciji pomoraca, potrebnim za njihov ukrcaj na plovni objekt.

Diplomski rad je organiziran u tri osnovne cjeline.

U prvoj cjelini se opisuju tehnologije potrebne za izradu ove vrste dinamičke web aplikacije kao i sama podjela i struktura razvoja web platformi. Kao što je istaknuto, riječ je o web aplikaciji čiji je prostor djelovanja World Wide Web. World Wide Web je klijentsko-serverski model razmjene informacija izgrađen na infrastrukturi interneta, te na HTTP(engl. *HyperText Transfer Protocol*) protokolu za prijenos podataka. Korisnici web-a pristupaju informacijama putem web preglednika (eng. *browser*-a) - klijentske aplikacije namijenjene preuzimanju i prezentaciji web dokumenata sa servera, navigaciji među dokumentima, te slanju povratnih, korisničkih informacija na server. HTTP poslužitelj Apache je programski paket koji će biti korišten unutar ovog diplomskog rada. Njegova je

osnovna zadaća prosljeđivanje sadržaja Web pregledniku. Apache je još uvijek dominantan u pogledu korištenih Web poslužitelja, naime, istraživanja pokazuju da između 60 i 70% svih Web poslužitelja na svijetu koriste upravo Apache programski paket . Osim što je besplatan, Apache korisnicima i administratorima donosi širok spektar, kako jednostavnih, tako i naprednih mogućnosti što ga definitivno stavlja na prvo mjesto izbora većine Web razvijatelja. Za upravljanje aplikacijom i komunikaciju između servera i baze podataka korišten je PHP(engl. PHP:Hypertext preprocessor) programski jezik. PHP je (eng.*open source*) programski jezik za razvoj Web aplikacija (odnosno dinamičko generiranje HTML(eng. *Hyper Text Markup Language*) koda. Kao baza podataka koristi se MySQL relacijska baza podataka.MySQL baza podataka je najrasprostranjenija baza podataka na internetu koja omogućava brz protok, lako upravljanje i veliku pouzdanost.Našom MySQL bazom upravljamo preko SQL jezika.SQL jezik je zamišljen kao jednostavan jezik,pun standardnih engleskih riječi koji se brzo uči i omogućuje primjenu na druge vrste baza podataka.

Za vizualni aspekt ovog rada koristi se standardni i u modernom radnom okruženju nezaobilazni,CSS(eng. *Cascading Style Sheets*) meta jezik te programski okvir(eng.*framework*)

Bootstrap 3.0.Preko Bootsrtap-a će bit omogućen moderan izgled aplikacije,te olakšano kodiranje responzivnosti(prikaza aplikacije neovisno o veličini ekrana).

U drugoj cjelini rada je prikazana struktura, te opis izrade dinamičke Web aplikacije za registar pomoraca.Prikazan je cijeli radni proces te su naglašeni sigurnosni aspekti Web aplikacije, isto tako su opisane mogućnosti koje sama aplikacija obavlja.

U posljednjoj cjelini navode se mogućnosti proširenja funkcionalnosti aplikacije,implementiranje iste u drugačije radne platforme te su dani najvažniji zaključci ovog rada.

## 2. PRIKAZ DINAMIČKOG SADRŽAJA NA WEB – U

### 2.1. DINAMIČKE WEB STRANICE

Kroz posljednji nekoliko godina se uočava trend prelaska sa statičkih Web stranica izvedenih u HTML-u (engl. *HyperText Markup Language*) na dinamičke. Razlog što se takve stranice nazivaju dinamičkim leži u promjenjivoj „naravi” sadržaja koje one prikazuju. Taj je sadržaj dinamički zbog vanjskih utjecaja na sam sadržaj. Među vanjske utjecaje spadaju promjene zapisa u bazi podataka, promjene tipa preglednika (*Google Chrome – Internet Explorer – Opera – Mozilla*), promjena korisnikove IP (engl. *Internet Protocol*) adrese (npr: promjena u prikazu jezika, ovisno o tome gdje se korisnik nalazi), promjena doba dana (npr: promjena sadržaja vremenske prognoze ili promjena jelovnika nekog restorana) i dakako korisnička interakcija (npr: individualizacija prezentacije i tipa sadržaja – prikaz podataka o korisnikovoj omiljenoj nogometnoj momčadi). Jasno je da osoba (točnije osobe) koje se bave izradom dinamičkih stranica moraju imati široki spektar znanja, zato što dinamičke Web stranice mogu biti izvedene u velikom broju različitih tehnologija koje većinom zahtijevaju programerske vještine (PHP (engl. *PHP:Hypertext Preprocessor*), Perl, JScript, VBScript (engl. *VisualBasic Script*), ActiveX, ASP (engl. *Active ServerPages*), JSP (engl. *Java Server Pages*) itd.), dizajniranje i izradu baze podataka (primjerice MySQL – jedna od mnogih relacijskih baza podataka ili za manju količinu podataka XML (engl. *eXtensible Markup Language*) uz podrazumijevano poznavanje HTML–a (ili barem snalaženje u radu sa HTML – om).

Ako se pobliže pogleda razvoj pristupa prikazivanju dinamičkih stranica, može se uočiti da je u prvoj generaciji rješenja najviše bio korišten CGI (engl. *Common Gateway Interface*). CGI je mehanizam koji je omogućio da se na poslužiteljima izvode vanjski programi koji su obavljali neke određene zadaće. Ti programi (ili skripte) su bili pisani najčešće u programskim jezicima C i Perl, a najveći problem ovog pristupa je taj što on nije skalabilan. Za svaku novu primjenu se morao pisati

novi kod, bez mogućnosti iskorištavanja i nadograđivanja starih programa koji već rade na serverima. Druga generacija rješenja donijela je znatne pomake u smislu da su tvrtke koje su razvijale poslužitelje, usporedno razvijale i API – je (engl. *Application Program Interface*) za te poslužitelje. Tako je primjerice Microsoft izdao ASP, no da bi ste ga koristili, morali ste imati i Microsoft – ov IIS (engl. *Internet Information Services*) poslužitelj. Uz ASP, pojavili su se i *servleti* koji su olakšali



pisanje poslužiteljskih aplikacija u programskom jeziku JAVA, što je donijelo jednostavnost naspram CGI skriptama. No i *servleti* se moraju pridržavati napiši – prevedi – pokreni životnog ciklusa kao i CGI skripte, pa je argument protiv njihovog korištenja isti – nisu skalabilni. Primjer treće generacije rješenja je JSP, pandan ASP-a, ali koji sadrži apsolutnu otvorenost, odnosno neovisnost o bilo kakvom komercijalnom rješenju i proizvođaču. JSP je temeljen na JAVI i omogućuje razvoj Web aplikacija te prikaz dinamičkog sadržaja.

S obzirom na postojanje velikog broja različitih Web stranica, treba obratiti pozornost na neke zajedničke odrednice tih stranica. Kada bi stranice grupirali po složenosti, podjela bi bila sljedeća:

1. Stranice sa malom složenošću podataka, implementacija i aplikacija koje (eventualno) koriste. Ovakve stranice se obično sastoje od nekoliko HTML dokumenata i obično im je svrha promidžba nekog proizvoda ili usluge. Izrađuju se uz pomoć široko rasprostranjenih HTML uređivača. Procjenjuje se da najveći dio stranica pripada ovoj kategoriji.
2. Stranice orijentirane na pružanje neke određene usluge poput pretraživanja ili Internet pošte mogu se temeljiti na složenim bazama podataka i aplikacijama koje služe za pružanje usluga, ali zadržavaju jednostavnost prikaza tih podataka iz razloga što ne postoji potreba za prikazom velike količine istih.
3. Stranice koje služe za prikaz velike količine podataka odlikuju se velikom složenošću strukture prikaza, ali ne pružaju nikakve ili vrlo ograničene usluge. Primjer ovakvih stranica su stranice raznih sveučilišta i škola, kojima je primarna funkcija prikaz velike količine podataka o studentima, profesorima, predmetima i organizacijama vezanim uz nastavu odnosno studij.

Informacijski sustavi temeljeni na Internetu su najsloženije stranice od svih navedenih. Nude pristup ogromnim količinama podataka i korištenje mnogih interaktivnih usluga.

## **2.2. WEB APLIKACIJE**

WWW je prvotno bio zamišljen samo kao medij za razmjenu informacija, pa je glavni zadatak pri njegovom razvoju bio taj da bude dovoljno jednostavan da razni autori mogu bezbrižno razmjenjivati dokumente. Pomisao o razvoju Web aplikacija se tada svodila na dizajniranje, povezivanje i prikaz takvih dokumenata. Daljnjim razvojem došlo je do napretka u dizajnu, prikazu i organizaciji dokumenata, no sama implementacija je ostala ista, većinom temeljena na HTML – u. Takvom pristupu i danas pogoduje činjenica da su Web preglednici (koji naravno prepoznaju HTML) prisutni na svim platformama i operacijskim sustavima. Velika popularnost Web – a ima za posljedicu da HTML aplikacije nastaju svakodnevno u velikim količinama. No, od svog nastanka do danas Internet se razvio u globalno okruženje (zapravo jednuveliku tržnicu roba i informacija, pri čemu je velika količina istih besplatna) za distribuciju raznolikih aplikacija, od malih, kratkoročno upotrebljivih servisa, do ogromnih sustava distribuiranih preko nekoliko velikih poslužitelja [2]. Jasno je da ovakve aplikacije ne mogu imati implementaciju orijentiranu samo na prikaz nekog skupa podataka, već se ipak prema njima treba odnositi kao “pravim” aplikacijama, s tom razlikom što Web aplikacije moraju biti prilagođene mediju na kojem djeluju. Vrlo dobar pristup razvoju Web aplikacija je upotreba objektno orijentiranog programiranja, zbog mogućnosti apstrakcije različitih problema, višestruke upotrebljivosti pojedinih komponenata i dijeljenja takvih komponenata između različitih aplikacija. Izbor programskog jezika za objektno orijentiran razvoj Web aplikacija je velik, a uključuje C++, Javu, C#, PHP, WCML (engl. *WebComposition Markup Language*) i slične. Dok su C+, Java i C# već dobro poznati i prihvaćeni objektno orijentirani jezici, WCML je novi jezik čija je baza *WebComposition* model koji definira objektno orijentirane komponente za razvoj Web entiteta na proizvoljnim razinama apstrakcije, a temelji se na XML – u (eng. *Extensible Markup Language*). Cilj razvoja tog jezika je da pomoću njega bude omogućen još lakši pristup razvoju Web aplikacija.

## 2.3. FRONT-END I BACK-END RAZVOJ

Razvoj web stranica i web aplikacija se razdvaja na dvije osnovne podjele. Razvoj na "prednjoj strani" (engl. *front-end*) i razvoj u "pozadini" (engl. *back-end*). Tako se razvijatelji (engl. *developer*), ovisno o tehnologijama koje koriste za razvoj, u odnosu na navedenu podjelu, dijele na front-end i back-end developere.

### 2.3.1. Front-end razvoj

Kao što sam naziv kaže " front-end " je dio koda koji se odnosi na prednju(vidljivu) stranu radne web platforme(stranice ili aplikacije) . Rezultati tog koda su vidljivi korisniku u obliku sučelja koje omogućuje interakciju korisnika i web platforme. Glavna svrha front-end koda je interakcija s korisnikom, te funkcionalna prezentacija podataka .Prezentacija podataka na što više "oku ugodan" način.

Sve ono što naše oko vidi na webu je mješavina HTML , CSS i JavaScript jezika .Navedeni su glavna tri jezika korištena za predstavljanje i prezentiranje naše web platforme na najbolji mogući način. HTML je odgovoran za stvaranje osnovne strukture naše web platforme koja se preko web preglednika ispravno prikazuje.Preko CSS-a uvodimo boje,pozadine, veličine fonta,margine,pozicije elemenata itd. JavaScript je najnapredniji jezik na klijentskoj strani(odvija se na web pregledniku) koji omogućuje interakciju s korisnikom u obliku klizača, padajućih izbornika, raznih efekata i još mnogo više interaktivnih elemenata.

Front-end developer je most koji spaja odnos dizajner - back-end developer. Posao front-end developera je da preuzme završni dizajn web platforme, te ga pravilno iskodira u ispravnu

strukturu, koja se proslijeđuje dalje back-end developeru. Tada back-end developer ima uvid u samu strukturu i izgled platforme, te oko tog koda izrađuje funkcionalnosti koje sama platforma nudi.

Osim glavnih navedenih elemenata u front-end okruženju (HTML, CSS, JavaScript), postoje i mnoge biblioteke (eng. *library*) te programski okviri (eng. *framework*) koji olakšavaju i ubrzavaju kodiranje. Sve naravno ovisi o potrebama projekta i klijenta. Pa tako možemo koristiti jQuery (JavaScript biblioteke), LESS (CSS biblioteka), Bootstrap (Front - End Framework) ili bilo koje druge biblioteke i programske okvire koji se svakodnevno razvijaju i pojavljuju na tržištu.

### **2.3.2. Back-end razvoj**

Back-end kod možemo protumačiti kao “mozak” svake web platforme. Back-end development je dio razvoja web platforme koji nikada nije vidljiv korisniku. Ova vrsta razvoja se odvija korištenjem poslužiteljskih (eng. *Server side*) programskih jezika koji “komuniciraju” s raznim vrstama baza podataka. Jednostavnije rečeno, front-end kod vrši interakciju s korisnikom u realnom vremenu, dok back-end kod služi za interakciju sa web serverom, da bi vratio rezultate “spremne” za korisnika.

Operacije back-end koda su malo kompleksnije od onoga na front-end-u. Developer razvija i gradi aplikaciju koristeći programske jezike kao što su PHP, Ruby, Python, NET itd. Preko tih jezika ostvaruje konekciju na bazu podataka (MySQL, SQL server, Access, itd.) da bi pretražio, spremio ili promjenio podatke, te ih na kraju vratio korisniku u formi front-end koda. Ova kompleksna struktura nam omogućuje da pretražujemo po internetu, koristimo internet trgovinu, koristimo socijalne mreže, ukratko, da obavljamo sve moderne radnje koje internet nudi. Ako smislimo bilo koju operaciju na web-aplikaciji kao slanje elektroničke pošte, prijavljivanje na primitak web-obrasca, učitavanje sadržaja na web-stranici i sl. možemo biti sigurni da je back-end kod zaslužan za te radnje.

Kao i kod front-end, tako i kod back-end razvoja postoje mnoge biblioteke i programski okviri koji olakšavaju i ubrzavaju razvoj dinamičkih web aplikacija. Najpoznatiji su Ruby on Rails, Symfony, Laravel, Node.js., Code Igniter.

## **3. FUNKCIONALNOST KAO MOTIVACIJA**

### **3.1. MOTIVACIJA**

Kroz praktični dio ovog diplomskog rada krajnji je cilj prikazat jedan moderan pristup i radni proces razvoja jedne funkcionalne i dinamičke web aplikacije. Radni proces, kroz koji je demonstrirana funkcionalnost web aplikacije, je u ovom slučaju registar pomoraca u koji se unose i održavaju podaci o pomorcu, kao i njegova potrebna dokumentacija za ukrcaj na plovni objekt.

Motivacije za odabir ovakvog radnog procesa su višestruke. Jedna vrsta motivacije predstavlja mogućnost praktične implementacije radnog procesa, kao i trenutni zahtjevi na tržištu. Današnje funkcionalno poslovanje uvelike ovisi o efektivnom utrošenom radnom vremenu, organizaciji i sigurnosti dobara kojima se upravlja. U ovom slučaju su to podaci o pomorcu. Jedna moderna pomorska agencija, da bi zadovoljila i pratila standarde tržišne utakmice, naprosto mora imati model poslovanja koji omogućava interakciju svojih zaposlenika, interakciju korisnika i prije svega ubrzani ali organizirani radni proces. U današnje vrijeme se navedeni zahtjevi rješavaju preko informacijskih sustava.

Glavni motiv ovog diplomskog rada, je upravo razvoj jednog takvog informacijskog sustava, web aplikacije, koja koristi i spaja više različitih web tehnologija, koja ima mogućnost višestruke implementacije i nadogradnje, te sustav koji je redundantan i siguran.

### 3.2. Temeljne funkcionalnosti web aplikacije za registar pomoraca

Funkcionalnost web aplikacije možemo podijeliti u dvije glavne cjeline:

- I. Aplikacija preko svog korisničkog sučelja omogućava agentu pomorske agencije jedan jasan pregled cjelokupne dokumentacije pomorca kao i pretraživanje dokumentacije po određenim kriterijima.

U navedenu dokumentaciju spadaju:

1. Slika pomorca
2. Osobni podaci pomorca
3. Podaci o putnoj i medicinskoj dokumentaciji
4. Podaci o položenim sigurnosnim tečajevima
5. Podaci o certifikatu sposobnosti
6. Podaci o GMDSS operateru
7. Podaci o prijašnjoj plovidbi pomorca

- II. Aplikacija preko svog korisničkog sučelja omogućava administratoru pomorske agencije upravljanje cjelokupne dokumentacije pomorca.

