Web Technology Final Assignment Report

Project Name: Booking Class System

Author

Ruoyan CHi	Huiyu Ran	Yijing Zhu
ix19849	pj19174	gh19117
Login and display users' information.	Insert and delete data(staff、student、teacher part)	Show data(staff 、 student 、 teacher part)

Introduction

The system is designed for classes such as gyms, toastmasters and interest classes that need to be booked in advance. And it has three interfaces for student,teacher and staff. For Students, they can browse the available class hours, get information about the course's teachers, and book classes for the ones they're interested in. Meanwhile, the class which booked by the student can be canceled. For Teachers, they can give a brief introduction to themselves and fill in their available time for the staff to adjust and schedule. For the staff, they can schedule, publish, and modify course schedules and information.

Project Schedule

We have created two sides for the system including client side and server side.

client side

Use HTML, CSS and js to display the static wedpage

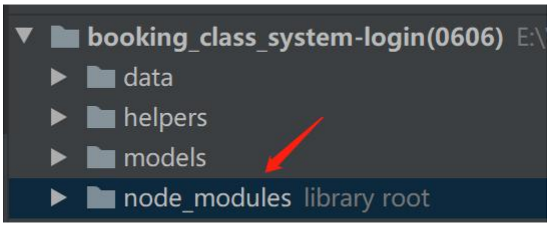
server side

start the server

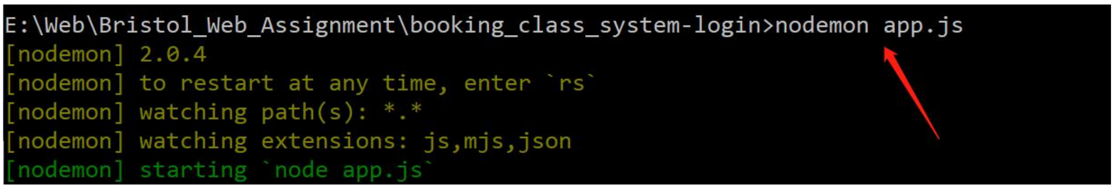
```
npm init -y
npm i -S express
npm install --save art-template npm install --save express-art-template
npm i -S bootstrap@3.3.7
npm install --save body-parser
```

nodemon app.js

(We have saved all the downloaded files into *booking_class_system-login/node_modules,*so you can input nodemon app.js directly after downloading.)



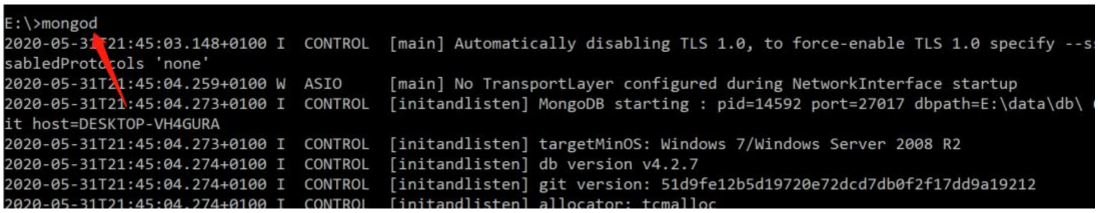
```
▼ booking_class_system-login(0606) E:\
  ▶ data
  ▶ helpers
  ▶ models
  ▶ node_modules library root
```



```
E:\Web\Bristol_Web_Assignment\booking_class_system-login>nodemon app.js
[nodemon] 2.0.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
```

start the mangodb

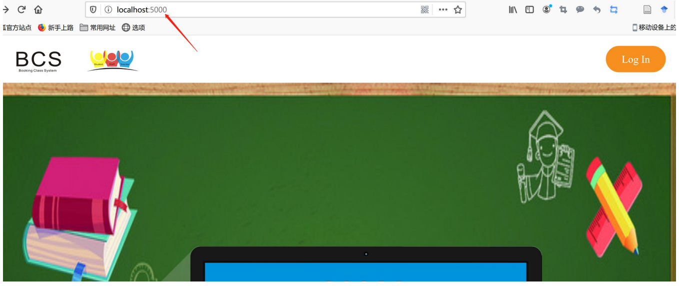
mongod



```
E:\>mongod
2020-05-31T21:45:03.148+0100 I  CONTROL  [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --s
sabledProtocols 'none'
2020-05-31T21:45:04.259+0100 W  ASIO     [main] No TransportLayer configured during NetworkInterface startup
2020-05-31T21:45:04.273+0100 I  CONTROL  [initandlisten] MongoDB starting : pid=14592 port=27017 dbpath=E:\data\db\
it host=DESKTOP-VH4GURA
2020-05-31T21:45:04.273+0100 I  CONTROL  [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2020-05-31T21:45:04.274+0100 I  CONTROL  [initandlisten] db version v4.2.7
2020-05-31T21:45:04.274+0100 I  CONTROL  [initandlisten] git version: 51d9fe12b5d19720e72dcd7db0f2f17dd9a19212
2020-05-31T21:45:04.274+0100 I  CONTROL  [initandlisten] allocator: tcmalloc
```

Open the browser

http://localhost:5000/



The system can only be used for one student, one teacher and one staff member at a time. During the test, a student, a teacher and a staff member should be registered, and the data will be stored in the database.

For example, using the following information to register, which include Email Address, Password, Full Name, Date of Birth, Phone Number and Status.

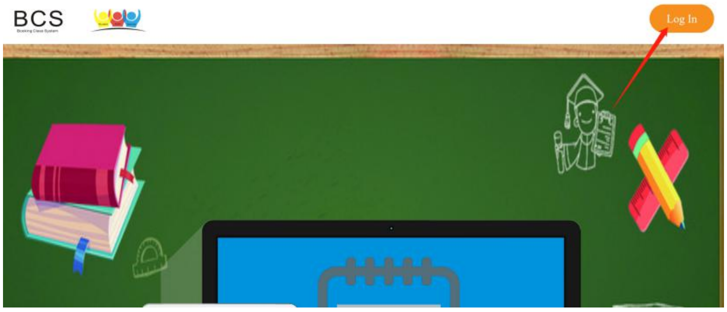
harry@student.com	**severus@teacher.com**	**albus@staff.com**
harry	serverus	albus
Harry Potter	Severus Snape	Albus Dumbledore
31/7/1980	9/1/1960	1/8/1881
7851892175	7917482487	7098472815

Use Mongodb to store the user information. In addition, these information will be shown in student-information page, teacher-information page and staff-information page.

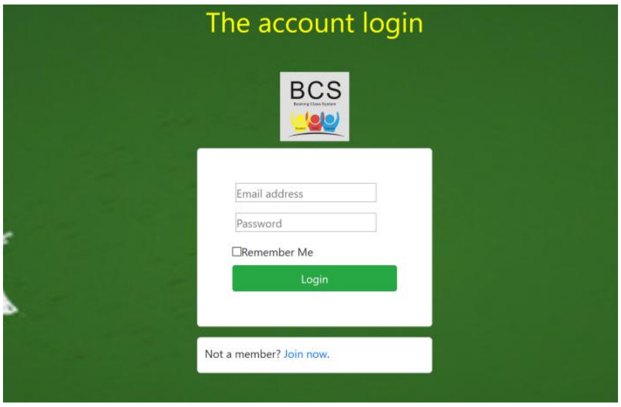
Page Introduction:

1. Login and Register

Click login to show login page.



