

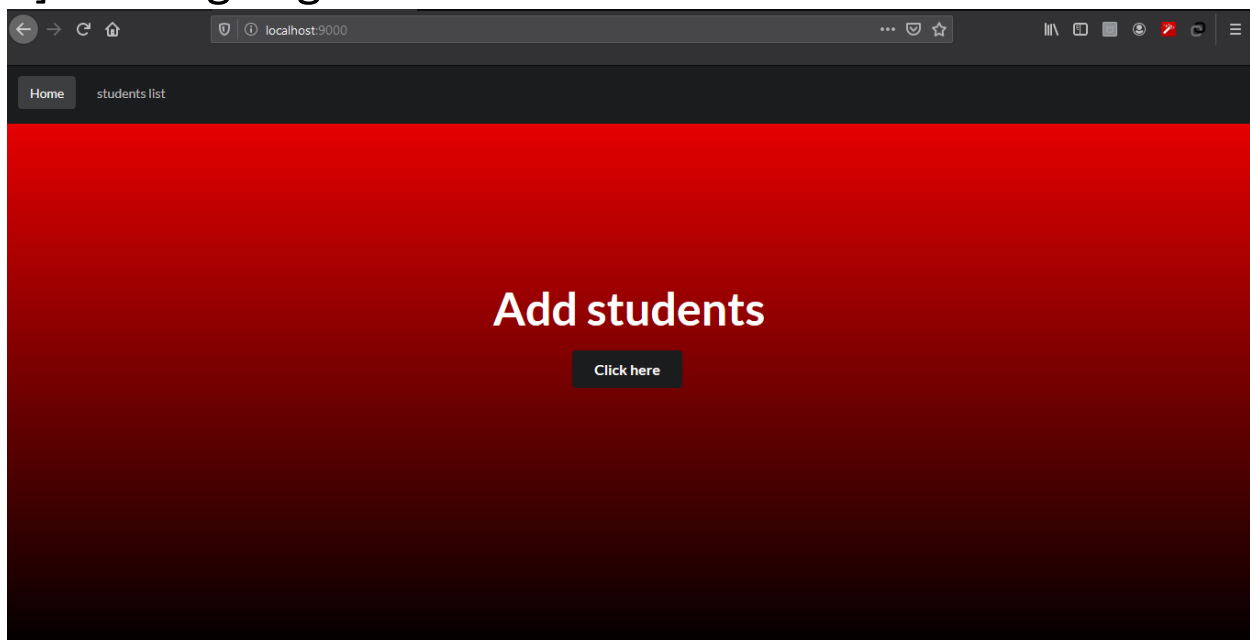
# Think201 challenge submission (Student entry web app)

I have built a web app as per the challenge requirement and in this pdf document there are two parts, part1 explaining the functionality of web app and frontend. Part 2 which consist of logic explanation (Backend functionality), The tech stack used-(HTML, CSS, JavaScript, Node.js, Express.js, Semantic-ui, Mongo dB)

## Part-1

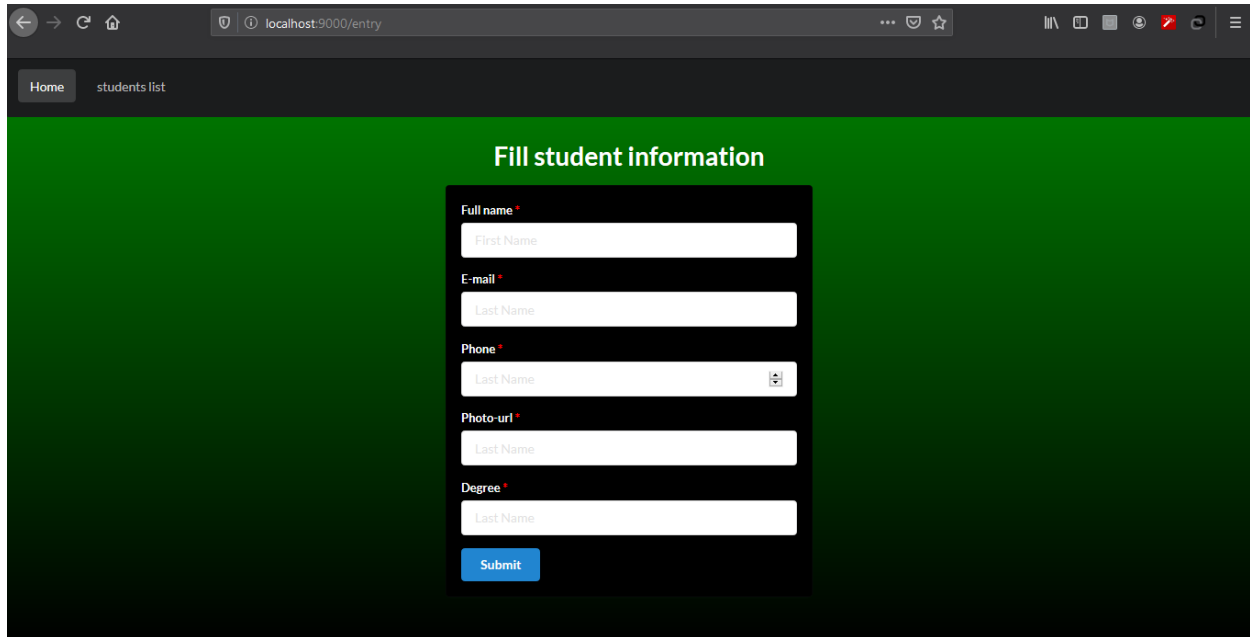
### Functionality of the web app (Frontend)