Upravljanje podacima se sastoji od sljedećih radnji :

- Pretraživanje podataka pomoraca po određenim kriterijima

**Unos :**

1. Slike pomorca
2. Osobni podaci pomorca
3. Podaci o putnoj i medicinskoj dokumentaciji
4. Podaci o položenim sigurnosnim tečajevima
5. Podaci o certifikatu sposobnosti
6. Podaci o GMDSS operateru
7. Podaci o prijašnjoj plovidbi pomorca

**Izmjena :**

1. Slike pomorca
2. Osobni podaci pomorca
3. Podaci o putnoj i medicinskoj dokumentaciji
4. Podaci o položenim sigurnosnim tečajevima
5. Podaci o certifikatu sposobnosti
6. Podaci o GMDSS operateru
7. Podaci o prijašnjoj plovidbi pomorca

**Brisanje :**

- cjelokupna dokumentacija se briše iz registra

Navedeni elementi čine osnovnu funkcionalnost ove web aplikacije. Važno je istaknuti da postoje dvije skupine korisnika aplikacije (agent i administrator) sa različitim ovlastima. Agent i administrator imaju odvojeno sučelje za pristup aplikaciji. Navedeni korisnici pristupaju aplikaciji sa vlastitim korisničkim imenom i zaporkom.

Agenti pomorske agencije nakon ulaza u sustav imaju mogućnost pretraživanja i pregleda dokumentacije pomoraca. Pretraživanje se vrši preko određenih kriterija. Tako agent ima mogućnost



pretrage registra na osnovi kriterija imena, prezimena, statusa ili kompanije pojedinog pomorca. Pod status spada da li je traženi pomorac na brodu ili na kopnu.

Administrator nakon ulaza u sustav ima navedene mogućnosti agenta ali, uz to, ima i već istaknute mogućnosti organizacije i upravljanja podataka unutar pomorske agencije.

## 4. RAZVOJNO OKRUŽENJE WEB APLIKACIJE

Kroz ovo poglavlje su navedene i objašnjene tehnologije korištene za razvoj web aplikacije za registar pomoraca.

Kod kreiranja obične HTML stranice dovoljna nam je aplikacija za kreiranje HTML stranica (npr. Dreamweaver) i web-preglednik. Kada je stranica gotova, spremimo je negdje na tvrdi disk našeg računala u obliku .html ili .htm datoteke. Putem web-preglednika pristupamo toj datoteci te na taj način možemo vidjeti rezultat našeg rada.

Kod izrade web-aplikacija i stranica u PHP-u situacija je drugačija. Osim aplikacije za pisanje samog koda i preglednika, potreban nam je i web-server.

Web-server je aplikacija koja je instalirana na poslužitelju na kojem imamo smještenu našu stranicu. Služi za izvođenje programskog koda i prikazivanje rezultata korisniku u web-pregledniku. Ovisno o programeru, web-server možemo instalirati na naše osobno računalo. Dobra praksa je, sebi na vlastitom računalu pripremiti okruženje u kojem se mogu izvoditi naše PHP skripte kako bi ih mogli pravilno testirati prije nego što ih postavimo na javni poslužitelj.

## 4.1. APACHE WEB-SERVER

Najpopularniji web server današnjice je Apache. Radi na principu da PHP kod (ili bilo koji drugi skriptni jezik) pohranjujemo u datoteke s ekstenzijom .php pomoću FTP (eng. *File transfer Protocol*) programa te ih smještamo na naš poslužitelj. Ako radimo na našem osobnom računalu s instaliranim i podešenim web-serverom, tada ih spremamo u direktorij koji je definiran u konfiguraciji web-poslužitelja. Naime, web-server će moći izvršiti samo datoteke koje se nalaze u tom direktoriju. Putem web-preglednika pristupamo tom direktoriju. Kada mi, kao korisnik, zatražimo da se prikaže neka PHP skripta u web-pregledniku, preglednik šalje naredbu web-serveru. Web-server dohvaća traženu PHP skriptu i izvršava je, a nama, kao korisniku prikazuje samo njen rezultat u obliku HTML stranice. Rezultat izvršavanja PHP skripte uvijek je u tekstualnom obliku. Ako rezultat kombinira i HTML elemente, tada će ih preglednik prevesti i prikazati.

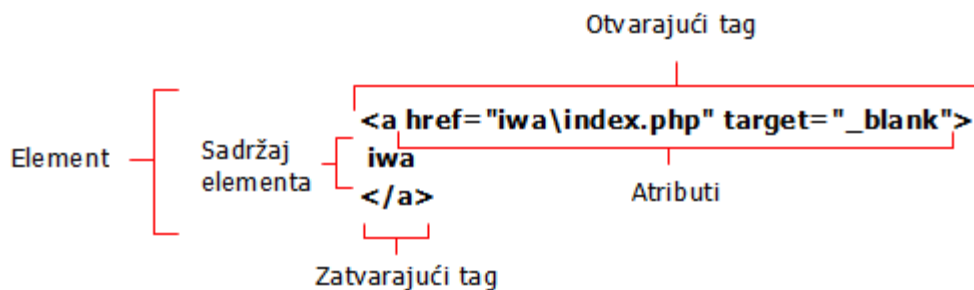
Apache je najčešće korišteni web poslužitelj na internetu s udjelom višim od 60%. Napisan je u C jeziku te sadrži potpuno konfigurabilno sučelje. Podržan je od više grafičkih sučelja koji imaju jednostavniji i lakši način konfiguracije samih poslužitelja. Apache razvija i održava otvorena zajednica programera pod vodstvom *Apache Software Foundation*.

## 4.2. HTML

HTML (engl. *Hyper Text Markup Language*) je meta jezik koje koriste preglednici kako bi odredili na koji način prikazati web-stranicu ili web aplikaciju. Kombinacija je običnog teksta i specijalnih markera (eng. *tags*). Ti markeri govore web-pregledniku kako prikazati sadržaj našeg web dokumenta. Marker se koristi svaki put kada želimo formatirati tekst, umetnuti sliku, prikazati tablicu s podacima, itd. Web preglednik nakon učitavanja web dokumenta parsira dobiven tekst, interpretira tagove i njihov sadržaj prema specifikaciji HTML standarda, te ga na kraju prikazuje kao cjelinu, to jest kao web dokument.

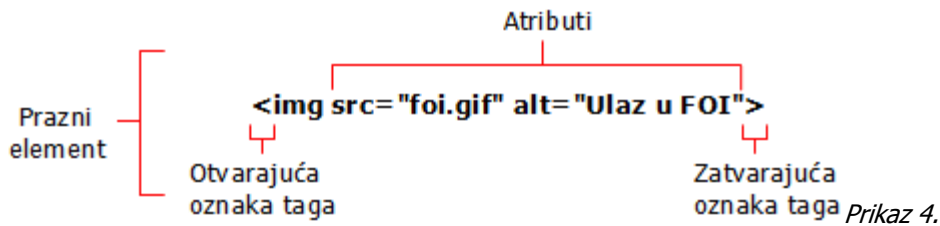
Osnovna strukturna jedinica HTML dokumenta jest HTML element, koji se sintaksno opisuje na sljedeći način:

1. počinje otvarajućim tagom (opening tag)
2. završava zatvarajućim tagom (closing tag)
3. sadržaj elementa je sve što se nalazi između otvarajućeg i zatvarajućeg taga
4. neki elementi imaju prazan sadržaj
5. prazni elementi (bez sadržaja) zatvoreni su u otvarajućem tagu
6. većina HTML elemenata može imati attribute koji dodatno opisuju specifičnosti pojedinih tagova



Slika 1. Sastavni dijelovi HTML elementa prikazani na primjeru `<a>` taga (anchor)

Neki tagovi ne mogu imati sadržaj, to su tzv. "prazni elementi" i kao takvi oni nemaju zatvarajući tag.



Slika 2. Sintaksa praznog elementa je ista uz izostavljanje zatvarajućeg taga

HTML elementi mogu imati više različitih atributa; svaki atribut sastoji se od imena i vrijednosti (ime="vrijednost"), a specificira se u otvarajućem tagu elementa. Atributi daju web pregledniku dodatne informacije, specifične za pojedinu vrstu elementa. Unutar HTML-a postoji mnoštvo tagova koji nam služe za formatiranje naslova, paragrafa, tzv. div elemenata, tablica, lista, tagova za ubacivanje slika (slika 2.), videa i još puno drugih. Velik broj istih bit će prikazani i objašnjeni u poglavlju koje će opisivat izradu same web aplikacije za registar pomoraca.

Sljedeći primjer prikazuje osnovnu strukturu jednog HTML dokumenta:

```

<html>

  <head>

    <title>Naslov stranice</title>

  </head>

  <body>

    Sadržaj stranice

  </body>

</html>

```

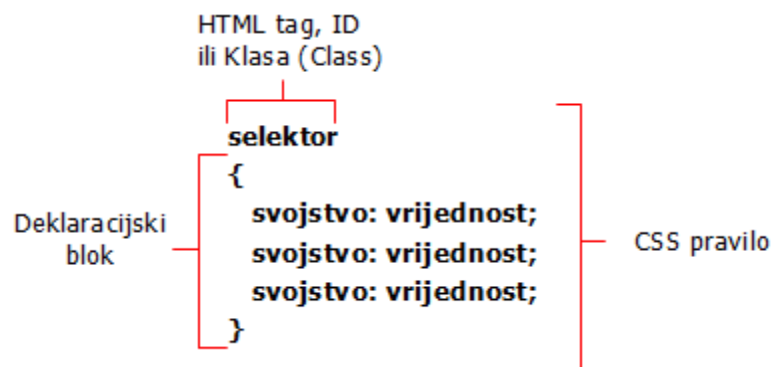
Primjer 1. Sintaksa strukture html dokumenta

### 4.3. CASCADING STYLE SHEETS(CSS)

CSS je meta-jezik kojim se definira prezentacija dokumenata kreiranih bilo kojim markup(jezik za označavanje podataka) jezikom, pa time i HTML dokumenata. Dodjeljivanjem stilova (fontova, boja, pozadina, margina, pozicija...) pojedinim elementima ili grupama elemenata definira se grafička prezentacija web dokumenta. Korištenjem CSS-a nad strukturiranim web dokumentima moguće je utjecati na prezentaciju dokumenata bez potrebe za dodavanjem novih HTML tagova, te bez žrtvovanja neovisnosti aplikacije. Dosljednom upotrebom CSS-a, postiže se potpuna separacija prezentacije web dokumenta od njegovog sadržaja.

Kada posjetitelj učita web stranicu, web preglednik šalje dodatne upite na web server za sve povezane i ugrađene objekte (kao što su slike, video, itd.) među kojima su i datoteke sa CSS kodom. Web preglednik posjetitelja će interpretirati taj kod koristeći svoj vlastiti mehanizam za iscrtavanje slike tzv.(eng. 'rendering engine'), da primijeni CSS na HTML i prikaže stranicu u web pregledniku.

Sintaksa CSS-a sastoji se od samo dva elementa: selektora i deklaracijskog bloka. Selektor označava HTML element ili skupinu elemenata nad kojima će biti primijenjen deklaracijski blok, a deklaracijski blok se sastoji od grupe svojstava i njihovih vrijednosti.



Slika 3. Osnovna sintaksa CSS-a

```

.table {

        position: absolute;

        top: 120px;

        left: 50px;

    }

```

Primjer 2. pozicioniranje HTML tablice preko CSS-a

Ostali elementi CSS-a kao što su margine, širina, dužina HTML bloka, fontovi, boje, pozadine, pozicije itd. , te njihova primjena su prikazani i objašnjeni u poglavlju izrade same web aplikacije za registar pomoraca. Preko CSS-a je izrađen kompletan raspored elemenata aplikacije.

#### 4.4. PHP PROGRAMSKI JEZIK

PHP je (slobodan softver, eng. *open source*) programski jezik za razvoj Web aplikacija (odnosno dinamičko generiranje HTML koda). Naziv PHP je skraćenica od engleskog „PHP: Hypertext preprocessor“. PHP je poslužiteljski (engl. *server-side*) jezik koji može biti integriran u HTML ili se koristiti kao samostojeća aplikacija (u tom se obliku koristi iznimno rijetko). Srodni jezici PHP – u su ASP i JSP. Iako prvotno nije bio namijenjen objektno orijentiranom razvoju, PHP vrlo brzo napreduje i implementira prijeko potrebne nadogradnje. Jedna od najbitnijih prednosti PHP – a jest njegova podrška za rad širokom paletom baza podataka. Podržava sve popularnije baze podataka kao što su MySQL, PostgreSQL, dBase, Oracle itd. [4]. PHP se izvršava na svim popularnijim web serverima i dostupan je za različite operacijske sustave. Funkcionalnost PHP-a je uključivanje dio koda unutar HTML bloka, te se nakon obrade takvih dokumenata svaki dio s PHP kodom zamijeni rezultatom koji se dobije njegovim izvršavanjem. Takav se dokument tada prikazuje korisniku u HTML obliku.

Kada PHP obrađuje dokument,prvo se traže početni i završni PHP marker koji govore PHP-u da interpretira kod između njih.

```
<?php  
  
    // PHP kod se postavlja između markera  
  
?>
```

### Primjer 3. Osnovna sintaksa unutar PHP-a

Baš ovaj način označavanja koda omogućuje da se PHP kod uključi u rezne vrste dokumenata.Između PHP markera,ovaj tekst se nalazi nakon ( //).Dvije kose crte su PHP-u jedan od načina za označavanje i pisanja komentara.Ta linija koda se ne izvršava,već služi razvijatelju web aplikacije kao napomena,objašnjenje što se odvija u pojedinoj komentiranoj liniji.Komentiranje koda je izvrsna praksa koja omogućava,samom razvijatelju ali i drugim programerima praćenje logike rada web aplikacije.

Većinu vremena vidjet ćemo da je PHP uključen i ugniježđen unutar HTML dokumenata kao na sljedećem primjeru :

```
<p>Ovo je obični HTML tekst</p>  
  
<?php echo 'Ovaj tekst je napisan u PHP-u' ;?>  
  
<p>Ovo je isto obični HTML tekst</p>
```

### Primjer 4. PHP ugniježđen unutar HTML dokumenta



PHP kod uvijek mora biti između markera. Kod koji nije postavljen između PHP markera neće se izvršiti već će bit protumačen kao običan tekst i ispisat će se na stranici.

Korištena je jezična konstrukcija `echo` koja u PHP-u služi za ispis teksta koji se nalazi između jednostrukih ili dvostrukih navodnika nakon ključne riječi `echo`. Naime, da bi se unutar PHP markera ispisao neki tekst, nije dovoljno kao u HTML-u samo napisati tekst već je potrebno koristiti posebne jezične konstrukcije ili ugrađene PHP funkcije za ispis teksta. Isto vrijedi i za ostale elemente PHP jezika kao što su varijable, polja, petlje, funkcije, objekti..., itd. U kratko, svi elementi PHP jezika imaju svoja pravila pisanja ili pozivanja (ako su u pitanju funkcije ili metode) što će biti pokazano i objašnjeno kroz poglavlje izrade same web aplikacije za registar pomoraca.

#### **4.5. BAZA PODATAKA MYSQL**

Baza podataka je organizirani skup podataka. Oduvijek postoji potreba za vođenjem i čuvanjem nekakvih podataka i informacija. Uzmimo za primjer jednu tvrtku koja vodi podatke o svom poslovanju, izdanim računima, zaposlenicima, plaćama itd. Pogledajmo i primjer jednog fakulteta koji vodi podatke o svim studentima i profesorima, dvoranama i učionicama kojima raspolaže, rasporedu predavanja i još mnogim drugim stvarima. Na samo ova dva sveprisutna primjera vidimo da postoje velike količine podataka koje treba spremati i organizirati.

Prije pojave računala ovakvi su se podaci bilježili u razne knjige. Sada zamislimo se i upitajmo kakav je posao pronaći podatke o zaposlenima iz npr. Zadarske županije. Možda i nije neki problem ako ih je 10. Ali što ako ih je 720? Isto tako u odnosu na drugi primjer. Pronaći na fakultetu od 1500 studenata samo one studente koji su položili određeni kolegij? Dolaskom računalne ere koju poznajemo danas razvila su se programska rješenja za pretraživanje i dohvaćanje podataka. Došlo je do stvaranja baza podataka, specijaliziranog računalnog programa koji se brine za skladištenje i dohvaćanje podataka koji su u njega upisani. Da bi se moglo pristupiti tim podacima, osmišljen je novi programski jezik čija je namjena komunikacija s bazom podataka.

Taj novi jezik nazvan je SQL i zamišljen je kao jednostavan jezik,pun standardnih engleskih riječi.SQL je postao standard za komunikaciju s bazama podataka. Unutar SQL jezika ima mnogo implementacija,što komercijalnih,što oblika slobodnog softvera(eng *open source*).

MySQL je *open source* poslužitelj baze podataka (engl. *database server*). Poslužitelju baze podataka se može pristupiti preko mreže, ali je najčešće on smješten u istoj lokalnoj mreži kao i HTTP poslužitelj (često i na istom računalu). Na poslužitelju baze podataka može postojati veći broj baza koje su potpuno samostalne, no,više baza podataka može sadržati podatke vezane u jednu te istu primjenu (npr.unutar jedne Web aplikacije). Na poslužitelju baze podataka nužno je definirati

korisničke račune. Svakom korisničkom računu na poslužitelju moguće je dodijeliti razna administracijska prava za cijeli poslužitelj ili pojedine baze. Neka od prava bi bila stvaranje novih baza, pravo pristupa postojećim bazama, pravo uređivanja (unosa ili izmjena podataka) postojećih baza itd.

Poslužitelj MySQL podržava više različitih načina rada s obzirom na povezanost podataka unutar baze podataka. Svaka baza podataka ima neke posebnosti: bolje organizira ili brže dohvaća podatke,zauzima manje resursa itd. Ono što im je zajedničko je upravo SQL jezik. To je upravo velika prednost za programere web aplikacija, jer kad se jednom nauči SQL jezik, može se komunicirat s velikim brojem baza podataka, na raznim platformama bez potrebe daljnjeg proučavanja baze i učenja njenog jezika.

