Univerzitet u Beogradu

Fakultet organizacionih nauka

Laboratorija za elektronsko poslovanje



Web aplikacija za turističku agenciju

Seminarski rad iz predmeta Internet tehnologije

Studenti:

Teodora Jovanović 102/15

Nemanja Stanojević 120/18

Nikola Dulović 177/17

Beograd, 2022.

Sadržaj

Contents

[2 Korisnički zahtev 3](#_Toc114670363)

[3 Opis Sistema 3](#_Toc114670364)

[3.1 Slučajevi korišćenja 3](#_Toc114670365)

[3.1.1 Prijava na sistem 4](#_Toc114670366)

[3.1.2 Prikaz putovanja 6](#_Toc114670367)

[3.2 Model podataka 7](#_Toc114670368)

[3.3 REST API 8](#_Toc114670369)

[4 Korišćene tehnologije 9](#_Toc114670370)

[5 Korisničko uputstvo 18](#_Toc114670371)

[5.1 Zajednički slučajevi korišćenja 18](#_Toc114670372)

[5.1.1 Prijava na sistem 18](#_Toc114670373)

[5.2 Administrator 19](#_Toc114670374)

[5.2.1 Prikaz putovanja 19](#_Toc114670375)

[5.3 Korisnik(Putnik) 21](#_Toc114670376)

[5.3.1 Pregled putovanja za koje može da napravi rezervaciju 21](#_Toc114670377)

[5.3.2 Rezervacija putovanja 21](#_Toc114670378)

[6 GitHub link ka repozitorijumu 23](#_Toc114670379)

# Korisnički zahtev

Potrebno je kreirati veb aplikaciju turističke agencije. Takođe jedan od zahteva je da postoje dva korisnika I da jedan od njih predstavlja administratora. Administratoru omogućiti pregled putovanja, izmenu i biranje. Pored toga omogućiti administratoru i unos novih aranžmana. Korisniku aplikacije omogućiti prijavljivanje na sistem, prikaz aranžmana i rezervaciju.

Slučajevi korišćenja:

1. Prijava korisnika na sistem

2. Unos putovanja

3. Brisanje putovanja

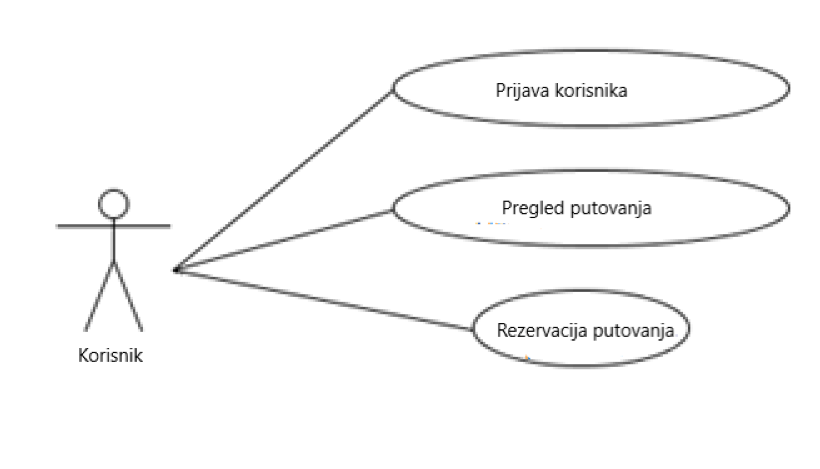
4. Izmena podataka o putovanju

5. Prikaz putovanja

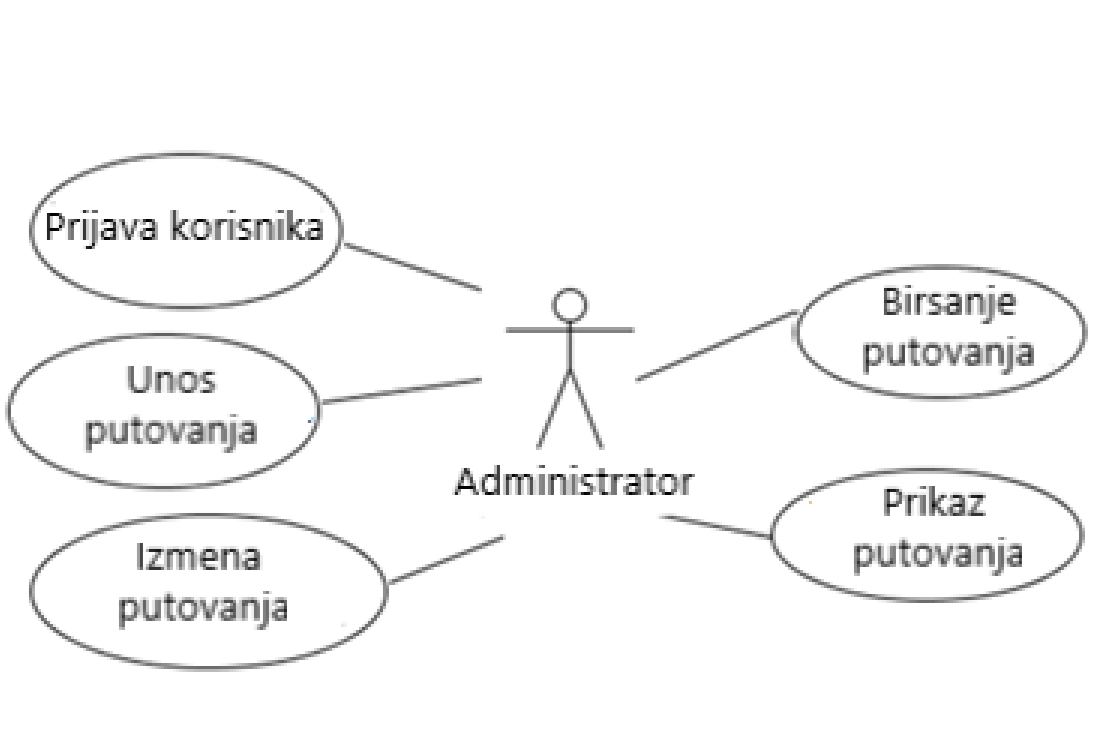
6. Rezervacija putovanja

# Opis Sistema

## Slučajevi korišćenja



Slika : Dijagram slučajeva korišćenja za korisnika



Slika : Dijagram slučajeva korišćenja za administratora

### Prijava na sistem

Naziv SK

Prijavljivanje korisnika

Aktori SK

Korisnik

Učesnici SK

Korisnik i sistem (aplikacija)

Preduslov: Sistem je uključen i otvorena je za forma za logovanje.

