

Time Zone Converter

Predmet: Klijent server sistemi

Profesor: Milos Kosanovic

Student: Lazar Nikolic Rer 1/19

Sadržaj

1. Uvod.....	3
1.1. Klijentski deo.....	3
1.2. Serverski deo.....	4
2. Struktura.....	4
3. Instalacija i podesavanje.....	4
3.1 Standardno pokretanje.....	5
3.2 Dev pokretanje.....	5
3.3 API.....	5
3.4 CORS.....	5
4. Klijentska funkcionalnost.....	6
4.1 Time zone converter.....	6
4.2 Time Calculator.....	10
5.3 Create Event.....	11
5. Serverska funkcionalnost.....	13
6. Literatura.....	18

1. Uvod

Projekat je izgradjen sa ciljem pruzanja jednostavnog prevodjenja vremena u izmedju dve razlicite lokacije, u footer-u se nalazi meni koji vodi korisnika ka ostalim funkcionalnostima aplikacije kao sto je lako racunanje vremena nakon unesenog broja sekundi, minuta ili sati kao i uzvracanja broja sekundi, minuta i sati izmedju dva razlicita vremena. Takodje aplikacija pruza mogucnost kreiranja eventa koji se moze podeliti sa drugima koji je u obliku pozivnice, naravno ova funkcionalnost radi samo lokalno, kako bi nasa aplikacija bila javna zahteva kupovinu domena za hosting nase aplikacije kako bi nasa aplikacija bila vidljiva na internetu i kako bi mogli javno da podelimo link do stranice naseg eventa.

1.1 Klijentski deo

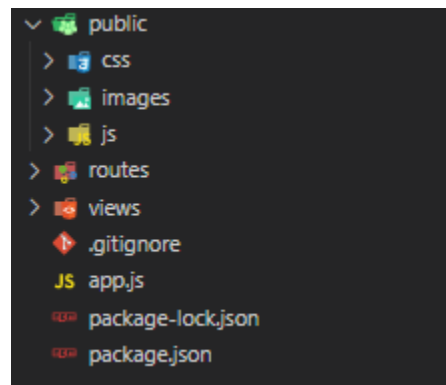
Na frontend-u su koriscene tehnologije HTML, Sass(CSS), JavaScript i jQuery. HTML (HyperText Markup Language) je osnova, sam kostur aplikacije koji definise njen sadrzaj i strukturu, koriscenjem takozvanih tagova i specijalnih elemanta HTML oznacava tekst, slike i sadrzaj za prikaz na web pretrazivacu. Sass je pretprocesorski script jezik koji se interpretira, odnosno kompajlira u CSS jezik, koriscen je zbog svojih prednosti prilikom rada kao sto su mogucnost podele po segmentima kao i zbog lakse i urednije kontrole nad tim segmentima, kao i uvođenja promenljivih uz pomoc kojih lako mozemo promeniti odredjene stilove koji se cesto ponavljaju, poput boje pozadine ili teksta. CSS (Cascading Style Sheets) je jezik koji se koristi stilizovanje, odnosno sminkanje dokumenata na Web stranicama, CSS je najpoznatija i najkoriscenija tehnologija koja se koristi na Webu. JavaScript, na dalje JS, je programski jezik koji predstavlja srz World Wide Web-a (www) zajedno sa HTML-om i CSS-om. Preko 97% websajtova koriste JS na klijentskoj strani kako bi dodali funkcionalnost stranici, cesto se koristi uz neku od gomilu biblioteka na raspolaganju koje konstantno JS zajednica unapredjuje i dodaje. U nasem slucaju uz JS je koriscen jQuery zbog olaksanog koriscenja i manipulacijom nad elementima i sleketorima.

1.2 Serverski deo

Na backend-u je koriscen Nodejs uz API komponentu, detaljnije o API-u bice receno u delu za podesavanje aplikacija, a link kao API ce biti dostavljen u literaturi. Node.js je viseplatformsko radno okruzenje koje služi za izvršavanje JS jezika izvan web pretraživaca.