#### **4.6. BOOTSTRAP 3.0**

Bootstrap je programski okvir(eng.*framework*) koji služi za izradu web stranica i web aplikacija. Sam framework sadrži HTML i CSS predloške prema kojima su definirane forme (forms), dugmići (buttons), navigacija (navigation), tipografija i ostale komponente sučelja. Važno je napomenuti da bootstrap nakon verzije 2.0 u potpunosti podržava izradu responzivnih web stranica (stranica prilagodljivih veličini ekrana). Ono što ovaj framework čini sve više popularnijim je njegova cijena,besplatan je,te dobra dokumentacija koja omogućava brzu i laku implementaciju.

Bootstrap je podjeljen na module, te je uglavnom sačinjen od serije LESS stylesheets(eng.stil) koji implementiraju različite komponente alata. Primjerice stylesheet koji se zove bootstrap.less sadrži komponente stilova koji su predefinjirani kroz framework. Developeri mogu nadograditi i prilagoditi Bootstrap sebi i svojim potrebama tako što mogu odabrati koje komponente žele koristiti u svom projektu a koje ne.Ta vrsta selektivnosti uvelike olakšava i proširuje primjenu ovog frameworka.

Bootstrap dolazi sa standardnim 960 px širokim rasporedom rešetki (eng. *grid layoutom*). Developeri također mogu sami odrediti širinu web stranice kroz CSS varijablu width layout. Za oba slučaja postoje četiri različite varijacije alata koji omogućuju različite rezolucije i tip uređaja kao što su: mobilni uređaji, tableti, računala itd. Svaka varijacija prilagođava širinu stupaca.

Framework dolazi sa setom stilova (stylesheets) koji nam pružaju osnovni izgled svih ključnih HTML komponenti. Stilovi pružaju jedinstven i moderan prikaz za oblikovanje teksta, tablica i elemenata obrazaca.Uz uobičajene HTML elemente, bootstrap sadrži i ostale vrlo često korištene elemente sučelja kao što su dugmići sa naprednim mogućnostima (padajuće opcije, navigacijske liste, horizontalne i vertikalne kartice, navigacija, obilježavanje stranica) poruke upozorenja,i još mnogo toga.

Bootstrap dolazi sa nekoliko Javascript komponenti u obliku jQuery programskog dodatka(eng. *plugin*). Oni nam omogućuju dodatne korisničke elemente kao što su dijaloški okviri (eng. *Dialog Boxes*), razni okviri za prikaz slika,interaktivni elementi sa predefinjiranim stilovima, itd. Osim nadogradnje novih korisničkih elemenata Javascript nam omogućuje i unapređenje već postojećih elemenata.Kako bi koristili Bootstrap u HTML stranici potrebno je preuzeti bootstrap CSS file sa njihove službene stranice <http://getbootstrap.com> i uključiti ga u HTML dokument.

```
<head>

  <link rel="stylesheet"
href="//netdna.bootstrapcdn.com/bootstrap/3.1.1/css/bootstrap.min.css">

  <link rel="stylesheet"
href="//netdna.bootstrapcdn.com/bootstrap/3.1.1/css/bootstrap-
theme.min.css">

  <link
rel="stylesheet"href="//code.jquery.com/ui/1.10.4/themes/jquery-ui.css">

</head>
```

Primjer 5. Uključivanje Bootstrap CSS-a u HTML dokument

## 5. PREGLED PROGRAMSKIH PAKETA I ALATA PRI IZRADI WEB APLIKACIJE

### 5.1. WAMP SERVER

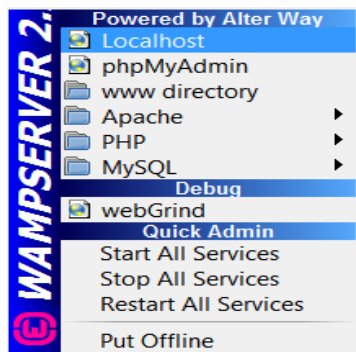
Kao što je navedeno, dinamičke web stranice ne možemo generirati iz bilo koje mape (eng. *folder*) na našem računalu. Za prikaz dinamičkih web aplikacija moramo imati web poslužitelj (eng. *server*) koji razumije i prevodi programski jezik te uspješno procesira našu web platformu.

Da bi dinamične stranice u potpunosti mogle ispravno funkcionirati potrebno je imati sljedeće elemente instalirane na računalu:

1. skriptni jezik u kojem je stranica napisana (npr. PHP)
2. server koji će sve to pokretati (npr. Apache)
3. server baze podataka (npr. MySQL)

Prije nekoliko godina, za izradu web aplikacije bilo je potrebno instalirati posebno PHP, posebno MySQL, posebno Apache. Danas, tu kompleksnu instalaciju možemo skratiti vrlo korisnim, programskim paketom zvanim WAMP (Windows Apache MySQL PHP). Preko WAMP programskog paketa u par jednostavnih koraka odjednom instaliramo sva tri navedena elementa na računalo.

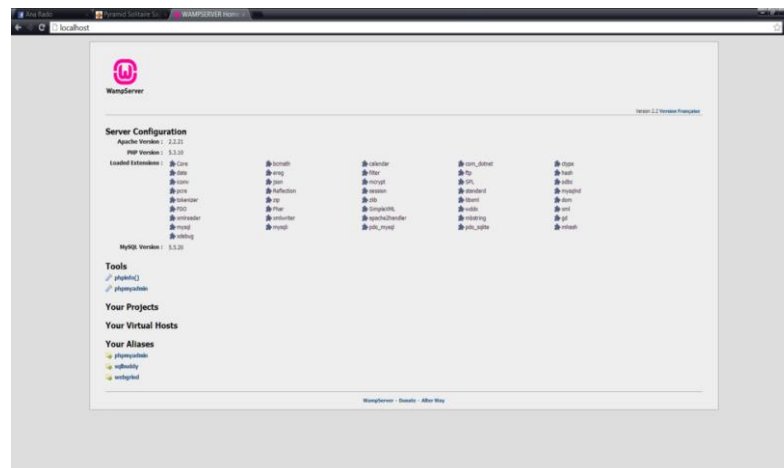
Nakon instalacije WAMP-a dobija se lokalni server kojem se u svakom trenutku može pristupiti klikom na Localhost u okviru WampServer panela. Localhost predstavlja lokalni server, server koji radi na korisnikovom računalu.



Slika 4. Pristup lokalnom serveru preko WAMP panela

Da bi se kreirane HTML stranice nalazile na lokalnom serveru one moraju biti sačuvane u www direktoriju koji se na računalu nalazi na adresi C:\wamp\www (ako su korištena predložena podešavanja prilikom instalacije WAMP-a). Dakle, sam folder www predstavlja lokalni server i u njega se smješta sav materijal koji treba biti na serveru i kojem će se pristupati preko browsera.

Nakon što unesemo “localhost” u naš web preglednik, prikazati će se ova stranica:



Slika 5. WAMP sučelje u lokalnom okruženju

Unutar ovog,localhost sučelja nalazi se pristup projektima i ostalim alatima za izradu web aplikacija,kao i prikaz posljednje verzije Apache web poslužitelja,verzije PHP-a i MySQL-a.

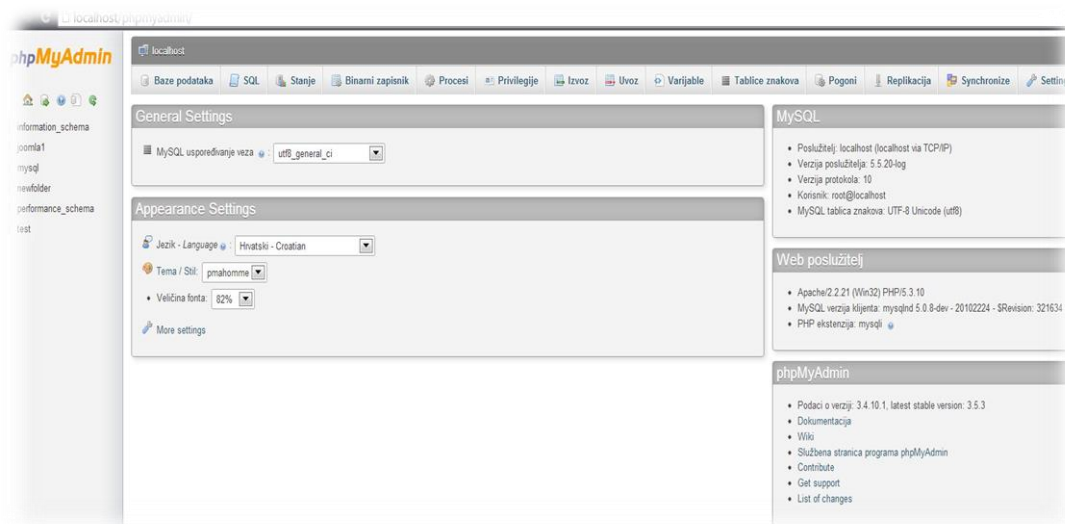
## **5.2. PHPMYADMIN**

Izvrstan alat kojem imamo pristup unutar našeg localhost sučelja,naziva se phpMyAdmin.

PhpMyAdmin je besplatna web aplikacija otvorenog,slobodnog koda pisana u PHP-u, a služi za upravljanje i administraciju MySQL baza podataka preko Web-a. Uz pomoć njega možemo izvršavati mnoge MySQL operacije putem korisničkog sučelja izravno u Internet pregledniku.

Možemo upravljati bazama podataka, tablicama, poljima, indeksima, korisnicima, dozvolama pristupa samim bazama, izvršavati vlastite upite na bazu preko sql jezika i još mnogo drugih stvari, navedene su radnje najčešće korištene.

Samoj aplikaciji pristupamo preko web preglednika na adresi tipa:  
<http://localhost/phpmyadmin> .



Slika 6. phpMyAdmin sučelje u lokalnom okruženju

Ovisno o postavkama poslužitelja, najčešće se ne zahtjeva lozinka za pristup aplikaciji, iako se takva opcija, a, i mnoge druge mogu promjeniti u konfiguraciji same aplikacije. Grafičko sučelje je prevedeno na 72 jezika i omogućava i ubrzava razvoj relacijskih baza podataka potrebnim za podršku i razvoj dinamičkih web aplikacija.



### 5.3. NETBEANS IDE

Za pisanje programskog koda potreban nam je tekstualni uređivač ili (eng. *text editor*). Dovoljno je da unutar text editor-a napisani kod spremimo u odgovarajućoj ekstenziji (.html, .php i sl.) i možemo ga pokrenuti preko našeg poslužitelja i web preglednika.

Profesionalni web razvijatelji radi ubrzanja radnog procesa, pregledne strukture projekata, prilagodbe vlastitom, osobnom stilu kodiranja koriste IDE (eng. *integrated development environment*). IDE je programska aplikacija koja omogućava niz mogućnosti za programere koje uvelike unaprijeđuju programski razvoj. Jedna od takvih programskih aplikacija je korištena u ovom diplomskom radu.

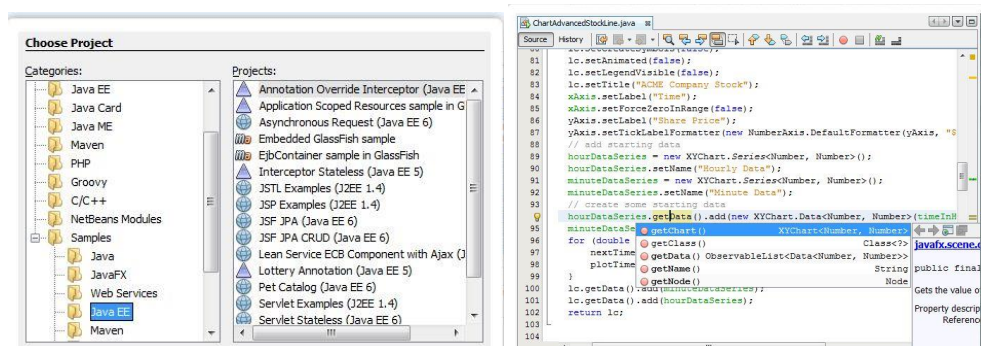
Netbeans je besplatan, moderan IDE sa velikom zajednicom programera diljem svijeta. Inicijalno je razvijen za Java programere, ali danas ima podršku za puno više ostalih programskih i meta jezika (PHP, C, C++, Ruby, HTML, CSS, JavaScript...). Netbeans nudi sve ono što jedan moderan IDE sadrži:

1. Organizacija koda u projekte
2. Podrška za više programskih jezika
3. Podrška za programske dodatke(plugin)
4. Bojanje teksta ovisno o sadržaju koda
5. Ugrađeni integrirani prevodioc

## 6. Napredne mogućnosti otkrivanja grešaka (debug)

## 7. Nadopunjavanje i memoriziranje koda (IntelliSense)

Netbeans je besplatan alat koji nudi organizirano radno okruženje i veliko ubrzanje radnog procesa, te odgovarajuću podršku i dokumentaciju. Ovaj besplatan alat se može preuzeti na službenoj stranici <https://netbeans.org/>.



Slika 7. Netbeans IDE

## 6. ARHITEKTURA WEB APLIKACIJE

Arhitekturu web aplikacije za registar pomoraca razdvajamo na osnovne elemente koji su potrebni za rad i funkcionalnost same aplikacije. Web aplikacija za registar pomoraca se temelji na prednostima PHP programskog jezika, poslužitelja baze podataka MySQL i programskog paketa (HTTP poslužitelja) Apache.

Apache je odabran iz više razloga, tj. zbog više mogućnosti koje programski paket nudi. Tako Apache sadrži implementiranu podršku za PHP programski jezik, nudi jednostavnost konfiguracije programskog paketa, a i ne zanemarujemo činjenicu da se Apache koristi kao HTTP poslužitelj na gotovo svim postojećim platformama. Aplikacijska logika i, dijelom, korisničko sučelje, su izvedeni u programskom jeziku PHP iz razloga što PHP nudi odličnu podršku za rad sa različitim vrstama baza podataka.

PHP programski jezik pojednostavljuje razvoj tako što razvijateljima nudi jednostavan način za umetanje dinamičkog sadržaja u statički HTML kod. Ti dinamički podaci su pohranjeni u bazi podataka. Kao poslužitelj baze podataka je odabran MySQL zbog svoje brzine odziva na zahtjeve PHP Web aplikacije i niskih hardverskih zahtjeva spram računala na kojem je instaliran.

Arhitektura sustava za registar pomoraca je zamišljena na način da se u bazi podataka MySQL spremaju podaci koji su potrebni PHP Web aplikaciji. U trenutku kada stigne HTTP zahtjev od korisnika do HTTP poslužitelja Apache, Apache upućuje zahtjev za HTML stranicom PHP Web aplikaciji. Prije nego pošalje traženu HTML stranicu poslužitelju Apache, Web aplikacija u HTML kod ubacuje dinamičke podatke, na za to unaprijed predviđena mjesta. Prije umetanja podataka u HTML kod, Web aplikacija podatke dohvaća iz baze podataka. U slučaju da je korisnik aplikacije putem sučelja „poslao“ podatke prema Web aplikaciji, ti podaci prolaze kroz poslužitelj Apache i Web aplikaciji postaju dostupni kroz HTTP varijable PHP skriptnog jezika. Nakon toga, Web aplikacija s tim podacima postupa u skladu sa željama korisnika.

## **7. IMPLEMENTACIJA**

### **7.1. ODABIR RADNE PLATFORME**

Prije opisa same implementacije potrebno je pojasniti razloge odabira platforme koja je poslužila pri razvoju aplikacije za registar pomoraca. Nakon proučavanja nekoliko različitih kombinacija platformi i alata za razvoj Web aplikacija odabrane su sljedeće tehnologije: kombinacija skriptnog jezika PHP, meta jezika HTML i CSS(za aplikacijsku logiku i osnovnu strukuru), baza podataka MySQL (za čuvanje svih podataka vezanih uz rad aplikacije) te front-end programski okvir Bootstrap (korisničko sučelje i organizacija podataka).

Za razvoj aplikacije su korišteni potpuno besplatni alati :

1. Wamp server 2.4
2. PHP prevodioc 5.4.16
3. HTTP poslužitelj Apache 2.4.4
4. poslužitelj za bazu podataka MySQL 5.6.12
5. phpMyAdmin 4.2.9
6. Bootstrap 3.0

## 7. Netbeans IDE 8.0

Ovi alati, iako besplatni, omogućuju jedno u potpunosti funkcionalno i profesionalno radno okruženje u kojem samo projekt i znanje razvijatelja uvjetuje stupanj kompleksnosti web platforme koja se razvija. WAMP server nam pruža Apache lokalni server na našem računalu, skriptni jezik PHP i podršku za bazu podataka MySQL. Preko Netbeans IDE uređivača koda ubrzavamo cjelokupni radni proces i strukturu projekta. Bootstrap CSS programski okvir nam omogućuje responzivnost i razvoj korisničkog sučelja. Sve ove alate preuzimamo sa službenih web stranica, gdje imamo pristup tehničkoj dokumentaciji. Kroz 30-ak minuta, koliko je potrebno za instalaciju navedenih alata dobivamo jedno moderno radno okruženje koje nam omogućuje razvoj i implementaciju web stranica i web aplikacija.