Osnovni scenario SK

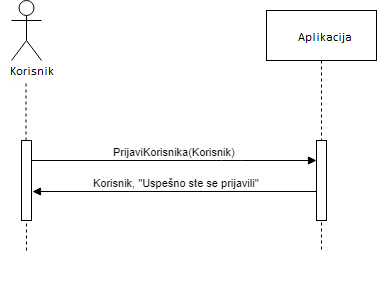
1. Korisnik unosi korisničko ime i lozinku. (APUSO)

2. Korisnik kontroliše da li je korektno uneo korisničko ime i lozinku. (ANSO)

3. Korisnik poziva sistem da se uloguje (proveri podatke). (APSO)

4. Sistem proverava podatke o korisniku. (SO)

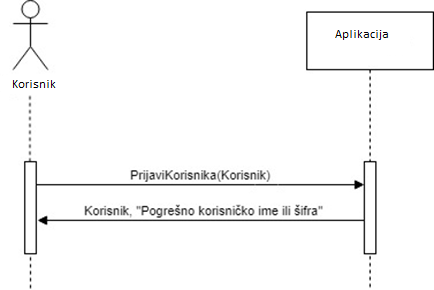
5. Sistem prikazuje korisniku poruku: “Uspešno ste se prijavili “. (IA)



**Slika 3: Dijagram sekvenci slučajeva korišćenja – Prijavljivanje korisnika (osnovni scenario)**

Alternativna scenarija

5.1. Ukoliko sistem ne može da nađe korisnika on prikazuje korisniku poruku: “Pogrešno korisničko ime ili šifra”. (IA)



**Slika 4: Dijagram sekvenci slučajeva korišćenja – Prijavljivanje korisnika (alternativni scenario)**

### Prikaz putovanja

Naziv SK

Prikaz putovanja

Aktori SK

Korisnik

Učesnici SK

Korisnik i sistem (aplikacija)

Preduslov: Sistem je uključen. Učitana je lista putovanja. Otvoren je prozor za pretragu.

Osnovni scenario SK

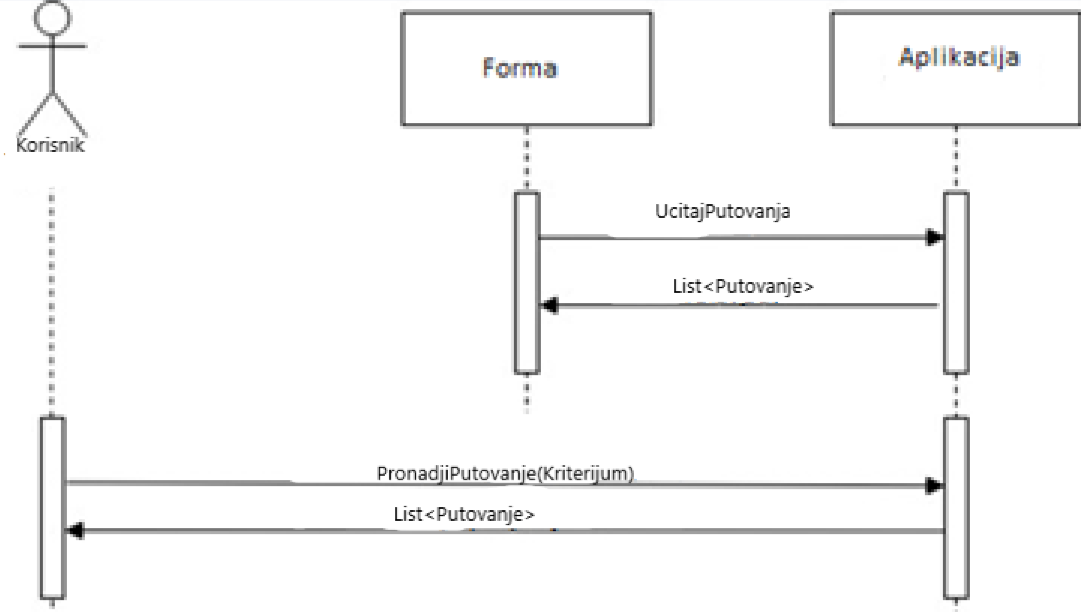
1. Korisnik unosi vrednost po kojoj pretražuje putovanje. (APUSO)

2. Korisnik poziva sistem da nađe putovanjee po zadatoj vrednosti. (APSO)

3. Sistem traži putovanje po zadatoj vrednosti i parametru. (SO)

4. Sistem prikazuje korisniku listu putovanja. (IA)

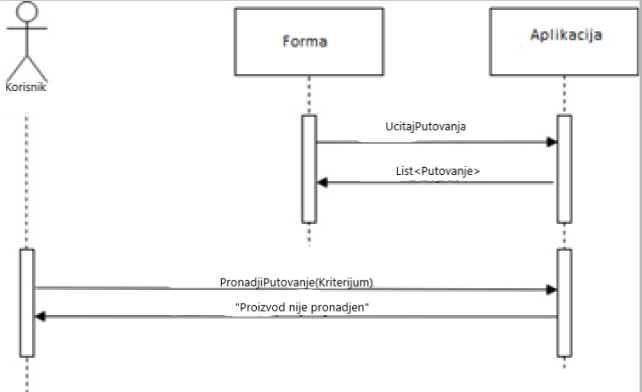
5. Korisnik bira putovanje koga želi da prikaže. (APUSO)



