SVEUČILIŠTE U ZAGREBU
GRAFIČKI FAKULTET

ZAVRŠNI RAD

Kristina Talić

Sveučilište u Zagrebu
Grafički fakultet

Smjer: Tehničko-tehnološki

ZAVRŠNI RAD

IZRADA INTERAKTIVNOG KALENDARA POMOĆU WEB TEHNOLOGIJA

Mentor:
doc. dr. sc. Tajana Koren Ivančević

Student:
Kristina Talić

Zagreb, 2021

SAŽETAK

U ovom završnom radu provodi se izrada interaktivne web stranice sa kalendarom te zanimljivostima vezanim za svaki pojedini datum. Ovisno o datumu, prilikom otvaranja stranice prikazuje se aktualni mjesec te klikom na određeni datum prikazuje se i zanimljivost vezana uz taj konkretan datum. Za izradu ove web stranice koriste se HTML i CSS tehnologije te JavaScript. Kroz HTML i CSS su izgrađene struktura i izgled stranice te animacije i tranzicije, dok se pomoću JavaScript-a povlače zanimljivosti vezane za svaki pojedini dan u godini. Te informacije se nasumično izvlače iz baze podataka i na taj način omogućavaju korisnicima da se svakodnevno upoznaju sa zanimljivostima iz čitavog svijeta.

KLJUČNE RIJEČI: kalendar, web stranica, HTML i CSS tehnologije, JavaScript, baza podataka

ABSTRACT

In this final paper, an interactive website with a calendar and interesting facts related to each date is created. Depending on the date, when you open the page, the current month is displayed, and by clicking on a specific date, an interesting fact related to that specific date is also displayed. HTML and CSS technologies and JavaScript are used to create this website. Through HTML and CSS, the structure and appearance of the page are built, as well as animations and transitions, while using JavaScript, interesting facts related to each day in the year are drawn. This information is randomly extracted from the database and thus allows users to get acquainted with interesting things from all over the world on a daily basis.

KEYWORDS: calendar, web page, HTML and CSS technologies, JavaScript, database

SADRŽAJ

1. UVOD

Web stranica (engl. *web page* ili *webpage*) je web dokument koji je pogodan za *World Wide Web* (informacijski prostor s otvorenim kodom gdje su dokumenti i ostali web resursi identificirani URL-ovima) i internetski preglednik (eng. *web browser*). Internetski preglednik prikazuje web stranicu na monitoru ili mobitelu pa je stoga vrlo važno da dizajner prilikom njezine izrade, pripazi na kompatibilnost stranice između navedenih uređaja.

Web stranica kao HTML dokument omogućava predstavljanje teksta i drugih multimedijskih parametara kao što su zvuk, slika, animacija itd. Izgled navedenih parametara, tipografija, pozadina itd. ostvareni su pomoću CSS instrukcija koje mogu biti integrirane u HTML dokument ili predate preko zasebne datoteke koja je izdvojena iz HTML-a. Također, danas gotovo sve stranice u sebi sadrže JavaScript koji web stranicama osigurava određene funkcionalnosti koje, bez njega, postanu ograničene ili nedostupne.

U današnje vrijeme, korištenje web stranica je svakidašnja te česta pojava kod većine ljudi, a njihova svrha može biti različita no ponajviše, nekolicina ljudi web stranice koristi za naučne svrhe.

1.1. CILJ I ZADATAK ZAVRŠNOG RADA

Cilj ovog završnog rada je napraviti web stranicu koja će, uz pomoć kalendara, svojim korisnicima prikazati i podučiti ih o zanimljivostima koja su se dogodila na određeni datum. Te zanimljivosti obuhvaćaju događaje iz cijelog svijeta te se za jedan određeni datum može prikazati veći broj činjenica tj. zanimljivosti.

2. TEORIJSKI DIO

2.1. HTML

Web stranica je sve ono što se prikazuje na našem monitoru dok je upaljen internetski preglednik, a napisana je HTML kodom (kratica za *HyperText Markup Language,* u prijevodu znači prezentacijski jezik za izradu web stranica) čija je glavna funkcija priskrbiti *hypertext* koji će se filtrirati prema ostalim web stranicama preko određenih linkova. Time nastoji da dokument izgleda isto, bez obzira o kojem se web pregledniku, računalu ili operativnom sustavu radi. [1]

HTML nije programski jezik, već nam on služi za opis hipertekstualnih dokumenata, a ekstenzije tih dokumenata su obično .html ili .htm.

Prva verzija HTML jezika predstavljena je 1993. godine kada je HTML jezik bio podosta ograničen, dok je zadnja, tj. najnovija verzija HTML5 standard (logo vidljiv na slici broj 1) prvi put predstavljena 1999. godine te se nastavila usavršavati sve do 2014. godine.

Slika 1. Službeni logo HTML5 standarda

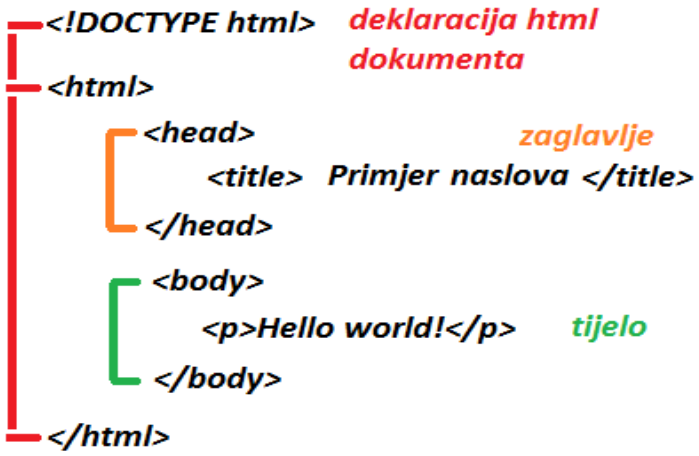
Izvor:
https://hr.wikipedia.org/wiki/HTML#/media/Datoteka:HTML5_logo_and_wordmark.svg

Jedna od najvećih prednosti HTML jezika je ta što je vrlo jednostavan za uporabu te se lako uči, stoga je veoma popularan i opće prihvatljiv.

2.1.1. STRUKTURA HTML DOKUMENTA

Svaki HTML dokument se sastoji od osnovnih HTML elemenata koji su sljedeći: u samom početku pisanja HTML dokumenta, preporuča se postaviti <!DOCTYPE> element koji označava deklaraciju vrste dokumenta (engl. *Document Type Declaration*) kojim se utvrđuje točna varijanta, koristi se za izradu HTML dokumenta. Nakon <!DOCTYPE> elementa dodajemo <html> element koji označava početak HTML dokumenta i u kojem se nalaze <head> i <body> elementi.

<head> element označava zaglavlje ili *header* HTML dokumenta u kojem se većinom definiraju lingvističke značajke HTML dokumenta, naslov web stranice te uz određene HTML elemente dodaju se i koncepcijska obilježja stranice. Pomoću <body> elementa oblikuje se sadržaj HTML dokumenta tj. stranica koja se prikazuje. Cjelokupna struktura HTML dokumenta vidljiva je na slici 2.



