BSc (Hons) in Information Technology Specializing in Software Engineering

Year 3 - 2022

SE3040 – Application Frameworks

Group Project: Technical Report

Group Name: SE3030_WE_68

Group Member Details

Student ID number	Name	SLIIT Email
IT20236564	Kumarathunga M.A.M.T	it20236564@my.sliit.lk
IT20226046	Kavinda M.A.I.N	it20226046@my.sliit.lk
IT20227036	M.M.M.B Moremada	it20227036@my.sliit.lk
IT20134280	S.Ishalini	it20134280@my.sliit.lk

Git Repository: https://github.com/kavindamain99/Research-Project-Management-Tool-frontend

Backend: https://github.com/kavindamain99/Research-Project-Management-Tool-backend

Group Presentation Video: Recordings - OneDrive_1.mp4

Project Start Date: 15 March 2022

Project End Date: 4 July 2022

Project Declaration:

I the undersigned solemnly declare that the project technical based on our group members own work carried out during the course of our study under the supervision of Mr. Thusithanjana Thilakarathna.

Group Leader:

Marking Rubric

Date	Weight	Marks (out of 5)			
Description		5	5	5	5
1. SRS Document -	**10**				
N/A	5				
N/A	5				
2. Frontend Development	**25**				
a. Viva	10				
b. demo 1. Implemented all the comonents [2] 2. UI/UX [08] 3. Components [5]	15				
3. Backend Development	**25**				
a. Viva	10				
b. Demo 1. Routes [5] 2. DAL [5] 3. DAO [5]	15				
4. Apps are hosted	**10**				
a. Frontend	5				
b. Backend	5				
5. Unit Tests	10				
a. Frontend	5				
b. Backend	5				
6. Test Cases	**5**				
7. Technical Report **a. Screenshots of the UI [1]** **b. Screenshots of the MongoDB [2]** **c. Description [2]**	**5**				

Total	100				
Comments					

Mark out of 5

0.0 - Section is not done

0.1 - There is something but entirely wrong or not relevant

1.0 - Relevant but wrong

2.0 - Slightly correct

3.0 - Average

4.0 - Good - Slight errors

5.0 - WOW (No errors)

Report Template.

1. Component Description (Member wise)
2. Screenshots of the Frontend (Member wise)
3. Screenshots/ Code Snippets of the Backend (Member wise)
4. Screenshots of the MongoDB Schemas (Member wise)
5. Proof of Hosting (Backend and Frontend)

When you write the report, create topics as given in the report template and subtopics from the member's SLIIT Student ID number. Please refer to the marking rubric as well.

Research Project Management Tool

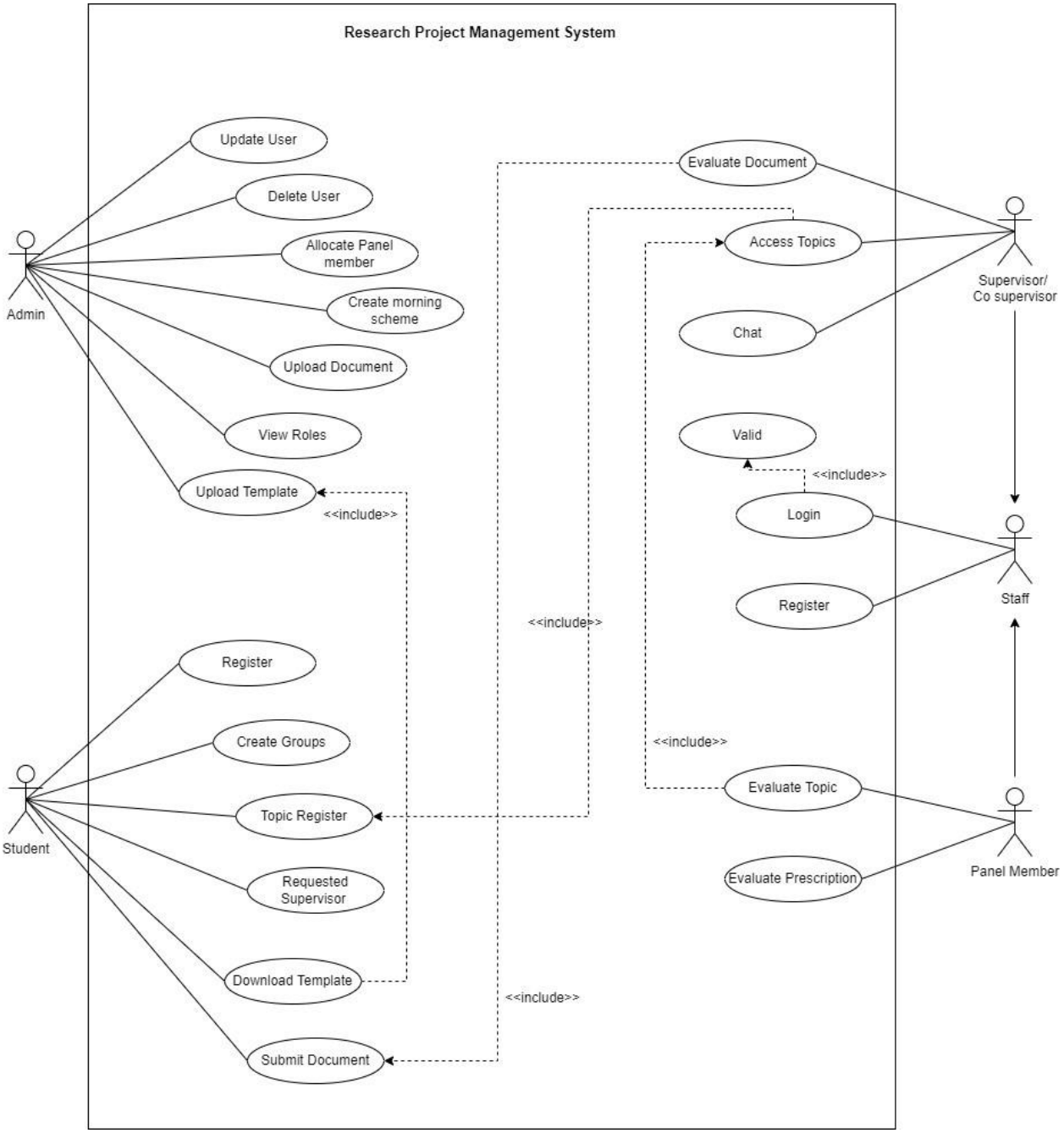
All SLIIT undergraduates must do a research project in their 4th year. This Research Project module is a 16 credit, two semesters long project. A student group must find a research topic in a specific research field and send it to a supervisor who has interests in the same research field. Once the supervisor accepted the topic, they must find a co-supervisor of the same research interest.

After finalizing the supervisor student must send a document including the topic details of the Research. Then topic evaluation panel will evaluate the topic and send feedback to the student group. If the topic is accepted, they can continue to do the project. If rejected, they must find a new topic and submit it back.