### **7.2. BAZA PODATAKA MYSQL**

Iako je korisničko sučelje prvo s čime se korisnik susreće, opis implementacije aplikacije za registar pomoraca će krenuti od opisa baze podataka. Sam opis ide u tom smjeru jer je to i najprirodniji put razvijanja jedne web aplikacije. Nakon početne ideje korisničkog sučelja, koje možemo skicirati i na komadu papira krećemo na izradu same baze podataka.

Baza podataka je skup međusobno povezanih podataka pohranjenih na vanjskoj memoriji, istodobno dostupnih raznim korisnicima i programima. Podaci koji su pohranjeni u nekoj bazi fizički su pohranjeni kao datoteke, pa se postavlja pitanje po čemu se baza podataka razlikuje od izravne datoteke. Ono što čini razliku između istih je postojanje sustava za upravljanje bazom podataka (engl. Database Management System, skraćeno DBMS).

Ključna razlika između baze podataka i datoteke je svojstvo nezavisnosti podataka koju pruža DBMS. DBMS je poslužitelj baze podataka koji se nalazi između korisnika baze podataka i same baze. On u ime korisnika obavlja sve operacije s podacima.

### 7.2.1. Modeliranje entiteta i veza

Entitet je objekt, pojava ili događaj o kojem želimo spremati podatke. To je nešto što se može identificirati (npr. student, kupac, proizvođač, polaznik tečaja, pomorac, agent) itd. Svaki entitet ima svoje značajke koje ga opisuju jednoznačno ili jedinstveno, a nazivaju se atributi. Na primjer, atributi pomorca su ime, prezime, visina, težina, adresa itd. Za svaki se entitet stvara tablica koju nazivamo relacija.

Veza je nešto što veže dva ili više entiteta. Razlikujemo tri vrste binarnih veza. Te veze nazivamo još i funkcionalnosti veze :

· 1 – 1 veza (ili 1:1) – jednom zapisu iz jedne tablice odgovara jedan i samo jedan zapis iz druge tablice (npr. Proizvod ima jamstveni list –svaki proizvod može imati samo jedan jamstveni list koji se odnosi upravo na taj proizvod)

· 1 – N (ili 1:∞) – jednom zapisu iz jedne tablice odgovara 0,1 ili više zapisa iz druge tablice

(npr. Profesor predaje kolegij – jedan profesor može predavati više kolegija na fakultetu)

· M – N (ili ∞:∞) – jedan zapis prvog entiteta može biti u vezi s 0,1 ili više primjeraka drugog entiteta, te također jedan zapis drugog entiteta može biti u vezi s 0,1 ili više zapisa prvog entiteta (npr. Pomorac upisuje tečaj – više pomoraca može upisati isti tečaj, a također jednom tečaju može prisustvovati više pomoraca)

Da bi naša web aplikacija uspješno “komunicirala” s našom bazom podataka i izvršavala zadatke koje zatraži korisnik, esencijalno je osigurati pravilno modeliranje (eng.*database design*) baze podataka. To se postiže ispravnim procesom modeliranja entiteta i veza.

### **7.2.2. Relacije i atributi web aplikacije za registar pomoraca**

#### **I. Relacije za korisnike web aplikacije :**

Tablica Administratori - unutar ove tablice nalaze se podaci o administratorima unutar web aplikacije. Sastoji se od sljedećih atributa :

1. *id* - atribut koje sadrži jedinstvenu korisničku oznaku,
2. *name* - atribut koje sadrži ime administratora,
3. *last\_name* - atribut koje sadrži prezime administratora,
4. *username* - atribut koje sadrži korisničko ime administratora,
5. *password* - atribut koje sadrži zaporku administratora

Tablica Agenti - unutar ove tablice nalaze se podaci o administratorima unutar web aplikacije. Sastoji se od sljedećih atributa :

1. *id* - atribut koje sadrži jedinstvenu korisničku oznaku,
2. *name* - atribut koje sadrži ime agenta,
3. *last\_name* - atribut koje sadrži prezime agenta,
4. *username* - atribut koje sadrži korisničko ime agenta,
5. *password* - atribut koje sadrži zaporku agenta

II. Relacije korištene za registar pomoraca:

Tablica *Certifikati* - unutar ove tablice nalaze se nazivi certifikata koje ima pojedini pomorac u registru :

1. *cert\_pk* - atribut koje sadrži jedinstvenu oznaku certifikata,
2. *qualification* - atribut koje sadrži ime certifikata



Tablica pomorac\_certifikat – pomoćna relacija (M:N veza) –unutar ove tablice nalaze se dodatni opisni podaci vezani za sam certifikat :

1. *pk* - atribut koje sadrži jedinstvenu oznaku,
2. *pomorac\_fk* - atribut koji sadrži pripadnu šifru pomorca,
3. *cert\_fk* - atribut koje sadrži pripadnu šifru certifikata,
4. *country* - atribut koje sadrži ime države u kojoj je certifikat izdan,
5. *place* - atribut koje sadrži ime mjesta gdje je certifikat izdan,
6. *date* - atribut koje sadrži datum kada je certifikat izdan,
7. *number* - atribut koje sadrži identifikacijski broj certifikata

Tablica matrikule – unutar ove tablice nalaze se podaci o knjižicama plovidbe pojedinog pomorca u registru :

1. *matrikula\_pk* - atribut koje sadrži jedinstvenu oznaku knjižice plovidbe,
2. *pomorac\_fk* - atribut koji sadrži pripadnu šifru pomorca,

3. *country\_issue* - atribut koje sadrži ime države u kojoj je knjižica izdana ,
4. *place\_issue* - atribut koje sadrži ime mjesta gdje je knjižica plovidbe izdana,
5. *date\_issue* - atribut koje sadrži datum kada je knjižica plovidbe izdana,
6. *date\_expiry* - atribut koje sadrži datum isteka knjižice plovidbe

Tablica medic – unutar ove tablice nalaze se podaci o lječničkim uvjerenjima pojedinog pomorca u registru :

1. *medic\_pk* - atribut koje sadrži jedinstvenu oznaku lječničkog uvjerenja,
2. *pomorac\_fk* - atribut koji sadrži pripadnu šifru pomorca,
3. *country\_issue* - atribut koje sadrži ime države u kojoj je uvjerenje izdano ,
4. *place\_issue* - atribut koje sadrži ime mjesta gdje je uvjerenje izdano,
5. *date\_issue* - atribut koje sadrži datum kada je uvjerenje izdano,
6. *date\_expiry* - atribut koje sadrži datum isteka lječničkog uvjerenja

Tablica operator – unutar ove tablice se nalazi naziv certifikata za operatera :

1. *cert\_pk* - atribut koje sadrži jedinstvenu oznaku certifikata,
2. *operator* - atribut koje sadrži ime certifikata za operatera

Tablica *pomorac\_operator* – pomoćna relacija (M:N veza) –unutar ove tablice nalaze se dodatni opisni podaci vezani za sam certifikat operatera :

1. *pk* - atribut koje sadrži jedinstvenu oznaku,
2. *pomorac\_fk* - atribut koji sadrži pripadnu šifru pomorca,
3. *cert\_fk* - atribut koje sadrži pripadnu šifru certifikata,
4. *country* - atribut koje sadrži ime države u kojoj je certifikat izdan,
5. *place* - atribut koje sadrži ime mjesta gdje je certifikat izdan,
6. *date* - atribut koje sadrži datum kada je certifikat izdan,
7. *number* - atribut koje sadrži identifikacijski broj certifikata

Tablica pomorci - unutar ove tablice nalaze se osobni podaci za pojedinog pomorca u registru :

1. *pomorac\_pk* - atribut koje sadrži jedinstvenu oznaku za pojedinog pomorca,
2. *name* - atribut koji sadrži ime pomorca,
3. *last\_name* - atribut koje sadrži prezime pomorca,
4. *year\_birth* - atribut koje sadrži datum rođenja pomorca,
5. *nationality* - atribut koje sadrži nacionalnost pomorca,
6. *city* - atribut koje sadrži ime grada u kojem je pomorac prijavljen,
7. *address* - atribut koje sadrži adresu prebivališta pomorca
8. *closest\_airport* - atribut koje sadrži ime najbližeg grada sa aerodromom u odnosu na prebivalište,
9. *br\_tel* - atribut koji sadrži kućni telefonski broj pomorca,
10. *br\_mob* - atribut koje sadrži broj mobitela pomorca,
11. *email* - atribut koje sadrži naziv elektroničke pošte pomorca,
12. *company* - atribut koje sadrži naziv kompanije pod kojom pomorac plovi,

13. *status*- atribut koje sadrži informaciju, je li pomorac na kopnu ili na moru

14. *file* - atribut koje sadrži pripadnu sliku pomorca

Tablica praksa – unutar ove tablice nalaze se podaci o prijašnjoj plovidbi pomorca u registru :

1. *peractice\_pk* - atribut koji sadrži jedinstvenu oznaku prakse pomorca,
2. *pomorac\_fk* - atribut koje sadrži pripadnu šifru pomorca,
3. *vessel\_name* - atribut koje sadrži naziv broda gdje je praksa služena,
4. *type* - atribut koje sadrži tip broda na kojem je praksa služena,
5. *flag* - atribut koje sadrži ime zastave pod kojem je pomorac plovio
6. *company* - atribut koje sadrži ime kompanije pod kojim je pomorac plovio
7. *gt* - atribut koje sadrži tip broda na kojem je praksa služena,
8. *from\_* - atribut koje sadrži datum početka plovidbe
9. *from\_* - atribut koje sadrži datum kraja plovidbe

Tablica putovnice – unutar ove tablice nalaze se podaci o putovnici pojedinog pomorca u registru:

1. *putovnice\_pk* - atribut koje sadrži jedinstvenu oznaku putovnice pomorca,
2. *pomorac\_fk* - atribut koji sadrži pripadnu šifru pomorca,
3. *country\_issue* - atribut koje sadrži ime države u kojoj je putovnica izdana,
4. *place\_issue* - atribut koje sadrži ime mjesta gdje je putovnica izdana,
5. *number* - atribut koje sadrži broj putovnice,
6. *date\_issue* - atribut koje sadrži datum kada je putovnica izdana,
7. *date\_expiry* - atribut koje sadrži datum isteka putovnice pomorca

Tablica tečajevi – unutar ove tablice nalaze se podaci o sigurnosnim tečajevima koje pomorac mora imati za ukrcaj na plovni objekt :

1. *tečaj\_pk* - atribut koje sadrži jedinstvenu oznaku tečaja,
2. *course* - atribut koji sadrži naziv pojedinog tečaja,
3. *number\_id* - atribut koji sadrži serijski broj sigurnosnog tečaja,

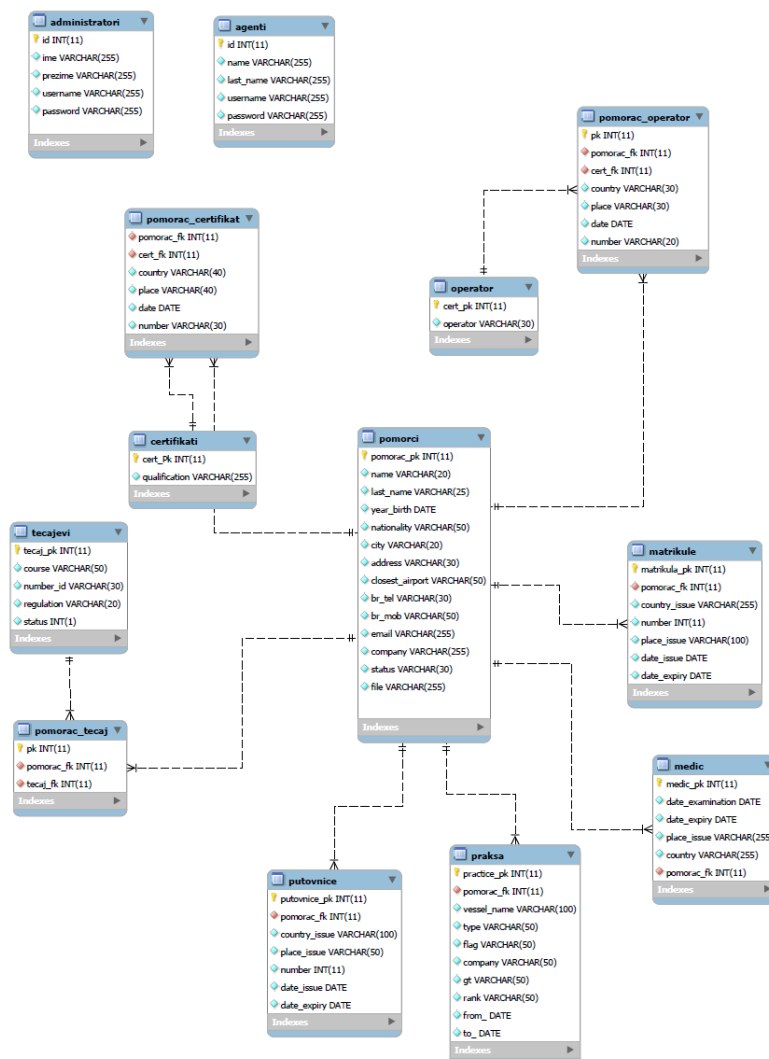
4. *regulation* - atribut koje sadrži naziv,šifru regulacije po kojoj je tečaj izdan
5. *status* - atribut koji omogućava dodatnu kontrolu pri izradi korisničkog sučelja

Tablica pomorac\_tecaj – pomoćna relacija (M:N veza) :

1. *pk* - atribut koje sadrži jedinstvenu oznaku,
2. *pomorac\_fk* - atribut koji sadrži pripadnu šifru pomorca,
3. *tecaj\_fk* - atribut koje sadrži pripadnu šifru tečaja

### 7.2.3. ER dijagram

ER dijagram(eng.*Entity Relationship Diagram*) je dijagram koji pokazuje relacije između entiteta u sustavu baze podataka.ER dijagram je dovoljno jednostavan da ga mogu razumjeti ljudi različitih struka.Zato ER dijagram služi za komunikaciju projektanta baze podataka i korisnika,i to u najranijoj fazi razvoja baze. Stvaranje ER dijagrama je iterativan proces,odnosno kako bismo došli do završne verzije,moramo “dizajnirati” više probnih verzija dok ne dođemo do odgovarajućeg zapisa,a time i do valjanog modela naše baze podataka.Sa prikupljanjem iskustva smanjuje se broj probnih dijagrama. Nakon što dobijemo naš završni ER dijagram,pristupamo izradi same baze podataka. Baza se iz ER dijagrama dobiva pretvorbom elemenata dijagrama u tablice.



Slika 8. ER dijagram baze podataka Agencija



Agencija je naziv baze podataka korištene u izradi web aplikacije i prikazana je na sljedećoj(slika 9)slici. Kao što vidimo, sve elemente dijagrama pretvaramo u tablice. Unutar tih tablica nalaze se svi podaci kojima aplikacija upravlja.Upravljanje podacima razvijatelj implementira korištenjem SQL jezika.

Table	Action	Rows	Type	Collation	Size	Overhead
administratori	Browse Structure Search Insert Empty Drop	~0	InnoDB	utf8_general_ci	16 KiB	-
agenti	Browse Structure Search Insert Empty Drop	~0	InnoDB	utf8_general_ci	16 KiB	-
certifikati	Browse Structure Search Insert Empty Drop	~27	InnoDB	utf8_general_ci	16 KiB	-
matrikule	Browse Structure Search Insert Empty Drop	~10	InnoDB	utf8_general_ci	32 KiB	-
medic	Browse Structure Search Insert Empty Drop	~10	InnoDB	utf8_general_ci	32 KiB	-
operator	Browse Structure Search Insert Empty Drop	~0	InnoDB	utf8_general_ci	16 KiB	-
pomorac_certifikat	Browse Structure Search Insert Empty Drop	~4	InnoDB	utf8_general_ci	48 KiB	-
pomorac_operator	Browse Structure Search Insert Empty Drop	~2	InnoDB	utf8_general_ci	48 KiB	-
pomorac_tecaj	Browse Structure Search Insert Empty Drop	~12	InnoDB	utf8_general_ci	48 KiB	-
pomorci	Browse Structure Search Insert Empty Drop	~10	InnoDB	utf8_general_ci	16 KiB	-
praksa	Browse Structure Search Insert Empty Drop	~2	InnoDB	utf8_general_ci	32 KiB	-
putovnice	Browse Structure Search Insert Empty Drop	~10	InnoDB	utf8_general_ci	32 KiB	-
tecajevi	Browse Structure Search Insert Empty Drop	~10	InnoDB	utf8_general_ci	16 KiB	-
<b>13 tables</b>	<b>Sum</b>	<b>97</b>	<b>InnoDB</b>	<b>utf8_general_ci</b>	<b>368 KiB</b>	<b>0 B</b>

Slika 9. Baza podataka Agencija unutar phpMyAdmin sučelja

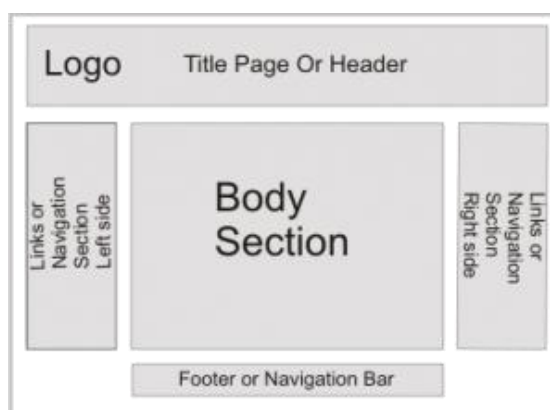
## 7.3. KORISNIČKO SUČELJE

Nakon modeliranja i izrade baze podataka dolazimo do dijela kada te iste podatke koje sadrži naša baza prikazujemo korisniku, a u sljedećem koraku procesa razvoja, omogućujemo korisniku da tim podacima i upravlja. Razvoj korisničkog sučelja možemo podijeliti u dvije faze :

1. Osnovna struktura web platforme

2. Završno grafičko korisničko sučelje

Najčešće i prije dizajniranja ER dijagrama, modeliranja baze podataka developer ima “plan”, ideju kako će web platforma biti prezentirana korisniku. Početak provođenja te ideje spada u osnovnu strukturu web platforme koja se razvija. Dakle, developer može skicirati i na papiru samo osnovni raspored (eng. *layout*) elemenata koja platforma nudi. To nikako nije završni dizajn web platforme nego samo svojevrsan “nacrt” sustava koji se razvija. Tako samo od prilike postavimo elemente, svojevrsne blokove za elemente naše aplikacije. Tako, primjerice postavimo zaglavlje (eng. *header*), mjesto za logotip, navigaciju (eng. *navigation*), glavni dio stranice ili tijelo (eng. *body*), podnožje (eng. *footer*), u sklopu same navigacije poveznice (eng. *link*) na druge elemente platforme i sl.