Slika 2. Struktura HTML dokumenta

Izvor: http://www.webtech.com.hr/img/html_structure.png

Za svaki od navedenih, standardnih HTML elemenata, postoji otvarajuća i zatvarajuća oznaka, dakle: <head></head>, no postoje i samozatvarajući HTML elementi kod kojih nema zatvarajuće oznake kao npr.

<link rel="stylesheet" href="css/style.css" />.

2.1.2. OSTALI HTML ELEMENTI

Osim već spomenutih, takoreći glavnih elemenata koji čine HTML dokument, postoje i druge vrste elemenata kao što su npr. tekstilni elementi pod koje spada *heading* tj. naslov stranice (moguće je postaviti 6 određenih razina dokumenta koji se određuju naslovima od <h1> gdje je najviše istaknut naslov stranice, sve do <h6> gdje je najmanje istaknut naslov stranice odnosno veličina fonta), paragraf tj. obično napisani tekst (definiramo tagom), link koji definira neku poveznicu (definiramo tagom <a> a nastavkom "href" ukazuje se na lokaciju zadanog linka) itd.

Elementi poput <title>, <link rel>, <style>, <script>, <meta> itd. su takvi elementi koji čine zaglavlje tj. nalaze se unutar <head> elementa i imaju svoje određene funkcije za to mjesto. <title> element ukazuje na naziv dokumenta. [2]

<link rel> element ima definiranu link poveznicu te uz atribut "rel" određuje vezu između tog trenutnog dokumenta i dokumenta na koji je povezan. <style> element se upotrebljava kako bi se definirao stil za unutarnji određeni element tj., *style* oznake unose CSS svojstva što zapravo predstavlja unutarnji CSS. <script> element upotrebljava se za utvrđivanje izvršavanja skripti unutar tog elementa, dakle unutar spomenute oznake se izjašnjavaju funkcije i varijable koje predstavljaju unutarnji JavaScript. <meta> element se odnosi na podatke o podatcima. Uz sebe, oznaka <meta> nosi atribute ″name″ i ″content″ ili ″charset″ koji deklarira znakovni set koji je na našem području UTF-8 jer sadrži hrvatske znakove.

Element kao što je nalazi se unutar <body> elementa te označava neku podjelu ili odjeljak unutar HTML dokumenta. Koristi se kao spremnik za HTML dokument koji se potom stilizira tj. *style*-a u CSS-u pomoću atributa "class" ili "id". Element <button> se također nalazi unutar <body> elementa i označava gumb koji se može kliknuti. Uz oznaku <button> većinom se navodi i atribut "type" koji govori preglednicima koja je vrsta gumba u pitanju. Između ostalog, u <body> elementu se još nalaze i već spomenute oznake za naslove (npr. <h1></h1>), elementi za obično napisani tekst ( ),
 oznaka koja označava prelazak u novi redak itd.

Elementi namijenjeni za multimediju također se nalaze u <body> elementu. Kod najnovije verzije HTML-a dakle HTML5 za video i zvuk koriste se <video> i <audio> oznake kod kojih stoji atribut "src" (eng. *source*) što označava izvor zvuka tj. videa te primjer koda za video i audio elemente je vidljiv na slici 3 i 4

```
<video src="video_datoteka.mpg" width="640" height="480" controls></video>
```

Slika 3. Primjer oznake za <video> element

Izvor: https://hr.wikipedia.org/wiki/HTML

kod kojeg *width* i *height* atributi označavaju širinu i visinu videa. [3]

```
<audio src="zvučna_datoteka.mp3" controls>
```

Slika 4. Primjer oznake za <audio> element

Izvor: https://hr.wikipedia.org/wiki/HTML

Kod dodavanja slika najčešće se koriste formati datoteka JPEG, PNG i GIF jer su one uglavnom najviše kompatibilne sa drugim operacijskim sustavima i računalima. JPEG (eng. *Joint Photographic Experts Group*) slike su slike koje podržavaju 24 bitne boje i prvenstveno se koriste za prikaz kompleksnih tj. realističnih slika fotografske kvalitete. Za razliku od JPEG-a, GIF (eng. *Graphics Interchange Format*) slike su slike koje su manje od ostalih formata, podržavaju manji spektar boja tj. točno 256 boja i pogodne su za više jednostavnije slike. PNG (eng. *Portable Network Graphics*) format slika je najnoviji format koji je implementiran kako bi zamijenio i poboljšao GIF i JPEG format. On podržava 24 i 32 bitne boje i prilično je omiljen među web dizajnerima.

No bez obzira na format, sve slike nose istu oznaku a to je <img> oznaka koja uz sebe također nosi "src" atribut koji se tada doživljava kao kompletna oznaka što je vidljivo na slici 5.

```
<body>
  <img src="link_lokacije_slike.jpg"/>
</body>
```

Slika 5. Primjer oznake za <img> element

Izvor: https://hr.wikipedia.org/wiki/HTML

2.2. CSS

CSS (eng. *Cascading Style Sheets*) služi za strukturiranje HTML dokumenata i definira tri vrste stilskih obrazaca: integrirani, linijski i vezani. Kod integriranog obrasca stilska obilježja se integriraju unutar <style> oznake i to na početku HTML dokumenta. Svaki stil koji je dodijeljen određenoj oznaci, implementira se na sve oznake te varijante unutar dokumenta. Primjer koda za integrirana stilska obilježja vidljiv je na slici 6.

```
<style type="text/css">
  .ime_stila {
    font-size: 36px;
    color: #000;
    background-color: #0F0;
    font-family: Arial, Helvetica, sans-serif;
  }
  .drugi_stil { color: #0F0; }
</style>
```

Slika 6. Primjer integriranog stilskog obilježja

Izvor: https://hr.wikipedia.org/wiki/HTML#Dodavanje_CSS_stilova

Kod linijskog stila stilska obilježja implementiraju se kroz cijeli HTML dokument te svakoj pojedinoj HTML oznaci možemo priključiti vlastita stilska obilježja kao što je vidljivo na slici 7.

```
<p style="font-family: Arial, Helvetica, sans-serif; font-size: 36px; color: #F00;">Označeni tekst stylom</p>
```

Slika 7. Primjer linijskog stilskog obilježja

Izvor: https://hr.wikipedia.org/wiki/HTML#Dodavanje_CSS_stilova

Kod vezanog stilskog obilježja, obilježja su pohranjena u zasebnoj datoteci koja se mogu povezati sa bilo kojim HTML dokumentom koristeći oznaku <link> i koja je smještena unutar <head> elementa. Primjer je vidljiv na slici 8.

```
<link href="http//www.pou.com/Unnamed Site 2/style.css" rel="stylesheet" type="text/css">
```

Slika 8. Primjer vezanog stilskog obilježja

Izvor: https://hr.wikipedia.org/wiki/HTML#Dodavanje_CSS_stilova

Korištenjem CSS obrazaca može se pratiti bilo koji dio segmenta na web stranici, a to može biti podešavanje pozadine tj. mijenjanje boje pozadine, uređivanje teksta, tablica, margina, promjena veličine elementa itd.