2. Struktura

Projekat je sagradjen na osnovu nodejs express strukture, hijerarhija se moze videti na slici ispod. U root folderu, se nalaze podfolderi public, routes, uploads i views, kao i .gitignore fajl, i glavni app.js fajl koji predstavlja glavnu logiku serverske strane aplikacije. Unutar views foldera se nalaze sve stranice aplikacije, dok u public folderu se nalaze slike/ ikonine unutar svojih odgovarajucih foldera, css fajl kao i javascript fajlovi.



Express je jedan od poznatijih frameworkova za razvoj web aplikacija zasnovanih na node.js programskom jeziku. Express predstavlja modul za node.js koji sadrži komponente za rutiranje, konfiguraciju, parsiranje zahteva i puno drugih stvari.

3. Instalacija i podesavanje

Unutar glavnog foldera aplikacije nalazi se package.json fajl, on sadrži informacije o projektu, a najvažnije sadrži “zavisnosti”, odnosno imena kao i verzije

paketa(modula) koji su potrebni za rad nase aplikacije. Preko komandnog prozora pokretanjem komande **npm install** kreirace se folder `node_modules` u kome ce instalirati svi potrebni moduli.

3.1 Standardno pokretanje

Komandom `npm start` pokrece se skripta iz `package.json` fajla koja ce pokrenuti server, da bi smo videli rad aplikacije potrebno je otici u bilo kom pretrazivacu na linku `"localhost:8000/"`, `localhost` predstavlja nasu lokalnu adresu na racunaru, dok broj 8000 oznacava port na kojoj aplikacija radi, port je moguće promeniti unutar `app.js` fajla izmenom vrednosti promenljive `"port"`.

3.2 Dev pokretanje

Aplikaciju je takodje moguće pokrenuti komandom `npm run dev`, gde se ce aplikacija pokrenuti uz pomoc `nodemon` modula koji je jako praktican tokom razvoja ili izmene aplikacije jer automatski ponovo pokrece server nakon svake izmene kako bi tu izmenu mogli odmah videti u pretrazivacu.

3.3 API

Kao baza podataka koja nam pruza sve informacije o vremenu kao i listu podrzanih mesta koriscen je API, `TimeZoneDB`. U pitanju je besplatna baza podataka za gradove sveta, sadrzi imena zemalja, vremenskih zona, skracenica i drugih informacija. API informacije su dostupne preko query pristupa, mozemo dobiti podatke o vremenu uz pomoc geografske duzine i visine. Za pristup API-u potrebno je besplatno se registrovati na sajtu `timezonedb.com`, nakon registracije dobitete vas API key (API kljuc) koji je potrebno uneti u `index.js` u config objektu kao vrednost promenljive kljuc `'key'`.

3.4 CORS problem

Moguće je naći problem pri komunikaciji sa API serverom, CORS (Cross-Origin Resource Sharing) greska se javlja kada server ne uzvрати HTTP headere po traženom CORS standardu. Ovo je ugrađeno unutar samog pretrazivaca kao bezbedonosna

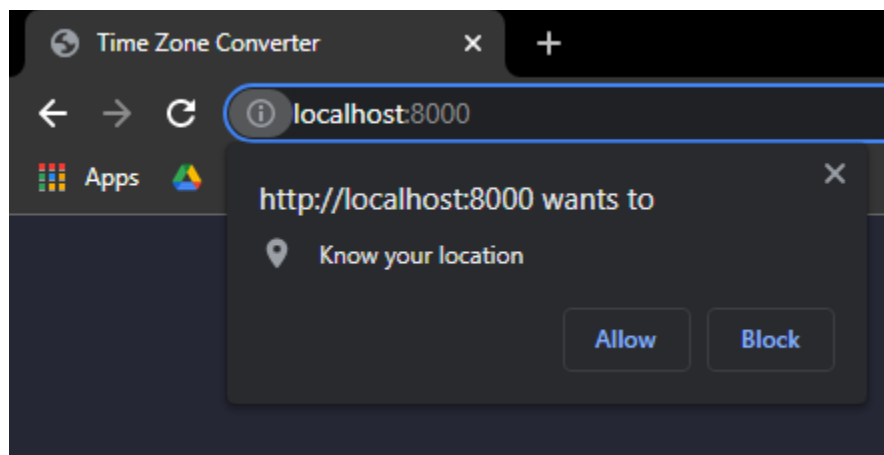
mera. Resenje, kako bi nasa aplikacija funkcionisala za nas loklano potrebno je instalirati CORS unblock ektenziju za pretrazivac, u nasem slucaju je koriscen Chrome pretrazivac i ekstenzija koja je koriscena bice postavljena u vidu linka u lteraturi. Eksentzija se dodaje na pretrazivac i ukljuci. Ukoliko bi smo imali funkcionalan domen i nas server zajedno sa aplikacijom hostovali ne bi bilo potrebe za ovim i aplikacija bi radila normalno.