Primjer 6. Osnovna struktura web platforme

Ovim postupkom developer već na sam pogled na “nacrt” ima jasniju sliku razvojnog procesa web platforme. Nakon što preko HTML i CSS meta jezika postavimo osnovnu strukturu, tj elemente naše web platforme možemo krenuti s aplikacijskom logikom, tj omogućujemo da naši elementi postanu funkcionalni.

Nakon što smo odredili osnovnu strukturu i izveli aplikacijsku logiku (PHP programski jezik) krećemo na izradu završnog korisničkog sučelja aplikacije. Kod izrade korisničkog sučelja u fokus stavljamo jednostavnost interakcije korisnika i web platforme. Tako vodimo računa o detaljnom rasporedu elemenata na našoj platformi, gdje ovisno o kompleksnosti sustava, omogućavamo korisnicima čim lakši, vizualni pristup svim funkcionalnostima koje aplikacija nudi. Nakon dostupne interakcije s korisnikom radimo na dizajnerskom aspektu platforme. Tako vodimo računa o ikonama, linkovima, slikama, padajućim izbornicima, klizačima (engl. *slider*), bojama, teksturama, raznim vrstama navigacije itd. Ovim postupkom dobivamo tzv GUI (engl. *graphical user interface*). Preko našeg GUI-a osiguravamo interaktivnost korisnika i web platforme, te sukladno kako je izveden postavljamo standard profesionalnosti cjelokupnog izgleda naše web platforme.

Korisničko sučelje nije napravljeno isključivo korištenjem PHP jezika, već je korišten i spomenuti HTML koji je standard za izradu Web stranica. PHP je korišten za sve napredne mogućnosti, odnosno dinamički prikaz podataka, a HTML i CSS za definiranje izgleda stranica. Korisničko sučelje podijeljeno u različite cjeline: administratorsko sučelje i sučelje za agente. Posebna je cjelina objektni dio aplikacije, koji obavlja specifičan posao koji ne spada u korisničko sučelje, ali je uz njega vrlo blisko vezan kao što će biti i pokazano u nastavku. Svaka se cjelina sastoji od nekoliko zasebnih datoteka, te će one ovdje biti objašnjene sa organizacijskog aspekta aplikacije i aspekta organizacije koda, dok će same funkcionalnosti (s vizualnim primjerima) biti pojašnjene u dodatku.

Kroz ovaj rad, osnovni elementi GUI-a su izvedeni i implementirani preko CSS programskog okvira Bootstrap 3.0.

## 7.4. APLIKACIJSKA LOGIKA

Kao što je navedeno u uvodu, PHP programski jezik ima podršku za objektno orijentirano programiranje (engl.kratice *OOP*). *OOP* je metoda programiranja kojoj je temeljni princip da se klasa definira kao jedinstvena programska cjelina. Unutar klase omogućavamo združivanja php funkcija i podataka s definiranim korisničkim sučeljem.Princip je taj da preko tih funkcija koje se u *oop*-u nazivaju metode omogućujemo komunikaciju s bazom podataka,a samim time i kompletno upravljanje podacima potrebnim za rad aplikacije.

Komunikaciju s bazom ostvarujemo preko SQL (engl. *Structured Query Language*) jezika. Izvršavaju se upiti na bazu gdje definiramo način na koji želimo upravljat podacima.Koristeći ključne elemente upita na bazu, definiramo da li želimo podatke izvlačit iz baze (*SELECT* dio upita),da li ubacujemo podatke u bazu (*INSERT* dio upita), da li radimo izmjene nad podacima(*UPDATE* dio upita) ili te iste podatke brišemo iz baze podataka (*DELETE* dio upita). Dovoljno je onda da “pozovemo” metodu koju smo definirali i kroz upravljanje podacima prikazujemo korisniku funkcionalnost aplikacije.

Kao što je navedeno, kodiramo datoteke u kojima definiramo klase, koje se ponašaju kao jedinstvene programske cjeline. Kroz razvoj aplikacije za registar pomoraca imamo niz klasa koje zasebno upravljaju određenim podacima.

Datoteke koje sadrže klase:

1. *Administrator.php* – klasa koja sadrži sve metode potrebne za manipulaciju podataka vezanih za administratore aplikacije(izvlačenje podataka o administratorima,unos administratora u bazu, brisanje iz baze, modifikacija njihovih podataka, upravljanje kontrolom pristupa itd.),

2. *Agent.php* – klasa koja sadrži sve metode potrebne za manipulaciju podataka vezanih za agente aplikacije(izvlačenje podataka o agentima,unos novih agenata u bazu, brisanje iz baze, modifikacija njihovih podataka,metode za autorizaciju agenta, itd.),
3. *Certifikat.php* – datoteka koja sadrži sve metode potrebne za manipulaciju certifikata kroz aplikaciju(izvlačenje certifikata iz baze,unos certifikata u bazu, brisanje iz baze, modifikacija njihovih podataka),
4. *Tecaj.php* – datoteka koja sadrži sve metode potrebne za manipulaciju sigurnosnih tečaja kroz aplikaciju(izvlačenje iz baze,unos tečajeva u bazu, brisanje iz baze, modifikacija njihovih podataka),
5. *Databse.php* – datoteka koja sadrži sve funkcije potrebne za sigurnu konekciju na bazu podataka,kao i metode za pripremu SQL upita,
6. *Matrikula.php* – datoteka koja sadrži sve metode potrebne za manipulaciju
7. knjižica plovidbe pomoraca (izvlačenje podataka iz baze,unos knjižica u bazu, brisanje iz baze, modifikacija njihovih podataka),
8. *Medic.php* – datoteka koja sadrži sve metode potrebne za manipulaciju
9. lječničkih uvjerenja pomoraca (izvlačenje podataka iz baze,unosu bazu, brisanje iz baze, modifikacija njihovih podataka),
10. *Pagination.php* – datoteka koja sadrži metode za omogućavanje organiziranog prikaza podataka kroz korisničko grafičko sučelje aplikacije,
11. *Pomorac.php* – datoteka koja sadrži sve metode potrebne za manipulaciju podataka o pomorcu (izvlačenje podataka iz baze,dodavanje podataka u bazu, brisanje iz baze,preuzimanje fotografije pomorca,itd.),

12. *Pomorac\_Certifikat.php* – datoteka koja sadrži sve metode potrebne za manipulaciju podataka pomorca i dodatnih detalja o pripadnom certifikatu (izvlačenje podataka iz baze,dodavanje knjižica u bazu, brisanje iz baze, modifikacija njihovih podataka),
  
13. *Pomorac\_Tecaj.php* – datoteka koja sadrži sve metode potrebne za manipulaciju podataka pomorca i dodatnih detalja o pripadnim sigurnosnim tečajevima (izvlačenje podataka iz baze,dodavanje u bazu, brisanje iz baze, modifikacija njihovih podataka),
  
14. *Praksa.php* – datoteka koja sadrži sve metode potrebne za manipulaciju
15. podataka o prijašnjoj plovidbi pomorca (izvlačenje podataka iz baze,unos u bazu, brisanje iz baze, modifikacija podataka),
  
16. *Praksa.php* – datoteka koja sadrži sve metode potrebne za manipulaciju podataka o putovnicama pomorca (izvlačenje podataka iz baze, unos u bazu, brisanje iz baze, modifikacija podataka),
  
17. *Session.php* – datoteka koja sadrži sve metode potrebne za ulaz (engl.*login*) i izlaz (engl.*logout*) administratora i agenta kroz sustav,kao i metode za ispis poruka upozorenja unutar aplikacije

Kroz ove navedene klase omogućavamo kontrolu podataka iz baze,razvijamo metode,preko kojih vršimo upravljanje i funkcionalnost web aplikacije.

```
<?php

class Pomorac {

    public static function findAll() {

        return self::find_by_sql('SELECT * FROM pomorci');

    }

}

?>
```

#### Primjer 7. Metoda klase pomorac

Iz primjera 7 je prikazana jedna metoda unutar klase Pomorac. Klasa Pomorac sadrži još metoda, a ovo je primjer metode gdje prikupljamo sve podatke iz tablice pomorac u bazi podataka. Koristi se SELECT upit na bazu preko kojeg izvlačimo podatke iz baze. Preko SQL upita koji je lako čitljiv jezik, kao da bazi kažemo “nađi sve, prikupi sve podatke iz tablice pomorac”. Instanciranjem objekta Pomorac, možemo u bilo kojem dijelu našeg koda pozvati ovu metodu i spremit navedene podatke u objekt. Tada iteracijom preko sintakse PHP programskog jezika prikazujemo i upravljamo podacima ovisno o funkcionalnosti web platforme.

Aplikacijska logika se sastoji od objektnog dijela, gdje definiramo klase preko kojih omogućujemo upravljanje podacima, i dodatnih procedura, tj php funkcija koje kodiramo da bi riješavali određene zadatke. Tako se preko php funkcija, omogućuje pretraživanje pomoraca po određenim kriterijima, padajući izbornici gdje će biti navedeni certifikati, sigurnosni tečajevi, status pomorca, funkcije za formatiranje datuma itd.

```
<?php

function certificate_dropdown() {

    $db = new Database();

    $result_set = Certifikat::findAll();

    echo '<select class="form-control" name="main_cert" >';

    echo "<option selected>Certificate of Competence</option>";

    foreach($result_set as $cert) {

        echo "<option value=" . $cert->cert_Pk . ">" . $cert->qualification . "</option>";

    }

    echo "</select>";

}

?>
```

Primjer 8. PHP Funkcija za padajući izbornik koji prikazuje certifikate



Primjer 8. prikazuje jednu php funkciju preko koje omogućujemo prikaz podataka, u obliku padajućeg izbornika (engl. *dropdown*). Imenujemo funkciju `certificate_dropdown`, instanciramo objekt `$db` za konekciju za bazu. Zatim u objekt `$result_set` spremamo podatke o certifikatima. Ovo činimo pozivajući klasu `Certifikat` i metodu `findAll()`. Zatim, unutar html dropdown-a obavljamo iteraciju preko php petlje `foreach` i ubacujemo potrebne podatke u padajući izbornik.

Funkcionalnost ovog pristupa je, što koristeći objekte i funkcije odvajamo aplikacijsku logiku od korisničkog sučelja. Time dobivamo jasniju, uređeniju organizaciju i strukturu koda i datoteka kroz cijeli projekt.

#### 7.4.1. ORGANIZACIJA KODA

Kao što je navedeno u prijašnjim poglavljima, PHP programski jezik se uključuje unutar HTML meta jezika. Tako se istovremeno postiže struktura web aplikacije i dinamičko upravljanje.

Slijedi primjer koda aplikacije, koji obavlja organiziranje dijelova korisničkog sučelja u smislenu cjelinu i prikaz jednog dijela dinamičkih podataka :

```
<?php $pomorci = Pomorac::findAll(); ?>

<table class="table table-hover">

<tr><th>Full name</th><th>Company</th><th>Status</th></tr>

<?php foreach ($pomorci as $pomorac): ?>
```

```

<tr>

<td><a href="report.php?pomorac=<?php echo $pomorac->pomorac_pk; ?>">

<?php echo '<h4>' . $pomorac->fullName() . '</h4>'; ?></a></td>

<td><?php echo '<h5>' . $pomorac->company . '</h5>'; ?></td>

<td><?php echo '<h5>' . $pomorac->status . '</h5>'; ?></td>

</tr>

<?php endforeach; ?>

</table>

```

#### Primjer 9. Kombinacija HTML i PHP elemenata

Predhodni primjer prikazuje kako se efektivno spajaju HTML i PHP pri čemu se postiže organizacija koda i odvajanje aplikacijske logike od korisničkog sučelja. Unutar varijable \$pomorci pozivamo definiranu klasu Pomorac i njenu metodu koja pronalazi sve podatke o pomorcu iz tablice pomorac unutar naše baze podataka. Zatim preko HTML tablice organiziramo podatke u smislenu cjelinu. Konstruiramo tablicu u koju spremamo podatke o pomorcima. U ovom primjeru, to su njihovo ime i prezime, kompanija pod kojom plove i status pomorca. Ovdje jasno vidimo kako koristimo HTML elemente za organizaciju i strukturu koda i PHP klasu za prikupljanje i prikaz podataka iz baze podataka.

## 7.4.2. OPIS I NAMJENE DATOTEKA APLIKACIJE

Objektni dio aplikacije i pripadne datoteke su navedene i opisane u poglavlju koje govori o aplikacijskoj logici. Sada će biti navedene i objašnjene sve ostale datoteke koje čine web aplikaciju za registar pomoraca. Možemo ih podijeliti na konfiguracijske datoteke, tj datoteke koje uvjetuju funkcionalno radno okruženje, korisničke datoteke, tj datoteke korisničkog sučelja i upravljačke datoteke, tj datoteke potrebne za sam rad aplikacije.

### I. Konfiguracijske datoteke :

1. *config.php* – datoteka koja sadrži PHP konstante definirane za spajanje na bazu podataka
2. *css.php* – datoteka koja sadrži CSS skripte od Bootstrap programskog okvira, kao i sve stilove i css pravila koja su korištena kroz aplikaciju (pozicioniranje elemenata, boje, fontovi, itd.)
3. *functions.php* – datoteka koja sadrži funkcije za obavljanje određenih zadataka korištene kroz aplikaciju
4. *initialize.php* – datoteka koja sadrži sve ostale datoteke potrebne za implementiranje korisničkog sučelja kao i datoteke za aplikacijsku logiku (Bootstrap skripte, css skripte, funkcije, klase)
5. *js.php* – datoteka koja sadrži JavaScript skripte od Bootstrap programskog okvira

## II. Korisničke datoteke :

1. *index.php* – početna datoteka agenta nakon ulaza u sustav s jasno definiranim funkcionalnostima preko grafičkog sučelja
2. *admin.php* – početna datoteka administratora nakon ulaza u sustav s jasno definiranim funkcionalnostima preko grafičkog sučelja
3. *report.php* – datoteka kojoj korisnik pristupa preko *index.php* datoteke koja sadrži sve podatke o pojedinom pomorcu
4. *navigation.php* – datoteka koja sadrži korisničku navigaciju datoteke *login.php*
5. *navigation2.php* – datoteka koja sadrži korisničku navigaciju datoteke *index.php*
6. *navigation\_admin.php* – datoteka koja sadrži korisničku navigaciju administratorskog dijela sučelja
7. *navigation\_report.php* – datoteka koja sadrži korisničku navigaciju datoteke *report.php*

### III. Upravljačke datoteke :

1. *login.php* – iako i korisnička datoteka, jer je ova datoteka prvo što korisnik vidi pri pokretanju aplikacije, *login.php* je datoteka u kojoj je izveden siguran ulaz agenta u sustav (provjera korisničkog imena i zaporke)
2. *logout.php* – datoteka preko koje agent izlazi iz sustava
3. *admin\_login.php* – iako i korisnička datoteka, *admin\_login.php* je datoteka u kojoj je izveden siguran ulaz administratora u sustav (provjera korisničkog imena i zaporke)
4. *admin\_logout.php* – datoteka preko koje administrator izlazi iz sustava
5. *admin\_insert.php* – datoteka preko koje se omogućava preuzimanje i unos slike pomorca kao i unos osobnih podataka, putne i medicinske dokumentacije
6. *admin\_insert2.php* – datoteka preko koje se omogućava unos sigurnosnih tečajeva, certifikata i prijašnje povidbe
7. *admin\_update.php* – datoteka preko koje se omogućava izmjene podataka o pomorcu (preuzimanje slike pomorca, unos osobnih podataka, putne i medicinske dokumentacije)
8. *admin\_update2.php* – datoteka preko koje se omogućava izmjena i nadopuna podataka o sigurnosnim tečajevima, certifikatima, prijašnjoj plovidbi

9. *admin\_delete.php* – datoteka preko koje se pomorac i svi njegovi podaci brišu iz registra
  
10. *form\_process.php* – datoteka koja sadrži sve upite na bazu, kao i upravljačku logiku za unos pomorca u registar
  
11. *update\_process.php* – datoteka koja sadrži sve upite na bazu, kao i upravljačku logiku za izmjenu pomorca unutar registra

## 8. RANJIVOST PHP APLIKACIJA I NJIHOVA ZAŠTITA

### 8.1. SQL INJECTION

Sigurnost naše web platforme, podataka u bazi i podataka samih posjetitelja najveća su briga programera. Neoprezno rukovanje podacima koje web platformi šalju korisnici preko web formi može dovesti do velikih šteta, od toga da napadač sazna povjerljive podatke korisnika, kao što su korisničko ime, zaporka ili broj kreditne kartice, do toga da onespobavi web platformu brisanjem podataka iz baze. Postoji veliki broj različitih napada i tehnika koje koriste napadači, a kod php aplikacija možemo izdvojiti SQL upad ili SQL injekciju (engl. *SQL injection*).