Style sheet u CSS-u se sastoji od nekolicine pravila od kojih se svako pravilo zatim sastoji od selektora i deklaracijskog bloka. Pomoću selektora (eng. *selector*) CSS odabire HTML elemente koje želimo stilizirati. Selektori mogu biti jednostavni, npr. svi h1 elementi, elementi određenog "id" (jedinstven element) ili "class" (obuhvaća više od jednog elementa) atributa te kombinacijski selektori tj. elementi u odnosu na druge elemente u DOM-u. Deklaracijski blokovi su uglate zagrade ({}) unutar kojih se nalaze deklaracije i kod kojih se svaka deklaracija zasniva na svojstvu, dvotočki i vrijednosti. Nakon svakog uzastopnog ponavljanja između dvije deklaracije stavlja se točka zarez. [4]

Vrijednosti mogu biti brojčane kao npr. 100 (debljina fonta), 100px tj. 100 piksela, 50% kao 50% širine prozora. Vrijednosti boja mogu biti izražene pomoću heksadecimalne vrijednosti (npr. #FF4500 ili #F10), RGB vrijednosti od 0 do 255 (npr. za plavu boju rgb(0, 0, 255)), RGBA vrijednosti koje uz već navedene vrijednosti obuhvaćaju i alpha prozirnost (npr. rgba(0, 0, 255, 0.3)) ili mogu biti i u HSL sustavu pa tako može glasiti npr. hsl (100%, 000, 30%) tj. hsla (0, 50%, 100%, 30%).

Tekstovi se mogu oblikovati pomoću više svojstava kao npr. *font-family* koji omogućava odabir fonta (npr. *font-family*: Arial;) te kod kojeg se može navesti i *font-style* koji može većinom biti *italic* ili *bold*. Pomoću *text-align* svojstva može se definirati poravnavanje teksta, a vrijednosti mogu biti *left* (lijevo), *right* (desno), *center* (centrirano) i *justify* (obostrano).

Jedno od novijih svojstava je *border-radius* koji omogućava da se rubovi elemenata zaoble. Veličine tih radijusa izražavaju se u pikselima ili postotcima. Četvrtaste div elemente može pretvoriti u zaobljene pravokutnike, elipse i krugove. Ako ne želimo da *padding* i rubovi utječu na traženu već utvrđenu visinu i širinu elementa tada se koristi *box-sizing* svojstvo i vrijednost *border-box* kako bi se određeni element zadržao u zadanim dimenzijama.

Jedno od važnijih stvari za dizajnere je način pozicioniranja elemenata na web stranici kod kojih koordinatni sustav dokumenata uvijek počinje u gornjem lijevom kutu.

Elementi se većinom pozicioniraju na više načina pomoću *position* svojstva. U većini slučajeva koriste se relativne ili apsolutne pozicije. Relativnim (*relative*) pozicioniranjem element se pomiče u odnosu na njegovo normalno stanje pomoću *left, right, bottom* i *top* svojstvima. Taj pomak ne utječe na elemente koji ga okružuju već ostaju u svom izvornom položaju. Apsolutnim (*absolute*) pozicioniranjem element se pomiče u odnosu na element koji ga sadrži, dakle on je u ovom slučaju izvučen iz normalnog pozicioniranja te se elementi koji bi ga okruživali jednostavno ponašaju kao da taj element ne postoji tj. ignoriraju prostor koji bi on trebao zauzeti.

Logo CSS-a, koji je vidljiv na slici 9, označen je plavom bojom te za razliku od HTML-a koji uz sebe nosi broj 5, nosi broj 3 kao zadnju noviju verziju pa je tako u punom nazivu CSS3 standard.

Slika 9. Logo CSS3 standarda

Izvor: https://image.shutterstock.com/image-vector/logo-vector-css-3-low-260nw-1902943426.jpg

2.3. JAVASCRIPT

JavaScript je najpopularniji programski jezik koji se izvršava u korisničkom web pregledniku. [5] Iako je napravljen na način da sliči Javi zbog lakšeg korištenja, temelji se na prototipu i tu završava svaka povezanost sa programskim jezikom Java. JavaScript skupa s AJAX (eng. *Asynchronous JavaScript and XML*) tehnikom olakšava web stranicama komunikaciju sa serverskim programom, a čini web aplikaciju dinamičnijom i jednostavnijom za korištenje. Logo JavaScript-a nosi žutu boju kao što je vidljivo na slici 23.

Slika 23. Logo JavaScript-a

Izvor: https://aws1.discourse-cdn.com/sitepoint/original/3X/e/b/eba3983204f9f9c01dab4c54e33b2b9d9a2731e0.png

U HTML-u, JavaScript je smješten između <script></script> oznaka. Primjer toga je *<script src="scripts/script.js"></script>.* Funkcija koja se događa u JavaScriptu je blok JavaScript koda koji se može izvršiti kada se ta funkcija pozove kao npr. može se pozvati kada se dogodi neki događaj, npr. kada korisnik klikne gumb. [6]

JavaScript se može postaviti u <body> ili <head> odjeljak HTML-a ili u oboje. U <head> elementu funkcija je smještena kao na slici 10:

```
<head>
<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>
</head>
```

Slika 10. Primjer funkcije u <head> elementu

Izvor: https://www.w3schools.com/js/js_whereto.asp

U <body> elementu funkcija je smještena kao na slici 11:

```
<body>

<h1>A Web Page</h1>
<p id="demo">A Paragraph</p>
<button type="button" onclick="myFunction()">Try it</button>

<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>

</body>
```

Slika 11. Primjer funkcije u <body> elementu

Izvor: https://www.w3schools.com/js/js_whereto.asp

Postavljanje skripti na dno <body> elementa poboljšava brzinu prikaza jer tumačenje skripte usporava prikaz. U oba slučaja funkcija se poziva kada se pritisne gumb.

JavaScript skripte se također mogu postaviti u vanjske datoteke kao i CSS. To se može napraviti na način koji je prikazan na slici 12.

```
<script src="myScript.js"></script>
```

Slika 12. Primjer vanjske datoteke JS

Izvor: https://www.w3schools.com/js/js_whereto.asp

Njihove datoteke imaju nastavak .js. One su uglavnom praktične kada se koristi isti kod za više različitih web stranica te olakšavaju čitanje i održavanje HTML-a i JavaScripta.

JavaScript ima više načina prikaza podataka a to su sljedeći: zapisivanje u HTML element pomoću *innerHTML*, zapisivanje u HTML-u izlazu pomoću *document.write()*, zapisivanje u okvir upozorenja pomoću *window.alert()* i zapisivanje u konzolu preglednika pomoću *console.log()*.