4. Klijnetska funkcionalnost

Glavna ideja prilikom pocetka izgradnje ove aplikacije je bila mogucnost lakog pretvaranja vremena iz jedne lokacija u drugu, uz to su dodate jos pare korisnih funkcionalnosti, pa cemo ih sve proci redom.

4.1 Time zone converter

Pri prvoj poseti na stranici browser ce od vas traziti dozvolu za citanje vase lokacije.



Klikom na “allow” dozvoljavamo browseru da pristupi informacijama o nasoj lokaciju i onda nastupa function `getCurrentLocation()` koja automatski postavlja naziv mesta gde se nalazimo i trenutno vreme na prvom unosnom polju, lokaciju i vreme naravno mozemo promeniti po zeli.

```
function geoLocation(){
  if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(showPosition);
  } else {
    alert('Geolocation is not supported by this browser.');
```

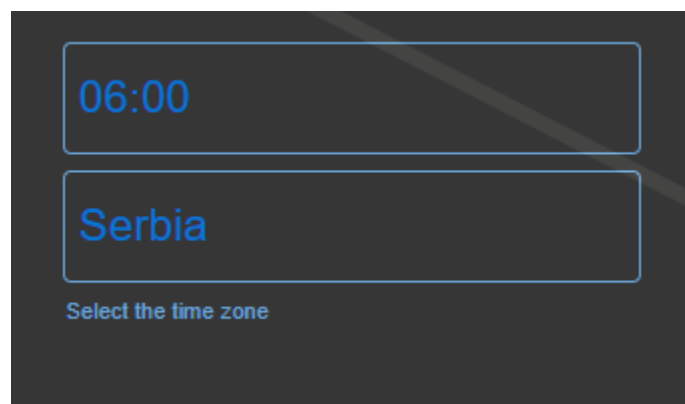
```
    }
    function showPosition(position) {
      const lat = position.coords.latitude;
      const long = position.coords.longitude;
      getCurrentLocation(lat, long)
    }
  }
}

Complexity is 3 Everything is cool!
function getCurrentLocation(lat, long){
  var url = `http://api.timezonedb.com/v2.1/get-time-zone?key=${config.key}&format=json&by=position&lat=${lat}&lng=${long}`;
  var xhr = new XMLHttpRequest();
  xhr.open("GET", url);

  xhr.setRequestHeader("accept", "application/json");

  xhr.onreadystatechange = function () {
    if (xhr.readyState === 4) {
      //console.log(xhr.status);
      //console.log(JSON.parse(xhr.response));
      var data = JSON.parse(xhr.response);
      let m = moment(data.formatted)
      dom.inputTime.value = m.format("HH:mm");
      dom.searchBar_left.value = data.countryName
    }
  };
  xhr.send();
}
```

Kod za funkciju `geoLocation()` je uzet sa w3schools sajta u HTML APIs sekciji, vrlo jednostavno radi i vraća nam objekat sa raznim informacijama o našoj poziciji, u našem slučaju su nam potrebne samo geografska širina i visina koje prosledjujemo funkciji `getCurrentLocation` kao parametre, `getCurrentLocation` funkcija zatim šalje GET zahtev našem API-u i kao odgovor dobija JSON fajl koji parsira, uzima potrebne podatke i direktno ih upisuje u input vrednosti naše aplikacije za konverziju.



