Vježba 8 - Ajax 2. dio i sesije

Zadatak 1. Dovršite zadatke 1 i 3 sa prethodne vježbe i pokrenite primjer 2!

Primjer 1. Koristeći sesije napravite igricu za pogađanje brojeva. Broj može biti od 1 do 100 i određuje se slučajno. Igrač ima ukupno 10 pokušaja, a pri svakom pokušaju ispisuje se poruka koliko je pokušaja preostalo, te da li je broj koji se traži veći ili manji. Broj pokušaja i broj koji se pogađa čuvajte u sesiji. Zadatak realizujte bez korištenja Ajax-a!

Sesija je način perzistencije podataka između zahtjeva klijenata prema serveru u određenom vremenskom intervalu kada korisnik šalje poruke prema serveru. Kako HTTP protokol ne posjeduje stanja potrebno je na neke od načina pamtiti podatke koji su bitni za nesmetan rad korisnika. Primjer kada se koriste sesije su kod prijave korisnika na sistem. Kada se korisnik prijavi na sistem u sesiju su upisani podaci dovoljni za prepoznavanje prijavljenog korisnika pri svim narednim zahtjevima. Ukoliko ne bi postojala sesija uz svaki zahtjev korisnik bi morao slati svoje korisničko ime i šifru.

Sesije kroz express možemo koristiti uključivanjem modula express-session. Podaci iz sesije se često čuvaju u cookie-u. Cookie je jedan od načina kako web preglednik spašava podatke koje mu server pošalje a koji su bitni za rad sesije. Server može poslati cookie u header-u odgovora prema klijentu; web preglednik će otvoriti header i spasiti navedeni cookie. Uz svaki sljedeći zahtjev klijenta prema serveru, web preglednik dodaje cookie kojeg mu je server ranije poslao. Na ovaj način je moguće ostvariti sesiju. Paket express-session ne čuva sve podatke u cookie-u nego samo id sesije. Podaci o sesiji se čuvaju na serveru, a na osnovu id sesije određuje se koji podaci pripadaju kojem korisniku.

Za funkcionalnosti našeg zadatka u sesiji se trebaju čuvati dva podatka, broj koji se pogađa i broj preostalih pokušaja. Više o express-sesion možete pročitati na linku.

Da bi koristili sesije trebate, instalirati paket express-session, uključiti ga i postaviti sesiju koristeći sljedeći dio koda:

```
app.use(session({
  secret: 'neka tajna sifra',
  resave: true,
  saveUninitialized: true
}));
```

***secret -** parametar kojim postavljamo tajnu sa kojom se cookie potpisuje. Potpisivanje cookie-a je potrebno da bi se osiguralo da niko nije mijenjao sadržaj cookie-a.*

***resave -** parametar kojim govorimo o tome da se podaci o sesiji trebaju snimiti u session store.*

saveUnitialized - *parametar kojim se podešava da li će nove sesije koje nisu modifikovane biti snimane u store.*

U sesiju podatke možemo upisati koristeći request objekat pozivajući ***req.session.podatak=”neka vrijednost”****, gdje je req request objekat. Na isti način čitamo podatke iz sesije. Ako je vrijednost podatka null to znači da sesija nije postavljena.*

Do sada smo html iz Node-a vraćali tako što smo koristili ili res.sendFile metodu ili smo upisivali direktno html kod u odgovor. Pristup kada direktno upisujemo html u odgovor nije praktičan i težak je za održavanje. Zbog toga se često koristi neki od template engine-a. Template engine omogućava da koristimo statičke fajlove, a da u njih u toku izvršavanja ubacujemo vrijednosti variabli, a klijentu se šalje html kod sa ažuriranim podacima. Na ovaj način spajamo statičke stranice i podatke koji se mijenjaju i klijentu se šalje gotov html dokument. Template engine kojeg ćemo koristiti u ovom primjeru je ***pug****. Stranice se zapisuju u pug formatu koji je dosta sličan html kodu, s tim da se umjesto tagova pišu samo nazivi tagova. Ugnježđavanje tagova je realizovano putem indentacije. Atribute tagova pišemo u zagradama. Variable u statičku stranicu ubacujemo sa* ***#{naziv_variable}****. U pug kodu možemo koristiti i petlje i uslove grananja. Više o ovome na linku.*

primjer1.js*:*

```
const express = require("express");
const session = require("express-session");
const bodyParser = require("body-parser");
const path = require("path");

const app = express();
app.use(bodyParser.urlencoded({ extended: true }));
app.use(session({
  secret: 'neka tajna sifra',
  resave: true,
  saveUninitialized: true
}));
app.set("view engine", "pug");
app.set("views", path.join(__dirname, "views"));

app.get('/',function(req,res){
  if(req.session.broj!=null) {
     req.session.broj =Math.floor((Math.random() * 100) + 1);
     req.session.broj_pokusaja=10;
  }
  res.render("pogodiBroj",{pokusaj:{broj_pokusaja:10,poruka:"Pogodite broj"}});
});


app.post('/',function(req,res){
  let broj,broj_pokusaja,poruka;
```

```
  if(req.session.broj!=null) {
     broj =req.session.broj;
     broj_pokusaja=req.session.broj_pokusaja;
  }
  else {
     broj=Math.floor((Math.random() * 100) + 1);
     broj_pokusaja=10;
     req.session.broj=broj;
     req.session.broj_pokusaja=broj_pokusaja;
  }

  if(broj_pokusaja>0){
     broj_pokusaja--;
     req.session.broj_pokusaja=broj_pokusaja;
     if(req.body['pokusaj']==broj){
        poruka="Pogodili ste broj!";
     }else if(req.body['pokusaj']>broj){
        poruka="Broj je manji!";
     }else if(req.body['pokusaj']<broj){
        poruka="Broj je veći!";
     }
  }else{
     poruka="Game over!";
     req.session.broj=null;
  }
  res.render("pogodiBroj",{pokusaj:{broj_pokusaja:broj_pokusaja,poruka:poruka}});
});
app.listen(3000);
```