Za pristupanje HTML elementu, JavaScript može koristiti *document.getElementById(id)* metodu gdje "id" atribut definira HTML element a *innerHTML* svojstvo definira HTML sadržaj. Primjer koda vidljiv na slici 13.

```
<body>

<h1>My First Web Page</h1>
<p>My First Paragraph</p>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = 5 + 6;
</script>

</body>
```

Slika 13. Primjer innerHTML()

Izvor: https://www.w3schools.com/js/js_output.asp

Ako je potrebno testirati prikaz podataka na stranici, može se koristiti *document.write()* kao što je prikazano na slici 14.

```
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
document.write(5 + 6);
</script>
```

Slika 14. Primjer document.write()

Izvor: https://www.w3schools.com/js/js_output.asp

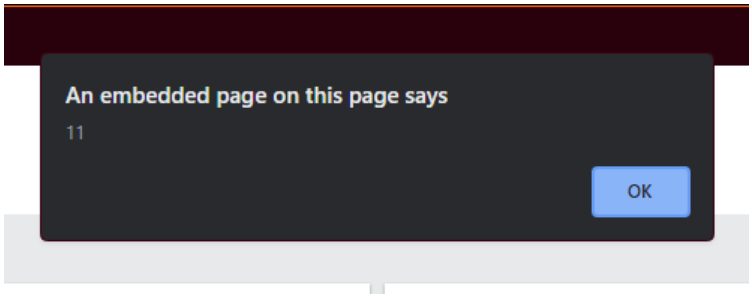
Za prikaz podataka, poruke ili upozorenja može se koristiti *window.alert()* kao što je prikazano na slici 15.

```
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
window.alert(5 + 6);
</script>

</body>
```



Slika 15. Primjer window.alert()

Izvor: https://www.w3schools.com/js/js_output.asp

2.3.1. JAVASCRIPT IZRAZI

JavaScript izrazi većinom počinju sa nekom ključnom riječju za identifikaciju JavaScript radnje koju je potrebno izvesti. Neke ključne riječi koje su korištene u ovom završnom radu:

- *var* – deklarira varijablu
- *const* – varijabla koja je konstantna tj. ne može se mijenjati
- *if* – označava blok naredbi koje trebaju biti izvršene pod nekim uvjetom
- *switch* – označava blok naredbi koje trebaju biti izvršene u različitim slučajevima
- *for* – označava blok naredbi koje treba izvesti u petlji
- *function* – deklarira funkciju
- *return* – izlaz iz funkcije

2.3.2 JAVASCRIPT FUNKCIJE

Funkcija u JavaScriptu je blok koda koji je dizajniran za izvršavanje određenog zadatka. Funkcija se izvršava kada je nešto pozove npr. kada se dogodi neki događaj (korisnik klikne na gumb), kada se pozove iz JavaScript koda ili automatski (samopozivanje). Kad JavaScript dosegne *return* izraz, funkcija će se prestati izvršavati. Funkcije također mogu imati povratne vrijednosti koje se isto tako vraćaju pomoću izraza *return*. Primjer koda vidljiv je na slici 16.

```
let x = myFunction(4, 3);

function myFunction(a, b) {
  return a * b;
}
```

Slika 16. Primjer funkcije i izraza return

Izvor: https://www.w3schools.com/js/js_functions.asp

2.3.3. JAVASCRIPT OBJEKTI

JavaScript varijable su spremnici za vrijednosti podataka. Objekti su također varijable koji mogu sadržavati mnoge vrijednosti. Uobičajeno se piše tj. deklarira objekt sa ključnom riječi "const". Na slici 17 vidljiv je primjer objekta opisan vrijednostima.

```
const car = {type:"Fiat", model:"500", color:"white"};
```

Slika 17. Primjer vrijednosti (Fiat, 500, bijela) i izraza const

Izvor: https://www.w3schools.com/js/js_objects.asp

2.3.4. JAVASCRIPT NIZOVI

JavaScript nizovi (eng. *array*) koriste se za spremanje više vrijednosti u jednu varijablu. Polja se također deklariraju sa ključnom riječi "const". Nizovi mogu sadržavati mnoge vrijednosti pod jednim imenom a vrijednostima se može pristupiti pomoću poziva na broj indeksa dok svaki indeksi niza počinje sa 0 kao što je vidljivo na slici 18.

```
const cars = ["Saab", "Volvo", "BMW"];
let x = cars[0];    // x = "Saab"
```

Slika 18. Element niza koji je pozvan preko broja indeksa

Izvor: https://www.w3schools.com/js/js_arrays.asp

Pomoću svojstva "length" može se dobiti duljina niza (broj elemenata). Vrijednost "length" je uvijek jedan više od najvišeg indeksa polja. Primjer na slici 19.

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.length;   // Returns 4
```

Slika 19. Svojstvo niza

Izvor: https://www.w3schools.com/js/js_arrays.asp

Najbolji način za prolaženje kroz niz je pomoću "for" petlje vidljive na slici 20.

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
let fLen = fruits.length;

text = "<ul>";
for (let i = 0; i < fLen; i++) {
  text += "<li>" + fruits[i] + "</li>";
}
text += "</ul>";
```

Slika 20. Svojstvo niza sa for petljom

Izvor: https://www.w3schools.com/js/js_arrays.asp

Pomoću metode "sort()" može se sortirati niz po abecednom redu koji je vidljiv na slici 21.

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.sort();        // Sorts the elements of fruits
```

Slika 21. Sortirani niz po abecednom redu

Izvor: https://www.w3schools.com/js/js_arrays.asp

2.3.5. JAVASCRIPT DATUMSKI OBJEKTI

Datumski objekti u JavaScriptu omogućavaju rad sa datumima. Prema zadanim postavkama, JavaScript će koristiti vremensku zonu preglednika i prikazivati datum kao niz cijelog teksta. *new Date()* stvara novi date objekt s trenutnim datumom i vremenom npr. *const d = new Date();*. *new Date(year, month, ...)* stvara novi date objekt s navedenim datumom i vremenom. Sedam brojeva navodi godinu, mjesec, dan, sat, minutu, sekundu i milisekundu (tim redoslijedom):

