

ADP Project

Counselling Session
Booking

G - 10

(B21136 , B21137 , B21138 , B21139 , B21140 , B21141)

Index

- Acknowledgement
- Abstract
- Introduction
- Making Process
- Result
- Conclusion

Acknowledgement :

We, the members of group G-10, would like to express our profound gratitude to Mr. Varun Dutt (Faculty Adviser SCEE , Course Instructor CS207) of School of Computing and Electrical Engineering , IIT Mandi, and all the TAs associated with the course Applied Database Practicum (CS 207) for their valuable contribution towards the completion of our project titled "**Counselling Session Booking**".

We would like to express our special thanks to our mentor Hemant (B20288@students.iitmandi.ac.in) for his guidance and counselling. Your useful advice and suggestions were really helpful to us during the project's completion. In this aspect, we are eternally grateful to you.

I would like to acknowledge that this project was completed entirely by the members of group G-10 which consists of me(B21140) , Shubham Kumar(B21136) , Smiti Oswal(B21137) , Somit Gond(B21138) , Sourav Sav(B21139) , Swati Shakya(B21141) and not by someone else.

Abstract :

Motive for making a counselling site is to provide counsel to every person who needs it. People have access to simple one-to-one interactions on this site. No more worrying about finding a counsellor first and then booking a session, here you get a counsellor of your own choice , who understands your problem and suggests to you what you should do.

Our job was to implement what we have learnt in the course ADP CS207 , and prepare a fully functioning system which would take costumer's registration and manage available sessions and booked sessions for Counselling purposes ; with an organised database system and an interactive interface (Website).

Introduction :

This project is about providing a platform for counselling.

This is an online platform where we provide counsellors to people.

Here , first people have to register their name for the counselling sessions then for booking a session they have to login on the site.

After login they got a list of different counsellors and the slots against each counsellor which are available.

These sessions start on the next day after booking.

Each person can book only one session a day.

Making Process :

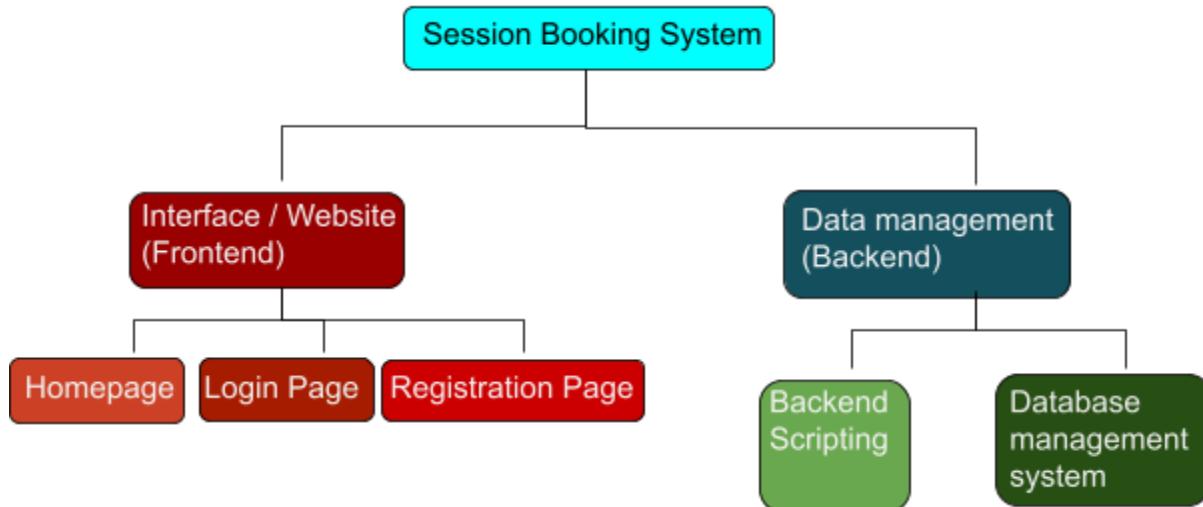
Programmes we are using for various purposes :

- HTML , CSS , Bootstrap template
- JavaScript
- NodeJS
- MongoDB

Editors and Proprietary consoles that are used :

- VS code , Sublime Text
- Node_js terminal shell
- Mongosh (MongoDB Shell)

We have divided our project into 2 parts - Frontend and Backend .