06:00

Serbia

Select the time zone

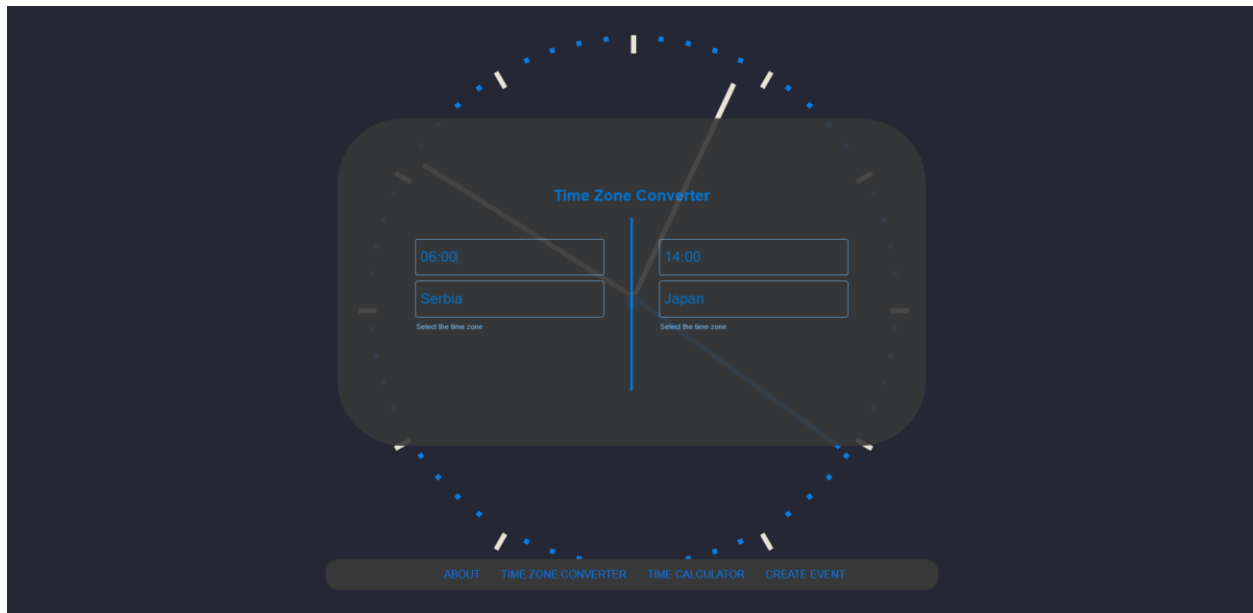


U desnom polju mozemo pretraziti sve podrzane lokacije, i klikom na zeljenu vreme ce se automatski prevesti u polju iznad. Naravno moguće je uneti vreme po zelji za obe lokacije.

Celu logiku pretraga vrši funkcija “pretragaZona()”, uz besplatan API koji je koriscen za izradu ove aplikacije postojala je mogucnost da se skine loklano lista svih podrzanih zona u json formatu, ova mogucnost je iskoriscena najpre radi brzog pristupa listi, ali takodje i radi prikaza rada i komunikacije izmedju nase klijentske i serverske strane. Lista zona se nalazi u routes folderu pod nazivom “list-time-zone.json”. Nazad na funkciju za pretrazgu zona, funkcija salje GET zahtev nasem Nodejs serveru

```
function pretragaZona() {  
    var url = "/AvailableTimeZones";  
    var xhr = new XMLHttpRequest();  
    xhr.open("GET", url);  
    xhr.setRequestHeader("accept", "application/json");  
  
    Complexity is 18 You must be kidding  
  
    xhr.onreadystatechange = function () {  
        if (xhr.readyState === 4) {  
            //console.log(xhr.status);  
            var jsonData = JSON.parse(xhr.responseText).zones;  
            var data = jsonData;  
        }  
    }  
}
```


Ukoliko dobijemo odgovor da je zahtev poslat, da je server završio uzvraćaj poruke i da je pretraživač završio preuzimanje odgovora, lista podržanih zona koju smo dobili se parsira i smesta u promenljivu data. Zatim preuzimaju eventListener-i koji pri svakom unosu u polje za pretragu prikazuju podudarajuće zone, i klikom na zeljenu zonu iz liste prikazanih se primenjuje konverzija. Detaljnije će biti opisan rad serverske strane kasnije.