```
const d = new Date(2018, 11, 24, 10, 33, 30, 0);
```

JavaScript broji mjesece od 0 do 11 pa će tako siječanj započinjat sa 0 a prosinac će završavati sa 11.

Metode datuma omogućuju da se dobiju i postave godina, mjesec, dan, sat, minuta, druga i milisekunda datumskih objekata, koristeći lokalno vrijeme ili UTC (univerzalno ili GMT) vrijeme. Kada se prikaže objekt datuma u HTML-u, on se automatski pretvara u niz s "toString()" metodom (primjer na slici 22). On pretvara datum u više čitljiviji oblik.

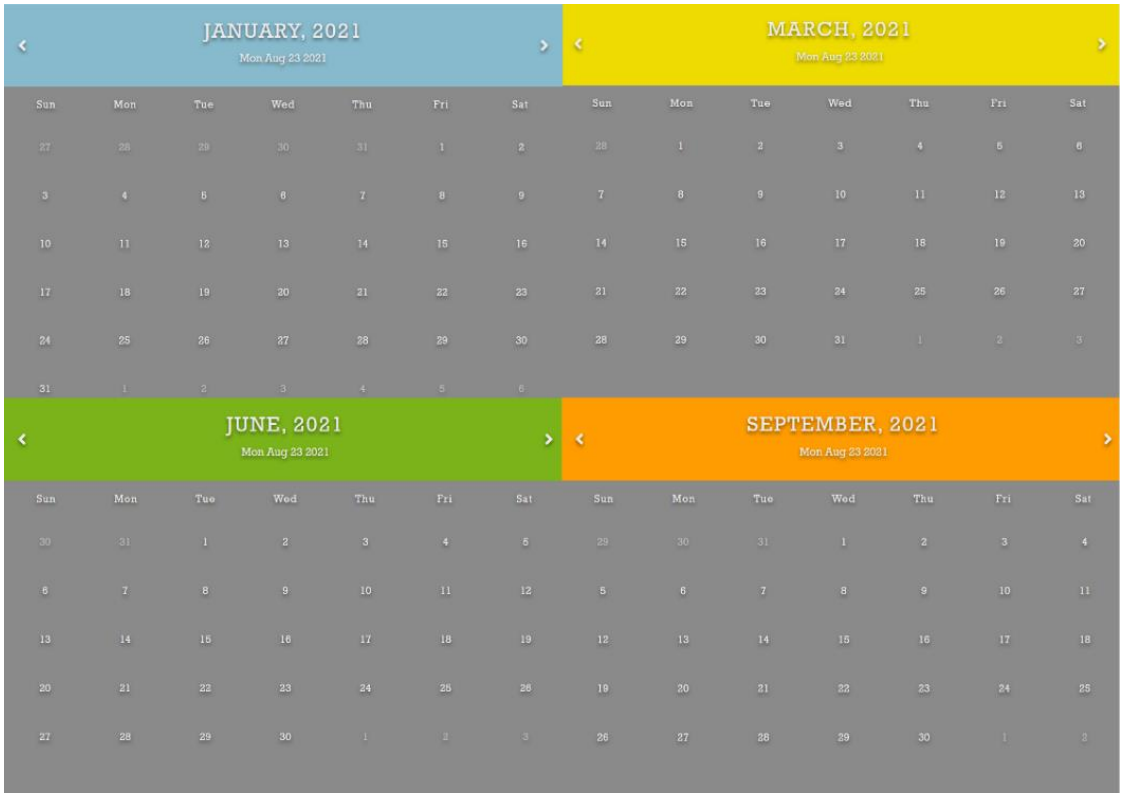
```
const d = new Date();
document.getElementById("demo").innerHTML = d.toString();
```

Slika 22. Niz sa toString() metodom

Izvor: https://www.w3schools.com/js/js_dates.asp

3. PRAKTIČNI DIO

Praktični dio ovog završnog rada započinje osmišljavanjem izgleda web stranice tj. kalendara koji se nalazi na njoj te funkcija koje će ova web stranica sadržavati. Primjer izgleda kalendara vidljiv je na slici 24.



Slika 24. Primjer izgleda kalendara

Zatim, za početak, potrebno je kreirati posebnu mapu koja će obuhvaćati sve ono što će se nalaziti na web stranici. Dakle, u mapi treba biti HTML dokument, podmapa koja sadrži CSS i podmapa koja sadrži JavaScript.

Kodiranje web stranice započinje .html datotekom i definiranjem vrste dokumenta korištenjem !DOCTYPE elementa. Otvaranjem <html> elementa definira se jezik HTML-a (u mojem slučaju engleski) te unutar njega postavljaju se <head> i <body> elementi koji se zatvaraju </html> oznakom. Unutar <head> elementa definira se meta koja deklarira znakovni set, skripte i linkove za vanjske datoteke tj. za lokacije CSS i JavaScript dokumenata i API-a koji povlači podatke iz baze podataka te naslov i ikona web kartice. Navedeni linkovi vidljivi su na na slici broj 25.

```
<head>
  <meta charset='utf-8' />
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <link href='css/bootstrap.css' rel='stylesheet' />
  <script src='scripts/bootstrap.js'></script>
  <link rel="stylesheet" href="css/style.css" />
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.13.0/css/all.min.css" />
  <title> Factlendar </title>
  <link rel="shortcut icon" href="favicon.png" type="image/x-icon">
</head>
```

Slika 25. Prikaz <head> elementa

U <body> elementu započinje stvaranje kalendara. Prvo što se nalazi u kodu prikazanom na slici 26, je navigacijska traka (eng. *navbar*) koja uobičajeno služi za popis veza no u ovom slučaju korištena je za promjenu boje pozadine.

```
<body>

  <nav class="navbar navbar-dark bg-dark">
    <a class="navbar-brand" href="#">
      <input type="color" id="background-color" name="background-color" value="#12121f">
      <label for="background-color">Background color</label>
    </a>
  </nav>
```

Slika 26. Prikaz koda za navigacijsku traku

Zatim slijedi kreiranje kalendara kod kojeg su stvorene pojedine div klase koje tvore cjelokupni kalendar i ono od čega se on sastoji (mjesec, tjedan, dani) te su dodani znakovi za navigiranje po kalendaru kao što je vidljivo na slici 27.

```
<div class="container">
  <div class="calendar">
    <div class="month" id="month">
      <i class="fas fa-angle-left prev"></i>
      <div class="date">
        <h1></h1>
        <p></p>
      </div>
      <i class="fas fa-angle-right next"></i>
    </div>
    <div class="weekdays">
      <div>Sun</div>
      <div>Mon</div>
      <div>Tue</div>
      <div>Wed</div>
      <div>Thu</div>
      <div>Fri</div>
      <div>Sat</div>
    </div>
    <div class="days"></div>
  </div>
</div>
```

Slika 27. Prikaz koda za kalendar

Unutar klase "container" nalaze se svi elementi kalendara. On sam po sebi ne služi za ništa osim da sadrži ostale elemente i da sadržaj stranice drži u željenoj širini (centrirano). Zatim, unutar klase "container" nalazi se klasa "calendar" u kojoj su ostali elementi kalendara, te koja služi za stiliziranje vanjskog dijela kalendara. Tako i sve ostale klase unutar klase "calendar" služe za stiliziranje određenog dijela kalendara. Na slici 28 prikazan je kod u CSS-u za klasu "calendar" i primjer kako izgleda, te je isto tako na slici 29 prikazan kod u CSS-u za klasu "month" i primjer kako to na web stranici izgleda.

```css
.calendar {
  width: 100%;
  height: 100%;
  background-color: #535353;
  box-shadow: 0 0.5rem 3rem rgba(88, 88, 88, 0.4);
}
```

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

Slika 28. Prikaz koda u CSS-u za kalendar

```css
.month {
  width: 100%;
  height: 12rem;
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 0 2rem;
  text-align: center;
  text-shadow: 0 0.3rem 0.5rem rgba(0, 0, 0, 0.5);
}
```

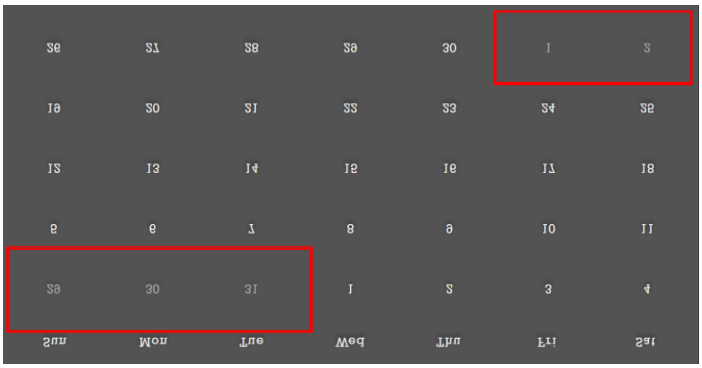
Slika 29. Prikaz koda u CSS-u za mjesec

Da bi kalendar funkcionirao, koristi se funkcija "renderCalendar" unutar JavaScript dokumenta u kojoj se odvija sva logika kalendara. Prvo se izrađuje i popunjava lista "months" s imenima mjeseci. Pomoću linije "const date = new Date();" dobiva se današnji tj. trenutni datum. Trenutni mjesec sprema se u varijablu "currentMonth" koja se koristi u sljedećoj liniji te kasnije u kodu. Zatim se odabire *header* element unutar "date" klase koristeći "querySelector" i tako se prikazuje trenutni mjesec. Na isti taj način odabire se paragraf element i prikazuje se cijeli današnji datum te primjer toga vidljiv je na slici 29 dok je primjer koda vidljiv na slici 30.