Frontend : For frontend we are using HTML , CSS and JavaScript. We have designed different templates for the Homepage , Registration page and Login page. Templates are designed using Bootstrap.

```

registers.js
const mongoose = require("mongoose");
const FormSchema = new mongoose.Schema({
  firstname: {
    type: String,
    required: true
  },
  lastname: {
    type: String,
    required: true
  },
  email: {
    type: String,
    required: true,
    unique: true
  },
  gender: {
    type: String,
    required: true
  },
  phone: {
    type: Number,
    required: true
  },
  age: {
    type: Number,
    required: true
  },
  password: {
    type: String,
    required: true
  },
  confirmPassword: {
    type: String,
    required: true
  }
});

session.js
const mongoose = require("mongoose");
const Schema = mongoose.Schema;
const sessionSchema = new Schema({
  cond: {type:Boolean,require:true},
  cond2:{type:Boolean,require:true},
  No:{type:Number,require:true},
  doctor:{type:String,require:true},
  sessionno:{type:String,require:true},
  sessiontime:{type:String,require:true},
  book:{type:String,require:true}
});
const Session = mongoose.model("Session",sessionSchema);
module.exports=Session;

```

Backend : For backend processes we are using NodeJS and MongoDB .
Here we have mainly 3 types of databases.

1. For keeping track of Counsellors' list : We are listing all the information of the Counsellors in JSON format.

```

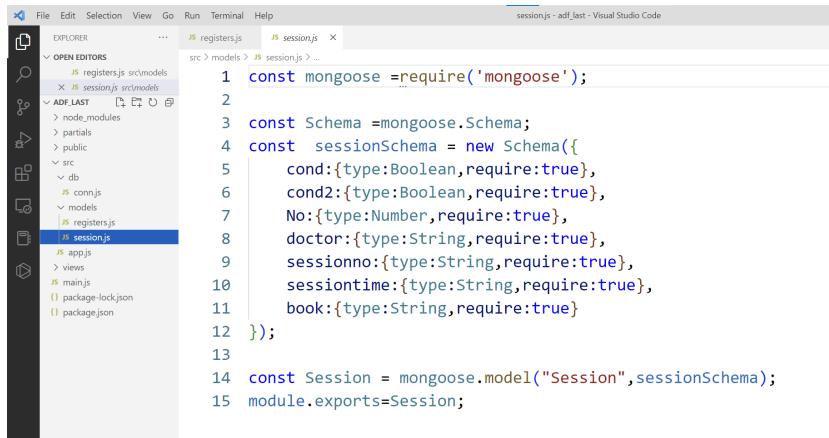
session.json
{
  "cond": "true",
  "cond2": "true",
  "No": 1234567890,
  "doctor": "Counselor",
  "sessionno": "1234567890",
  "sessiontime": "10:00-10:30 AM",
  "book": "Book One"
}

registers.js
const mongoose = require("mongoose");
const FormSchema = new mongoose.Schema({
  firstname: {
    type: String,
    required: true
  },
  lastname: {
    type: String,
    required: true
  },
  email: {
    type: String,
    required: true,
    unique: true
  },
  gender: {
    type: String,
    required: true
  },
  phone: {
    type: Number,
    required: true
  },
  age: {
    type: Number,
    required: true
  },
  password: {
    type: String,
    required: true
  },
  confirmPassword: {
    type: String,
    required: true
  }
});

```

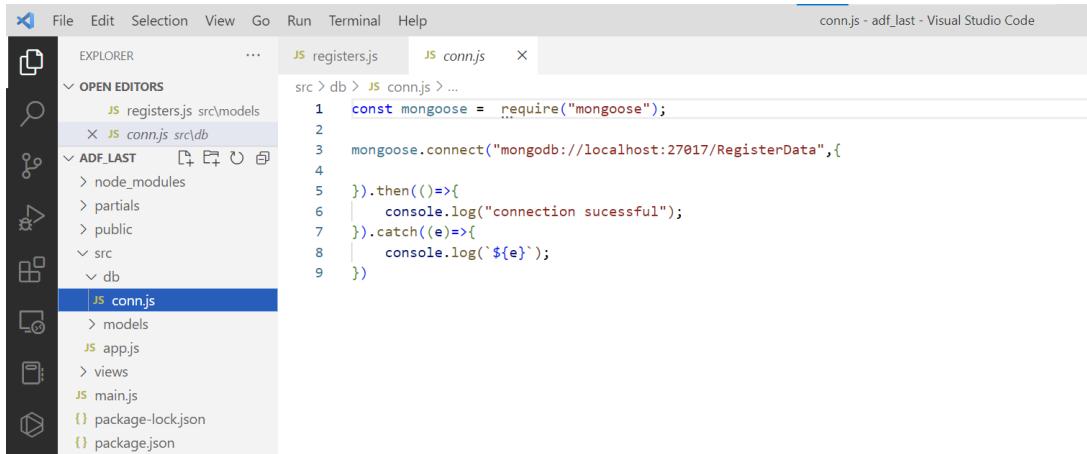
2. For Collecting Registration Data : In NodeJS script we are implementing MongoDB format data insertion.