Login page includes two parts: login function and register function.



The account login

BCS

Email address

Password

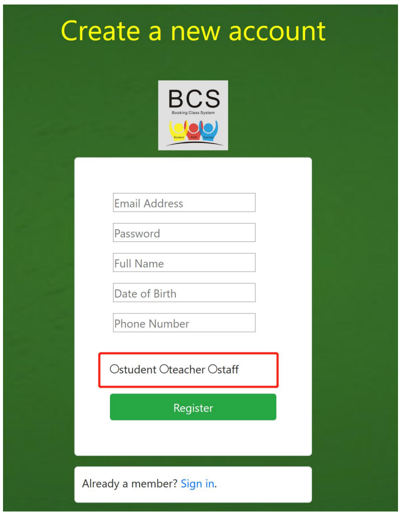
□Remember Me

Login

Not a member? Join now.

Click [Join now] to register an account.

There are three status available[student, teacher, staff]. Different status leads to corresponding page.



Create a new account

BCS

Email Address

Password

Full Name

Date of Birth

Phone Number

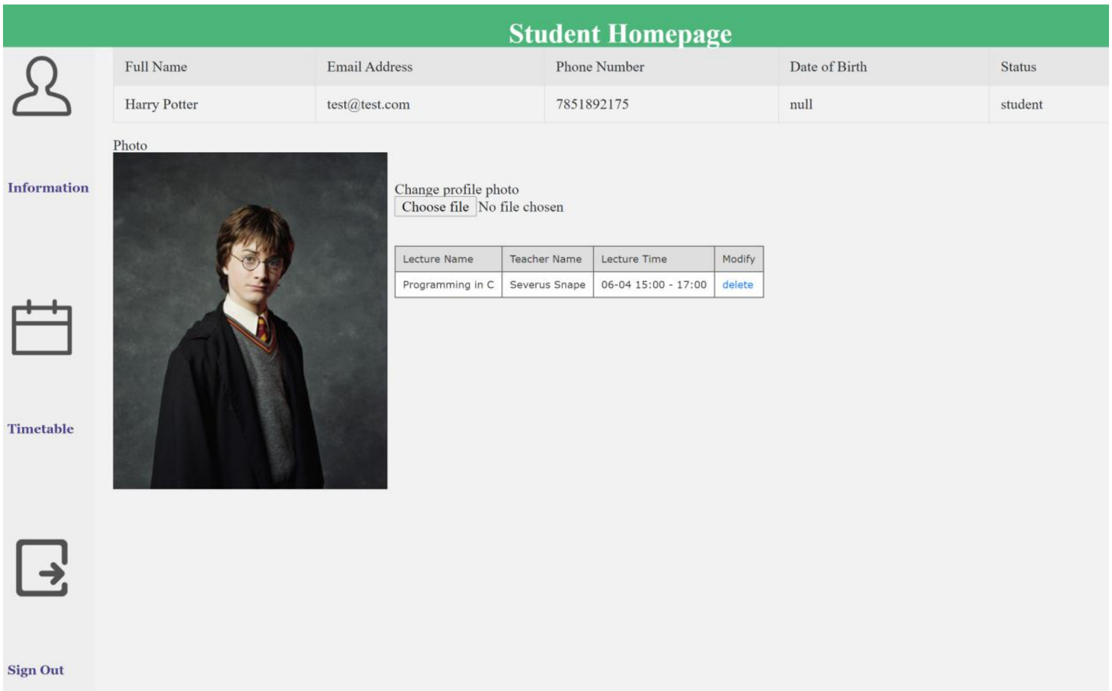
○student ○teacher ○staff

Register

Already a member? Sign in.

2. Student page

This is the Information part of the student page:



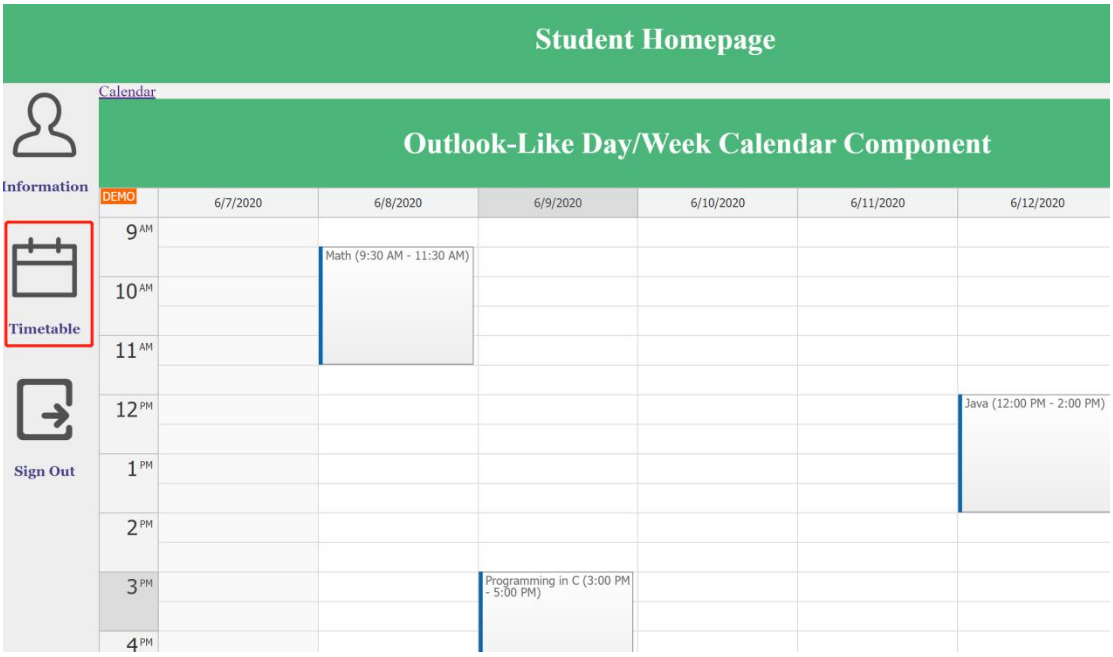
Full Name	Email Address	Phone Number	Date of Birth	Status
Harry Potter	test@test.com	7851892175	null	student

Lecture Name	Teacher Name	Lecture Time	Modify
Programming in C	Severus Snape	06-04 15:00 - 17:00	delete

In the top of this page shows the detail of student user(Full name, Email Address, Phone Number, Date of Birth,and Status).

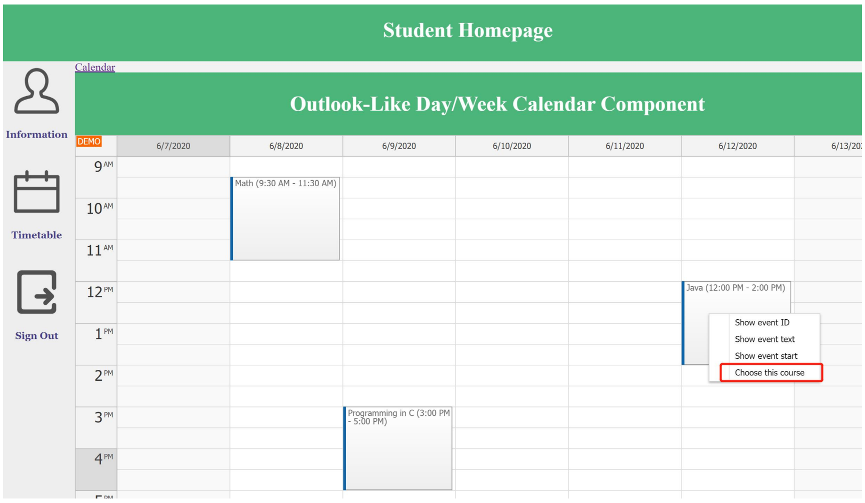
In the middle of this page shows the photo of student.User can click [choose file] button to change a new one. Below this button shows the lecture that user selected.