```
const months = [
  "January",
  "February",
  "March",
  "April",
  "May",
  "June",
  "July",
  "August",
  "September",
  "October",
  "November",
  "December",
];
const date = new Date();
var currentMonth = months[date.getMonth()];
document.querySelector(".date h1").innerHTML = currentMonth + ", " + date.getFullYear();
document.querySelector(".date p").innerHTML = new Date().toDateString();
```

Slika 30. Prikaz koda za prikazivanje trenutnog mj. i datuma

Potom se kreće sa prikazivanjem dana u kalendaru. Najprije se uzima zadnji i prvi dan trenutnog i prošlog mjeseca da bi se dinamički prikazao točan broj dana za svaki mjesec te se uz to prikazuju i određeni dani iz prošlog/sljedećeg mjeseca da bi se popunile rupe u tjednu. Primjer toga prikazan je na slici 31 gdje su crvenim pravokutnicima označena polja koja prikazuju datume iz prošlog/sljedećeg mjeseca, dok je na slici 32 prikazan kod spomenute radnje.



Slika 31. Prikaz dana prošlog/sljedećeg mjeseca

```
const monthDays = document.querySelector(".days");

const lastDay = new Date(
  date.getFullYear(),
  date.getMonth() + 1,
  0
).getDate();

const prevLastDay = new Date(
  date.getFullYear(),
  date.getMonth(),
  0
).getDate();

const firstDayIndex = date.getDay();

const lastDayIndex = new Date(
  date.getFullYear(),
  date.getMonth() + 1,
  0
).getDay();

const nextDays = 7 - lastDayIndex - 1;
```

Slika 32. Prikaz koda za dohvaćanje prvih i zadnjih dana u mj.

Dalje u kodu kreće prikazivanje dana. Prva "for" petlja popunjava rupe u tjednu prije prvog dana trenutnog mjeseca tako što uzima isti taj prvi dan i smanjuje se za jedan dok ne popuni redak tj. tjedan. Unutar druge "for" petlje vrti se kroz dane u mjesecu i u "if" bloku provjerava se za svaki dan da li je to današnji tj. trenutni datum te ako je, dodaje se tom danu klasa "today" koja je stilizirana da prikaže označen dan, a ako nije, prikazuje se "običan" dan. Treća tj. zadnja petlja radi isto što i prva, samo što popunjava rupe u tjednu poslije zadnjeg dana u trenutnom mjesecu tako što uzme isti taj zadnji dan i poveća ga za jedan dok ne popuni taj redak tj. tjedan do kraja. Kod za sve tri navedene "for" petlje prikazan je na slici 33.

```
let days = "";

for (let x = firstDayIndex; x > 0; x--) {
  days += `<div class="prev-date">${prevLastDay - x + 1}</div>`;
}

for (let i = 1; i <= lastDay; i++) {
  if (
    i === new Date().getDate() &&
    date.getMonth() === new Date().getMonth() && new Date().getFullYear() === date.getFullYear()
  ) {
    days += `<div class="today">${i}</div>`;
  } else {
    days += `<div>${i}</div>`;
  }
}

for (let j = 1; j <= nextDays; j++) {
  days += `<div class="next-date">${j}</div>`;
}
```

Slika 33. Prikaz koda za "for" petlje

Sljedeći dio koda stilizira *header* kalendara u odnosu na godišnja doba. Dakle, prvo se pune liste godišnjih doba s njihovim mjesecima te se zatim provjerava kojem godišnjem dobu pripada trenutni mjesec i u odnosu na to "boja" se *header* kalendara. Primjer koda prikazan je na slici 34, dok je izgled prikazan na slici 35.

```
const spring = ["March", "April", "May"];
const summer = ["June", "July", "August"];
const autumn = ["September", "October", "November"];
const winter = ["December", "January", "February"];

if(spring.includes(currentMonth)){
  document.getElementById('month').style.backgroundColor = "#EEDB00"
}else if (summer.includes(currentMonth)){
  document.getElementById('month').style.backgroundColor = "#7BB31A"
}else if (autumn.includes(currentMonth)){
  document.getElementById('month').style.backgroundColor = "#FF9C00"
}else if (winter.includes(currentMonth)){
  document.getElementById('month').style.backgroundColor = "#88BACD"
}
```

Slika 34. Prikaz koda za "obojani" header kalendara

Slika 35. Prikaz headera u odnosu na godišnje doba

Dalje što slijedi su navigacijske strelice koje su vidljive na slici broj 35 te koje navigiraju prethodni i sljedeći mjesec. Koristeći "querySelector" dodaje se element ".prev" tj. gumb za nazad, "addEventListener" koji "sluša" i čeka klik kojim se navigira na prethodni mjesec. Istim takvim postupkom koristeći "querySelector" dodaje se elementu ".next" gumb za naprijed, "addEventListener" koji također "sluša" i čeka klika te potom navigira na sljedeći mjesec. Kod za taj postupak je prikazan na slici 36.

```
document.querySelector(".prev").addEventListener("click", () => {
  date.setMonth(date.getMonth() - 1);
  renderCalendar();
});