**Slika 5: Dijagram sekvenci slučajeva korišćenja – Prikaz putovanja (osnovni scenario)**

Alternativna scenarija

4.1. Ukoliko sistem ne može da nađe putovanje sistem prikazuje korisniku poruku: “Putovanje nije pronađena!”(IA)



**Slika 6: Dijagram sekvenci slučajeva korišćenja – Prikaz putovanja (alternativni scenario)**

Korišćeni akronimi označavaju sledeće:

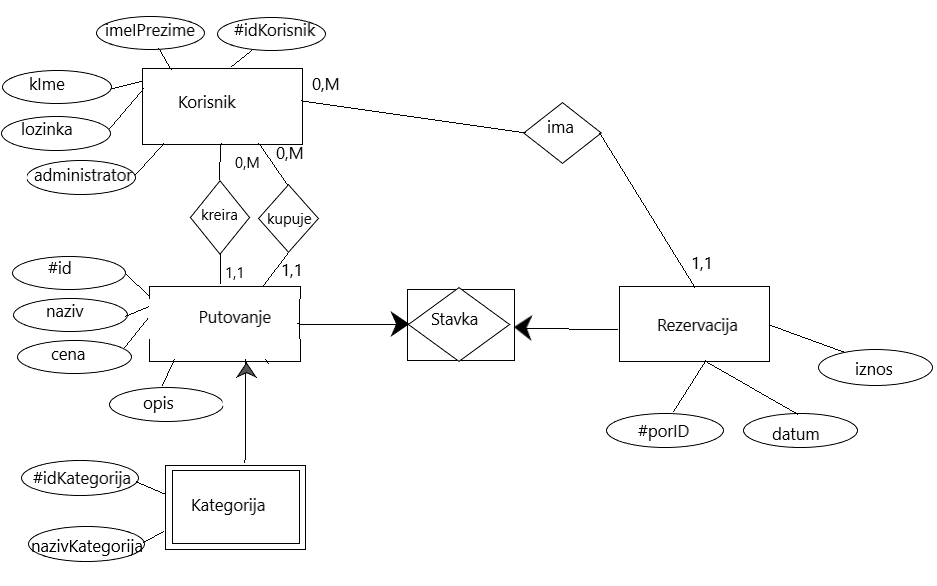
1. APUSO – Akter Priprema Unos za Sistemsku Operaciju,

2. APSO – Akter Poziva Sistemsku Operaciju,

3. SO – sistem izvršava Sistemsku Operaciju, i

4. IA – sistem prikazuje Izlazne Argumente sistemske operacije.

## Model podataka



Slika 7: Model podataka (PMOV)

## REST API

|  |  |
| --- | --- |
| Opis funkcije | Pregled putovanja |
| HTTP metoda | GET |
| URL | (nema) |
| URL parametri | (nema) |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | <putovanje>  <id>1</id>  <naziv>Jahorina</naziv>  </putovanje>  <putovanje>  <id>2</id>  <naziv>Kopaonik</naziv>  </putovanje>  <putovanje>  <id>3</id>  <naziv>Maldivi</naziv>  </putovanje> |
| Format izlaznih parametara | application/xml |

|  |  |
| --- | --- |
| Opis funkcije | Pregled svih kategorija |
| HTTP metoda | GET |
| URL | /kategorije.json |
| URL parametri | (nema) |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | Niz JSON objekata. Svaki element niza ima atribute kategorijaID[int], nazivKategorije[string] |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Pregled svihputovanja |
| HTTP metoda | GET |
| URL | /putovanja.json |
| URL parametri | (nema) |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | Niz JSON objekata. Svaki element niza ima atribute id[int], naziv[string], opis[string], cena[int], kategorijaID[int], nazivKategorije[string] |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Pregled svihporudzbina |
| HTTP metoda | GET |
| URL | /porudzbina.json |
| URL parametri | (nema) |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | Niz JSON objekata. Svaki element niza ima atribute porudzbinaID[int], datumPorudzbine[date], ukupanIznos[int], korisnikID[int], imeIPrezime[string], kIme[string], lozinka[string], administrator[int] |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Pregled odredjenog proizvoda |
| HTTP metoda | GET |
| URL | /proizvodi/@id.json |
| URL parametri | Id[int] – identifikacioni broj proizvoda |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | Niz JSON objekata. Svaki element niza ima atribute porudzbinaID[int], datumPorudzbine[date], ukupanIznos[int], korisnikID[int], imeIPrezime[string], kIme[string], lozinka[string], administrator[int] |
| Format izlaznih parametara | application/json |

# Korišćene tehnologije

U našem radu koristili smo sledeće tehnologije:

1. HTML

2. CSS

3. PHP

4. AJAX

5. JSON

6. JavaScript

7. XML

8. JQuery

4.1. HTML

HyperText Markup Language, je standarizovani jezik koji se koristi pri strukturiranju tekstova, medija i ugrađenih objekata u web stranice i elektronsku poštu. Kao modifikovanu i pojednostavljenu verziju SGML jezika, HTML standarizuje i održava World Wide Web Consortium (W3C). Originalnu verziju HTML-a kreirao je Tim Berners-Lee, a prva zvanična verzija je izašla juna 1993 godine.

HTML jezik je sačinjen od običnog teksta i tagova. Iako se HTML znakovi (tags) često nazivaju kodom, tehnički HTML nije kod jer računarski kod su instrukcije koje od računara traže da izvrši određenu operaciju. Svrha HTML znakova je da se se struktura dokumenta "označi" tako da bi korisnikov agent (user agent), tj. internet pretraživač mogao da prepozna strukturu dokumenta i ispravno je prikaže u prozoru internet pretraževača kojeg korisnik koristi.