#### 1] Landing Page



The landing page consist of nav-bar with home, students list options and then there is an add student button which on clicking will redirect the user to the form where they will add student details.

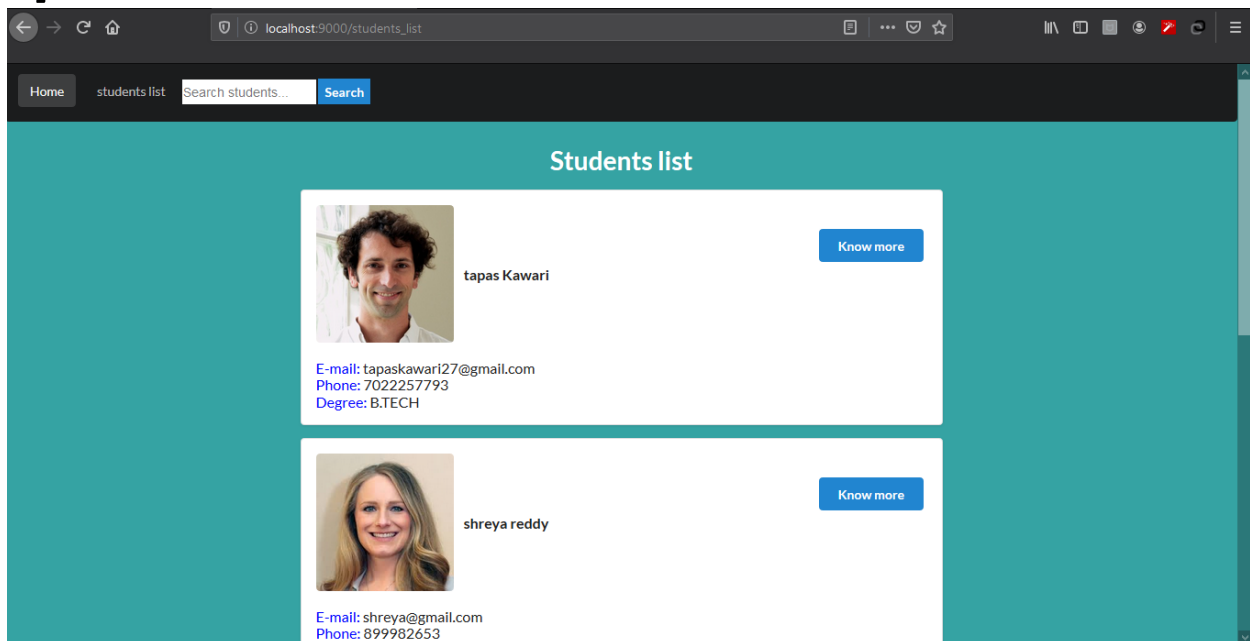
## 2] Form to fill student information



The screenshot shows a web browser at localhost:9000/entry. The page has a dark header with 'Home' and 'students list' links. The main content area has a green background with the title 'Fill student information'. A black form box contains the following fields: 'Full name' (with a sub-field 'First Name'), 'E-mail' (with a sub-field 'Last Name'), 'Phone' (with a sub-field 'Last Name' and a dropdown arrow), 'Photo-url' (with a sub-field 'Last Name'), and 'Degree' (with a sub-field 'Last Name'). A blue 'Submit' button is at the bottom of the form.

This page contains a form where user can fill student information it consist input fields for Name, Email, Phone, Photo-url, Degree. And once the user submits the form he will be redirected to students list page.