3. For Assigning Sessions : This is done in NodeJS.



```
File Edit Selection View Go Run Terminal Help
EXPLORER OPEN EDITORS ...
src > models > JS session.js > ...
src > models > JS session.js src\models
src > models > JS session.js src\models
src > ADF_LAST > ...
> node_modules
> partials
> public
> src
> db
> models
> registers.js
> session.js
> app.js
> views
> main.js
{} package-lock.json
{} package.json
1 const mongoose = require('mongoose');
2
3 const Schema = mongoose.Schema;
4 const sessionSchema = new Schema({
5   cond:{type:Boolean,require:true},
6   cond2:{type:Boolean,require:true},
7   No:{type:Number,require:true},
8   doctor:{type:String,require:true},
9   sessionno:{type:String,require:true},
10  sessiontime:{type:String,require:true},
11  book:{type:String,require:true}
12 });
13
14 const Session = mongoose.model("Session",sessionSchema);
15 module.exports=Session;
```

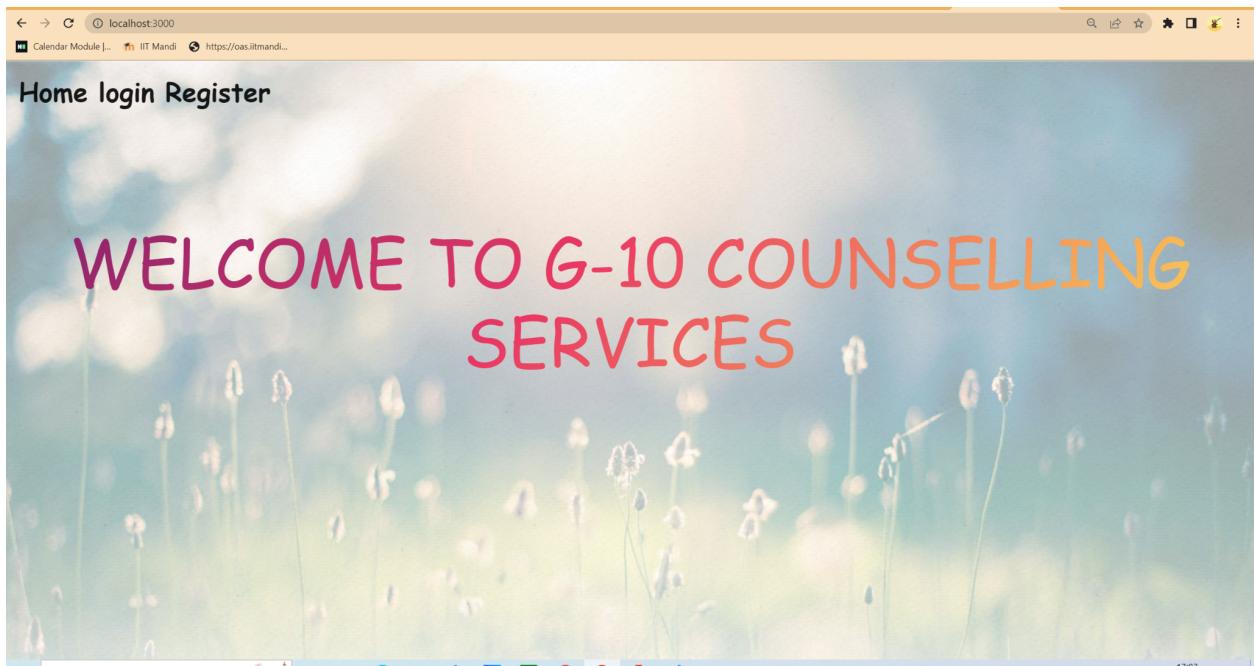
Then we are connecting MongoDB to NodeJS for creating databases. We are installing some NodeJS dependencies.