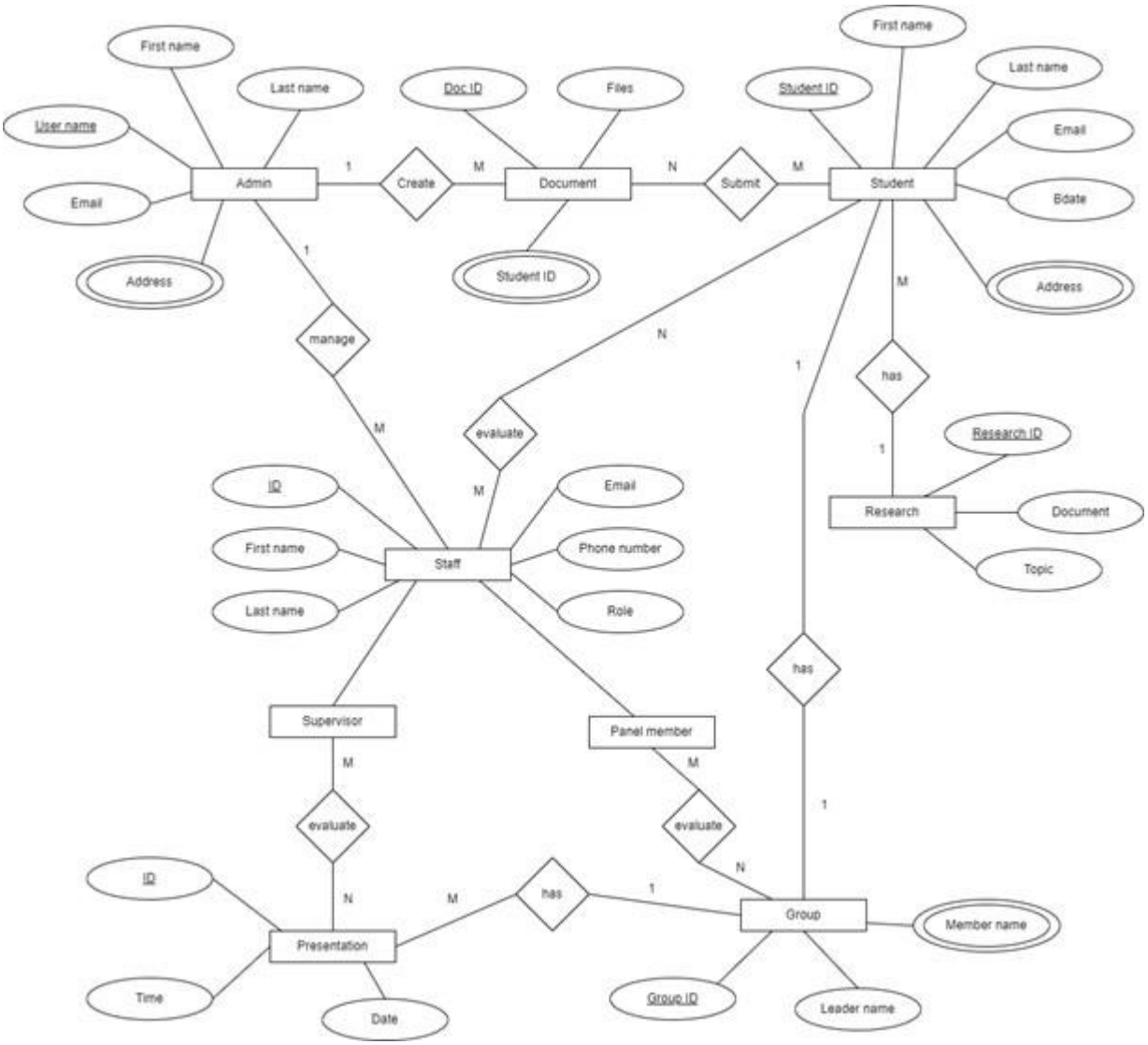
There are several evaluation stages. Document submissions are evaluated by the supervisor or the co-supervisor. Presentations will be evaluated by a separate panel. The final Thesis will be Double evaluated by the supervisor and a blind reviewer.

SLIIT Research project team is looking for a system, which has the capability of managing the Research project and automating certain tasks.

Use Case Diagram:



ER Diagram


First name
Last name
User name
Email
Admin
Address
1
Create
M
Doc ID
Files
Document
Student ID
N
Submit
M
Student ID
First name
Last name
Email
Bdate
Address
Student
1
manage
M
evaluate
M
N
1
has
M
1
Research ID
Research
Document
Topic
ID
First name
Last name
Email
Phone number
Role
Staff
has
Supervisor
Panel member
M
M
evaluate
evaluate
1
N
N
M
has
1
ID
Time
Presentation
Date
Group
Member name
Group ID
Leader name