Izraz SQL injection odnosi se na postupak ubacivanja SQL upita koji se izvršava u bazi podataka baz znanja programera. Ovi se upadi najčešće događaju putem formi preko kojih korisnici unose podatke, npr. korisničko ime i zaporku ili e-mail adrese. Zbog lošeg pripremljenog SQL upita u programskoj skripti moguće je pisanjem dodatnih SQL naredbi promijeniti izgled prvotnog SQL upita i tako postići da se izvrši nešto drugo od onoga što je programer zamislio. Najčešće se ovakvi napadi upotrebljavaju radi prikupljanja autorizacijskih podataka, ali i rušenja stranice brisanjem baze podataka.

Obrazac ovog napada je u načelu jednostavan i sastoji se od velikog broja pokušaja i pogreški te učenju na tim pogreškama. Napadač preko forme za unos podataka na stranici pokušava zapisati razne SQL upite ili samo neke dijelove tih upita. Ako PHP skripta nije dobro napisana i nema dio za provjeru pristiglih podataka, tada se događa da se taj tekst ugrađuje u SQL upit koji se šalje bazi podataka. Napadač prati informacije koje dobiva te prema tome prilagođava naredbe radi dohvaćanja nekih trajnih podataka ili brisanja podataka iz baze.

## 8.2. NAČIN ZAŠTITE

Postoji nekoliko načina zaštite protiv ovakve vrste napada. Trebamo biti svjesni da ne možemo vjerovati niti jednom unesenom podatku koji korisnik dostavlja i da za svaki predani podatak putem forme trebamo napraviti provjeru. Ovo možemo raditi "ručno" izdvajanjem podataka ili prepustiti to mehanizmima PHP-a preko pripremljenih upita.

### 8.2.1. Izdvajanje podataka

Izdvajanje podataka je metoda koja podrazumjeva da predane podatke iz forme izdvojimo u nekakve nove varijable koje prvo provjeravamo nizom ugradbenih PHP funkcija, te ako zadovoljavaju sve provjere, ugrađujemo SQL upite.

Tako imamo primjer dobre i loše prakse, gdje prikupljamo podatke od korisnika koje su uneseni preko web forme :

```
<?php
$email = $_POST['email'];

$query = "SELECT polja FROM tablica WHERE polje = '$email'";

?>
```

Primjer 10. Ranjivost na SQL injekciju



U predhodnom primjeru vidimo jedan jednostavan princip prikupljanja podataka iz web forme. U varijablu \$email spremamo globalnu varijablu \$\_POST sa ključem ['email'] ,tj spremamo podatak koji je unesen od strane korisnika preko web forme.Za primjer,neka to bude email adresa. Zatim u varijabli \$query imamo spremljen upit na bazu gdje kao uvjet SQL upita koristimo podatak koji je unesen od strane korisnika.

Kao što je navedeno na početku poglavlja 8, ovaj pristup nije ispravan jer sa lako ovaj SQL upit može nadograditi drugim,i time otkriti neželjene podatke ili dovesti do brisanja podataka.

Ideja izdvajanja podataka je sljedeća :

```
<?php

$clean['email'] = $_POST['email'];

// niz provjera

$query = "SELECT polja FROM tablica WHERE polje = '$clean[email]'";

?>
```

#### Primjer 11. Izdvajanje podataka

U prvoj se liniji izdvaja vrijednost kojhu je poslao korisnik u novu varijablu ili polje kao u ovom slučaju.Polje je bolje riješenje ako postoji više polja u formi koje korisnik mora popuniti. Nakon toga treba ugraditi programski kod koji radi niz provjera ispravnosti vrijednosti podataka u toj varijabli. Ovo vrijedi za bilo koji SQL upit.

Programski kod koji ostvaruje provjere pišemo pomoću ugradbenih PHP funkcija preko kojih provjeravamo i uvjetujemo kakva vrsta podatka može proć u naš SQL upit :

```
<?php

$clean['email'] = $_POST['email'];

if (ctype_alnum($clean['email'])) {

    //programski kod

}

?>
```

#### Primjer 12. Izdvajanje podataka preko PHP funkcije

Funkcija `ctype_alnum` vraća istinu(engl.*true*) ako se predana varijabla ili polje sastoji samo od slova i brojeva ; inače vraća laž(engl.*false*).Upotrebaljava se za provjeru poslanog podatka, gdje je podatak sastavljen isključivo od brojeva i slova.Ako je provjera *true* program se izvodi,tj prelazi na sljedeći korak,ako je provjera *false* program prekida izvođenje.

Uz ovu navedenu funkciju postoji i još niz drugih koje možemo ubacit u provjeru varijable ili polja.Tako imamo :

### 1. Funkcija `cctype_alpha`

Vraća *true* ako se predana varijabla sastoji samo od slova, inače vraća *false*. Poslani podatak mora biti samo tekst. Ovaj podatak ne smije sadržavati razmake jer inače funkcija vraća *false*. Ova vrsta provjere je idealna kada očekujemo samo jednu riječ iz web forme.

### 2. Funkcija `cctype_digit`

vraća *true* ako se predana varijabla sastoji samo od brojeva;  
inače vraća *false*

### 3. Funkcija `is_numeric`

provjera je li predana varijabla broj, te ako je, vraća *true*; inače vraća *false*

### 4. Funkcija `is_string`

provjera je li predana varijabala niz znakova te ako je, vraća *true*; inače vraća *false*.

### 5. Funkcija `strlen`

vraća broj znakova u predanoj varijabli. Ovo možemo iskoristiti za kontrolu duljine predanog podatka. Ako je duljina podatka u prihvatljivim granicama, koje odredimo, tada ga možemo propustiti do SQL upita; inače ga moramo srezati ili izbaciti.

### 6. Funkcija `trim`

uklanja razmake s početka i kraja niza znakova. Upotrebljava se kada korisnici upisuju podatke u formu zabunom zabunom stave jedan ili više razmaka na kraj upisa. Na ovaj način se uklanjaju kako bi podatak u bazi bio što cjelovitiji

### 8.2.2. Pripremljeni upiti

Pripremljeni upiti jedna su od novosti koju donosi MySQLi klasa. Slovo *i* u nazivu ove klase označava poboljšanje (engl. *improved*), čime se htjelo dati do znanja da je stvoren novi poboljšani, brži i sigurniji način za komunikaciju i manipulaciju bazom podataka. Ova je klasa sastavni dio PHP-a i korištena je kroz ovaj diplomski rad. Dostupna je u svim verzijama PHP-a od 4.1.3 .

Pripremljeni upiti programerima pružaju mogućnost da pišu SQL upite koji su puno sigurniji, prilikom izvođenja imaju bolje performanse te su lakši za pisanje.

Sva funkcionira preko tzv. vezanih parametara. Vezani parametri u pripremljenim upitima omogućuju programeru da stvori predložak SQL upita i zatim ga pohrani na SQL server. Kada se upit treba izvršiti, podaci koji trebaju popuniti već kreirani predložak šalju se SQL serveru. SQL upit formira se na server i zatim izvodi nad bazom podataka.

Tako možemo vidjeti primjer jednog sql upita gdje tražimo imena i prezimena svih pomoraca kojima je status 'onboard', tj. svi pomorci koji su u registru trenutno na moru :

```
"SELECT ime, prezime FROM pomorci WHERE status = 'onboard'"
```

Preko pripremljenog upita prebacujemo upit u SQL predložak i sada zapisujemo :

```
"SELECT ime, prezime FROM pomorci WHERE status = ?"
```

Dakle, na mjesto podatka koji se spremaju u bazu postavljen je znak ? koji predstavlja rezervirano mjesto za podatke koji će poslije doći na to mjesto. Ovaj znak upitnika upotrebljava se na svim mjestima koji sadržavaju podatke. Bilo podatke koje dobivamo od korisnika ili podatke iz naše baze podataka. Ovime postizemo sigurnost podataka, jer prije izvođenja upita definiramo kakve je vrste podatak koji je u predlošku.

Cijeli proces jednog pripremljenog upita:

```
<?php
```

```
$db = new Database();
```

```
$query = "SELECT ime, prezime FROM pomorci WHERE status = ?";
```

```
$status = "onboard";
```

```
if ($stmt = $db->prepare($query)) {
```

```
    $stmt->bind_param('s', $status);
```

```
    $stmt->execute();
```

```
    $stmt->bind_result($safe_q);
```

```
    $stmt->fetch();
```

```
    echo $safe_q;
```

```
$stmt->close();  
  
}  
  
?>
```

### Primjer 13. Pripremljeni SQL upit

U prvoj liniji koda instanciramo objekt `$db` i spajamo se na bazu. Zatim stvaramo predložak SQL upita koji traži sve pomorce po određenom statusu. Sadržaj varijable `$status` će kasnije doći na rezervirano mjesto u predlošku. Zatim se izvršava **prepare** metoda MySQLi klase. Kao parametar predaje joj se varijabla sa SQL predloškom. Rezultat ove metode je novi objekt koji se pohranjuje u varijablu `$stmt`. Ovaj objekt ima vlastite metode. Prvo poziva metodu **bind\_param** preko koje šalje serveru podatke koje treba pohraniti u predložak. Nakon toga se poziva metoda **execute** koja pokreće formiranje SQL upita i njegovo izvršavanje. Metoda **bind\_result** opisuje gdje će se pohraniti vrijednosti koje vraća SQL upit. U ovom primjeru to je varijabla `$safe_q`. Metodom **fetch** se naređuje da se dohvati rezultat izvršavanja SQL upita. Nakon toga, rezultat tog upita možemo i ispisati ali to je opcionalno. Bitno je da znamo da je varijabla ili polje `$safe_q` ispunjena sigurnim podacima. Na kraju upita zatvaramo objekt `$stmt` metodom **close**.

Bitno je istaknuti da kod **bind\_param** metode određujemo koji je tip podatka koji šaljemo u SQL predložak. Slovo određuje tip podatka. Tako je :

`i` - tip podatka je cijeli broj (engl. *integer*)

`d` – tip podatka sa je decimalni broj (engl. *double* ili *float*)

`s` – upotrebljava se za sve ostale tipove podataka

Metodom pripremljenih upita možemo sigurno upravljati podacima kroz čitavu aplikaciju. Ova metoda je do sada najsigurniji način obrane od SQL injekcije. Uz pripremljene upite dobra je praksa i voditi računa o enkripciji zaporki unutar baze podataka.

## **9. DODATNE FUNKCIONALNOSTI WEB APLIKACIJE ZA REGISTAR POMORACA**

Postoji nekoliko dodatnih funkcionalnosti koje nisu implementirane, a vjerojatno bi se u radu aplikacije pokazale vrlo korisnima. Iako trenutna verzija aplikacije funkcionalno obavlja svoj zadatak, omogućava administraciju dokumentacije pomoraca, postoji još par ideja kako bi se funkcionalnost podigla na još veći nivo. Jedna od tih funkcionalnosti je da se omogući agentu prebacivanje kompletne dokumentacije pojedinog pomorca u datoteku spremnu za ispis. Tako bi, preko PHP-a trebalo razviti mogućnost prebacivanja kompletne dokumentacije u primjerice .csv , .doc , .exc format. Tada bi agent pomorske agencije osim pristupa podacima imao i mogućnost jednostavnog ispisa istih.

Sljedeća funkcionalnost koja bi se mogla implementirati je povećanje kontrole podataka administratora. Pri tome se osvrćem na mogućnost proširenja broja sigurnosnih tečajeva za proces unosa, izmjene i brisanja. Naime, u trenutnoj verziji aplikacije imamo mogućnost unosa 8 različitih sigurnosnih tečajeva. Tih sigurnosnih tečajeva je puno više, a i njihov broj raste. Bila bi dobra funkcionalnost kada bi administratoru omogućili dodatne elemente forme gdje bi proizvoljno mogao dodavati još sigurnosnih tečajeva, ako za to postoji potreba.

Zadnja funkcionalnost koja bi, po implementaciji imala i komercijalnu upotrebu, iako ona malo odstupa iz teme ovog diplomskog rada, jest implementacija cijele web aplikacije na jednu web stranicu. Time bi se zaokružio cijeli proces dinamičke naravi aplikacije. Naime, pri implementaciji ove aplikacije na web stranicu morali bi dodat još jednu korisničku rolu osim ove dvije koje trenutno aplikacija koristi. Tako bi uz agenta i administratora imali i same pomorce kao korisnike. Svaki pomorac koji bi posjetio našu web stranicu imao bi mogućnost dobivanja jedinstvenog korisničkog imena i zaporke. Ovaj proces bi se odvio preko web forme gdje bi pomorac unio svoje podatke tipa ime, prezime i email adresu. Nakon poslanog zahtjeva bi na svoj email dobio vlastito korisničko ime i zaporku. Nakon toga bi pomorac imao mogućnost ulaza u sustav, te sam unos svojih podataka. Time



bi se postigle dvije stvari, dokumentacija bi bila primljena direktno preko web stranice i odmah bi se u startu mogla osigurati interakcija između pomorca i agencije.

Ovo su samo neke od funkcionalnosti koje bi se mogle implementirati u samu aplikaciju. Kako vrijeme prolazi, broj mogućih funkcionalnosti bi i dalje rastao. Razvoj web aplikacije nikada nije u potpunosti završen proces, već se funkcionalnosti iste mogu proširivati i smanjivati ovisno o potrebama korisnika.

## 10. ZAKLJUČAK

S obzirom na činjenicu da je razvoj dinamičke web aplikacije (za registar pomoraca) tema za izradu diplomskog rada sa vrlo naglašenim praktičnim dijelom, odnosno samom implementacijom aplikacije, potrebno je u ovom zaključku osvrnuti se na zahtjeve koji su bili postavljeni spram aplikacije te na razinu do koje su ti zahtjevi ispunjeni.

Tijekom izrade programske web aplikacije za registar pomoraca korišteni su besplatni alati preko kojih se osiguralo jedno profesionalno radno okruženje. Iako su ti alati u potpunosti besplatni, omogućuju jednom programeru radne uvjete i okruženje koji mogu zadovoljiti sve standarde jednog modernog radnog procesa. Upravo je to prednost razvoja sa slobodnim kodom i slobodnim, tj. besplatnim alatima.

Kroz ovaj diplomski rad cilj je bio prikazati proces razvoja web aplikacije za registar pomoraca gdje će biti zadovoljeni zahtjevi unapređenja jednog radnog procesa, što je u ovom slučaju funkcionalno dokumentiranje podataka o pomorcima kao i upravljanje tim istim podacima. Tako korisnici kroz aplikaciju mogu pretraživati, unositi, mijenjati i brisati dokumentaciju potrebnu za ukrcaj pomoraca na plovni objekt. Važno je istaknuti da je aplikacija intuitivna. Za korištenje iste nije potrebno nikakvo dodatno školovanje, već je sve vidljivo a time i jasno korištenjem grafičkog sučelja koje je prilagođeno različitim ekranima, od mobitela, tableta do laptopa i računala.

Za poboljšanje i olakšavanje izvedbe ovog radnog procesa korištene su i prikazane različite web tehnologije koje ne zaostaju za današnjim trendovima razvoja različitih web platformi. Koristeći PHP programski jezik, objektno orijentirano programiranje, postižući responzivnost tj prilagodbu aplikacije na razne veličine ekrana, možemo reći da je praktični rad zadovoljio jedan moderan standard razvoja jedne web aplikacije. Važno je istaknuti da je aplikacija u potpunosti funkcionalna i što je jako važno sigurna. Iako je sama aplikacija razvijena u lokalnom okruženju, mi istu možemo sa sigurnošću preseliti i u okruženje web-a. Zbog korištenja sigurnosnih tehnika i tehnologija pri programiranju ova aplikacija bi i sa sigurnosnog aspekta bila na profesionalnoj razini.

Što se tiče funkcionalnosti, već je navedeno da razvoj jedne web aplikacije nikada nije u potpunosti gotov proces. Funkcionalnosti se mijenjaju ovisno o zahtjevima korisnika i novim tehnologijama koje se svakodnevno razvijaju. Ova aplikacija radi kao jedan funkcionalni sustav za upravljanje podacima ali se funkcionalnosti iste itekako mogu proširiti. Neke od funkcionalnosti su i

navedene u predhodnom poglavlju. Iz razloga što je ova aplikacija funkcionalna, sigurna i razvijena korištenjem modernih različitih tehnologija gdje postoji prostor za dodatnu nadogradnju, možemo reći da je diplomski rad na temu razvoja dinamičke web aplikacije uspješno izveden.

