**CS387 Project: Railway Reservation System**

**Team:**

Aanya Verma – 200050001

 Pragati Gupta – 200050063

Gracy Paul – 200050079

 Yasaswi Nannapaneni – 200050085

 Sirigudi Meghana – 200050135.

This project is about the Train-Ticket-Reservation-System, which is used to view Train Schedule, search trains, Seat availability, Train timings. We can get information about train between two stations. We can book seats online. This provides a safe and secure seat reservation system.

**Users:**

Admin: Adds new trains, updates old train route, timings, no. of seats, etc. Cancels trains. Has higher privilege levels than public.

Customer (Public): Enquires about train routes, timings, seat availability. Books tickets. Checks PNR status.

**Model:**

**userProfile:**

userName: {type: String, required: true},

gender: {type: String},

email: {type: String,required: true},

createPassword: {type: String, required: true},

confirmPassword: {type: String, required: true},

termsAndConditions: {type: Boolean, required: true},

signupDate: {type: Date, default: Date.now},

isUserExist: {type: Boolean}

**adminProfile:**

userName: {type: String, required: true},

gender: {type: String},

email: {type: String, required: true},

createPassword: {type: String, required: true},

confirmPassword: {type: String, required: true},

termsAndConditions: {type: Boolean, required: true},

signupDate: {type: Date, default: Date.now},

isUserExist: {type: Boolean}

**Train:**

trainNumber: {type: Number, unique: true, required: true},

trainName: {type: String, unique: true, required: true},

from: {type: String, uppercase: true, required: true},

to: {type: String, uppercase: true, required: true},

totalSeat: {type: Number, required: true},

fair: {type: Number, required: true}

**Ticket:**

trainName: {type: String, required: true},

trainNumber: {type: Number, required: true},

from: {type: String, required: true},

to: {type: String, required: true},

fair: {type: Number, required: true},

passengerName: {required: true},

passengerGender: {type: String, required: true},

user: {type: Schema.Types.ObjectId, ref: 'user'}

**Application structure:**

**Backend:**

User related:

router.post('/signUp', services.signUp);

router.get('/authentication', services.authentication);

router.get('/getMyProfile/:userId', services.getMyProfile);

router.get('/currentUser/:email/:password', services.currentUser);

router.get('/searchTrain', services.deleteTrain);

router.get('/bookTrain/:trainNumber', services.bookTrain);

router.post('/confirmTicket/:userId', services.confirmTicket);

router.get('/myTicket/:userId', services.myTicket);

Admin related:

router.post('/adminSignUp', services.AdminSignUp);

router.get('/adminAuthentication', services.adminAuthentication);

router.get('/getMyAdminProfile/:userId', services.getMyAdminProfile);

router.get('/currentAdmin/:email/:password', services.currentAdmin);

router.post('/createTrain', services.createTrain);

router.get('/getStation', services.getStation);

router.get('/listTrain', services.listTrain);

router.get('/editTrain/:id', services.editTrain);

router.post('/updateTrain/:id', services.updateTrain);

router.get('/deleteTrain/:id', services.deleteTrain);

router.get('/resultTrain/:from/:to', services.resultTrain);

**Implemented:**

* Admin and User (public) role for users.
* Authentication: Sign in and Sign Up, with Captcha security
* Train Search using initial and destination station as input
* User (public) can book a train, check their train status
* Admin can add, update, and delete train along with other privileges

**Frontend Structure:**

* Homepage: admin, user
* Loginpage: sign in (username, password, captcha), sing up
* Userpage: Search Train, Book Train, Check Bookings
* Adminpage: Search Train, Update button which opens form to edit information

**Features that can be implemented:**

* Booking status: confirm/waitlist
* Seats for different classes and fair, and payment for booking (abstract)
* Cancel Ticket privilege for users
* Train routes and schedule

**Learnt:**

* Using mongoose model with NodeJS. Flexibility of MongoDB (NoSQL)
* Modularity of ReactJS and NodeJS.