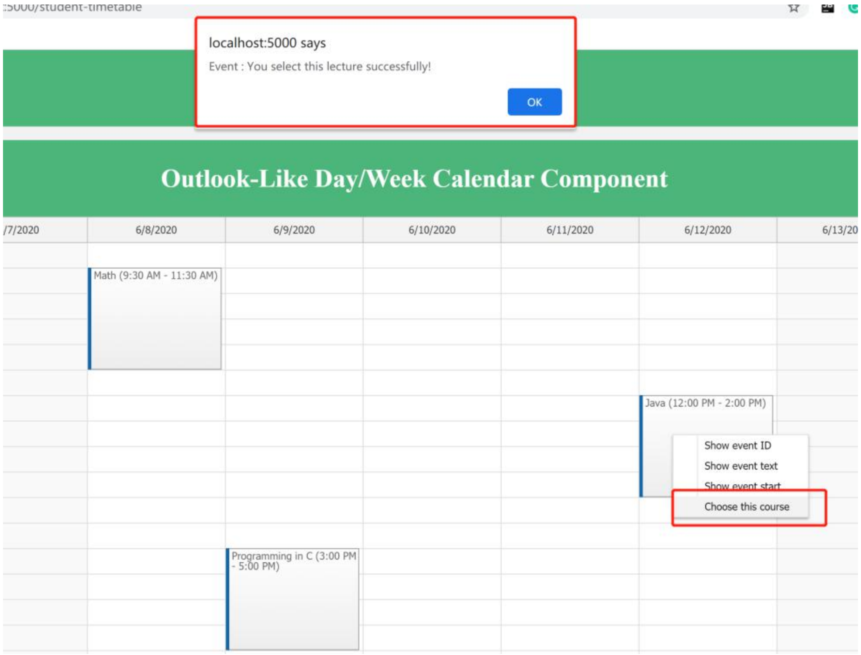
This is the Timetable part of the student page:



	6/7/2020	6/8/2020	6/9/2020	6/10/2020	6/11/2020	6/12/2020
9 AM		Math (9:30 AM - 11:30 AM)				
10 AM						
11 AM						
12 PM						Java (12:00 PM - 2:00 PM)
1 PM						
2 PM						
3 PM			Programming in C (3:00 PM - 5:00 PM)			
4 PM						

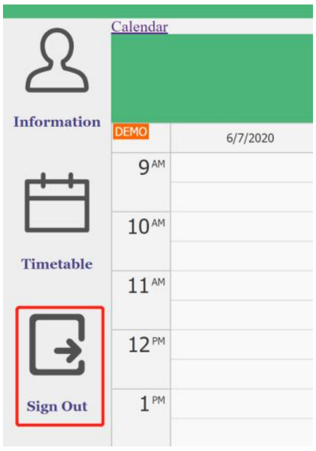
This page is a calendar showing the available lectures user can select.
User can click to select lecture.




localhost:5000 says

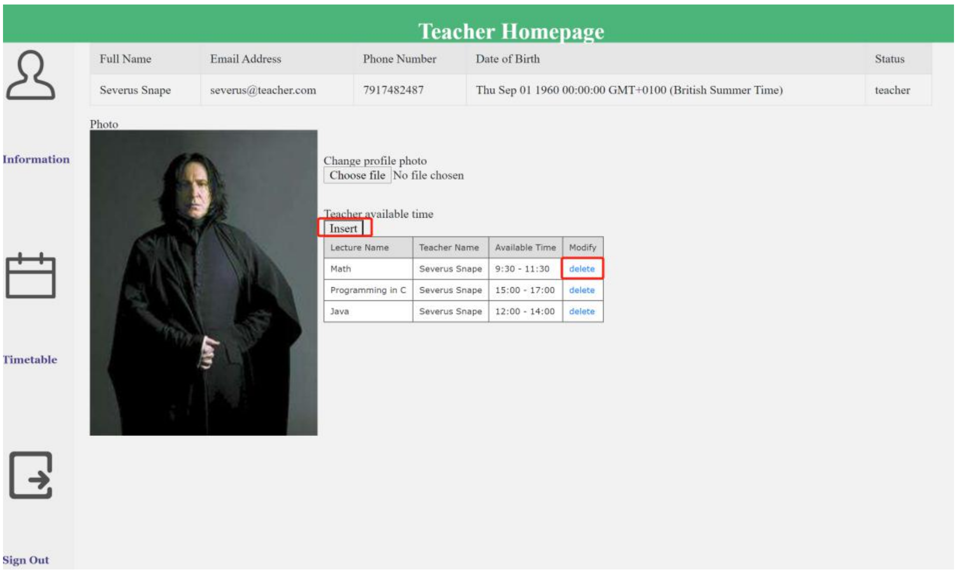
Event : You select this lecture successfully!

Click [sign out] button to log out.



3. Teacher Page

This is the Information part of the teacher page.



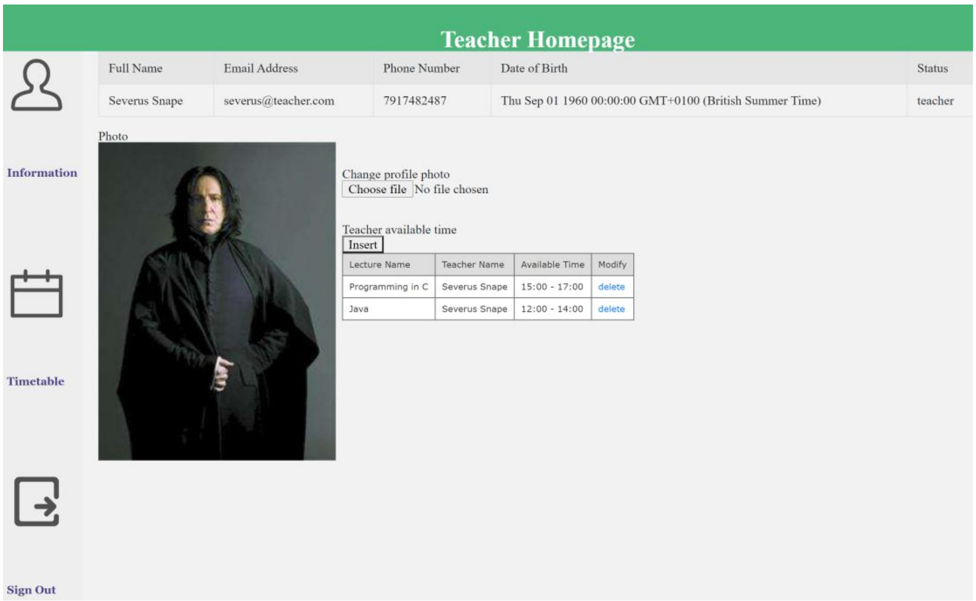
Teacher Homepage

Full Name	Email Address	Phone Number	Date of Birth	Status
Severus Snape	severus@teacher.com	7917482487	Thu Sep 01 1960 00:00:00 GMT+0100 (British Summer Time)	teacher