Oba input polja rade preko eventListener-a, i pri svakom unosu automatski pozivaju funkciju getTimeWhen() koja vrši prevod vremena uz komunikaciju sa API-jem.

```
Complexity is 3 Everything is cool!  
dom.inputTime.addEventListener('input', function(){  
  if(dom.inputTime.value != "" || dom.resultTime.value != ""){  
    dom.resultTime.value = getTimeWhen(dom.searchBar_left.value, dom.searchBar_right.value, dom.inputTime.value)  
  }  
});  
  
Complexity is 3 Everything is cool!  
dom.resultTime.addEventListener('input', function(){  
  if(dom.inputTime.value != "" || dom.resultTime.value != ""){  
    dom.inputTime.value = getTimeWhen(dom.searchBar_left.value, dom.searchBar_right.value, dom.resultTime.value)  
  }  
});
```

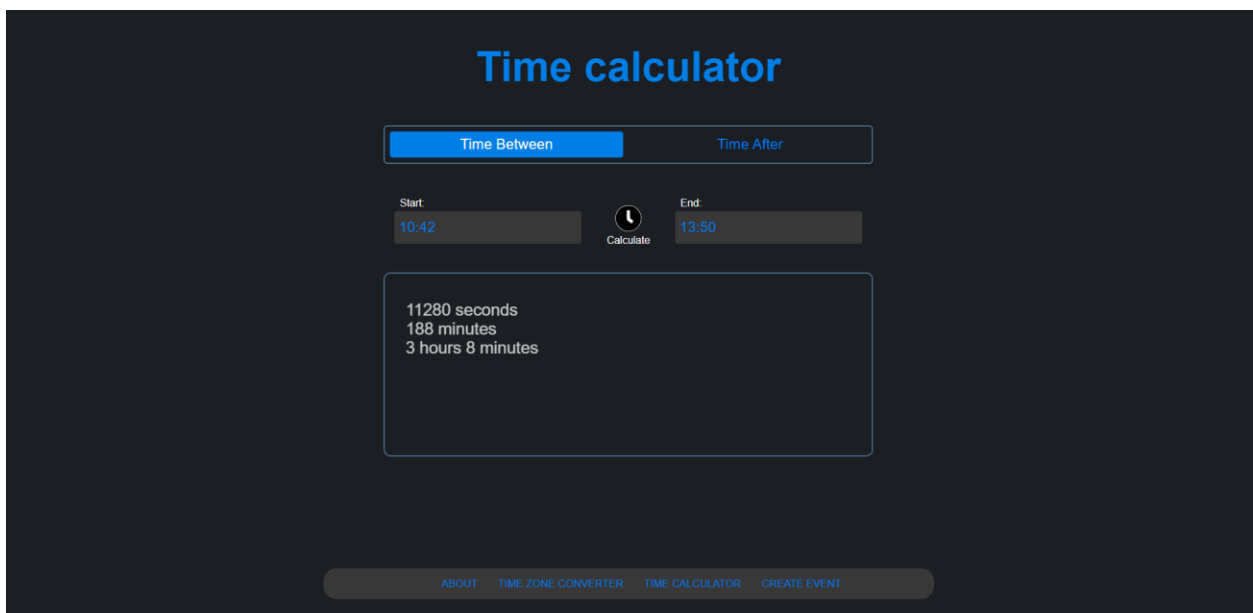
U zavisnosti gde je izmenjena vrednost polja u suprotnom polju se prilagodjava vreme za odabrane lokacije. Dakle eventListener za polja gde se unosi vreme poziva funkciju `getTimeWhen()` koja vraca vreme za zeljenu vremneksu zonu, odnosno lokaciju.

4.2 Time calculator

Sledeca funkcionalnost aplikacije jeste time calculator koji je podeljen na dva dela. Time between i Time After.

Time between

Ideja je da za dva uneta razlicita vremena dobijemo odgovor kolika je razlika izmedju njih u minutima ili satima npr.