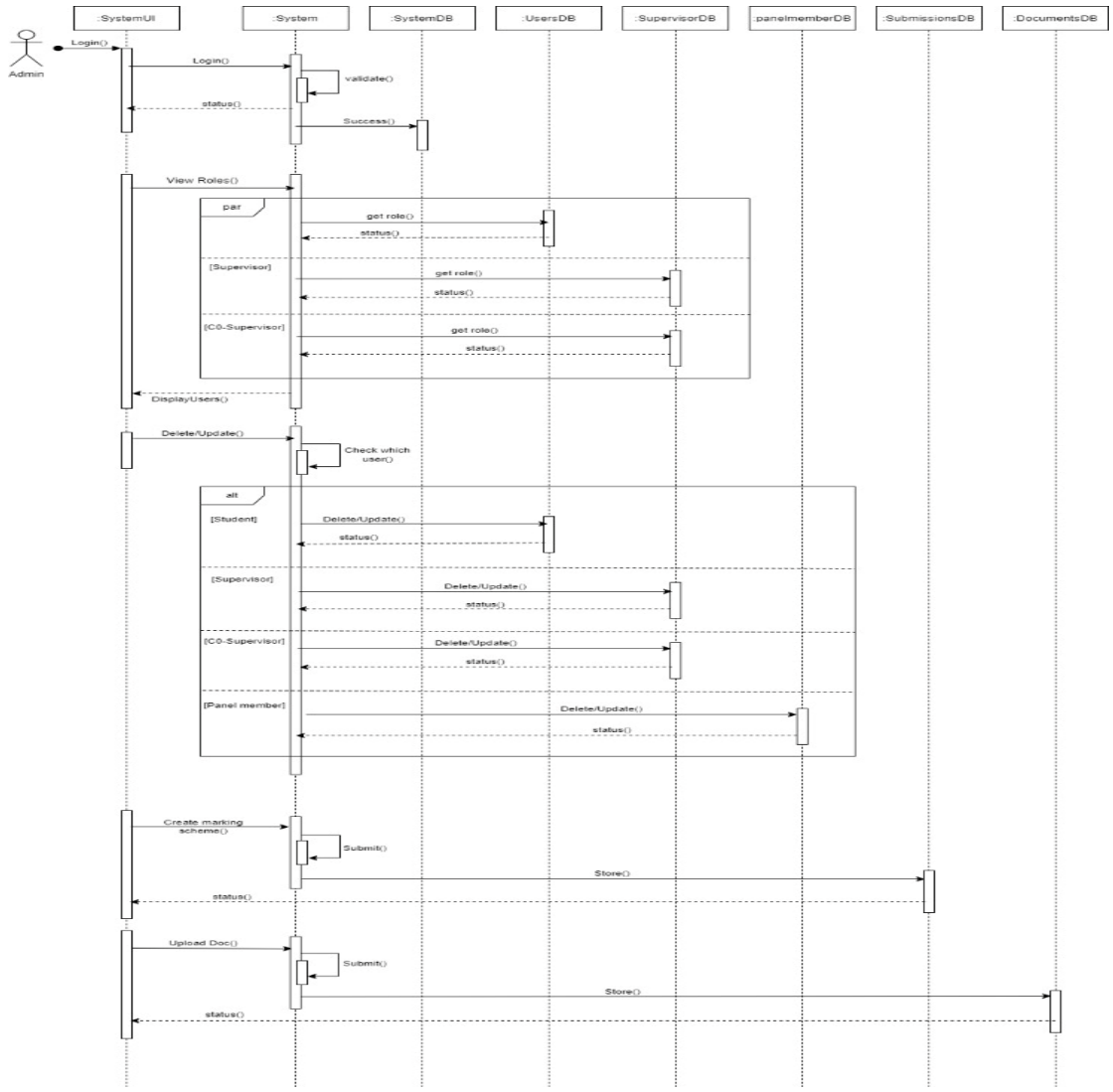
Component Description

IT20226046 Kavinda M.A.I.N

Description:

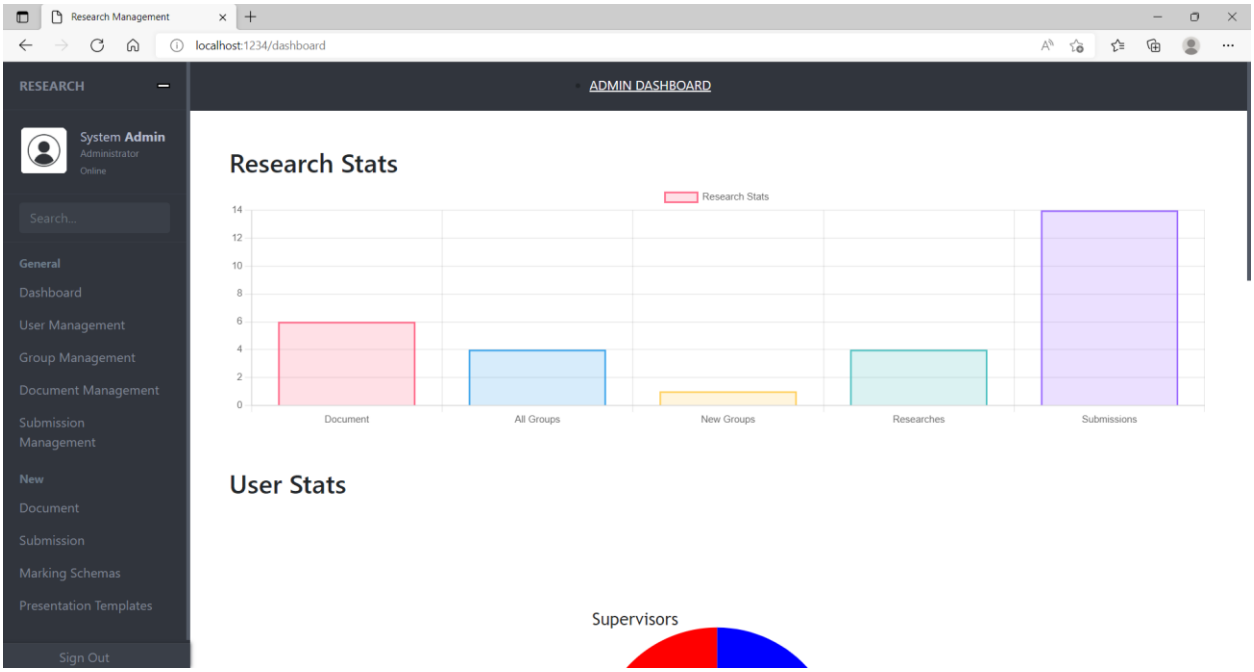
Admin Role

Admin is the owner of this system and who can view all of users and update or delete them. And also, their main function is the creation of submission part. Admin can create a submission type and create marking scheme for them. they can upload document or presentation templates to the system. Allocate panel members to student group is also a function of the administrator.