Za dodatno modifikovanje web stranica koriste se sledeće tehnologije:

• CSS ili Cascading Style Sheets za izgled i položaj prezentacije

• Skriptni jezici (Javascript, VBScript) za omogućavanje dinamičnosti i interaktivnosti na web prezentacijama,

• DOM ili Document Object Model, koji označava vezu između skripte i elementa na stranici

Gornji dodaci, u paketu sa HTML jezikom, ponekad se nazivaju DHTML ili Dinamični HTML.

Svaki HTML dokument mora početi tagom <html> i završiti se tagom </html>. HTML dokument sadrži uobičajeno dva različita dela, head i body. Head sadrži informacije o dokumentu koje se ne prikazuju na ekranu. Body sadrži sve ostalo što se prikazuje kao deo web stranice.

HTML5 je jezik koji se koristi za strukturiranje i prezentovanje sadržaja za World Wide Web i predstavlja osnovnu tehnologiju Interneta. HTML5 je peta revizija HTML standarda koja se i dalje dopunjuje i razvija. Osnovni ciljevi razvijanja ove poslednje verzije HTML-a su poboljšanje jezika sa podrškom za multimedije koji će biti razumljiv za ljude, ali isto tako i za kompjutere i uređaje (web pretraživače, parsere...). HTML5 je potencijalni kandidat za „cross-platform“ mobilne aplikacije pošto su mnoge osobine HTML5 pravljene tako da budu u mogućnosti da se pokrenu na uređajima male snage kao što su smart telefoni i tableti.

HTML5 je namenjen da prevaziđe ne samo HTML4, nego XHTML1 i HTML DOM. WHATWG je radio na web formama i aplikacijama, dok je W3C radio na XHTML 2.0. 2006. godine su počeli zajedničku saradnju. Tako da se HTML5 može smatrati mešavinom karakteristika i specifikacija HTML-a i XHTML-a. Ovome je doprinela zajednička praksa, kao i mnoge greške u postojećim web dokumentima, ovo je takođe i pokušaj da se definiše jedinstveni markup jezik, koji se moţe pisati i u HTML-u i u XHTML-u. Ovo uključuje detaljne procesne modele da bi ohrabrilo još interoperabilnih implementacija. To proširuje, unapređuje i racionalizuje označavanje dokumenata, kao što je omogućen i API (application programming interfaces), za kompleksne web aplikacije. Trenutno postoje dve paralelne verzije HTML 5, jedna sluţbena na kojoj radi W3C i nesluţbenu na kojoj rade stručnjaci iz Applea, Mozille, Opere i Googlea, koji imaju velik uticaj na razvoj Interneta, iako iza sebe nemaju formalnu organizaciju kao što je W3C. Ipak, urednici HTML 5 specifikacije za oba tima su isti ljudi, što znači da grupe meĎusobno saraĎuju i da ćemo na kraju ipak imati jedan standard. Urednici HTML 5 specifikacije su Ian Hickson iz Googlea i David Hyatt koji radi za Apple.

HTML5 uvodi mnoge nove sintaksne promene. One uključuju nove <video>, <audio> i <canvas> elemente, kao i integraciju SVG (Scalable Vector Graphics) sadržaja (koji zamenjuju korišćenje <object> tagova). Ove karakteristike su dizajnirane tako da lako uključuju i rukuju multimedijalnim i grafičkim sadržajem na web-u bez potrebe uključivanja plugin-ova i API-ja. Drugi novi elementi, kao što su na primer <section>, <article>, <header> i <nav> su kreirani da obogate semantički sadržaj dokumenta. Sa druge strane, neki elementi i atributi su uklonjeni, kao na primer <font> i <center> koji su prevaziđeni korišćenjem mnogo moćnijeg css-a.

4.2. CSS

Iako koristi da se njime opisuju web stranice i korisnički interfejsi napisani u HTML-u i XHTML-u, CSS-om se takođe mogu opisati bilo koje vrste XML dokumenata uključujući i sam XML, SVG ili XUL. Uz korišćenje HTML i Javascript jezika, CSS daje vizuelnu strukturu i opis web sajtova, web aplikacija kao i aplikacija za mobilne telefone.

Tri osnovne karakteristike CSS jezika su mogućnost za definisanje klasa za izgled, boje i fontova. Ovi elementi omogućavaju pristupačniji i fleksibilniji sadržaj kao i kontrolu web dizajnera nad određenom grupom HTML elemenata u sadržaju. Na primer, znamo da HTML ima pojedine tagove kojima mogu da se definišu pojedini elementi, kao što je slučaj sa tagom <bold>. On omogućava podebljanost teksta na stranici. Ukoliko želimo da nam svaki naslov bude podebljan, korišćenjem CSS-a ćemo izbeći konstantno ponavljanje <bold> tagova na svakom mestu gde se nalazi naslov tako što ćemo definisati klasu za izgled <h1> tagova unutar CSS-a i samo pozvati tu klasu prilikom ispisivanja naslova.

CSS3 je najnoviji standard CSS-a. CSS3 specifikacija je i dalje u razvoju od strane W3C organizacije. Većina css3 svojstava je implementirana u modernim web pretraživačima.

CSS3 je podeljen na module. Stara specifikacija je podeljena na manje delove i dodati su novi delovi. Neki od najvažnijih modula su:

• Box model

• Backgrounds i borders

• Tekstualni efekti i web fontovi

• 2D/3D transformacije

• Animacije

• Multiple column layout

• User interface

• Selektori

4.3. PHP

PHP je open source jezik koji se koristi za razvoj server-side aplikacija, kao i dinamičkog Web sadržaja. PHP dozvoljava interakciju sa velikim brojem relacionih baza podataka kao što su MySQL, Oracle, IBM D2, Microsoft SQL Server, PostgreSQL i SQLite. PHP radi na većini operativnih sistema današnjice, kao što su UNIX, Linux, Windows i Mac OS i može da interaguje sa većinom Web servera.