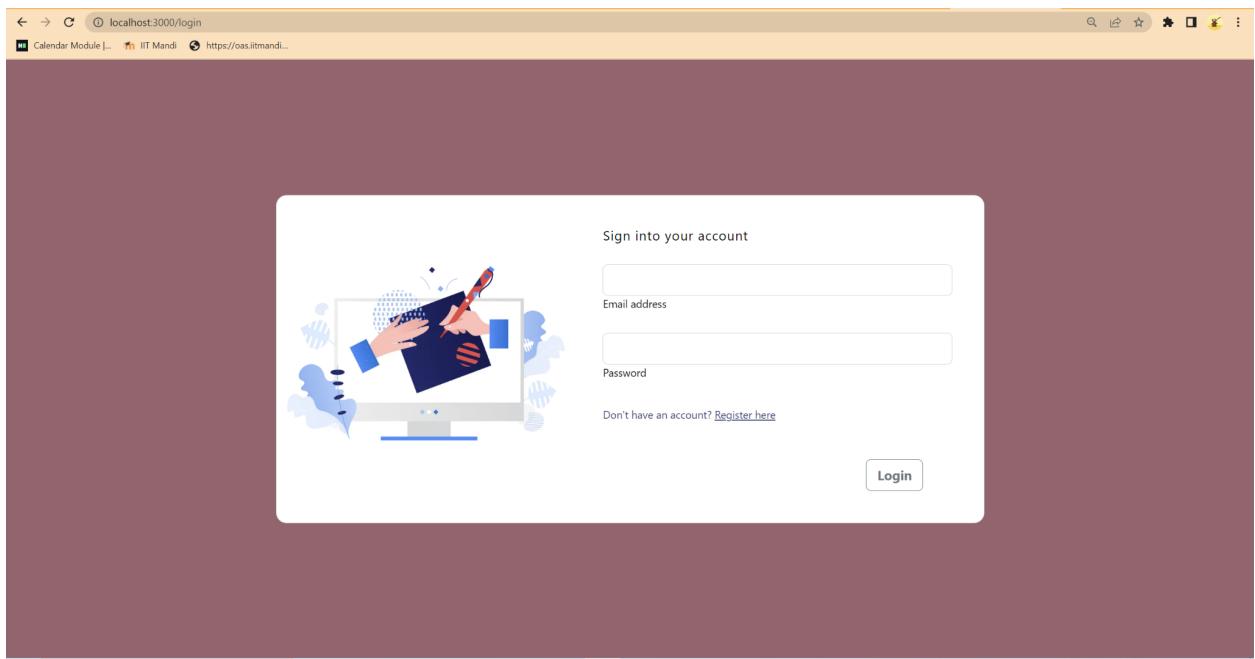
```
File Edit Selection View Go Run Terminal Help
EXPLORER OPEN EDITORS ...
src > db > JS conn.js > ...
src > db > JS conn.js src\db
src > db > JS conn.js src\db
src > ADF_LAST > ...
> node_modules
> partials
> public
> src
> db
> conn.js
> models
> app.js
> views
> main.js
{} package-lock.json
{} package.json
1 const mongoose = require("mongoose");
2
3 mongoose.connect("mongodb://localhost:27017/RegisterData",{
4
5   }).then(()=>{
6     console.log("connection sucessful");
7   }).catch((e)=>{
8     console.log(` ${e}`);
9   })
```

Overview / Result :

Homepage :



Login Page :



Registration Page :

localhost:3000/register

Calendar Module | IIT Mandi | https://oas.iitmandi...

REGISTRATION FORM

Firstname* Lastname

Phone no.* Age

Gender: Female Male Other

Email ID*

Password* ConfirmPassword*

already have an account? [Login here](#)

Register

Session viewing page :

Home Register

Counsler 8:00AM-10:00AM Booked	Counsler 8:00AM-10:00AM Book now	Counsler 8:00AM-10:00AM Book now	Counsler 8:00AM-10:00AM Book now	Counsler 8:00AM-10:00AM Book now
12:00PM-2:00PM Book now	12:00PM-2:00PM Booked	12:00PM-2:00PM Book now	12:00PM-2:00PM Book now	12:00PM-2:00PM Book now
4:00PM-6:00PM Book now				
Counsler 8:00AM-10:00AM Book now				
12:00PM-2:00PM Book now				
4:00PM-6:00PM Book now				

Conclusion :

Making a database system with an interactive interface is challenging for new learners. During pulling this project off we learnt a valuable lesson of teamwork and how to build up a complexly connected system of programmes . The course of CS 207 helped a lot in this regard . It's easy to lose track of progress in a big project . Here we managed to work effectively and efficiently . That's how we mould the session booking system to the final version.

I hope our work can compete with the best of session booking systems out there if not more efficiently.