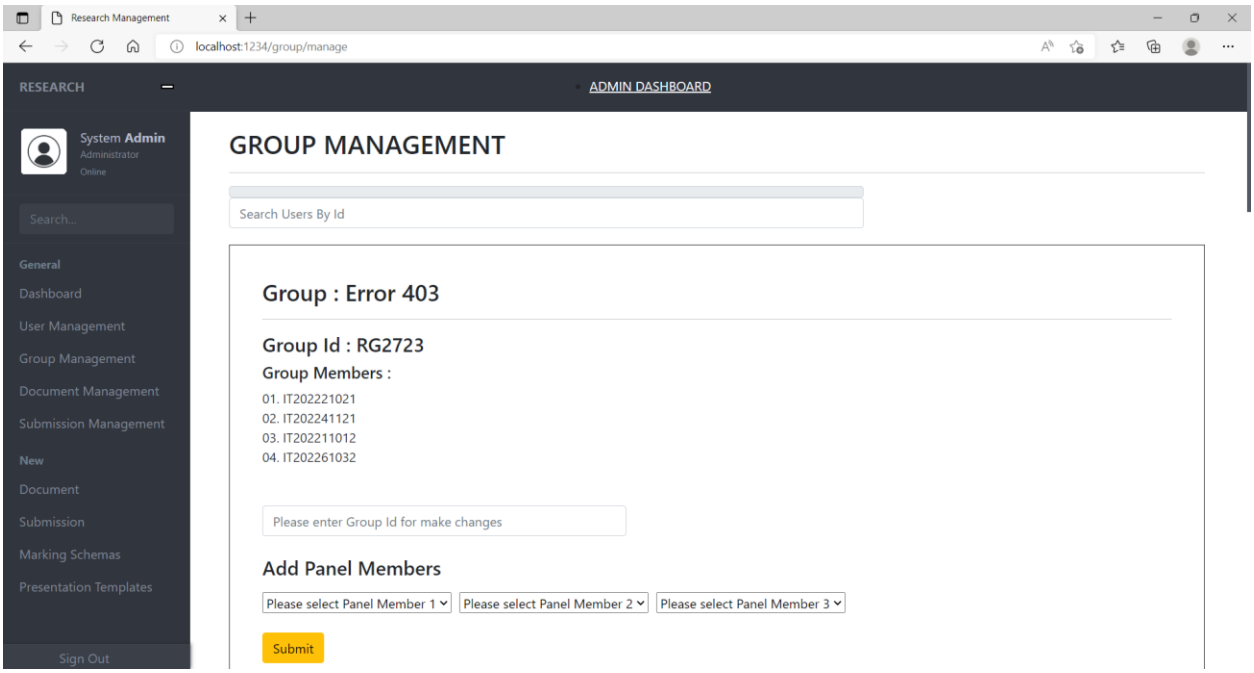


Screenshots of the Frontend

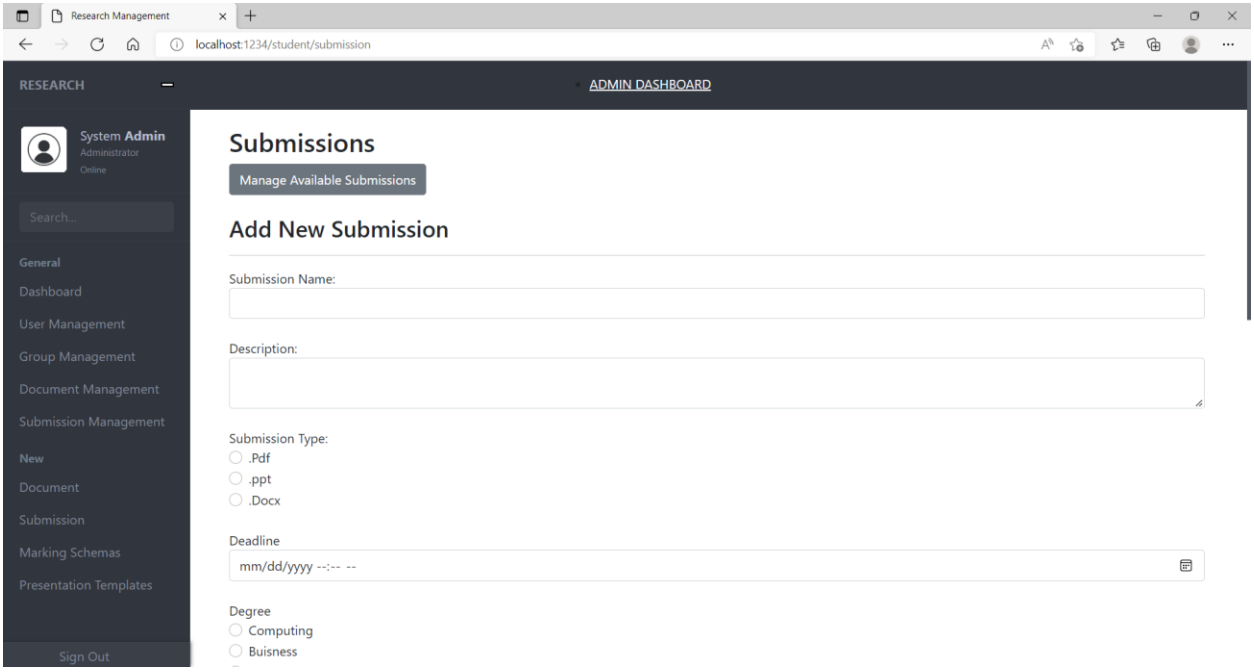
Admin dashboard



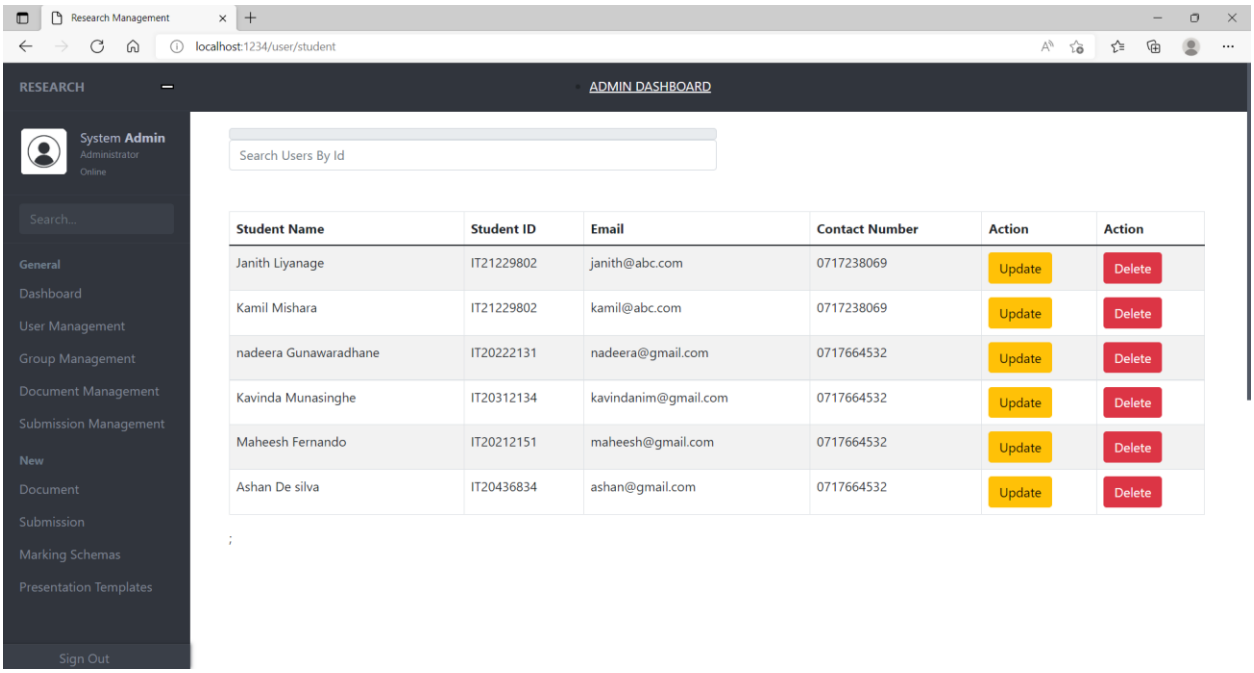
Allocate panel members



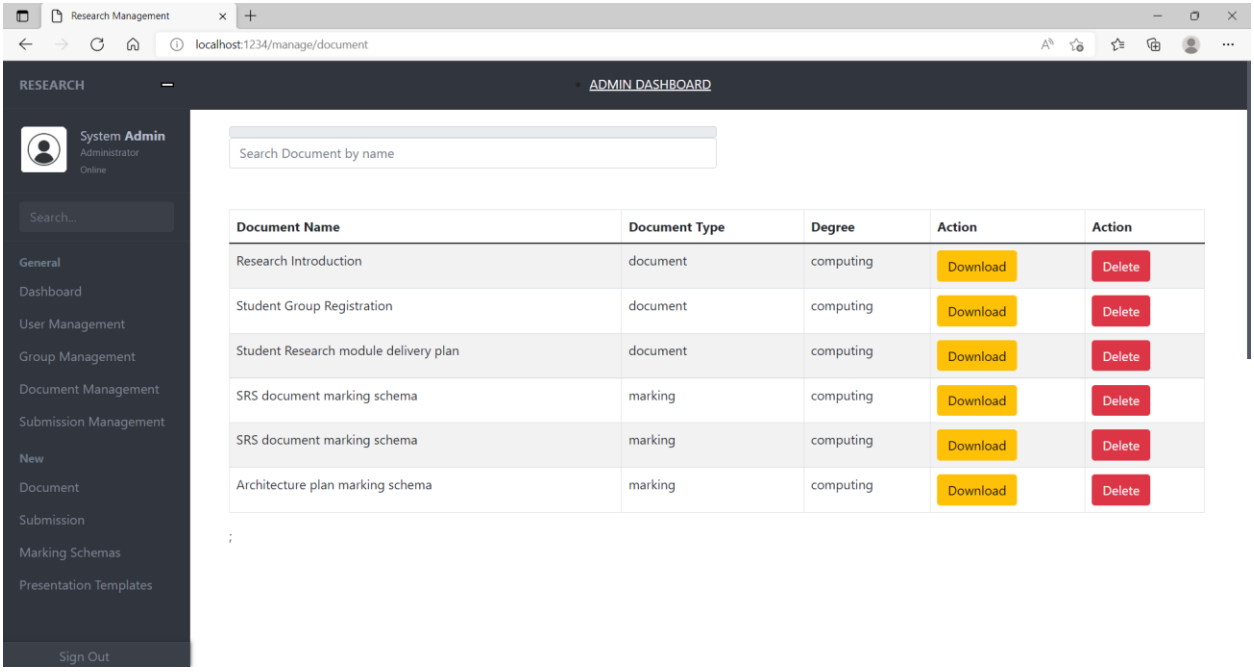
Create submission type



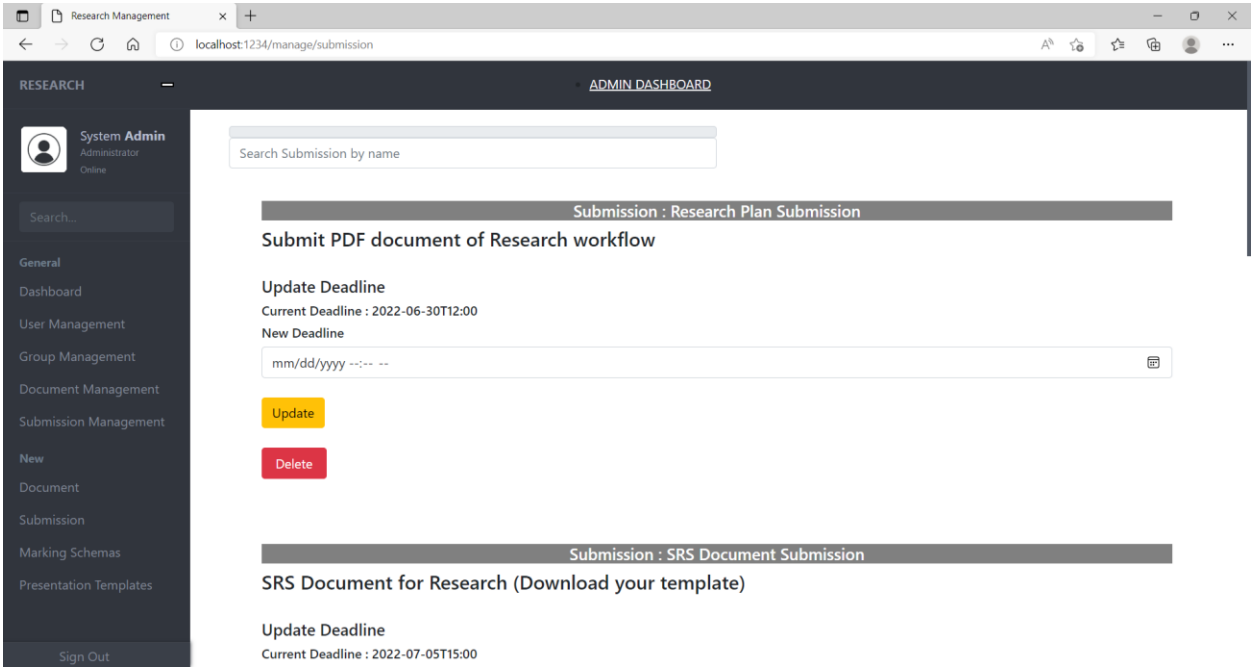
Delete / Update Users



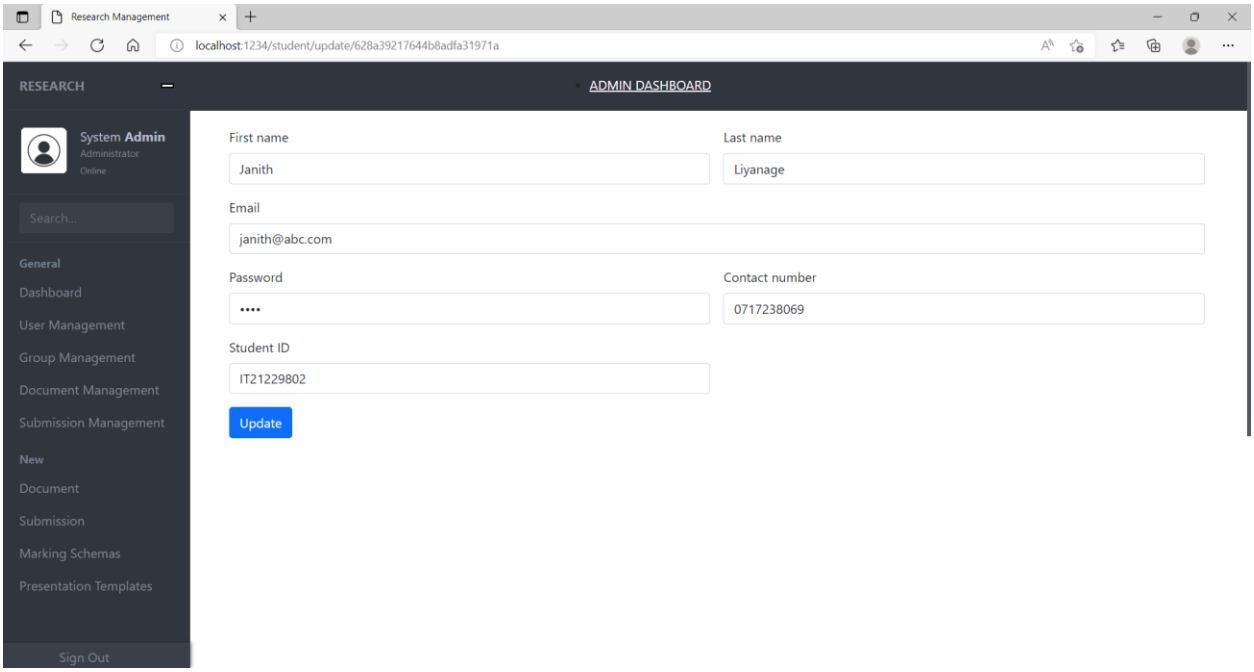
Document management



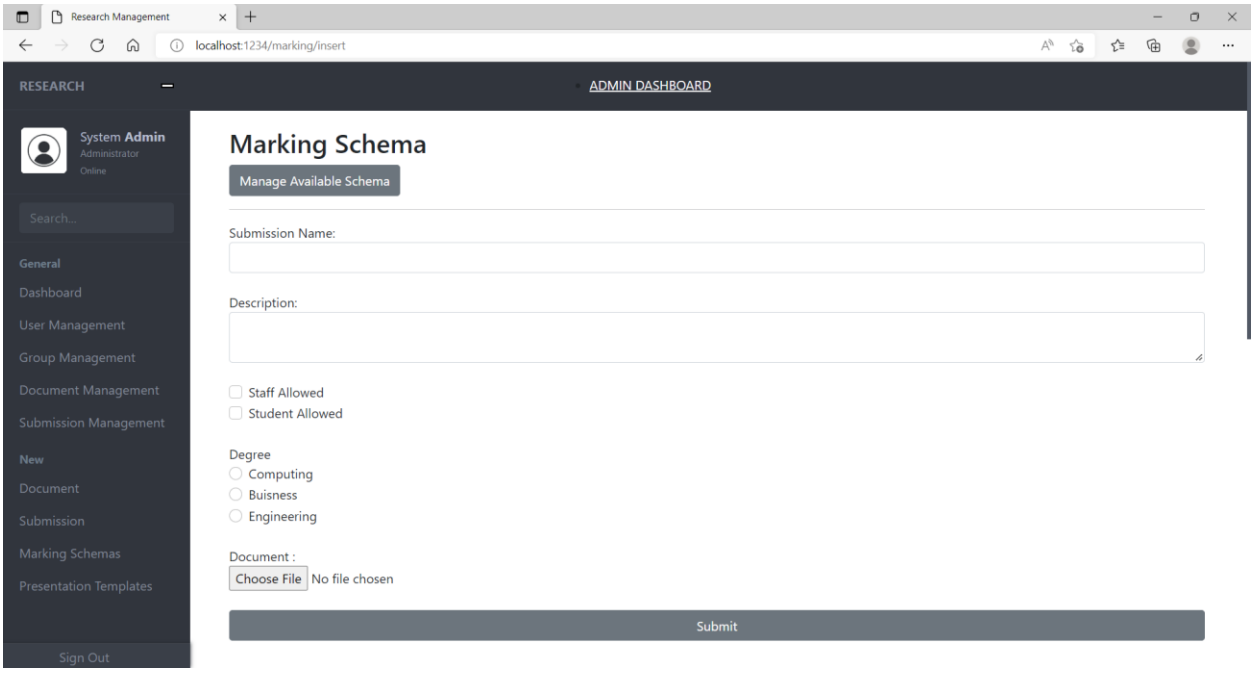
Submission management



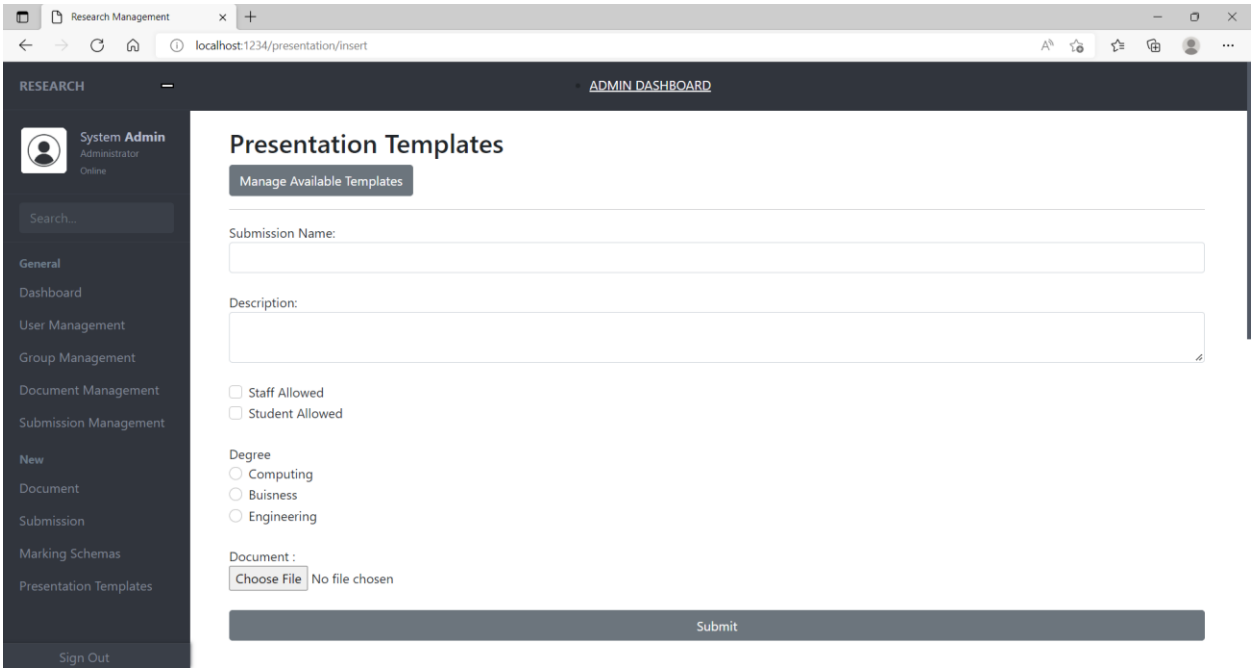