PHP je stekao popularnost zbog svoje jednostavnosti i sintakse nasleđene iz programskog jezika C. Tokom vremena jezik se proširivao i sticao mogućnosti za objektno orijentisano programiranje, naročito od verzije 5.0. Nalikuje jeziku C++ u smislu da dozvoljava i čisto-proceduralno programiranje ali omogućava i korišćenje klasa i drugih koncepata objektno orijentisanog programiranja (nasleđivanje, apstraktne metode, interfejsi itd.).

Neke od osnovnih odrednica PHP – a su:

• PHP je akronim za Hypertext Preprocessor

• server – side skriptni programski jezik

• skripte se izvršavaju na serveru

• podržava rad sa različitim bazama podatama (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, itd.)

• open source softver (OSS)

• radi na različitim platformama (Windows, Linux, Unix i td.)

• kompatibilan je sa skoro svim serverima koji se danas koriste (Apache, IIS, itd. )

• lak je za učenje i efikasno funkcioniše na strani servera.

Za PHP fajlove, bitne su sledeće činjenice:

• mogu da sadrže tekst, HTML tagove i skripte

• vraćaju se klijentskom browser – u kao HTML tekst (plain HTML)

• imaju ekstenziju: “.php”, “.php3” ili “.phtml”

• Program koji se napiše u PHP-u ne zahteva prevođenje (kompajliranje), nego se interpretira pri svakom izvršavanju. PHP interpreter može raditi po CGI principu, odnosno tako što će interpreter postojati kao eksterna aplikacija koja se poziva da izvrši datu skriptu svaki put kad bude zahtevana od nekog korisnika, a može biti instaliran i kao modul veb-servisa. Druga varijanta je danas u najvećoj upotrebi jer pruža znatno veću brzinu izvršavanja - interpreter je na taj način uvek učitan u memoriju te se ne mora pozivati spoljašnji program.

Uobičajen scenario po kom se izvršavaju PHP skripte na web serveru je sledeći:

• klijent (korisnik Interneta koji koristi neki brauzer) zahteva PHP stranicu sa servera

• server prosleđuje zahtev servisu za veb (program veb-server na serveru)

• veb-server prepoznaje da se traži PHP datoteka

• ne šalje njegov sadržaj klijentu, nego ga izvršava kao program pomoću PHP modula

• izlazni tekst programa (standardni izlaz) se šalje klijentu kao rezultat zahteva

• klijent prepoznaje vrstu rezultata (HTML kod, slika, PDF sadržaj, arhiva itd.)

rezultat se prikazuju klijentu na odgovarajući način

4.4. AJAX(Asynchronous JavaScript And XML)

AJAX je skraćenica od “Asynchronous JavaScript and XML”. Termin AJAX prvi put je upotrebljen u februaru 2005. godine, kada je Džesi Džejms Garet (Jesse James Garret), dizajner informacionih sistema i direktor kompanije Adaptive Path, pokušao da nađe odgovarajuću skraćenicu za grupu tehnologija koju je predlagao svom klijentu. Tehnologije potrebne za postojanje AJAX -a su postojale i ranijih godina ali je glavni razlog za slabiju upotrebu svih tehnika korišćenih u AJAX-u, nedostatak podrške od strane web čitača. Mnogi su godinama pre nje koristile slične principe, poput Microsoftovog Remote Scriptinga ili veoma raširenog DHTML-a. Najbitnija stvar koja je uticala na brz razvoj AJAX-a jeste trenutak na tržištu u kojem su veliki igrači Internet industrije želeli da težište korišćenja računara prenesu sa desktopa na web stranice, za šta im je bila potrebna upravo tehnologija.

Web aplikacije imaju brojne prednosti u odnosu na desktop aplikacije. Web aplikacije dosežu do većeg broja ljudi, lakše se razvijaju, implementiraju i održavaju.

Najveći nedostatak Web aplikacija u odnosu na desktop aplikacije je taj što su desktop aplikacije „bogatije” u smislu funkcija i informacija koje nude. Međutim, taj nedostatak je u velikoj meri otklonjen uz pomoć AJAX tehnologije.

Web aplikacije su one aplikacije čijom funkcionalnošću se upravlja preko web servera i dostavljaju se korisnicima preko mreže kao što je internet ili intranet. I pored mnogostrukih prednosti, klasične web aplikacije imaju određene neostatke, kada je u pitanju interakcija sa korisnicima:

• Spor odgovor na zahteve

• Gubitak sadržaja prilikom ponovnog učitavanja stranice

• Gubitak informacija na ekranu

• Gubitak pozicije skrolovanja

• Bez trenutnog odgovora na korisničke akcije

• Korisnik mora da čeka na učitavanje sledeće strane

• Klikni, čekaj, refresh princip rada na stranici

• Stranica se ponovno učitaca sa servera za sve akcije korisnika

• Sinhroni način rada.

Pre svega, svaki put kada se učitava nova stranica, postoji određeno vreme kada aplikacija «stoji». Veoma često se prilikom malih izmena ili zahteva za delovima stranice, ponovo učitava cela stranica, iako je najveći deo nove stranice u potpunosti identičan sa prethodnom.

Da bi se shvatio koncept AJAX tehnologija, neophodno je objasniti razliku izmenu sinhronog i asinhronog prenosa podataka između klijenta i servera.

U sinhronom načinu rada procesi se izvode sekvencijalno. Između dva procesa postoji jaz koji je jednak vremenu izvođenja drugog procesa. Dakle, osnovni proces komunikacije između klijenata i servera može se opisati na sledeći način: klijent šalje zahtev ka serveru, podaci se prenose ka serveru, server obrađuje podatke I zatim ih vraća klijentu. Ono što je primetno sa slike dole je da postoji vremenski interval između pravljenja zahteva i odgovora na isti. Za vreme tog intervala klijent čeka, bez mogućnosti da pravi nove zateve ili nastavi svoje korišćenje web aplikacije. Može se zaključiti da kod sinhronog prenosa podataka ne postoji mogućnost za eliminisanje čekanja zato što se procesi obavljaju jedan za drugim, čak i ukoliko za to nema potrebe.