Teacher available time

Lecture Name	Teacher Name	Available Time	Modify
Math	Severus Snape	9:30 - 11:30	delete
Programming in C	Severus Snape	15:00 - 17:00	delete
Java	Severus Snape	12:00 - 14:00	delete

Teacher user can delete/insert his available time:



Teacher Homepage

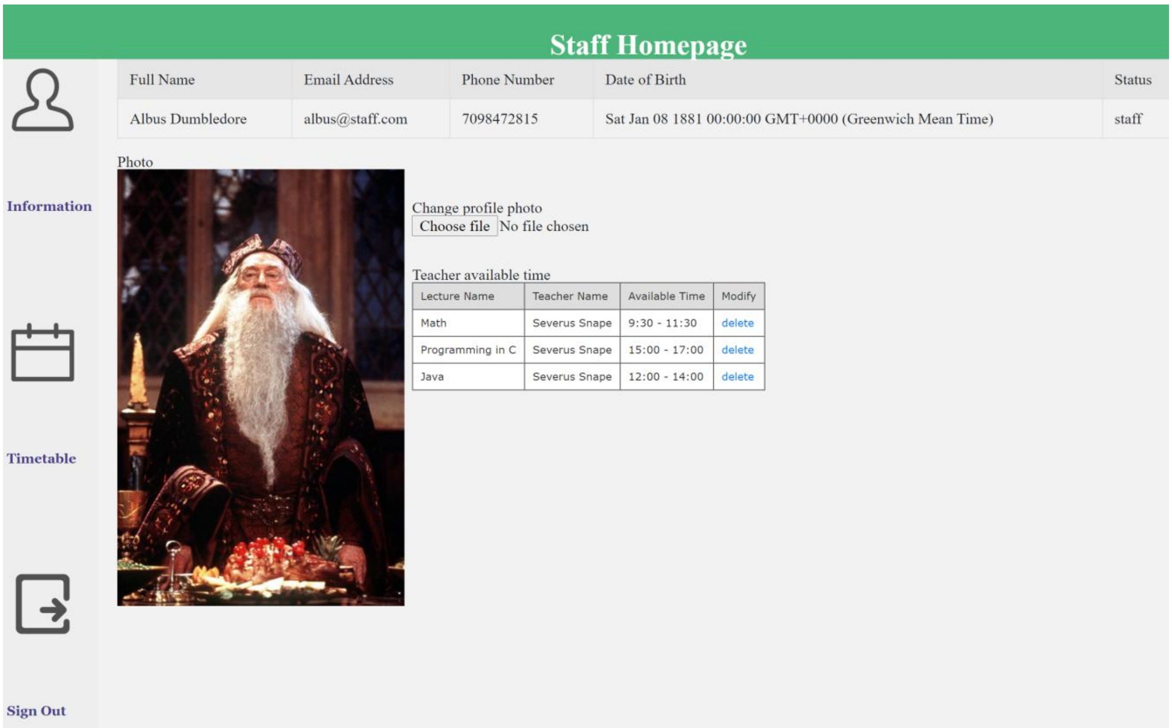
Full Name	Email Address	Phone Number	Date of Birth	Status
Severus Snape	severus@teacher.com	7917482487	Thu Sep 01 1960 00:00:00 GMT+0100 (British Summer Time)	teacher

Teacher available time

Lecture Name	Teacher Name	Available Time	Modify
Programming in C	Severus Snape	15:00 - 17:00	delete
Java	Severus Snape	12:00 - 14:00	delete

4. Staff Page

This is the Information part of the staff page.
Staff is able to schedule the lectures for teachers.



Staff Homepage

Full Name	Email Address	Phone Number	Date of Birth	Status
Albus Dumbledore	albus@staff.com	7098472815	Sat Jan 08 1881 00:00:00 GMT+0000 (Greenwich Mean Time)	staff

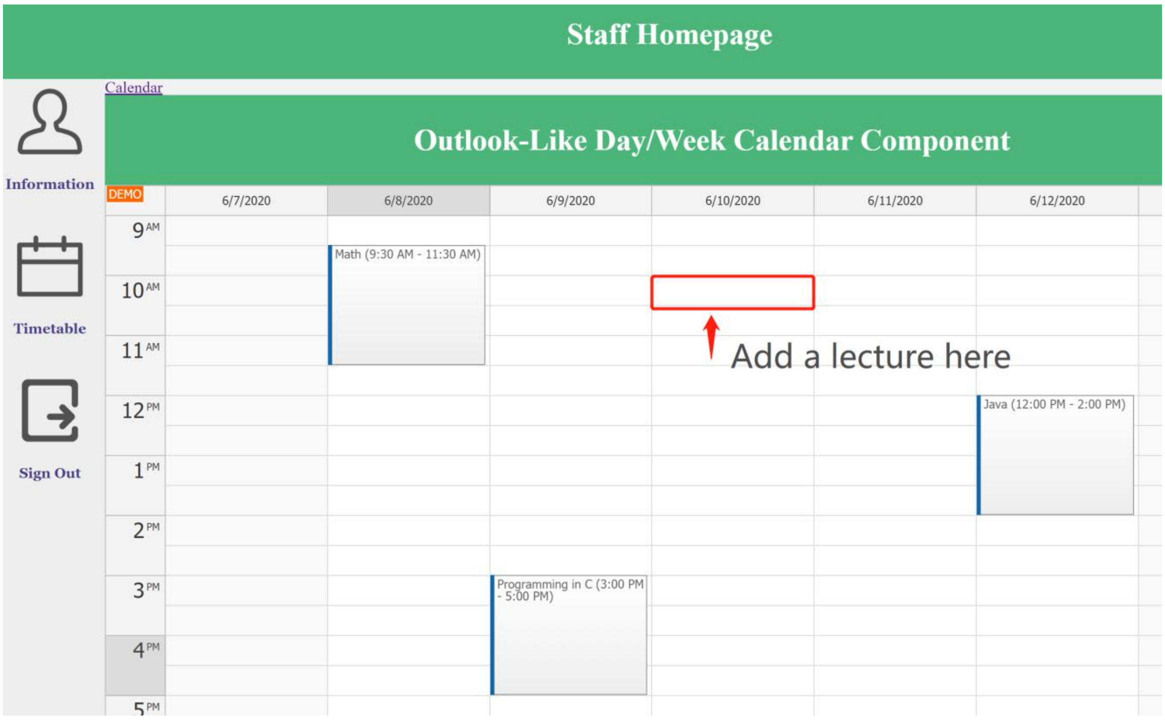
Photo

Change profile photo
Choose file No file chosen

Teacher available time

Lecture Name	Teacher Name	Available Time	Modify
Math	Severus Snape	9:30 - 11:30	delete
Programming in C	Severus Snape	15:00 - 17:00	delete
Java	Severus Snape	12:00 - 14:00	delete

This is the Timetable part of the staff page:

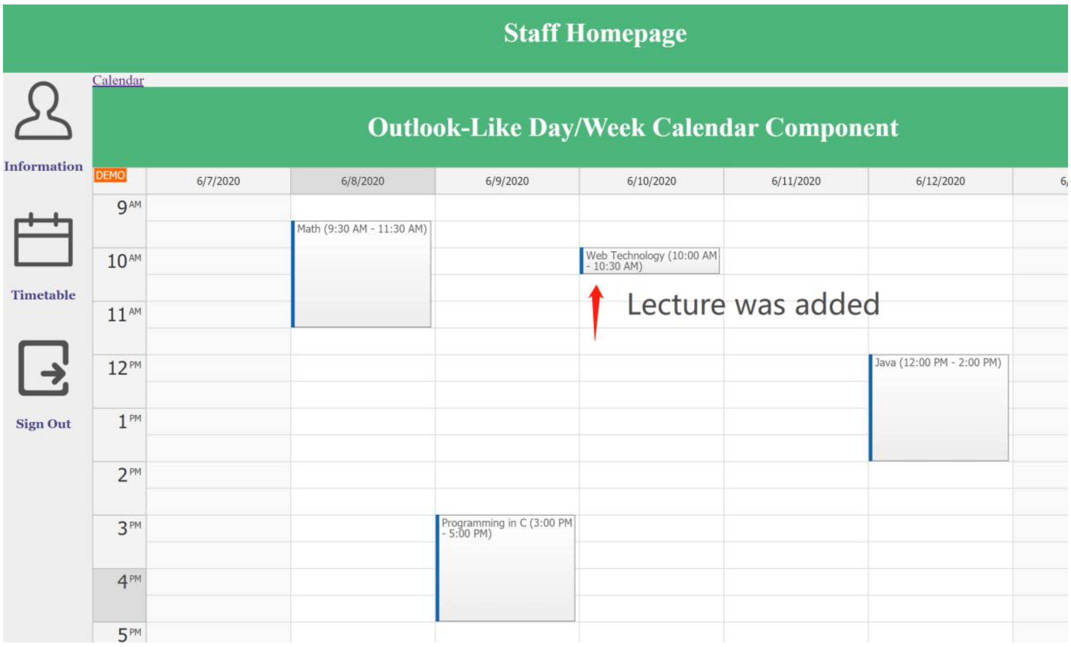


Staff Homepage

Calendar

Outlook-Like Day/Week Calendar Component

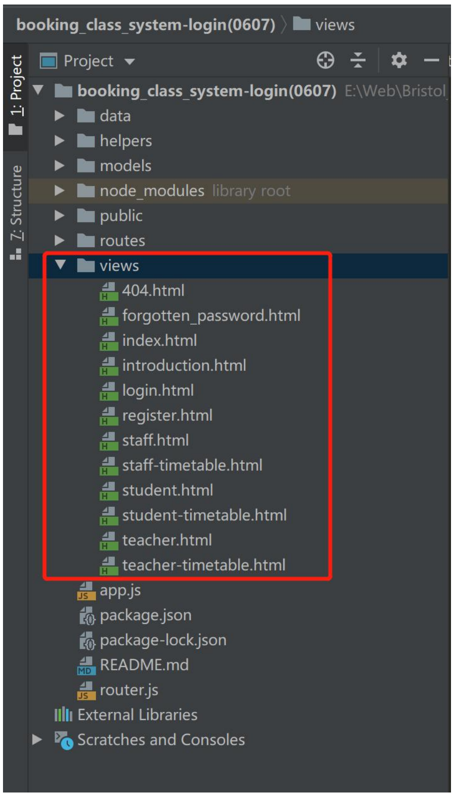
The added lecture will show in staff Timetable page as well as in related teacher page.


Staff Homepage
Calendar
Outlook-Like Day/Week Calendar Component
Information
Timetable
Sign Out
DEMO
6/7/2020
6/8/2020
6/9/2020
6/10/2020
6/11/2020
6/12/2020
9 AM
10 AM
11 AM
12 PM
1 PM
2 PM
3 PM
4 PM
5 PM
Math (9:30 AM - 11:30 AM)
Web Technology (10:00 AM - 10:30 AM)
Lecture was added
Java (12:00 PM - 2:00 PM)
Programming in C (3:00 PM - 5:00 PM)

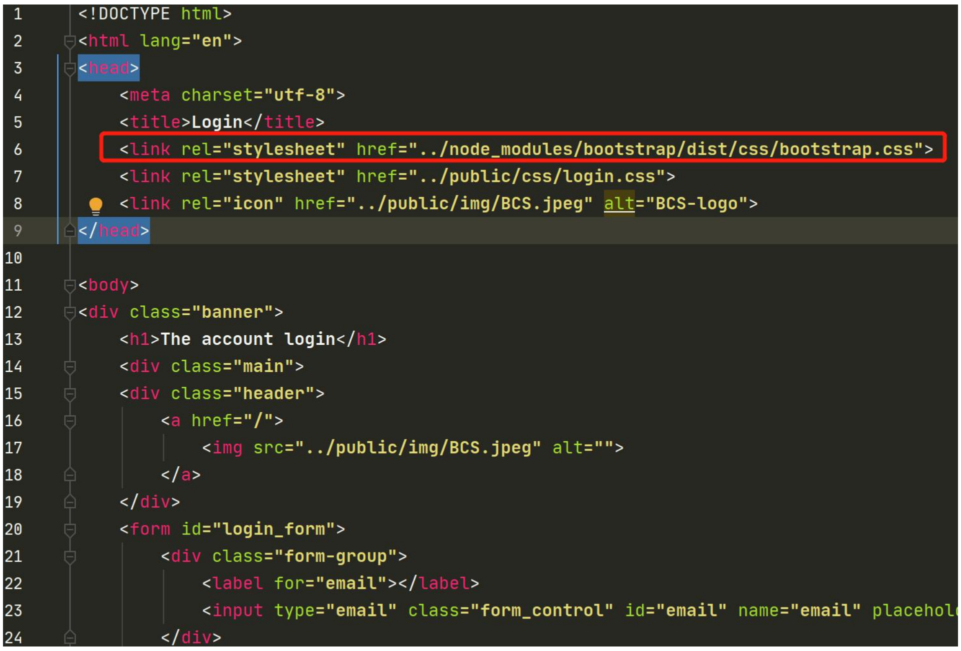
Code Introduction

HTML:

In the front-end, we introduce bootstrap to design our page.



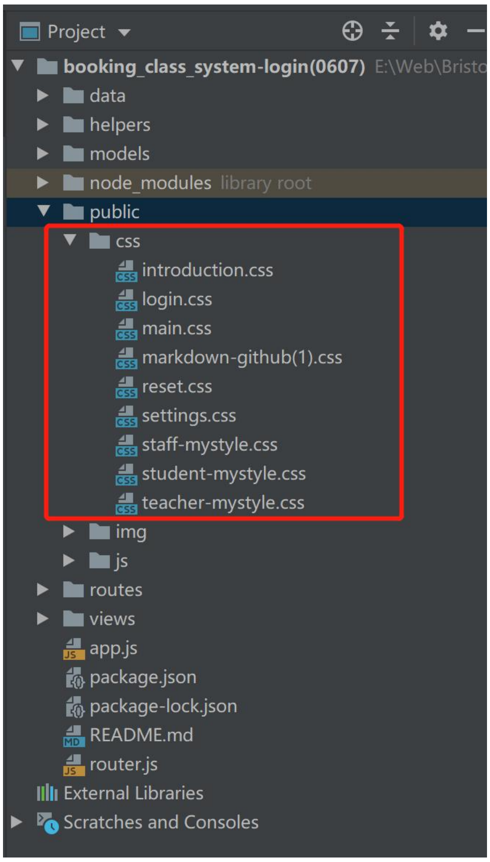
```
booking_class_system-login(0607) 〉 views
Project
▼ booking_class_system-login(0607)  E:\Web\Bristol
  ▶ data
  ▶ helpers
  ▶ models
  ▶ node_modules  library root
  ▶ public
  ▶ routes
  ▼ views
      404.html
      forgotten_password.html
      index.html
      introduction.html
      login.html
      register.html
      staff.html
      staff-timetable.html
      student.html
      student-timetable.html
      teacher.html
      teacher-timetable.html
    app.js
    package.json
    package-lock.json
    README.md
    router.js
  External Libraries
▶ Scratches and Consoles
```