Update user



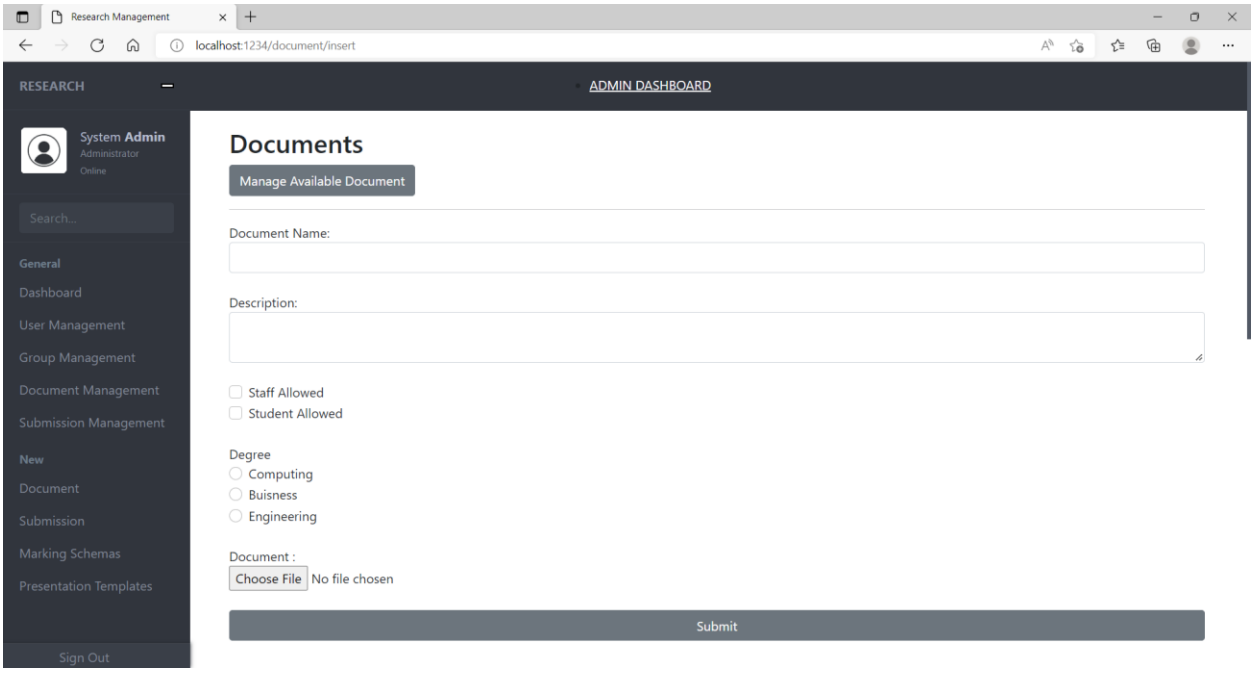
Upload marking scheme



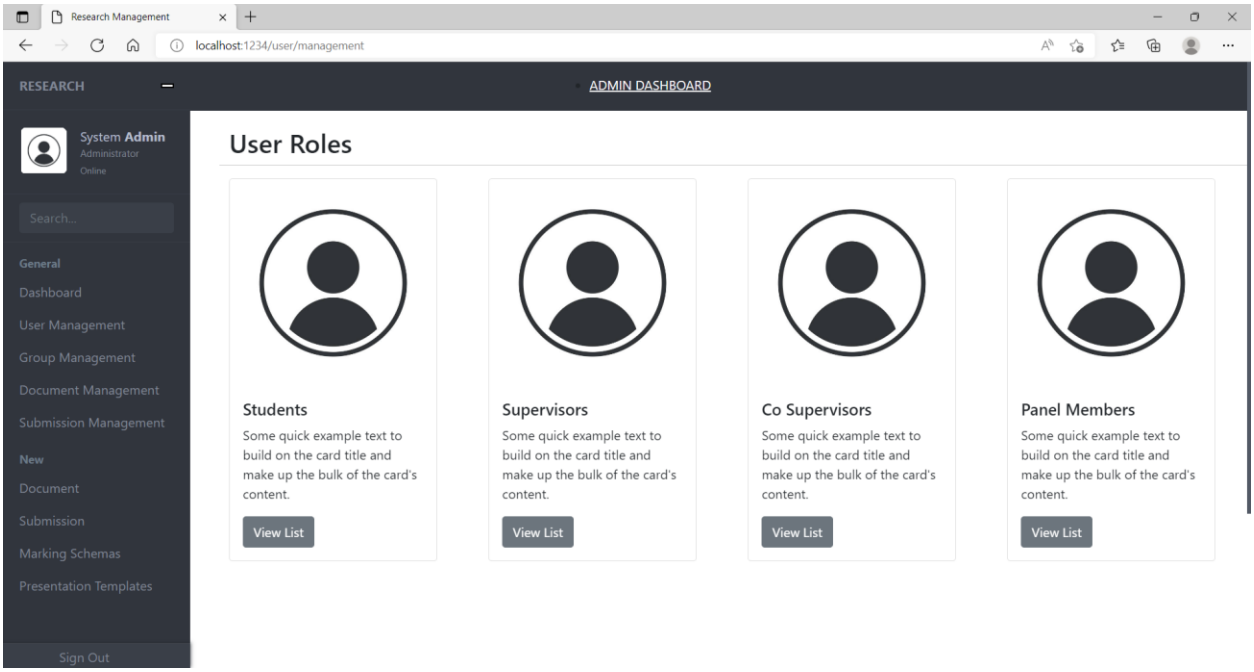
Upload presentation template



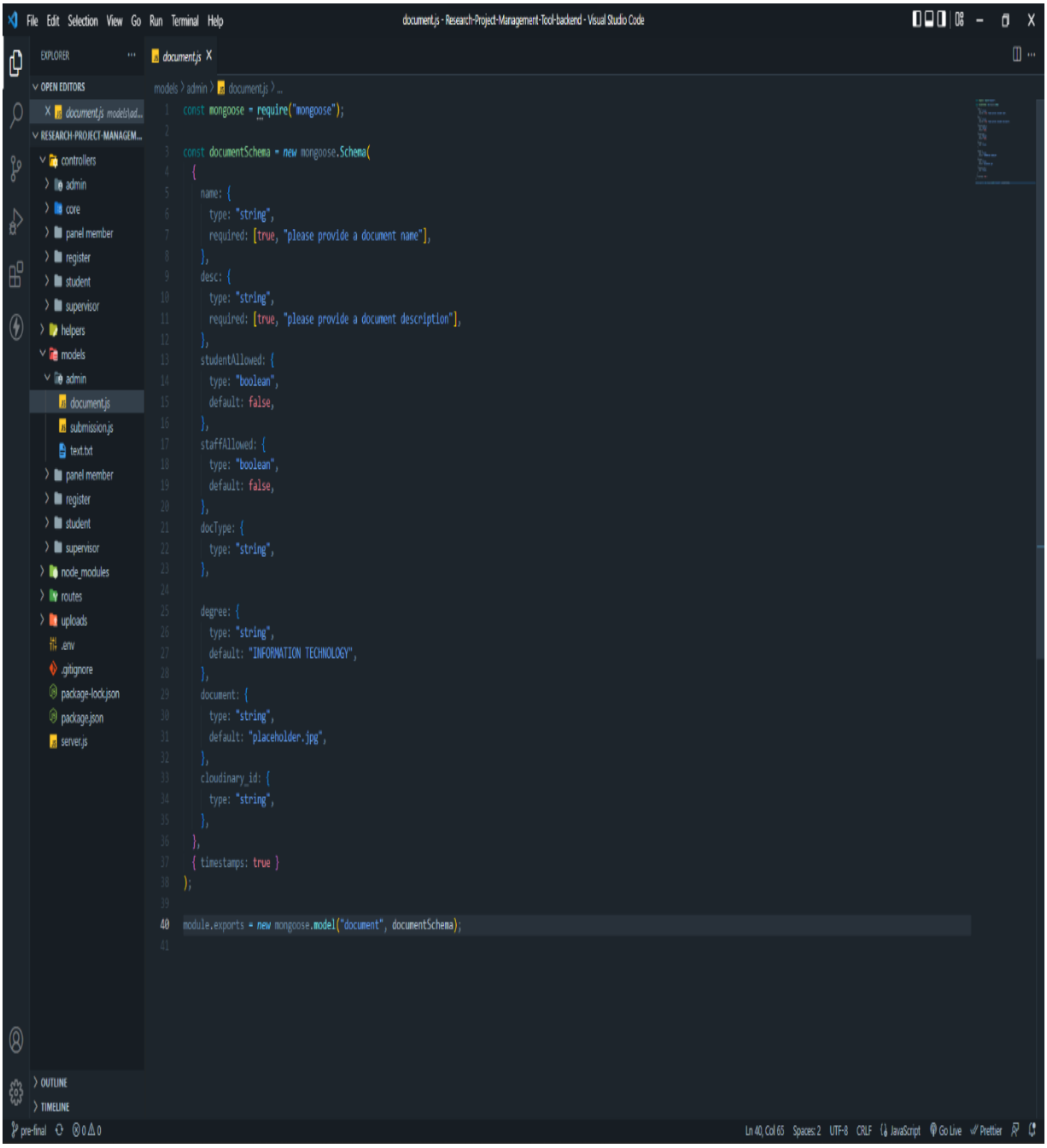
Upload documents



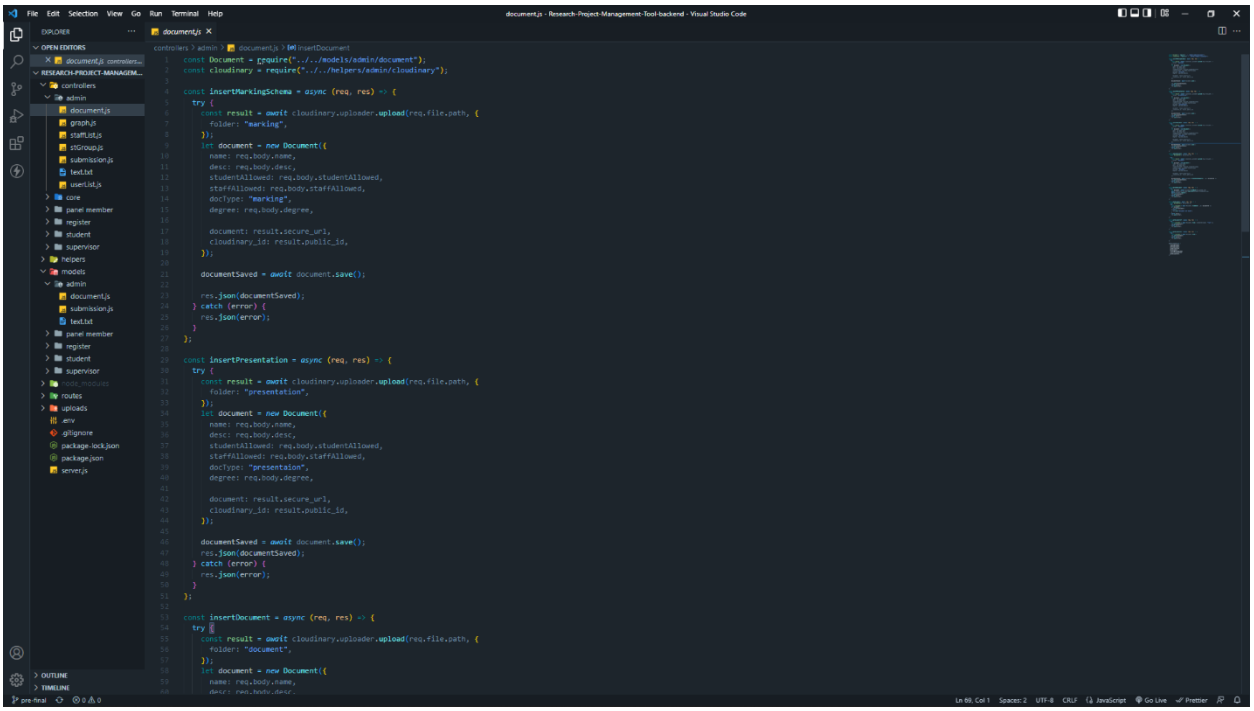
View roles


Research Management
localhost:1234/user/management
RESEARCH
ADMIN DASHBOARD
System Admin
Administrator
Online
Search...
General
Dashboard
User Management
Group Management
Document Management
Submission Management