document.querySelector(".next").addEventListener("click", () => {
  date.setMonth(date.getMonth() + 1);
  renderCalendar();
});
```

Slika 36. Prikaz koda za navigacijske strelice

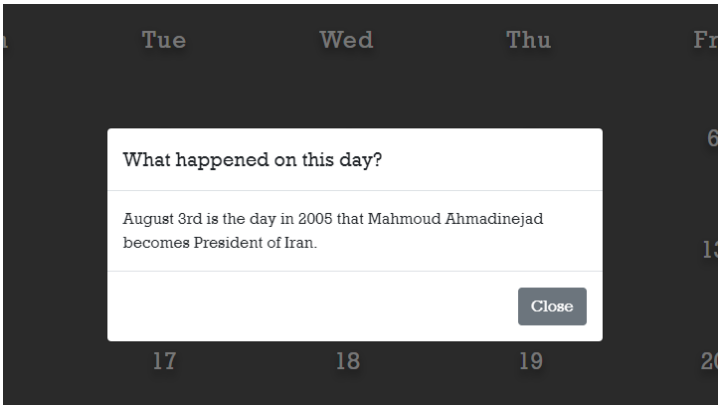
Nadalje, započinje prikazivanje i povlačenje podataka tako što se šalje *GET* zahtjev na "URL NumbersAPI" za odabrani datum pomoću *jQuery*. *jQuery* je brza i "lightweight" JavaScript biblioteka koja olakšava kretnju i manipulaciju HTML dokumenata, rukovanje događajima i animaciju. Kombinacijom fleksibilnosti i proširivosti, *jQuery* je promijenio i olakšao način na koji ljudi pišu JavaScript.

Prvo je svim danima na kalendaru dodijeljen "addEventListener" koji omogućava da se na njih može kliknuti te tim klikom se poziva funckija "dayClick". Shodno tome, "ulaskom" u funkciju, definira se varijabla "url" sa linkom od *NumbersAPI* stranice preko koje se povlače podatci tj. *event*-i (događaji). Varijablom "month" sprema se trenutni datum i povećava ga za jedan jer u JavaScriptu mjeseci kreću od 0. Zatim se oblikuje "url" pomoću kojeg se obavlja "GET" zahtjev. "Url" se tvori od primarnog linka na koji se nadodaje odabrani mjesec i dan te nastavak "/date" zbog toga što *NumbersAPI* nudi i druge brojčane funkcionalnosti. Primjer linka izgleda ovako: *http://numbersapi.com/8/26/date*. U sljedećoj liniji poziva se spomenuti "GET" zahtjev na navedeni "url". "GET" zahtjev vraća zanimljivosti za taj dan u "plain text" obliku koji se zatim šalje u funkciju gdje zapravo ispisuje zanimljivosti u element iskočnog prozora te ga na kraju i prikazuje. Prikaz koda vidljiv je na slici 37 dok je prikaz iskočnog prozora na web stranici prikazan na slici 38.

```
document.querySelector(".days").addEventListener("click", dayClick);

function dayClick(event){
    var url = "http://numbersapi.com/";
    var month = date.getMonth() + 1;
    var getUrl = url + month + "/" + event.target.innerHTML + "/date";
    $.get(getUrl, function (data) {
      document.getElementById("modalBody").innerHTML = data;
      $("#exampleModal").modal('show');
    });
}
```

Slika 37. Prikaz koda za povlačenje podataka



Slika 38. Prikaz iskočnog prozora

Za iskočni prozor koristio se *bootstrap*-ov template koji sadrži klasu "modal-body" u kojem se zapisuju povučeni podatci tj. zanimljivosti. *Modal* je dijaloški okvir/iskočni prozor koji se prikaže kada ga se pozove te kojem je dodijeljen naslov "What happened on this day?". Primjer toga nalazi se na slici 38, dok je na slici 39 kod za taj spomenuti *modal*.

```
<!-- Modal -->
<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel"
  aria-hidden="true">
  <div class="modal-dialog modal-dialog-centered" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModalLabel">What happened on this day?</h5>
      </div>
      <div class="modal-body" id="modalBody">
        ...
      </div>
      <div class="modal-footer">
        <button id="btnHideModal" type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
      </div>
    </div>
  </div>
</div>
```

Slika 39. Prikaz koda za modal

Pred kraj je dodana mogućnost promijene boje pozadine kako bi svatko mogao prilagoditi kalendar po svojoj želji tj. u ovom slučaju, boji.

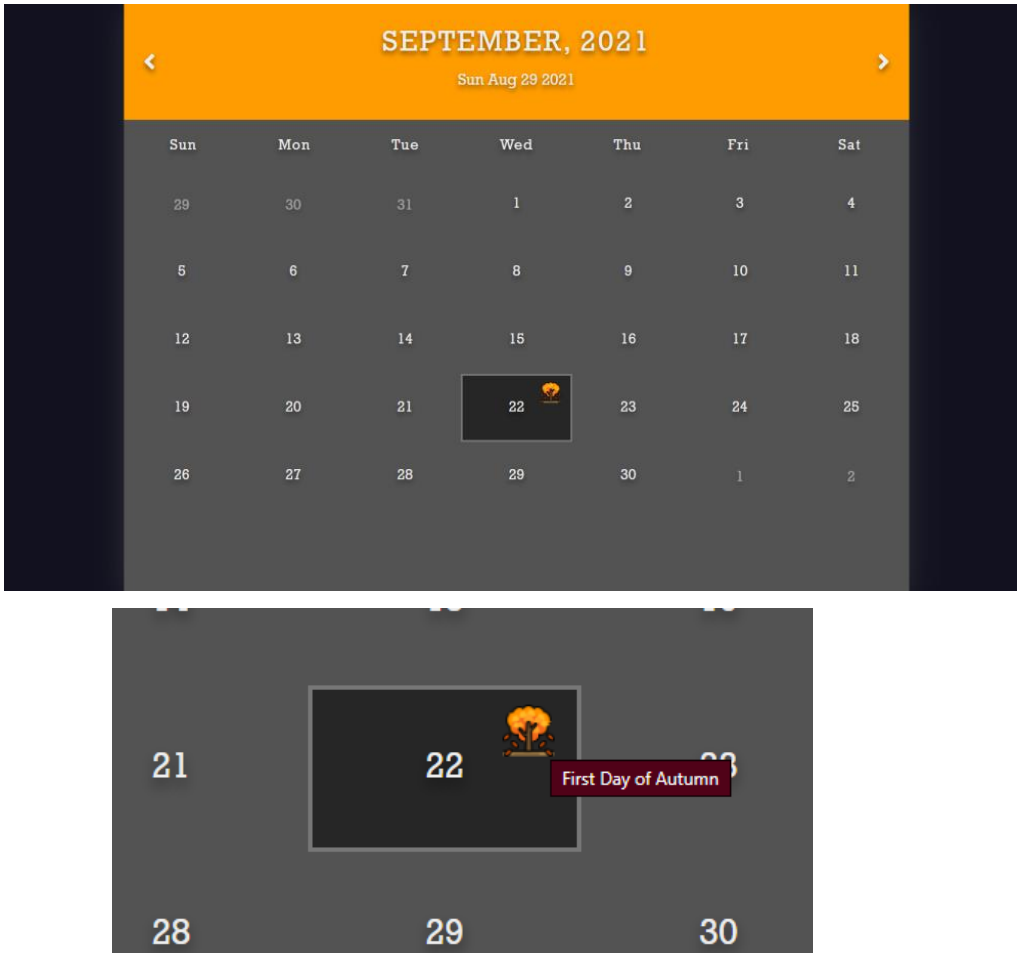
```
function changeBackground(){
document.body.style.backgroundColor = document.getElementById('background-color').value;
}
```

Slika 40. Prikaz koda za promijenu boje pozadine

Za kraj, kalendaru su dodane male ikone kako bi se bolje dočarali određeni datumi te kako bi kalendar zanimljivije izgledao. Na već spomenutu drugu "for" petlju nastavljaju se "if" provjere koje provjeravaju koji je od navedenih datuma praznik, te ako je, u tom slučaju postavljaju se određene tj. predviđene ikone za taj datum tj. praznik. Uz dodatak "title" prelaskom miša preko ikone, navodi se naziv navedenog praznika. Prikaz koda za dodavanje ikona vidljiv je na slici 41 dok je primjer izgleda na web stranici za spomenutu radnju prikazan na slici 42.