4.5. JSON

JSON (JavaScript Object Notation) predstavlja tekstualni format za serijalizaciju podataka. Izveden je od formata za zapis objekta skriptnog jezika JavaScript, međutim danas ovaj format koristi većina savremenih programskih jezika. Osnovna svrha JSON-a je prenos podataka i može se reći da predstavlja svojevrsnu alternativu XML-u. JSON je u memorijskom smislu manje zahtevan od XML-a, što znači da JSON fajl sa određenim podacima zauzima manje memorijskog prostora od XML fajla sa istim podacima.

JSON je zasnovan na JavaScript standardu Standard ECMA-262 3rd Edition - December 1999. Iako je JSON usko povezan sa JavaScript programskim jezikom, danas praktično svi programski jezici podržavaju JSON zapis podataka. Među njima su JavaScript, PHP, Java, C, C++, Pyhon, Ruby i drugi.

JSON se zasniva na dve strukture:

• Zbirka parova ime / vrednost. Na raznim jezicima, to je realizovano kao objekat, zapis, struktura, rečnik, heš tabela, lista sa ključevima ili asocijativni niz.

• Uređena lista vrednosti. U većini jezika, ovo je realizovana kao niz, vektor, lista ili sekvenca.

JSON format razlikuje više tipova podataka. Promenljive se u JSON-u nazivaju vrednosti. Od prostih tipova podataka, to su string, number, boolean i null, a od složenih tipova u pitanju su object i array.

4.6. JavaScript

JavaScript je objektno zasnovan skriptni jezik. Uključujemo ga u web stranicu da bi je učinili dinamičnijom. HTML (osnovni kod web stranica) se koristi samo za oblikovanje i uređivanje elemenata stranice (tekst, forme, linkove i tabele), ali nema šanse da diktiramo kako će se ti elementi ponašati. Mogućnost uključenja JavaScript skripte daje nam mnogo veću kontrolu kako se web stranica ponaša. Kombinovan sa HTML-om i CSS-om JavaScript čini DHTML (Dinamic HTML).

Objektno je zasnovan jer programer ne definiše samo tip podataka, nego i vrstu operacija (funkcija) koje se mogu primjeniti na strukture podataka. Na ovaj način, struktura podataka postaje objekat koji uključuje i podatke i funkcije. Pored toga, programeri mogu da kreiraju odnose između jednog i drugog objekta. Na primjer, objekti mogu da pridobiju karakteristike od drugih objekata.

Skriptni je jezik jer se sastoji od serije komandi koje se očitavaju u interpreteru (program prevodioc), a da se predhodno ne kompajlira sadržaj (compiler- program prevodioc). Odnosno ne prevodi se u mašinski jezik (binarni kod- 1 i 0) iz koga nikada nećemo saznati originalni jezik, nego se komande direktno "čitaju" iz koda (source code ili bytecode). Zbog ove karakteristike JavaScript se izvršava na strani korisnika (client side), tj. na računaru na kojem je pokrenut sadržaj sa JavaScript-om. Sam po sebi, HTML dozvoljava posjetocu da pošalje podatke ka serveru na obradu. Nažalost ako ti podaci nisu validni cijeli proces se mora ponoviti sve dok se ne unesu validni podaci. Ovo je jedan od osnovnih razloga nastanka JavaScript-a koji provjerava vjerodostojnost podataka na klijentovom pregledniku (browser-u) i tako olakšava posao na web-u.

Rani skript jezici su se često nazivali batch jezici. Ovo su neki skrip jezici : ASP, JSP, PHP, Perl, Tcl, Python itd.

JavaScript je najpopularniji skriptni jezik na Internetu kojeg podržavaju svi poznatiji preglednici (Internet Explorer, Mozilla Firefox, Netscape, Opera). Evo nekoliko stvari koje je svaki prosječan surfer na internetu vidio: padajući meni, neobični pokazivač miša, iskakajući prozor, sat ... Međutim postoje ozbiljnije primjene JavaScript-a kao što su:

Detekcija preglednika -Određuje se tip preglednika koji se koristi pri pregledu vaše web stranice. Zavisno od preglednika, može se prikazati drugačija stranica posebno dizajnirana za taj preglednik.

Kontrola prozora -mogu se kontrolisati dimenzije, meniji, dugmadi, vrijeme prikaza itd.

Web kolačići (cookies) -pohranjeni podataci na korisnikovom računaru, koji se automatski preuzimaju sljedeci put kada korisnik posjeti vašu web stranicu

Provjera sadržaja -dobar primjer za ovo je provjeravanje da li je korisnik upisao znak @ u polje za upis email adrese.

4.7. JQuery

jQuery je biblioteka čija je svrha da olakša upotrebu JavaScript-a na vašim web stranicama.

jQuery slogan „write less, do more“ dosta govori – sa jednom linijom jQuery koda možete pozvati funkcionalnosti za koje bi vam trebalo mnogo više rada i truda, ako bi koristili klasični JavaScript metod kodiranja.

jQuery obuhvata mnoge zadatke koji zahtevaju veliki broj linija JavaScript-a i omotava ih u metode koje možete pozvati sa samo jednom linijom.

jQuery takođe pojednostavljuje i druge komplikovane stvari u JavaScript-u kao što su AJAX i DOM manipulacija.

JQUERY BIBLIOTEKA SADRŽI SLEDEĆE FUNKCIONALNOSTI:

• HTML/DOM manipulacija

• CSS manipulacija

• HTML event metodi

• Efekti i animacije

• AJAX

Dodatno: jQuery poseduje plugin-e sa kojima možete postići gotovo svaki zahtev koji zamislite.

Najvažniji razlog tolike poularnosti jQuery biblioteke je ekstezivnost. Veliki broj najvećih svetskih kompanija kao što su Google, Microsoft, IBM ga koriste.