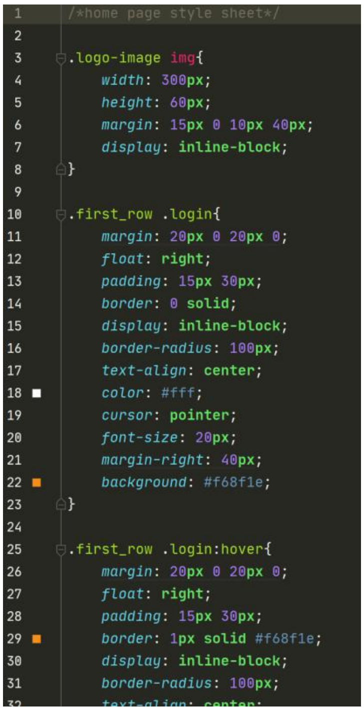
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>Login</title>
    <link rel="stylesheet" href="../node_modules/bootstrap/dist/css/bootstrap.css">
    <link rel="stylesheet" href="../public/css/login.css">
    <link rel="icon" href="../public/img/BCS.jpeg" alt="BCS-logo">
</head>

<body>
<div class="banner">
    <h1>The account login</h1>
    <div class="main">
    <div class="header">
        <a href="/">
            <img src="../public/img/BCS.jpeg" alt="">
        </a>
    </div>
    <form id="login_form">
        <div class="form-group">
            <label for="email"></label>
            <input type="email" class="form_control" id="email" name="email" placehol
        </div>
```

CSS:

We use CSS file to design the style of our html code.



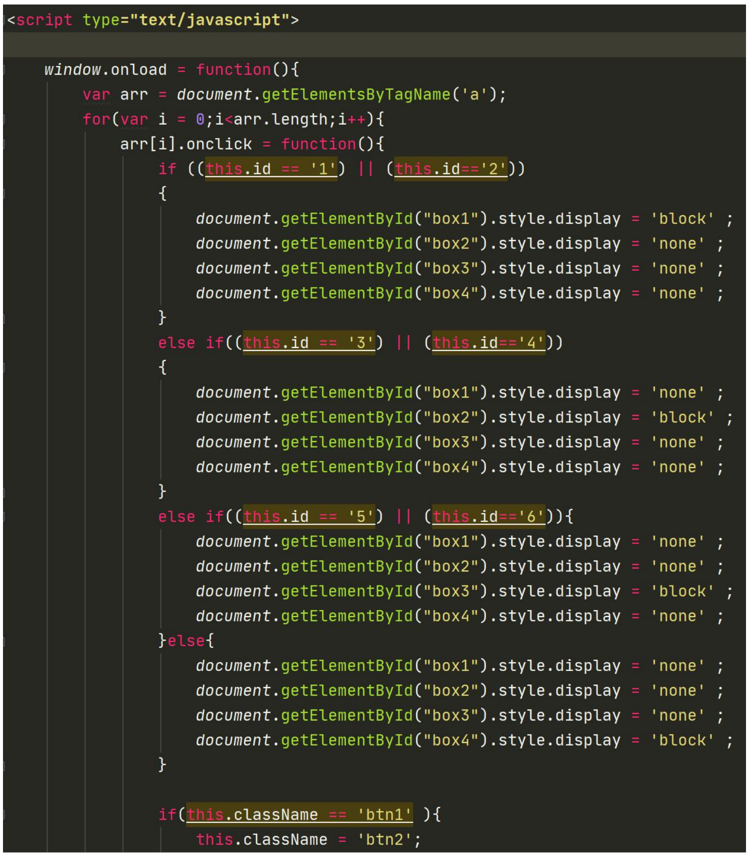
```
Project
booking_class_system-login(0607) E:\Web\Bristo
    data
    helpers
    models
    node_modules library root
    public
        css
            introduction.css
            login.css
            main.css
            markdown-github(1).css
            reset.css
            settings.css
            staff-mystyle.css
            student-mystyle.css
            teacher-mystyle.css
        img
        js
    routes
    views
    app.js
    package.json
    package-lock.json
    README.md
    router.js
External Libraries
Scratches and Consoles
```



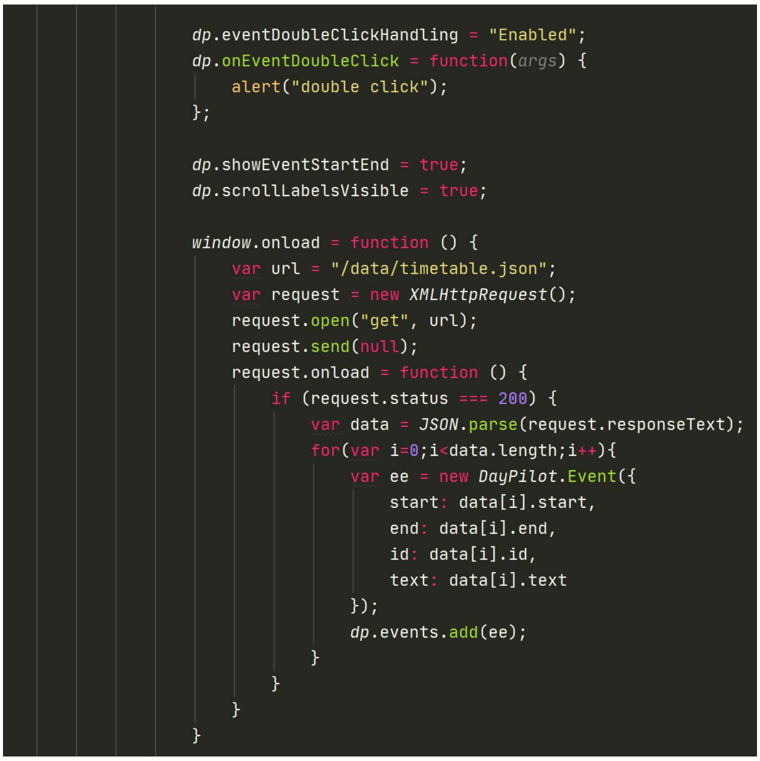
```css
/*home page style sheet*/

.logo-image img{
    width: 300px;
    height: 60px;
    margin: 15px 0 10px 40px;
    display: inline-block;
}

.first_row .login{
    margin: 20px 0 20px 0;
    float: right;
    padding: 15px 30px;
    border: 0 solid;
    display: inline-block;
    border-radius: 100px;
    text-align: center;
    color: #fff;
    cursor: pointer;
    font-size: 20px;
    margin-right: 40px;
    background: #f68f1e;
}