*Napravite folder views i u njemu napravite **pogodiBroj.pug**:*

```
html
  body
     h1 Pogodi broj
     p #{pokusaj.broj_pokusaja}
     h2 #{pokusaj.poruka}
     form(action="/",method="POST")
        input(type="text",name="pokusaj")
        input(type="submit",value="Pogodi")
```

*Isprobajte primjer pokretanjem **node primjer1.js**, otvorite **http://localhost:3000**. Otvorite developer konzolu i probajte pronaći da li se negdje nalazi broj kojeg trebati pogoditi. Na vaš repozitorij postavite screen shot-ove sa vašim pokušajima.*

Primjer 2. Uradite primjer 1 tako da se prilikom pokušaja stranica ne reload-a već da se samo ažuriraju podaci o poruci i broju preostalih pokušaja.

Pokušajte uraditi sami, kada ne uspijete rješenje se nalazi ispod. U rješenju postoji greška koju trebate otkriti.

Rješenje:

---

primjer2.js:

```
const express = require("express");
const session = require("express-session");
const bodyParser = require("body-parser");

const app = express();
app.use(bodyParser.json());
app.use(session({
   secret: 'neka tajna sifra',
   resave: true,
   saveUninitialized: true
}));
app.use(express.static(__dirname+"/public"));
app.get('/',function(req,res){
   res.sendFile(__dirname+"/public/pogodiBroj.html");
});
app.post('/',function(req,res){
   let broj,broj_pokusaja,poruka="";
   if(req.session.broj!=null) {
      broj =req.session.broj;
      broj_pokusaja=req.session.broj_pokusaja;
   }
   else {
      broj=Math.floor((Math.random() * 100) + 1);
      broj_pokusaja=10;
      req.session.broj=broj;
      req.session.broj_pokusaja=broj_pokusaja;
      poruka="Pogodite broj!"
      res.json({pokusaj:{broj_pokusaja:broj_pokusaja,poruka:poruka}});
   }

   if(broj_pokusaja>0&&poruka.length==0){
      broj_pokusaja--;
      req.session.broj_pokusaja=broj_pokusaja;
      if(req.body['pokusaj']==broj){
         poruka="Pogodili ste broj!";
      }else if(req.body['pokusaj']>broj){
         poruka="Broj je manji!";
      }else if(req.body['pokusaj']<broj){
         poruka="Broj je veći!";
      }
   }else{
      poruka="Game over!";
      req.session.broj=null;
   }
   res.json({pokusaj:{broj_pokusaja:broj_pokusaja,poruka:poruka}});
});
app.listen(3000);
```

Sljedeće fajlove stavite u folder *'public'*!

pogodiBroj.html:

```
<html>
  <head>
     <title>Pogodite broj</title>
  </head>
  <body>
     <h1>Pogodite broj</h1>
     Broj pokušaja: <p id="brojPokusaja"></p>
     <h2 id="poruka"></h2>
     <form action="#">
        <input type="text" name="pokusaj" id="pokusaj">
        <input type="button" value="Pošalji" id="posalji">
     </form>
  <script src="posaljiAjax.js"></script>
  </body>
</html>
```

posaljiAjax.js:

```
var porukaElement,brojPokusajaElement;
var pokusaj;
window.onload=function(){
  porukaElement=document.getElementById("poruka");
  brojPokusajaElement=document.getElementById("brojPokusaja");
  pokusaj=document.getElementById("pokusaj");
  pokusajAjax(null,ispisi);

document.getElementById("posalji").addEventListener("click",function(){pokusajAjax(pok
usaj.value,ispisi);})
}
function ispisi(poruka,brojPokusaja){
  porukaElement.innerHTML=poruka;
  brojPokusajaElement.innerHTML=brojPokusaja;
}
function pokusajAjax(pokusaj,fnCallback){
  var ajax = new XMLHttpRequest();
  ajax.onreadystatechange = function() {// Anonimna funkcija
     if (ajax.readyState == 4 && ajax.status == 200){
        var jsonRez = JSON.parse(ajax.responseText);
        fnCallback(jsonRez.pokusaj.poruka,jsonRez.pokusaj.broj_pokusaja);
     }
     else if (ajax.readyState == 4)
        fnCallback(ajax.statusText,null);
  }
  ajax.open("POST","http://localhost:3000",true);
  ajax.setRequestHeader("Content-Type", "application/json");
  ajax.send(JSON.stringify({pokusaj:pokusaj}));
}
```

Pokrenite primjer i snimite zahtjeve i odgovore u developer tools-u. Slike zahtjeva i odgovora postavite na vaš repozitorij!