## LITERATURA :

1. Quentin Zervaas : Practical Web 2.0 Applications with PHP,Apress 2007.
2. Michael Kofler : The Definitive Guide to MySQL 5,Paperback 2005.
3. Vikram Vaswani : PHP Programming Solutions,Paperback 2007.
4. David Sklar, Adam Trachtenberg : PHP Cookbook,Paperback 2003.
5. Gavin Powell : Beginning Database Design,Paperback 2005.
6. Paul Wilton , John W.Colby : Beginning SQL,Paperback 2005.
7. Andi Gutmans, Stig Sæther Bakken, Derick Rethans :  
  
PHP 5 Power Programming,Paperback 2004.
8. Josh Hill, James A.Brannen : HTML5 and CSS3.
9. ,CET 2011.
10. [http://www.w3schools.com/css/css3\\_borders.asp](http://www.w3schools.com/css/css3_borders.asp)
11. <http://jquery.com/download/>
12. <http://www.htmldog.com/articles/suckerfish/dropdowns/>

13. [http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/php2\\_web\\_apps.html](http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/php2_web_apps.html)
14. <https://netbeans.org/features/index.html>
15. <http://stackoverflow.com/questions/16772076/how-to-make-several-form-fields-required-in-php>
16. <http://php.net/manual/en/>
17. <http://getbootstrap.com/components/>
18. <http://www.wampserver.com/en/>

## **SKRAĆENICE :**

WWW – engl. World Wide Web

HTML – engl. HyperText Markup Language

IP – engl. Internet Protocol

PHP – engl. PHP:Hypertext Preprocessor

VBScript – engl. VisualBasic Script

ASP – engl. Active Server Pages

JSP – engl. Java Server Pages

XML – engl. eXtensible Markup Language

CGI – engl. Common Gateway Interface

IIS – engl. Internet Information Services

WCML – engl. WebComposition Markup Language

MVC – engl. Model/View/Controller

HTTP – engl. HyperText Transfer Protocol

PDF – engl. Portable Document Format

JPEG – engl. Joint Picture Expert Group

GIF – engl. Graphics Interchange Format

MPEG – engl. Moving Pictures Expert Group

URI – engl. Uniform Resource Identifier

SQL – engl. Structured Query Language

## **POPIS PRILOGA :**

Dodatak 1. datoteke za ulaz u sustav

Dodatak 2A. index.php

Dodatak 2B. Pretraga unutar index.php

Dodatak 3A. report.php

Dodatak 3B. report.php

Dodatak 4. admin.php

Dodatak 5. pretraga unutar admin.php

Dodatak 6A. admin\_insert.php

Dodatak 6B. admin\_insert2.php

Dodatak 6. uspješan unos novog subjekta

Dodatak 7A. admin\_update.php

Dodatak 7B. admin\_update2.php

Dodatak 7C. uspješana izmjena subjekta

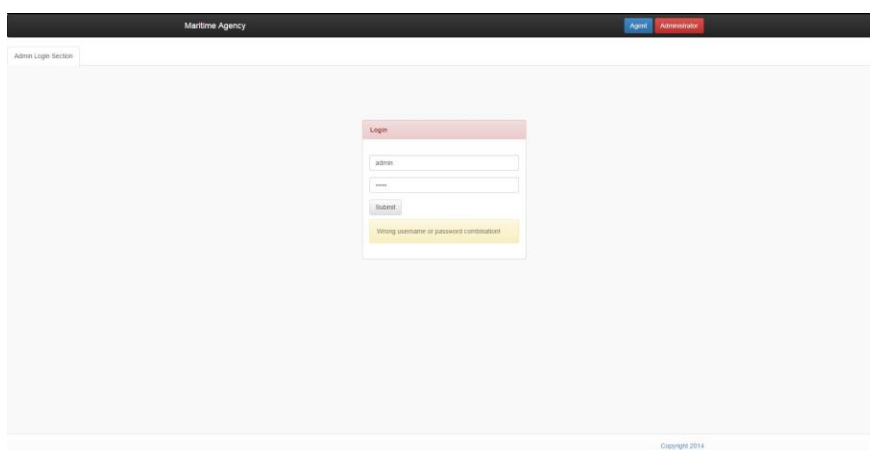
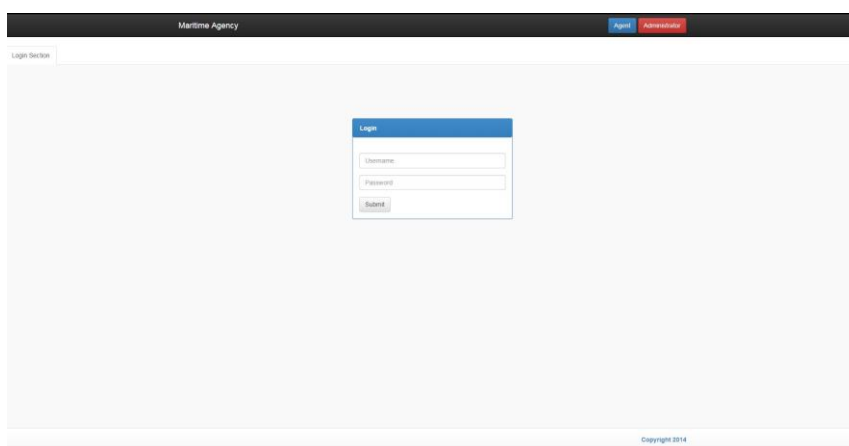
Dodatak 8. brisanje subjekta iz registra

Dodatak 9. responzivnost sučelja

## PRILOG - KORIŠENJE APLIKACIJE :

Preko sljedećih primjera i slika prikazane su i pojašnjene osnovne korisničke datoteke web aplikacije za registar pomoraca :

1. login.php
2. admin\_login.php



Dodatak 1. datoteke za ulaz u sustav



Dodatak 1. prikazuje datoteke za ulaz u sustav web aplikacije. Preko elemenata navigacije dolazi se do web forme pripadajuće korisničke role. Tako imamo odvojenu formu za ulaz agenta i formu za ulaz administratora u sustav. Ako korisnik unese krivo korisničko ime ili zaporku ispisuje se upozorenje i odgovarajuća poruka.

- index.php

Full name	Company	Status
<a href="#">Tomislav Jukopila</a>	CMA	On board
<a href="#">Ivan Pečarić</a>	Atlas Maritime	On board
<a href="#">Vjekoslav Rubeša</a>	Jadrolinija	On board
<a href="#">Nebojša Dovođa</a>	Royal Caribbean	Off-duty
<a href="#">Marko Genda</a>	Royal Caribbean	On-board
<a href="#">Denis Miškulin</a>	China Navigation	On board

## Dodatak 2A. index.php

Nakon uspješnog ulaza agenta u sustav dolazimo na index.php datoteku. Ovdje se nalazi registar pomoraca. Kao što vidimo, unutar jedne tablice imamo popis pomoraca. Izlistano je i istaknuto njihovo ime i prezime kao i pripadna kompanija i status, tj. informacija da li se nalaze na kopnu ili na moru. Ime i prezime je istaknuto zato jer su to poveznice, linkovi koji vode na drugu datoteku gdje se nalaze detaljni podaci o pripadnom pomorcu.

Ispod tablice se nalazi paginacija, korisnički element preko kojeg elegantnije sortiramo naš registar. U slučaju da imamo puno subjekata u registru morali bi se spuštati na dno stranice i tražiti odgovarajućeg pomorca, ovako imamo bolje organiziraniju vizualnu strukturu. Tako svaka stranica paginacije sadrži po 6 subjekata registra, tj 6 pomoraca. Klikom na drugu stranicu izlistava se sljedećih 6 subjekata. Ovaj broj stranica se može i povećavati i smanjivati.

U gornjem se dijelu navigacije nalaze dva elementa. Imamo mogućnost pretrage registra po određenim kriterijima. Tako možemo petraživati subjekte po kriteriju imena, prezimena, kompanije i statusa.

Na desnoj strani navigacije nalazi se dugme za izlaz iz sustava. Interakcija s ovim elementom nas vraća na stranicu za ulaz u sustav.

Na sljedećoj slici je prikazano pretraživanje na osnovi statusa pomorca :

Agency

Main Section

Skaman's register		
Full name	Company	Status
Ivan Magaš	Saipem	On board
Luka Selac	CMA	On board
Matija Zubović	China Navigation	On board
Marin Peharac	Saipem	Off-duty
Robert Mostarac	CMA	On board
Diego Merle	Atlas Maritime	Off-duty

1 2 3 »

Search result: 10

- Ivan Magaš
- Luka Selac
- Matija Zubović
- Robert Mostarac
- Tomislav Jakopita
- Ivan Pežanić
- Vjekoslav Rubeša
- Marko Genda
- Denis Mikutin

Copyright 2014

Nakon što unesemo odgovarajući kriterij u formu navigacije izlistaju nam se rezultati pretrage. Formira se tablica koja sadrži imena i prezimena subjekata koja odgovaraju traženom kriteriju, u ovom slučaju su to pomorci koji su ukrcani na plovni objekt. Izlistani subjekti, tj njihovo ime i prezime, također je istaknuto i predstavlja poveznicu na drugu datoteku. U zaglavlju tablice za pretragu ispisuje se broj subjekata koji odgovaraju zadanom kriteriju.

Kao što je navedeno, imena i prezimena su unutar registra istaknuta zato što su poveznice na drugu datoteku gdje su detaljni podaci pojedinog pomorca. Ta datoteka je :

- [report.php](#)

Maritime Agency Logout

Personal info Certificates/Service

**Personal Details**

First name	Last name	Date of birth	Nationality
Vjekoslav	Rubesa	1988-11-09	Croatian
City	Address	Email	Phone number
Kastav	Aleja Velikana 14	vjekoslav.rubesa@mail.hr	+38551691097
Mobile number	Company	Status	Closest airport
+385955647879	Jadrolinija	On board	Pula

**Travel documents and Medical Examination**

	Country of issue	Place of issue	Number	Date of issue	Date of expiry
Passport	HR, Croatia	Rijeka	33665	2011-04-23	2021-04-23
	Country of issue	Place of issue	Number	Date of issue	Date of expiry
Seaman's Book	HR, Croatia	Rijeka, Croatia	14914454	2014-08-04	2024-08-04
Health Certificate	Date of examination	Date of expiry	Place of issue	Health certificate issued by	
	2013-12-17	2023-12-17	Dom Zdravja Rijeka	Croatia	

[Back](#) Copyright 2014

### Dodatak 3A. report.php

Nakon što napravimo interakciju s bilo kojim subjektom unutar registra, klikom na njegovo ime odlazimo na datoteku report.php. Unutar te datoteke nalaze se svi dostupni podaci o pomorcu. U gornjem lijevom kutu ispod glavne navigacije unutar koje se nalazi element za izlaz iz sustava, imamo

drugu navigaciju sa odjeljcima(engl.tabs) za različite informacije o subjektu.Tako prvi tab(personal info)sadrži osobne podatke pomorca i pripadnu putničku i medicinsku dokumentaciju.

Kao što vidimo na slici,sučelje je podijeljeno na dvije tablice.Prva sadrži osobne podatke pomorca,a druga tablica sadrži podatke o putovnici,knjižici plovidbe i liječničkom uvjerenju.Pored prve tablice nalazi se mjesto za pripadnu slika pomorca.Na podnožju se nalazi donja fiksna navigacija unutar koje je element preko kojeg se vraćamo na registar pomoraca,tj na index.php.

Interakcijom s drugim tabom (Certificates/Service) navigacije odlazimo na dio report.php datoteke koja nam prikazuje podatke o sigurnosnim tečajevima, certifikatu sposobnosti, certifikatu gmdss operatora i prijašnjoj plovidbi pripadnog pomorca.

Maritime Agency Logout

Personal info Certificates/Service

SAFETY CERTIFICATES		
Name of Certificate / Course	Identification Number	Regulation
BASIC TRAINING	UPI-144-35/16-04/50.550-01	STCW(VI/1)
SECURITY AWARENESS	UPI-342-35/11-01/50.530-06	VIS-1
MEDICAL FIRST AID		
FIRE PREVENTION AND FIREFIGHTING		
ADVANCED FIRE-FIGHTING		
PERSONAL SAFETY	I-342-35/11-01/50.530-04/02	STCW(A-VI/1)
CROWD MANAGEMENT TRAINING		
SURVIVAL CRAFT AND RESCUE BOATS		

Certificate of Competence (COC)				
Name of Certificate	Country of Issue	Identification Number	Place of Issue	Date of Issue
3rd Officer	IT,Italy	11455977-23	Ancona	2012-10-05

General Operator Certificate (GOC)				
Name of Certificate	Country of Issue	Identification Number	Place of Issue	Date of Issue
GMDSS	HR,Croatia	007445-36	Maritime Faculty Rijeka	2014-10-15

PREVIOUS SEA SERVICE							
VESSEL'S NAME	FLAG	RANK	TYPE OF VESSEL	COMPANY	GT or KW	FROM DATE	TO DATE
SAIPEM 3000	Bahamas	5th Officer	Crane ship	Saipem	20639 t	2013-02-09	2013-06-11

← Back Copyright 2014

### Dodatak 3B. report.php

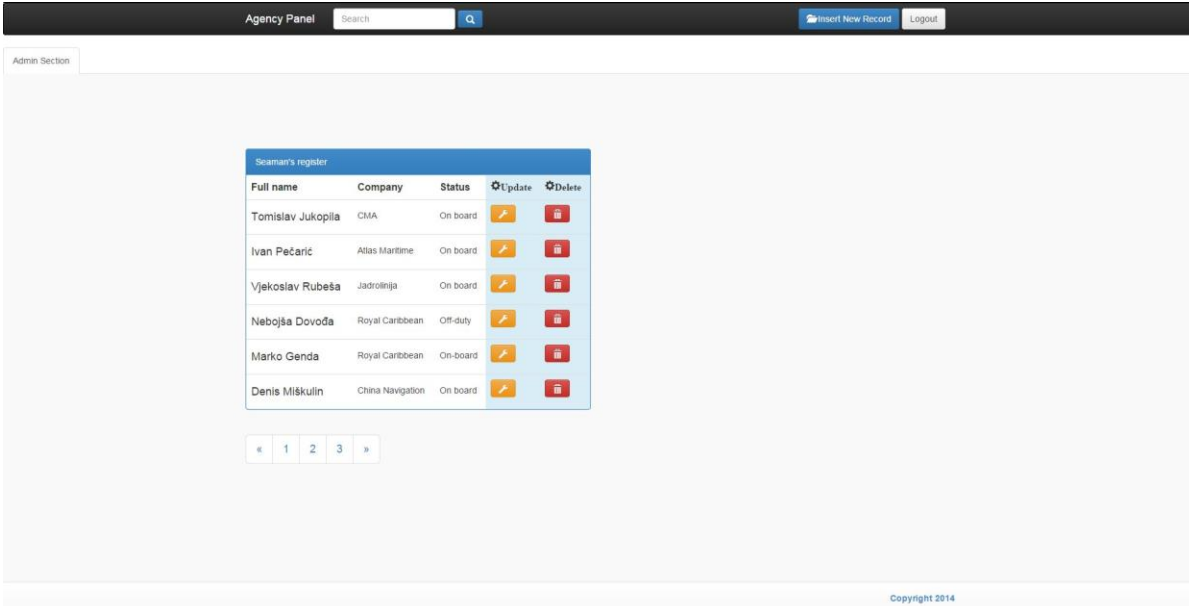
Kao što vidimo na slici, podaci su raspoređeni u četiri tablice. U prvoj tablici na lijevoj strani nalazi se popis sigurnosnih tečajeva s pripadnim identifikacijskim brojevima i regulacijskim standardima kojima pripadaju.Oni identifikacijski brojevi i regulacijski standardi koji su vidljivi

predstavljaju položeni tečaj pomorca. Zatim imamo tablicu u kojoj se nalaze podaci o certifikatu sposobnosti sa pripadnim detaljima, ispod iste, nalazi se tablica s detaljima certifikata gmdss operatora, kao i tablica s detaljima prijašnje plovidbe.

Na ovom tabu također imamo fiksnu navigaciju preko koje se interakcijom izlazi iz sustava, klikom na logotip, kao i klikom na navigaciju u podnožju, vraćamo se na index.php. Klikom na personal info vraćamo se na prijašnji tab.

Kao što je navedeno, dvije su korisničke role kroz rad aplikacije. Tako, administrator može upravljati svim ovim prikazanim podacima. Nakon ulaza administratora u sustav, prva datoteka na koju nailazi je admin.php.