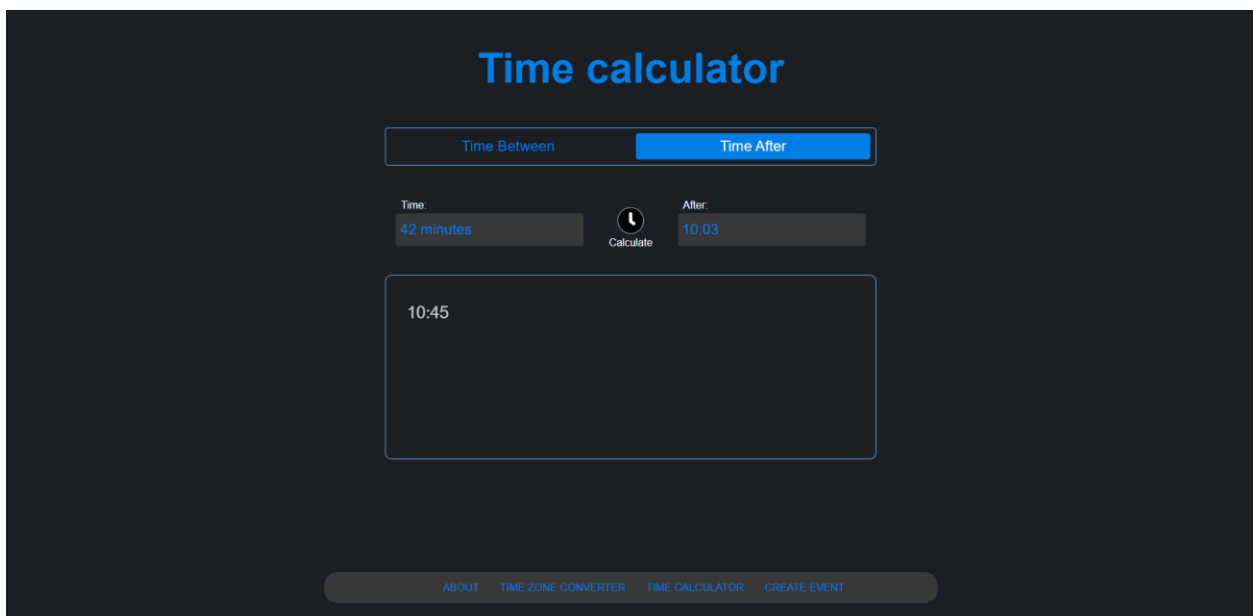


The screenshot shows a web application titled "Time calculator" with a dark theme. It features two tabs: "Time Between" (active) and "Time After". Under "Time Between", there are input fields for "Start" (10:42) and "End" (13:50), a "Calculate" button with a clock icon, and a result box displaying "11280 seconds", "188 minutes", and "3 hours 8 minutes". At the bottom, a navigation bar includes links for "ABOUT", "TIME ZONE CONVERTER", "TIME CALCULATOR", and "CREATE EVENT".

Potrebno je uneti vremena izmedju koja zelimo da izracunamo razliku, na klijentskoj strani se vremena pretvaraju u sekunde radi lakseg slanja, zatim se ti podaci salju nasem serveru i tu se vrshi racunanje, pakovanje podataka i cuvanje u fajlu, zatim se klikom na dugme sa ikonicom sata salje GET zahtev, na klijentskoj strani ti podaci preuzimaju i upisuju na stranici. Detaljnija desavanja ce biti obradjena u segmentu vezane sa serverske rute.

Time After

Na ovoj stranici se racuna koje je vreme nakon odredjenog perioda koji je unesen. Na levom polju za unos mozemo uneti sekunde, minute ili sati, a na desnom polju vreme u odnosu na koje zelimo da izracunamo, klikom na dugme calculate, oba vremena se salju u obliku sekundi ka nasem serveru gde se vrši racunaca i konverzija u formatirano vreme i upisuje u json fajl, klijent zatim salje zahtev za povracaj tog tajla odakle uzima informaciju koja je potrebna i rezultat upisuje na stranici uz pomoc funkcije appendData().



The screenshot shows a web application titled "Time calculator" with a dark theme. At the top, there are two tabs: "Time Between" and "Time After", with "Time After" being the active tab. Below the tabs, there are two input fields. The left field is labeled "Time:" and contains the text "42 minutes". The right field is labeled "After:" and contains the text "10:03". Between these two fields is a circular icon with a clock face and the word "Calculate" below it. Below the input fields is a large rectangular box displaying the result "10:45". At the bottom of the interface, there is a horizontal navigation bar with four links: "ABOUT", "TIME ZONE CONVERTER", "TIME CALCULATOR", and "CREATE EVENT".