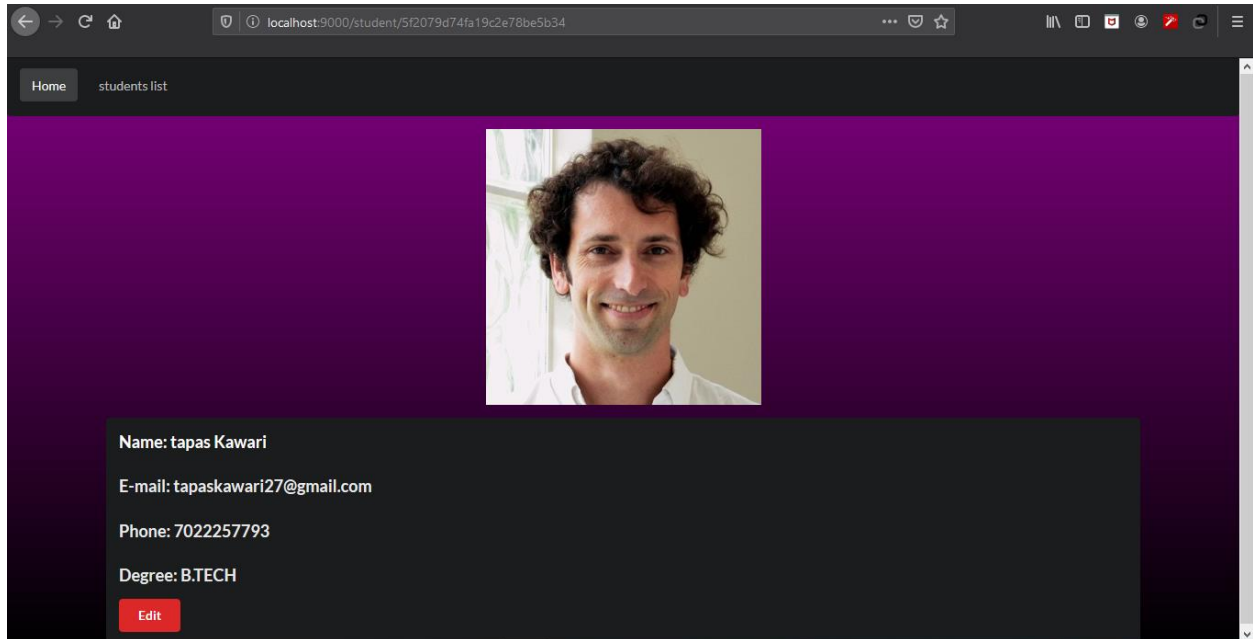
## 3] Students list



The screenshot shows a web browser at localhost:9000/students\_list. The page has a dark header with 'Home' and 'students list' links, and a search bar with the text 'Search students...' and a 'Search' button. The main content area has a teal background with the title 'Students list'. It displays two student profiles in white boxes. The first profile is for 'tapas Kawari' with a photo, email 'tapaskawari27@gmail.com', phone '7022257793', and degree 'B.TECH'. The second profile is for 'shreya reddy' with a photo, email 'shreya@gmail.com', and phone '899982653'. Both profiles have a blue 'Know more' button.

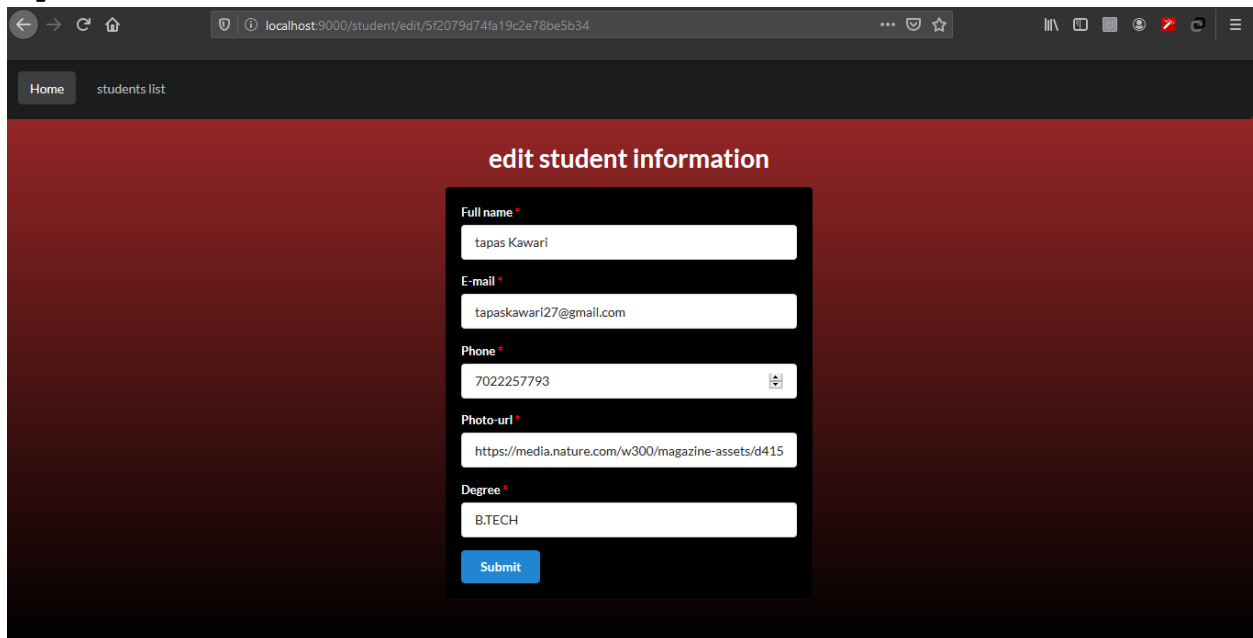
This page shows the information of all the available students in the database and we have know more button which will redirect to individual student details page