Tri stvari su značajne i izdvojile su jQuery u odnosu na druge konkurente:

• korišćenje CSS selektora

• unobtrusive (nenametljiv) JavaScript

• jednostavnost upotrebe

jQuery se pojavio u vreme kada su se front-end developeri navikavali na kreiranje web stranica sa boljim semantičkim obeležavanjem, a učenje CSS-a je bilo zastupljeno širom web-a. jQuery koristi potpuno iste CSS selektore koje developeri koriste za stilizovanje web stranica da bi dodali ponašanje i nema potrebe za promenom konteksta kada se referencira na iste elemente. „Unobtrusive JavaScript“ je metodologija koja definiše napredan način inplementiranja JavaScript koda na web stranicama. Suština ovog pristupa je u tome da se odvaja funkcionalnost (sloj ponašanja) od strukture/sadržaja i prezentacije.Na ovaj način izbegavaju se standardni problemi JavaScript kodiranja kao što su nekonzistentnost na browser-ima i nedostatak skalabilnosti.

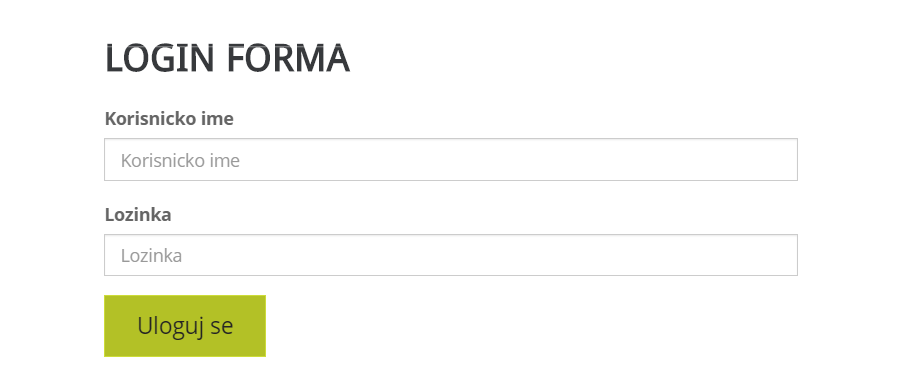
jQuery je napravljen tako da bude krajnje jednostavan za developere koji ga koriste.

# Korisničko uputstvo

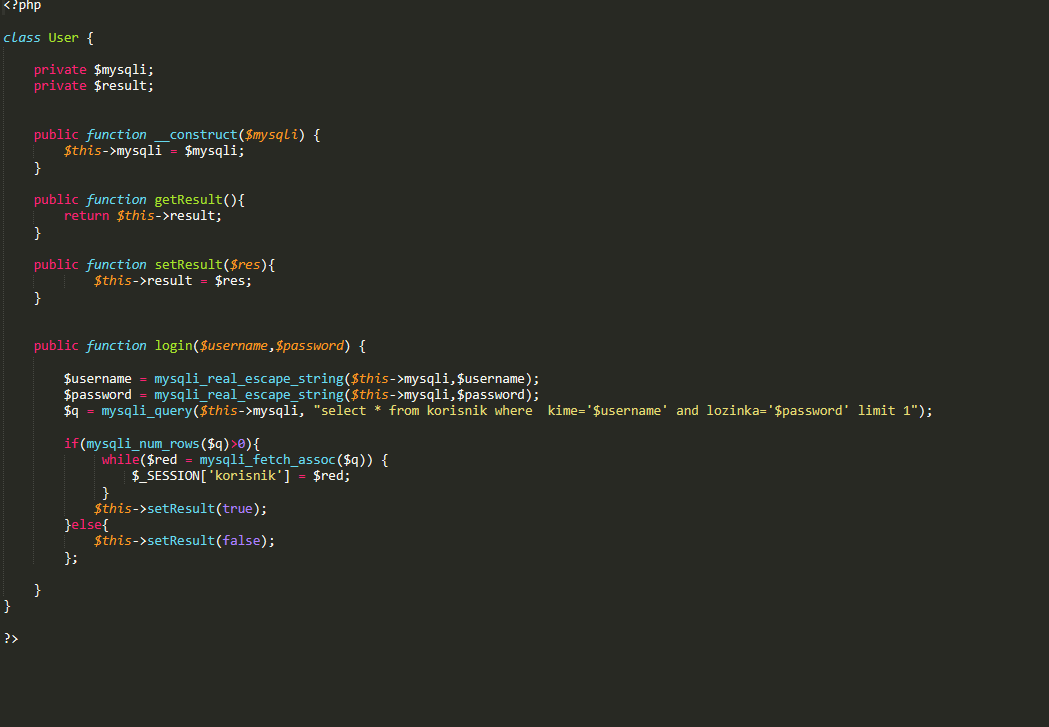
## Zajednički slučajevi korišćenja

### Prijava na sistem

Korisnici pristupaju svom profilu, kada se uloguju pomoću svoje lozinke i korisničkog imena. Kada pristupe svojim profilima imaju mogućnost da vrše izmene u bazi podataka.