```
for (let i = 1; i <= lastDay; i++) {
  if (
    i === new Date().getDate() &&
    date.getMonth() === new Date().getMonth() && new Date().getFullYear() === date.getFullYear()
  ) {
    days += `<div class="month-day today"><p>${i}</p></div>`;
  } else {
    if(i == 25 && currentMonth == "December"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/bozic.png" title="Christmas"></div>`
    }else if(i == 22 && currentMonth == "April"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/danzemlje.png" title="Earth Day"></div>`
    }else if(i == 31 && currentMonth == "October"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/halloween.png" title="Halloween"></div>`
    }else if(i == 14 && currentMonth == "February"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/valentinovo.png" title="Valentine's Day"></div>`
    }else if(i == 8 && currentMonth == "March"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/zena.png" title="Women's Day"></div>`
    }else if(i == 20 && currentMonth == "March"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/proljece.png" title="First Day of Spring"></div>`
    }else if(i == 21 && currentMonth == "June"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/ljeto.png" title="First Day of Summer"></div>`
    }else if(i == 22 && currentMonth == "September"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/jesen.png" title="First Day of Autumn"></div>`
    }else if(i == 21 && currentMonth == "December"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/zima.png" title="First Day of Winter"></div>`
    }else if(i == 1 && currentMonth == "January"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/novagodina.png" title="New Year"></div>`
    }else if(i == 27 && currentMonth == "April"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/dizajn.png" title="Graphic Communication's Day"></div>`
    }else if(i == 8 && currentMonth == "May"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/majka.png" title="Mother's Day"></div>`
    }else if(i == 1 && currentMonth == "August"){
      days += `<div class="month-day important"><p>${i}</p><img class="corner" src="ikone/prijatelj.png" title="Friendship Day"></div>`
    }else
      days += `<div class="month-day"><p>${i}</p></div>`;
  }
}
```

Slika 41. Prikaz koda za dodavanje ikona



Slika 42. Prikaz ikone i naslova ikone

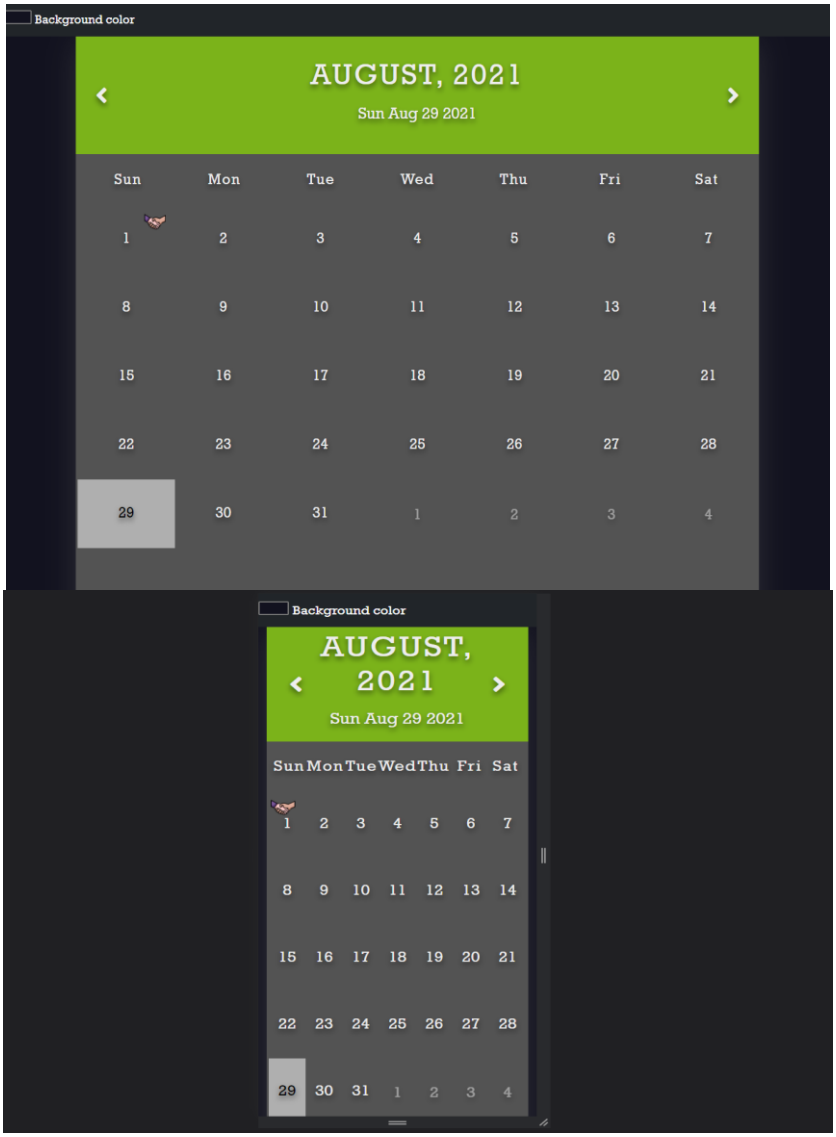
4. ZAKLJUČAK

Dakle, da se zaključiti kako interaktivne web stranice mogu postati tj. biti izvor znanja a time i zabave za njihove korisnike. Što je web stranica interaktivnija, time privlači više korisnika. Dodavanje interakcija ili sadržaja, npr. sadržaj za personalizirano stiliziranje web stranice, jedno je od pozitivnih strana korištenja takvih web stranica jer suzbija određenu rutinu koju većina web stranica posjeduje. U današnje vrijeme sve više malih ili velikih tvrtki daje izraditi interaktivne web stranice kako bi imali što bolje i kvalitetnije korisničko iskustvo te time uspješnije poslovali.

Naravno, za kvalitetnije korisničko iskustvo, osim dodavanja interaktivnog sadržaja, potrebno je i prilagoditi web stranice na različite uređaje kako bi web stranica mogla biti svuda dostupna. Tako je ova web stranica uz jednostavan dodatak koda:

(<meta name="viewport" content="width=device-width, initial-scale=1">)

postala prilagođena i za mobilne uređaje kao što se vidi na slici 43.



Slika 43. Prikaz kompatibilnosti web stranice za različite uređaje (monitor i mobitel)

5. LITERATURA

1) *HTML*. URL:https://www.w3schools.com/html/default.asp (Pristupljeno 11.7.2021.)
2) *Web Tech.* URL:http://www.webtech.com.hr/html.php (Pristupljeno 9.8.2021)
3) Prelec, A. (2015) *Izrada responzivne internetske stranice upotrebom HTML5 i CSS3 tehnologija*. Završni rad. Zagreb: Sveučilište u Zagrebu, Grafički fakultet
4) *CSS.* URL: https://www.w3schools.com/css/default.asp (Pristupljeno 11.8.2021)
5) *Kako uključiti JavaScript u web pregledniku*. URL:https://www.enable-javascript.com/hr/ (Pristupljeno 29.7.2021.)
6) *JavaScript.* URL: https://www.w3schools.com/js/js_whereto.asp (Pristupljeno 22.8.2021.)