## 4] individual student info



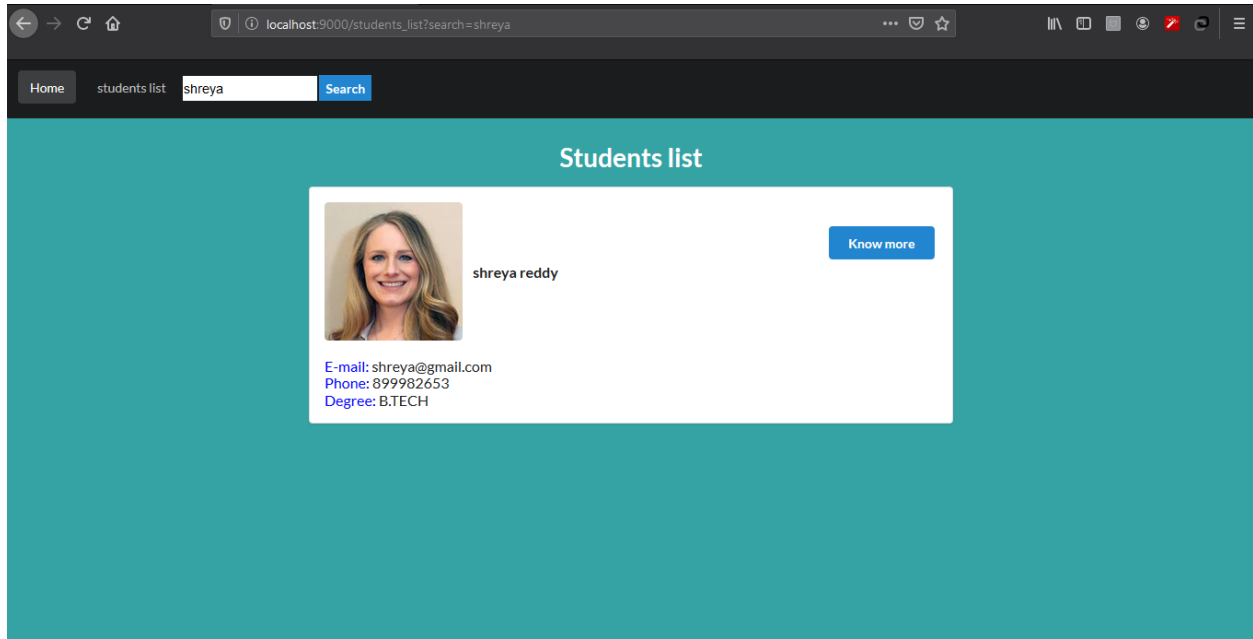
When clicked on know more button it will redirect to this page which consists of individual student information as we can see in the above photo it all consist the edit button which will redirect to edit page

## 5] Edit student info



In this page you can edit the student info and update the info in database

## 6] Search bar



User can search for a student and it will show the searched student as shown in the above photo

## Part-2

### Backend functionality

```
var mongoose=require("mongoose");

var student = new mongoose.Schema({
  name:String,
  email:String,
  phone:Number,
  photo:String,
  degree:String
})

var Student= mongoose.model("Student",student);
module.exports=Student;
```

The above photo shows the data schema/model for collecting the data from the user

## 1]"/" Route

This route renders the landing page(landing.ejs) which consist of add student button which redirects to the form page to add student details.

## 2]"/entry" Route

This route renders the form to fill the student information(entry.ejs) once the user submits the information there is a http post request and the data saves into mongo dB database and then the user is redirected to the students list page were the user can see the updated students list.

## 3]"/students\_list" Route

This route consist of all the students information(students\_list.ejs) listed down in cards , here the client makes a http get request to the server to fetch all the data available then once we get data we render the data on the client side.

## 4]"/student/:id" Route

This route has the individual student information(student.ejs) we make a http get request which finds students from the database using there id as we know mongo dB when a new data is created it assigns unique id to each data so using that id we can access individual student data then render it on client side.

## 5]"/student/edit/:id" Route

This route renders the edit form(edit.ejs) so here i access individual student's data by finding by id and updating the student information.

## 6]Searching students

Here basically i use the req.query.search and i find the user by name and then once the i find the user in the database it will be rendered on to the screen