.first_row .login:hover{
    margin: 20px 0 20px 0;
    float: right;
    padding: 15px 30px;
    border: 1px solid #f68f1e;
    display: inline-block;
    border-radius: 100px;
    text-align: center;
```

Javascript:

We use javascript to realize the shift of different section in one page and the design of our calendar:



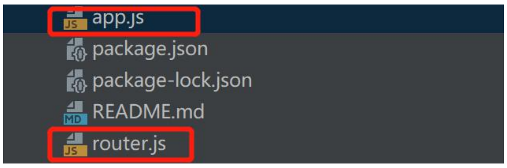
```javascript
<script type="text/javascript">

    window.onload = function(){
        var arr = document.getElementsByTagName('a');
        for(var i = 0;i<arr.length;i++){
            arr[i].onclick = function(){
                if ((this.id == '1') || (this.id=='2'))
                {
                    document.getElementById("box1").style.display = 'block' ;
                    document.getElementById("box2").style.display = 'none' ;
                    document.getElementById("box3").style.display = 'none' ;
                    document.getElementById("box4").style.display = 'none' ;
                }
                else if((this.id == '3') || (this.id=='4'))
                {
                    document.getElementById("box1").style.display = 'none' ;
                    document.getElementById("box2").style.display = 'block' ;
                    document.getElementById("box3").style.display = 'none' ;
                    document.getElementById("box4").style.display = 'none' ;
                }
                else if((this.id == '5') || (this.id=='6')){
                    document.getElementById("box1").style.display = 'none' ;
                    document.getElementById("box2").style.display = 'none' ;
                    document.getElementById("box3").style.display = 'block' ;
                    document.getElementById("box4").style.display = 'none' ;
                }else{
                    document.getElementById("box1").style.display = 'none' ;
                    document.getElementById("box2").style.display = 'none' ;
                    document.getElementById("box3").style.display = 'none' ;
                    document.getElementById("box4").style.display = 'block' ;
                }

                if(this.className == 'btn1' ){
                    this.className = 'btn2';
```



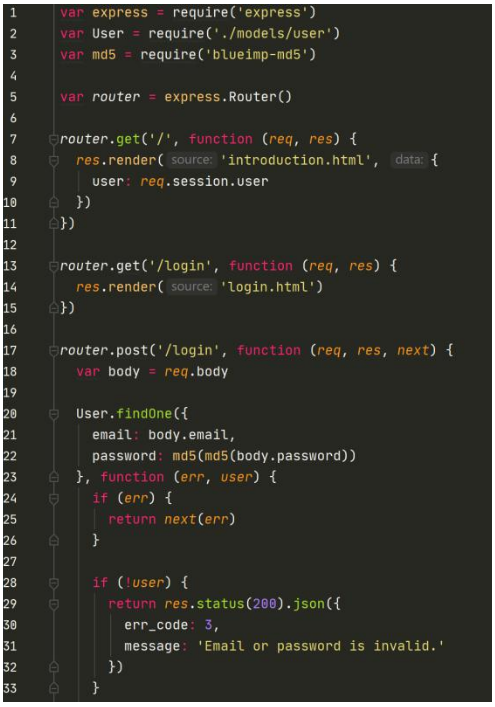
```
dp.eventDoubleClickHandling = "Enabled";
dp.onEventDoubleClick = function(args) {
    alert("double click");
};

dp.showEventStartEnd = true;
dp.scrollLabelsVisible = true;

window.onload = function () {
    var url = "/data/timetable.json";
    var request = new XMLHttpRequest();
    request.open("get", url);
    request.send(null);
    request.onload = function () {
        if (request.status === 200) {
            var data = JSON.parse(request.responseText);
            for(var i=0;i<data.length;i++){
                var ee = new DayPilot.Event({
                    start: data[i].start,
                    end: data[i].end,
                    id: data[i].id,
                    text: data[i].text
                });
                dp.events.add(ee);
            }
        }
    }
}
```

Node.js

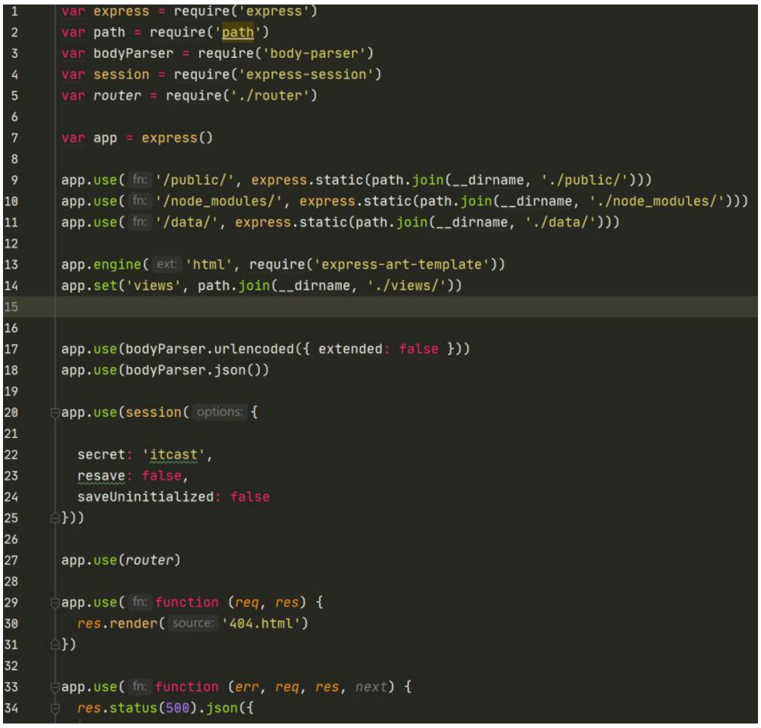
We use node.js to realize the shift of different pages and the getting/posting of data.



app.js
package.json
package-lock.json
README.md
router.js



```
var express = require('express')
var User = require('./models/user')
var md5 = require('blueimp-md5')

var router = express.Router()

router.get('/', function (req, res) {
  res.render( source: 'introduction.html',  data: {
    user: req.session.user
  })
})

router.get('/login', function (req, res) {
  res.render( source: 'login.html')
})

router.post('/login', function (req, res, next) {
  var body = req.body

  User.findOne({
    email: body.email,
    password: md5(md5(body.password))
  }, function (err, user) {
    if (err) {
      return next(err)
    }

    if (!user) {
      return res.status(200).json({
        err_code: 3,
        message: 'Email or password is invalid.'
      })
    }
```

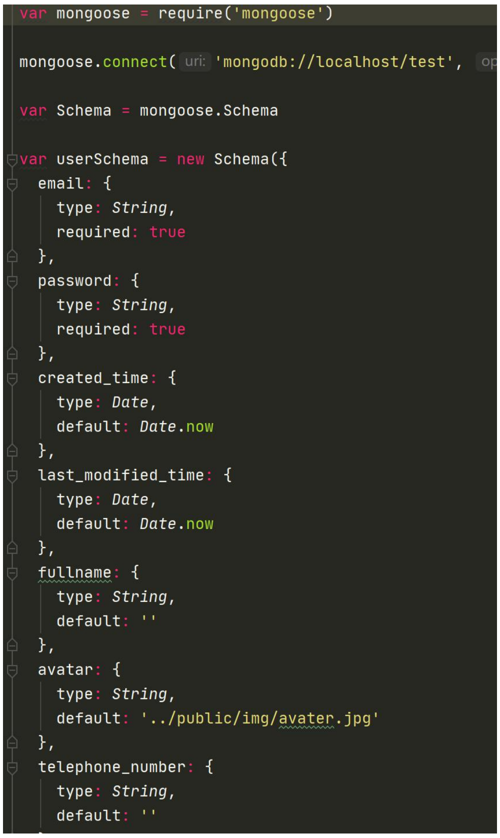