4.3 Create Event

Treća i finalna funkcionalnost ove aplikacije jeste mogućnost kreiranja Eventa ili događaja, Korisnik unosi informacije o nekom sastanku ili događaju, vreme početka, vreme kada se završava, datum, opis, link vezan za događaj i slično, takođe može odabrati i boje teksta, pozadine kao i hiperlinka po izboru, klikom na dugme Create podaci se šalju serveru i čuvaju.

The screenshot shows a 'Create Event' form on a dark-themed web application. The form is titled 'Create Event' in blue text. Below the title is a link 'Share with others'. The form fields include: 'Starts at' (text input), 'Ends at' (text input), 'Date' (text input with a calendar icon and placeholder 'mm/dd/yyyy'), 'Title' (text input), 'Location' (text input), 'Event URL' (text input), and 'Description' (text area). At the bottom of the form are 'Custom colors' options for 'Text', 'Link', and 'Background', each with a color picker. A blue 'Create' button is at the bottom right of the form. A navigation bar at the bottom of the page contains links: 'ABOUT', 'TIME ZONE CONVERTER', 'TIME CALCULATOR', and 'CREATE EVENT'.

Odlaskom na link localhost:8000/event-page, podaci se sa klijentske strane zahtevaju od servera, obraduju i prikazuju na stranici u vidu pozivnice.

The screenshot shows an event invitation page. The title is '2. Kolokvijum' in blue text. Below it is the subtitle 'Drugi kolokvijum iz predmeta Klijent server sistemi.' The event details are displayed in a light gray box: 'Start time: 14:00', 'End time: 16:00', 'Date: 2022-01-11', and 'Location: Aleksandra Medvedeva, 20'. A navigation bar at the bottom of the page contains links: 'ABOUT', 'TIME ZONE CONVERTER', 'TIME CALCULATOR', and 'CREATE EVENT'.

5. Serverska funkcionalnost

5.1 Tabela ruta

Metoda	Ruta	Parametri	Opis
get	/	/	Glavna stranica aplikacije
get	/AvailableTimeZones	/	vraca listu svih podrzanih zona u json formatu.
post	/create-event-form	startTime, endTime, date, location, title, eventURL, desc, textColor, linkColor, bgColor	Uzima podatke iz forme poslate sa klijentske strane i cuva ih u json formatu unutar foldera routes.
post	/time-calculate-between	startS, endS	Od klijentske strane dobija podatke o unesenim vremenima izracunava razliku vremena odovjeno po vrednostima u sekundama, minutima I satima I zapisuje ih u json fajl koji se cuva unutar routes foldera.
Post	/time-calculate-after	startS, endS	Racuna koliko ce sati biti nakon odredjeng vremena, formatira ga u citljivom obliku I upisuje u json fajl koji se cuva unutar routes foldera.
Get	/eventData	/	Vraca json fajl sa podacima o eventu
Get	/timeCalcData	/	Vraca json fal sa podaci o izracunatom vremenu.
get	/create-event	/	Vraca stranicu za kreiranje eventa.
get	/evet-page	/	Vraca stranicu kreiranog eventa.
get	/time-calculator-between	/	Vraca stranicu za racunanje izmedju vremena

get	/time-calculator-after	/	Vraca stranicu za racunanje vremena nakon unetog zeljenog perioda.
get	/download	/	Preuzima pdf dokumentaciju o ovom projektu

5.2 Objasnjenje ruta

ruta /create-event-form

```

6. app.post('/create-event-form', function(req, res){
7.     var data = {
8.         "startTime": req.body.startTime,
9.         "endTime": req.body.endTime,
10.        "date": req.body.date,
11.        "location": req.body.location,
12.        "title": req.body.title,
13.        "eventURL": req.body.eventURL,
14.        "desc": req.body.desc,
15.        "textColor": req.body.eventTextColor,
16.        "linkColor": req.body.eventLinkColor,
17.        "bgColor": req.body.eventBgColor,
18.    }
19.    var jsonData = JSON.stringify(data)
20.    fs.writeFile (__dirname + "/routes/events.json", jsonData,
    function(err) {
21.        if (err) throw err;
22.        //console.log('Event created / updated');
23.    }
24.    );
25.    res.redirect("/event-page")
26. });