- admin.php



The screenshot shows a web application interface for an agency panel. At the top, there is a navigation bar with 'Agency Panel', a search box, and buttons for 'Insert New Record' and 'Logout'. Below this is an 'Admin Section' header. The main content area displays a table titled 'Seaman's register'. The table has columns for 'Full name', 'Company', 'Status', 'Update', and 'Delete'. The data rows are as follows:

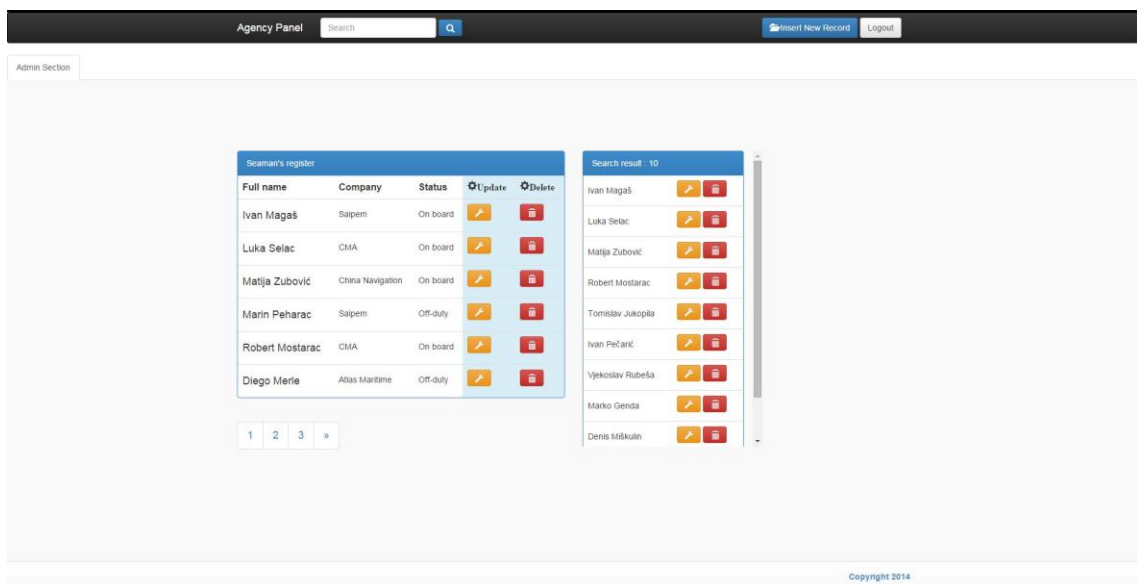
Full name	Company	Status	Update	Delete
Tomislav Jukopila	CMA	On board		
Ivan Pečarić	Atlas Maritime	On board		
Vjekoslav Rubeša	Jadrolinija	On board		
Nebojša Dovođa	Royal Caribbean	Off-duty		
Marko Cenda	Royal Caribbean	On-board		
Denis Miškulin	China Navigation	On board		

Below the table is a pagination control showing '« 1 2 3 »'. At the bottom right of the page, there is a 'Copyright 2014' notice.

Dodatak 4. admin.php

Dodatak 4 prikazuje korisničko sučelje korisnika s administratorskim ovlastima. Kao što je vidljivo na slici, na strani administratora imamo svojevrsnu kopiju registra pomoraca, uz dodatne funkcionalnosti za upravljanje podacima. Tako ovdje, kao i na index.php datoteci imamo mogućnost pretrage pomoraca po određenim kriterijima, kao i paginaciju za organiziraniju vizualnu strukturu. Uz pretraživanje registra administrator ima i mogućnost za unos, izmjenu i brisanje podataka. Na desnoj strani navigacije nalaze se dva elementa, dugme koje nas vodi na datoteku za unos novog pomorca u registar, i pored njega dugme za izlaz iz sustava. Ako pogledamo sam registar vidimo da imamo dva dodatna reda u tablici. Tako pored pripadajućeg imena pomorca, kompanije i statusa, imamo ikone za izmjene i brisanje podataka u registru.

Iste mogućnosti nam se nude i ako napravimo pretraživanje po određenom kriteriju:



Dodatak 5. pretraga unutar admin.php

Kao što je navedeno, interakcijom s elementom navigacije odlazimo na datoteku za unos novog pomorca u registar, tj. bazu podataka. Kada kliknemo na unos novog subjekta dobivamo sljedeću datoteku :

- admin\_insert.php

The screenshot shows a web application interface titled "Agency Panel". At the top right, there are buttons for "Insert New Record" and "Logout". The main content area is titled "Insert Section" and contains four distinct form sections, each with a blue header:

- Insert Personal Details:** Includes fields for first name, last name, date of birth, city, nationality, address, email, phone number, mobile number, company, status (a dropdown menu), and closest airport.
- Insert Health Certificate details:** Includes fields for date of examination, date of expiry, place of issue, and certificate issued by.
- Insert Passport Details:** Includes fields for country of issue, number, place of issue, date of issue, and date of expiry.
- Insert Seaman's Book details:** Includes fields for country of issue, place of issue, number, date of issue, and date of expiry.

At the bottom right of the form area, there is a "Next Step (2)" button with a right-pointing arrow. At the bottom left, there is a "Back" button with a left-pointing arrow. The footer of the page contains the text "Copyright 2014".

#### Dodatak 6A. admin\_insert.php

Unutar admin\_insert.php datoteke nalazi se web forma za unos osobnih podataka pomoraca, kao i forme za unos putne i liječničke dokumentacije. Unutar forme za osobne podatke imamo i padajući izbornik za status, gdje biramo opciju da li je pomorac na brodu ili na kopnu. Svi elementi forme su obvezni za ispuniti osim podaci o pomorskoj knjižici pomorca. U slučaju da korisnik nije ispunio sve podatke, ispisuje se poruka upozorenja. Ovo se odvija kada kliknemo na dugme za sljedeći korak.

Ako su podaci ispravno uneseni odlazimo na drugi korak, odnosno na datoteku za unos certifikata i podataka o prijašnjoj plovidbi.

- [admin\\_insert2.php](#)

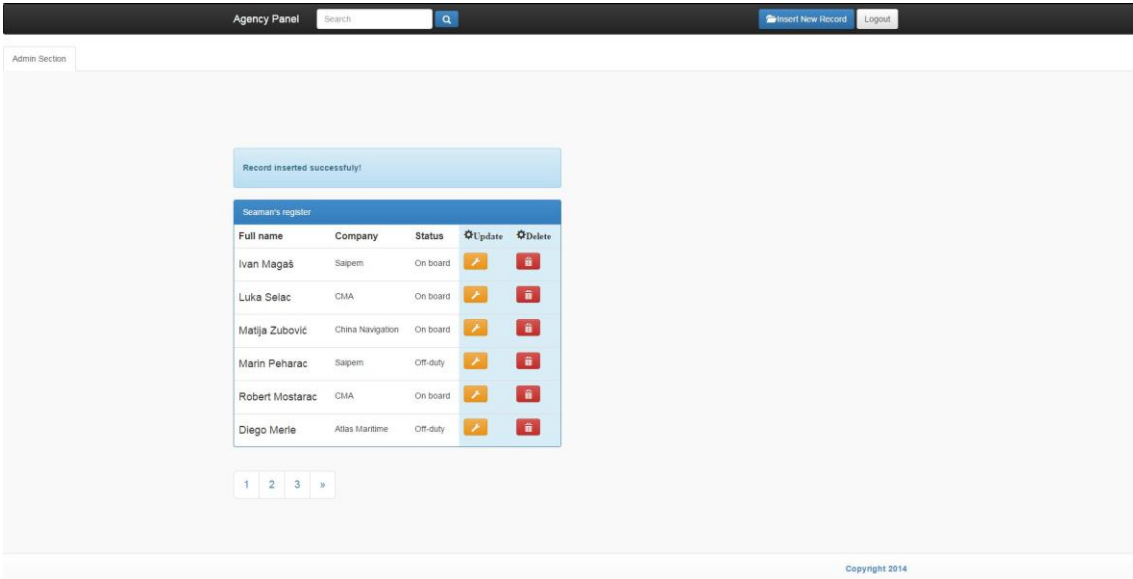
## Dodatak 6B. [admin\\_insert2.php](#)

U ovoj datoteci spremamo sigurnosne certifikate(check-box sa lijeve strane),certifikate o sposobnosti,certifikat gmdss operatora, podatke o prijašnjoj plovidbi, te pripadnu sliku pomorca. Ova je datoteka drugi korak unosa podataka i predstavlja svojevrsnu veliku formu s različitim elementima. Tako odjednom možemo spremit više sigurnosnih tečajja, s time da je tečaj o osnovnoj obuci unaprijed označen, spremamo prijašnju praksu pomorca, imamo elemente s padajućim izbornicima, element za preuzimanje datoteke s našeg računala (preuzimanje slike) i element za povratak na prvi korak koji se nalazi na fiksnoj navigaciji u podnožju stranice.



Sve je isprogramirano tako, da odjednom, u ova dva koraka spremamo sve podatke o pomorcu. Upravljačka logika i upiti na bazu nalaze se u datoteci form\_process.php. Kada kliknemo na dugme za spremanje svih ovih podataka, upravljačka logika se odvija na spomenutoj datoteci te se po izvršenju automatski prebacujemo na početnu datoteku admin.php, gdje se ispisuje poruka o uspješnosti ili neuspješnosti zadanih upita.

Uspješni unos donosi i odgovarajuću poruku :



The screenshot displays a web interface for an Agency Panel. At the top, there is a search bar and buttons for 'Insert New Record' and 'Logout'. Below the search bar, a message box states 'Record inserted successfully!'. The main content area features a table titled 'Seaman's register' with the following data:

Full name	Company	Status	Update	Delete
Ivan Magaš	Saipem	On board		
Luka Selac	CMA	On board		
Matija Zubović	China Navigation	On board		
Marin Peharac	Saipem	Off-duty		
Robert Mostarac	CMA	On board		
Diego Merle	Atlas Maritime	Off-duty		

Below the table, there are pagination controls showing '1 2 3 »'. At the bottom right of the page, the text 'Copyright 2014' is visible.

Dodatak 6C. uspješan unos novog subjekta

Na sličan način funkcionira i izmjena(engl.*update*) podataka.Tako kada se klikne na žutu ikonu za izmjenu podataka odlazi se se na datoteku `admin_update.php`.

- `admin_update.php`

The screenshot shows a web application interface with a dark header bar containing 'Agency Panel', a search bar, and buttons for 'Insert New Record' and 'Logout'. Below the header is a 'Update Section' tab. The main content area contains four update forms:

- Update Personal Details:** A form with fields for Name (Matja), Surname (Zubovic), Date of Birth (1996-12-16), Location (Rijeka), Nationality (Croatian), Address (Mail put 25), Email (zubovic.matja@mail.com), Phone (+385954871254), Company (China Navigation), Status (On board), and Port (Pula).
- Update Health Certificate details:** A form with fields for Issue Date (2013-07-24), Validity Date (2014-07-24), Issuing Institution (Institut zdravlja Zadar), and Country (Croatia).
- Insert Passport Details:** A form with fields for Nationality (Croatia), ID Number (245558), Location (Rijeka), Issue Date (2012-08-16), and Validity Date (2016-02-16).
- Update Seaman's Book details:** A form with fields for Issuing Authority (HR,Croatia), Location (Rijeka,Croatia), ID Number (6966721), Issue Date (2012-05-08), and Validity Date (2022-05-08).

At the bottom of the form area, there is a 'Next Step (2)' button with a right-pointing arrow. At the very bottom of the page, there is a 'Back' button and a 'Copyright 2014' notice.

Dodatak 7A. `admin_update.php`

Kada kliknemo na odgovarajućeg pomorca,kod kojeg želimo mijenjati podatke, opet nam se pokazuje forma sa osobnim podacima, putničkom i liječničkom dokumentacijom, s time što su sad već prikazani i “povučeni” podaci iz baze. Forma je ispunjena. Sada korisnik, tj. administrator može mijenjati podatke. Ako su podaci izmijenjeni, ili ako nema potrebe za promjenama, klikom na dugme odlazimo na drugi korak:

- admin\_update2.php

### Dodatak 7B. admin\_update2.php

Kao što vidimo na slici, nakon što smo otišli na drugi korak prikazani su nam i označeni sigurnosni tečajevi, prikazane su opcije iz padajućih izbornika, kao i ostali elementi. Sada možemo napraviti izmjene, nadopuniti prijašnju plovidbu te zamijeniti i preuzeti novu sliku pomorca.

Upravljačka logika i upiti na bazu nalaze se u datoteci update\_process.php. Kada kliknemo na dugme za spremanje svih ovih podataka, upravljačka logika se odvija na spomenutoj datoteci te se po izvršenju automatski prebacujemo na početnu datoteku admin.php, gdje se ispisuje odgovarajuća poruka:

Agency Panel

Admin Section

Record updated successfully!

Seaman's register				
Full name	Company	Status	Update	Delete
Ivan Mogaš	Saipem	On board	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
Luka Selac	CMA	On board	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
Matija Zubović	China Navigation	On board	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
Marin Peharac	Saipem	Off-duty	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
Robert Mostarac	CMA	On board	<input type="button" value="Update"/>	<input type="button" value="Delete"/>
Diego Merle	Atlas Maritime	Off-duty	<input type="button" value="Update"/>	<input type="button" value="Delete"/>

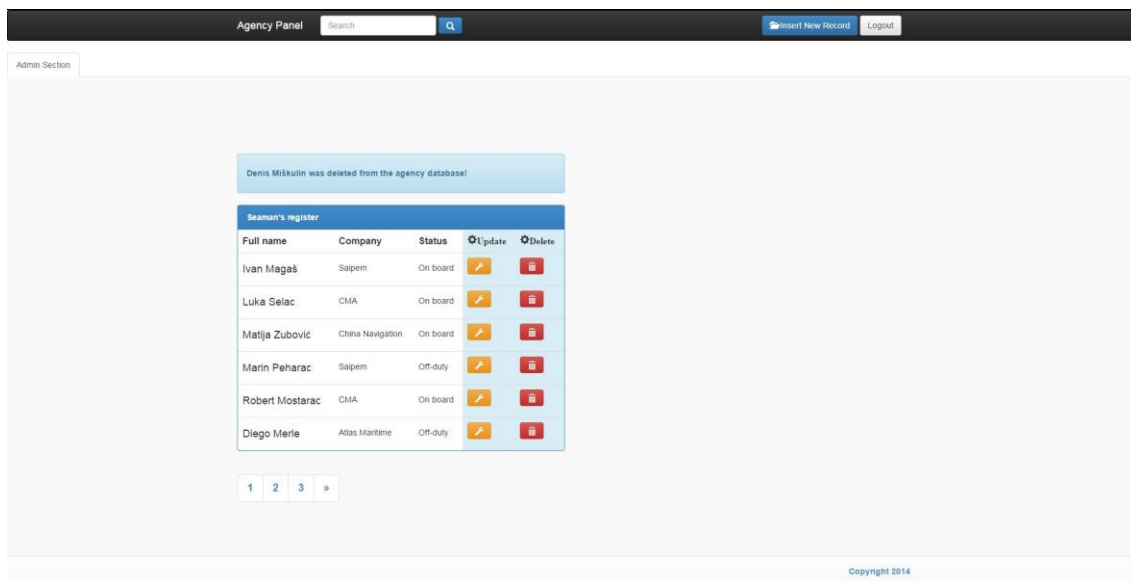
1 2 3 >

Copyright 2014

Dodatak 7C. uspješana izmjena subjekta

Ako želimo ukloniti subjekt iz registra izvršimo interakciju s crvenom ikonom koja nas vodi na datoteku za brisanje podataka.

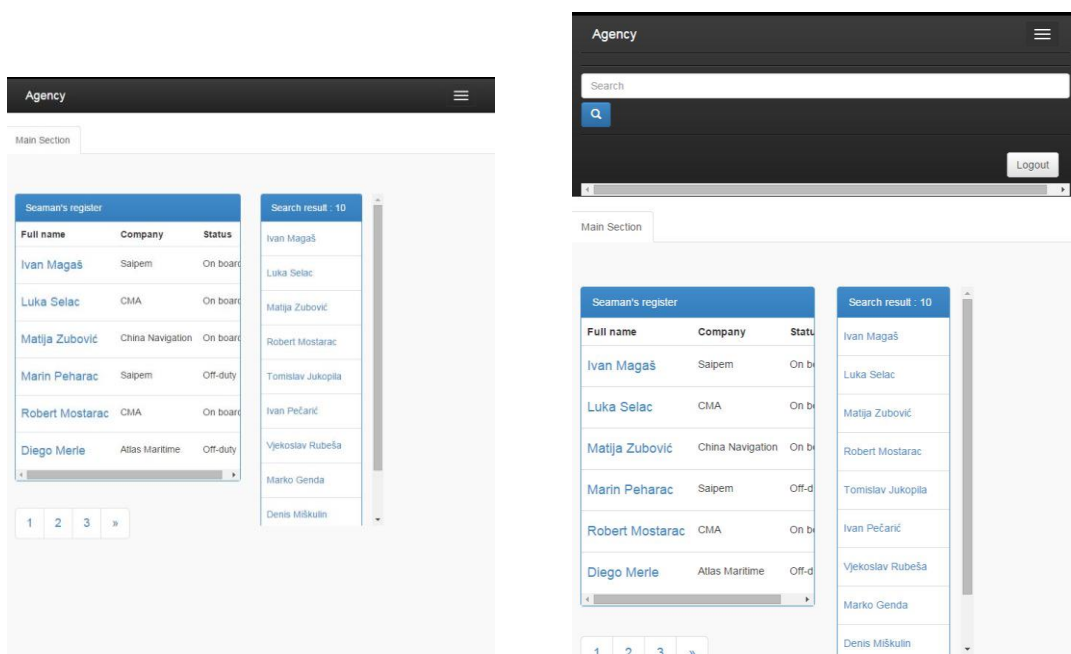
- `admin_delete.php`



#### Dodatak 8. brisanje subjekta iz registra

Prije nego što se subjekt izbriše prikaže nam se prozor gdje smo upitani da potvrdimo brisanje. Ako je odgovor potvrđan, subjekt se briše iz baze i ispisuje se poruka obavjesti.

Uz navedene osnovne korisničke datoteke i funkcionalnosti, treba obratiti pozornost i na responzivnost aplikacije. Cijela je aplikacija prilagođena različitim veličinama ekrana, od ekrana za računala, laptope, do ekrana za tablete i mobitele :



### Dodatak 9. responzivnost sučelja

Dodatak 9 prikazuje kako se sučelje ponaša kada se ekran smanji na veličinu mobitela. Na slici s lijeve strane vidimo da su se tablice automatski prilagodile manjem ekranu. U gornjem desnom kutu je formirana ikona za elemente navigacije.

S desne strane vidimo da se interakcijom sa ikonom otvara prozor gdje su prikazani naši elementi navigacije. Možemo izvršavati pretraživanje, kao i izlazak iz sustava.