```
var express = require('express')
var path = require('path')
var bodyParser = require('body-parser')
var session = require('express-session')
var router = require('./router')

var app = express()

app.use( fn: '/public/', express.static(path.join(__dirname, './public/')))
app.use( fn: '/node_modules/', express.static(path.join(__dirname, './node_modules/')))
app.use( fn: '/data/', express.static(path.join(__dirname, './data/')))

app.engine( ext: 'html', require('express-art-template'))
app.set('views', path.join(__dirname, './views/'))


app.use(bodyParser.urlencoded({ extended: false }))
app.use(bodyParser.json())

app.use(session( options: {

  secret: 'itcast',
  resave: false,
  saveUninitialized: false
}))

app.use(router)

app.use( fn: function (req, res) {
  res.render( source: '404.html')
})

app.use( fn: function (err, req, res, next) {
  res.status(500).json({
```

Database

Mongodb

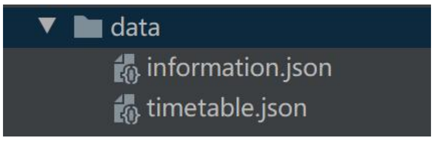
We use mongodb to store user’s details.

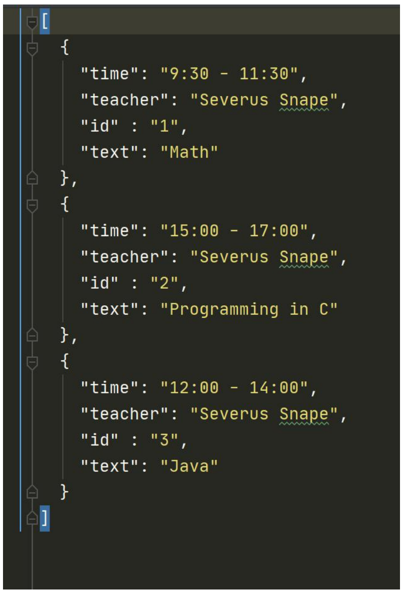


```
var mongoose = require('mongoose')

mongoose.connect( uri: 'mongodb://localhost/test',  op

var Schema = mongoose.Schema

var userSchema = new Schema({
  email: {
    type: String,
    required: true
  },
  password: {
    type: String,
    required: true
  },
  created_time: {
    type: Date,
    default: Date.now
  },
  last_modified_time: {
    type: Date,
    default: Date.now
  },
  fullname: {
    type: String,
    default: ''
  },
  avatar: {
    type: String,
    default: '../public/img/avater.jpg'
  },
  telephone_number: {
    type: String,
    default: ''
```

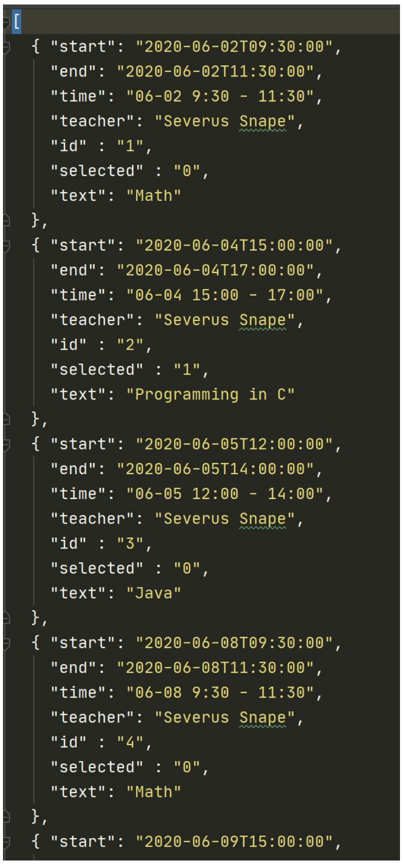
JSON

We use json file to store the data of timetable.





```
[
  {
    "time": "9:30 - 11:30",
    "teacher": "Severus Snape",
    "id" : "1",
    "text": "Math"
  },
  {
    "time": "15:00 - 17:00",
    "teacher": "Severus Snape",
    "id" : "2",
    "text": "Programming in C"
  },
  {
    "time": "12:00 - 14:00",
    "teacher": "Severus Snape",
    "id" : "3",
    "text": "Java"
  }
]
```



```
[
  { "start": "2020-06-02T09:30:00",
    "end": "2020-06-02T11:30:00",
    "time": "06-02 9:30 - 11:30",
    "teacher": "Severus Snape",
    "id" : "1",
    "selected" : "0",
    "text": "Math"
  },
  { "start": "2020-06-04T15:00:00",
    "end": "2020-06-04T17:00:00",
    "time": "06-04 15:00 - 17:00",
    "teacher": "Severus Snape",
    "id" : "2",
    "selected" : "1",
    "text": "Programming in C"
  },
  { "start": "2020-06-05T12:00:00",
    "end": "2020-06-05T14:00:00",
    "time": "06-05 12:00 - 14:00",
    "teacher": "Severus Snape",
    "id" : "3",
    "selected" : "0",
    "text": "Java"
  },
  { "start": "2020-06-08T09:30:00",
    "end": "2020-06-08T11:30:00",
    "time": "06-08 9:30 - 11:30",
    "teacher": "Severus Snape",
    "id" : "4",
    "selected" : "0",
    "text": "Math"
  },
  { "start": "2020-06-09T15:00:00",
```

Project Evaluation

1. Self Evaluation

A for HTML
A for CSS
A for JS
A for node.js
A for Database

2. Project Success

Measuring the success of a project once it’s brought to completion is a valuable practice. It provides a learning opportunity for future undertakings, and, the opportunity to assess the true effectiveness of the project. In order to have a holistic view, objective and subjective criteria need to be considered.

(1)Scope:
Our project has achieved its objectives (develop a booking class System) within the limited programming languages.

(2)Schedule:
Our team completed the project on time. We have three iteration during two months. We finished this project in 7th,June (Deadline: 10th, June).

(3)Team satisfaction:
Our team leader give us comprehensive information about project goal and support our individual and team motivation. we set and communicate clear goals for all of our discussions and consistently follow those goals. we respect each other personally and in terms of expertise. we listen to each other by clarifying unclear points and asking appropriate questions during this development. we include all necessary team members in the decision making processes.

3. Communication

Affected by this special situation of COVID-19, it’s impossible for us to have a face-to-face meeting since April. We did our best to keep in touch online weekly. Wechat is our main communication tool, on which we made discussion, assigned tasks and reported individual task progress.

4. Code Sharing

Github is a good platform for storing and sharing code. Our github link is

https://github.com/cry0324/booking_class_system/tree/login

5. How Coronavirus has affected our Project?

Owing to the Corona Virus Disease 2019 (COVID 19), we cancelled our face-to-face meeting and one of our team member went back to China in April, there are seven hours time difference between us,which has affected the development and implementation of our projects. We cannot communicate with each other deeply, which cause the different understanding on some parts(e.g.,the data insert and delete part). So we have to extend our ddl and we finally completed this website.