```

Sa stranice na ruti /create-event prikuplja podatke koji su postali sa klijentske strane i zapisuje ih u data objekat. zatim se promenljiva data uz pomoc JSON.stringify() metode smesta u promenljivu jsonData, I na kraju se kreira JSON podatak pod nazivom "events" u unutar njega cuvaju prikpljeni podaci. Na kraju se korisnik redirektuje na rutu /event-page.

ruta /time-calculate-between

```
app.post('/time-calculate-between', function(req, res){
  var startS = req.body.startS;
  var endS = req.body.endS;
  if(endS < startS){
    endS += 86400
  }
  var rezS = endS - startS;
  var rezM = Math.floor(rezS / 60);
  var rezH = Math.floor(rezM / 60);

  var data = {
    start: req.body.startS,
    end: req.body.endS,
    returnS: rezS,
    returnM: rezM,
    returnH: rezH,
  }
  var jsonData = JSON.stringify(data)
  fs.writeFile (__dirname + "/routes/timeCalcData.json", jsonData,
function(err) {
  if (err) throw err;
  //console.log('time data created / updated');
}
);
res.end();
});
```

Sa klijentske strane se salju podaci o unetim vremenima, podaci se salju u sekundama, server prima zahtev, ukoliko je krajnje vreme vece od pocetkog smatra se da je u pitanju sledeci dan, na serverskoj strani se racuna razlika vrmena u sekundama, minutima i satima posebno i zatim se upisuje u kreiran json fajl unutar routes foldera.

ruta /time-calculate-after

```
app.post('/time-calculate-after', function(req, res){
  var startS = req.body.startS;
  var endS = req.body.endS;

  var returnS = endS + startS;
  var m = moment();
  m.set({hour:0,minute:0,second:0,millisecond:0});
  var returnTime = m.add(returnS, 'seconds').format("HH:mm");

  var data = {
    start: startS,
    end: endS,
    return: returnTime,
  }
  var jsonData = JSON.stringify(data)
  fs.writeFile (__dirname + "/routes/timeCalcData.json", jsonData,
function(err) {
  if (err) throw err;
  //console.log('time data created / updated');
}
);
  res.end();
});
```

Slican postupak, podaci se sa klijentske strane salju serveru u sekundama, server oduzima dve vrednosti, i uz pomoc “moment” modula formatira vreme u citljivom obliku i zatim rezultat upisuje u json fajl unutar routes foldera.

rute /eventData i /timeCalcData

```
app.get("/eventData", function(req, res) {
  var eventInfoRaw = fs.readFileSync(__dirname + "/routes/events.json");
  res.setHeader('Content-Type', 'application/json');
```



```
        res.end(eventInfoRaw);
    });

    app.get("/timeCalcData", function(req, res) {
        var timeCalcData = fs.readFileSync(__dirname + "/routes/timeCalcData.json");
        res.setHeader('Content-Type', 'application/json');
        res.end(timeCalcData);
    });
```

Obe rute pristupaju json odgovarajucim json fajlovima gde se nalaze rezultati prethodnih ruta o izracunavanju vremena.

Ostale rute su klasicne rute za uzvracaj stranica, ili preuzimanje datoteke

```
app.get('/create-event',function(req,res){
    res.sendFile(path.join(__dirname+'/views/createEvent.html'));
});

app.get('/event-page',function(req,res){
    res.sendFile(path.join(__dirname+'/views/event-page.html'));
});

app.get('/time-calculator-between',function(req,res){
    res.sendFile(path.join(__dirname+'/views/time-calculator-between.html'));
});

app.get('/time-calculator-after',function(req,res){
    res.sendFile(path.join(__dirname+'/views/time-calculator-after.html'));
});

app.get("/download", function(req, res){
    res.download(__dirname + "/uploads/izvestaj.pdf")
});
```

6. Literatura

1. **API:** <https://timezonedb.com/api>
2. **CORS:** <https://chrome.google.com/webstore/detail/cors-unblock/lfhmikememgdcahcdlaciloancbhjino?hl=en>
3. https://www.dropbox.com/home/2021_Klijent%20Server%20Sistemi?preview=2020_Praktikum_V1.pdf
4. <https://momentjs.com/docs/#/-project-status/>
5. https://www.w3schools.com/html/html5_geolocation.asp