**Slika 8:Prijavljivanje korisnika**

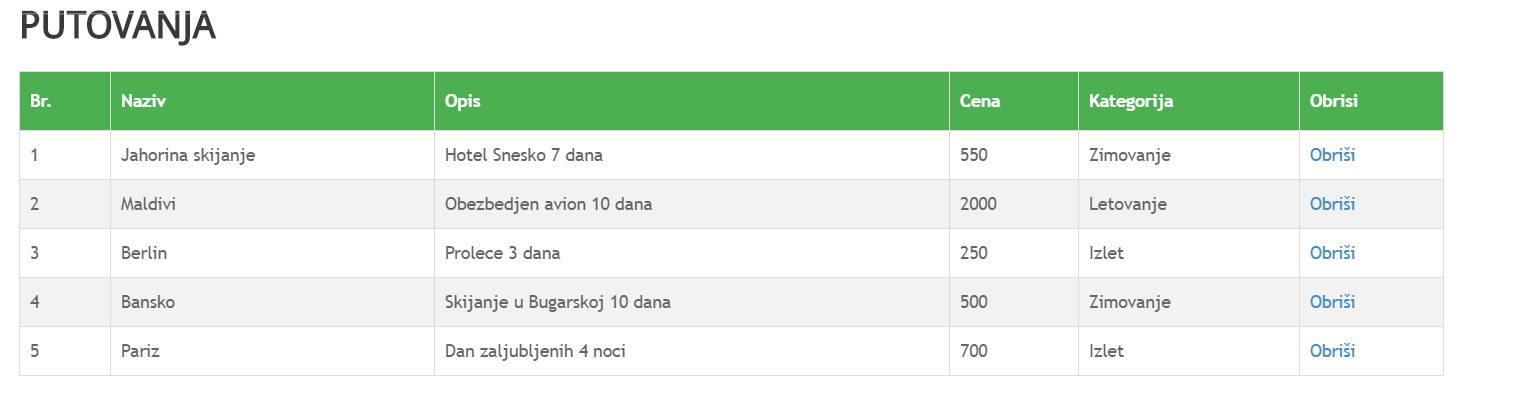


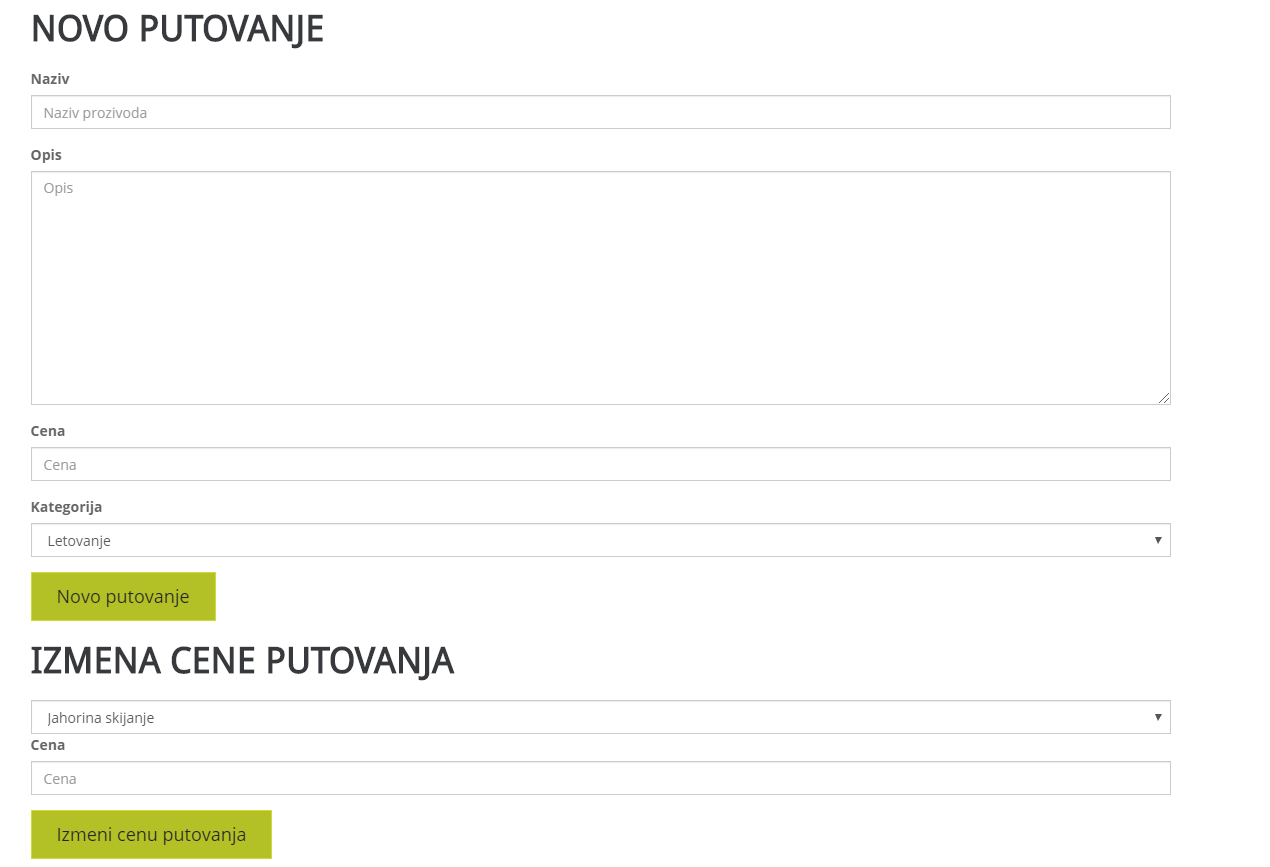
**Slika 9: Kod za prijavu korisnika**

## Administrator

### Prikaz putovanja

Administrator je u mogućnosti da unosi nova putovanja, vrši izmenu podataka o putovanjima, kao i da ih obirše





**Slika 10: Prikaz, unos i izmena putovanja**



**Slika 11: Kod za prikaz, unos i izmenu putovanja**



**Slika 12:Kod za brisanje putovanja**

## Korisnik(Putnik)

### Pregled putovanja za koje može da napravi rezervaciju

Svakom korisnku su prikazana putovanja, koja on može da pretražuje po kateogriji I da ih rezerviše.



**Slika 13:Prikaz putovanja**



**Slika 14:Kod za prikaz putovanja**

### Rezervacija putovanja

Prilikom rezervacije putovanja, korisnik može da unese broj osoba za koje rezerviše putovanje. Rezervacija se tada čuva u bazi.



**Slika 17:Rezervacija putovanja**



**Slika 15: Kod za rezervaciju putovanja**

# GitHub link ka repozitorijumu

https://github.com/nika998/iteh-turisticka-agencija