New
Document
Submission
Marking Schemas
Presentation Templates
Sign Out
User Roles
Students
Some quick example text to build on the card title and make up the bulk of the card's content.
View List
Supervisors
Some quick example text to build on the card title and make up the bulk of the card's content.
View List
Co Supervisors
Some quick example text to build on the card title and make up the bulk of the card's content.
View List
Panel Members
Some quick example text to build on the card title and make up the bulk of the card's content.
View List

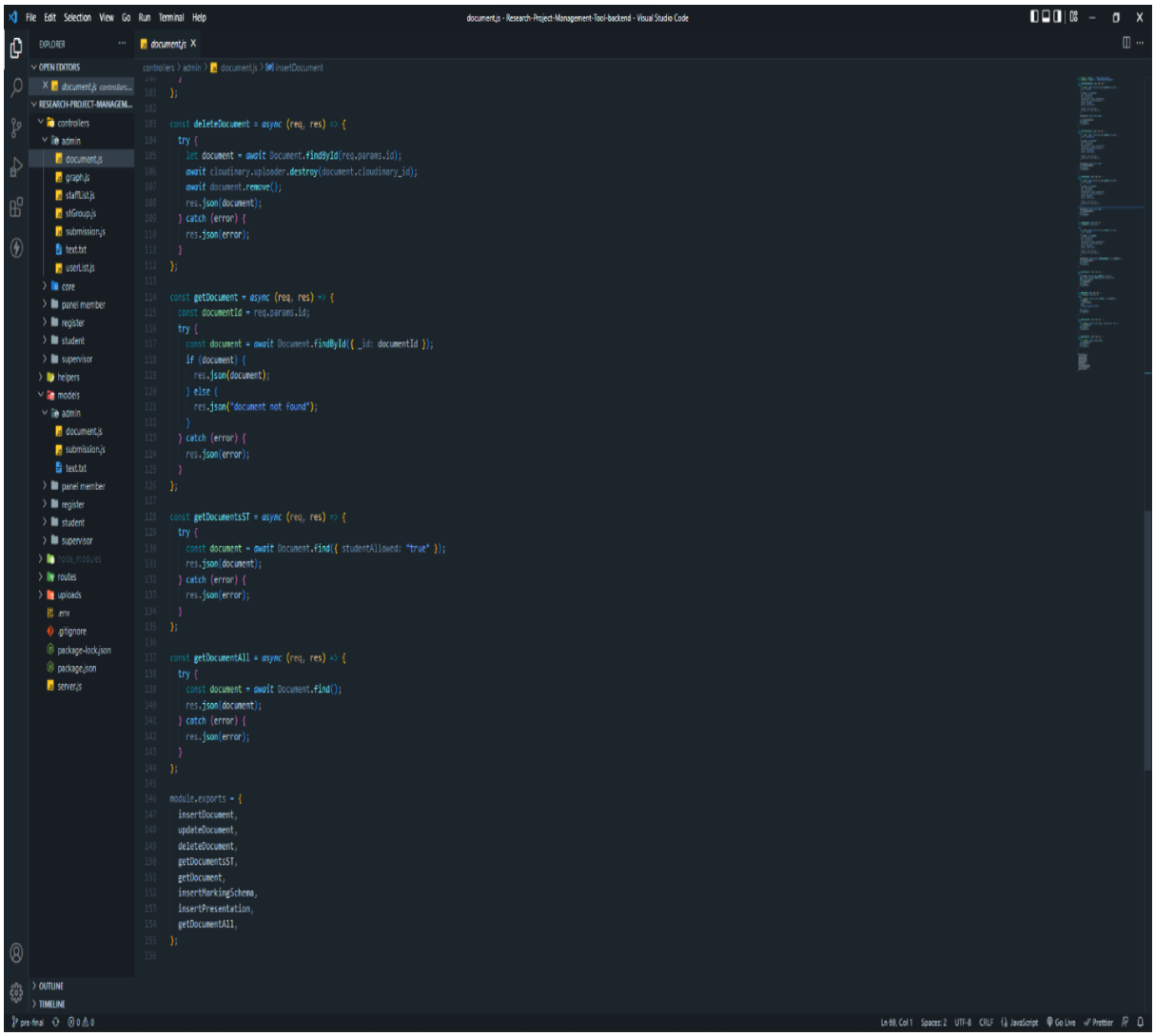
Screenshots/ Code Snippets of the Backend:




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

const documentSchema = new mongoose.Schema(
  {
    name: {
      type: "string",
      required: [true, "please provide a document name"],
    },
    desc: {
      type: "string",
      required: [true, "please provide a document description"],
    },
    studentAllowed: {
      type: "boolean",
      default: false,
    },
    staffAllowed: {
      type: "boolean",
      default: false,
    },
    docType: {
      type: "string",
    },

    degree: {
      type: "string",
      default: "INFORMATION TECHNOLOGY",
    },
    document: {
      type: "string",
      default: "placeholder.jpg",
    },
    cloudinary_id: {
      type: "string",
    },
  },
  { timestamps: true }
);

module.exports = new mongoose.model("document", documentSchema);
```

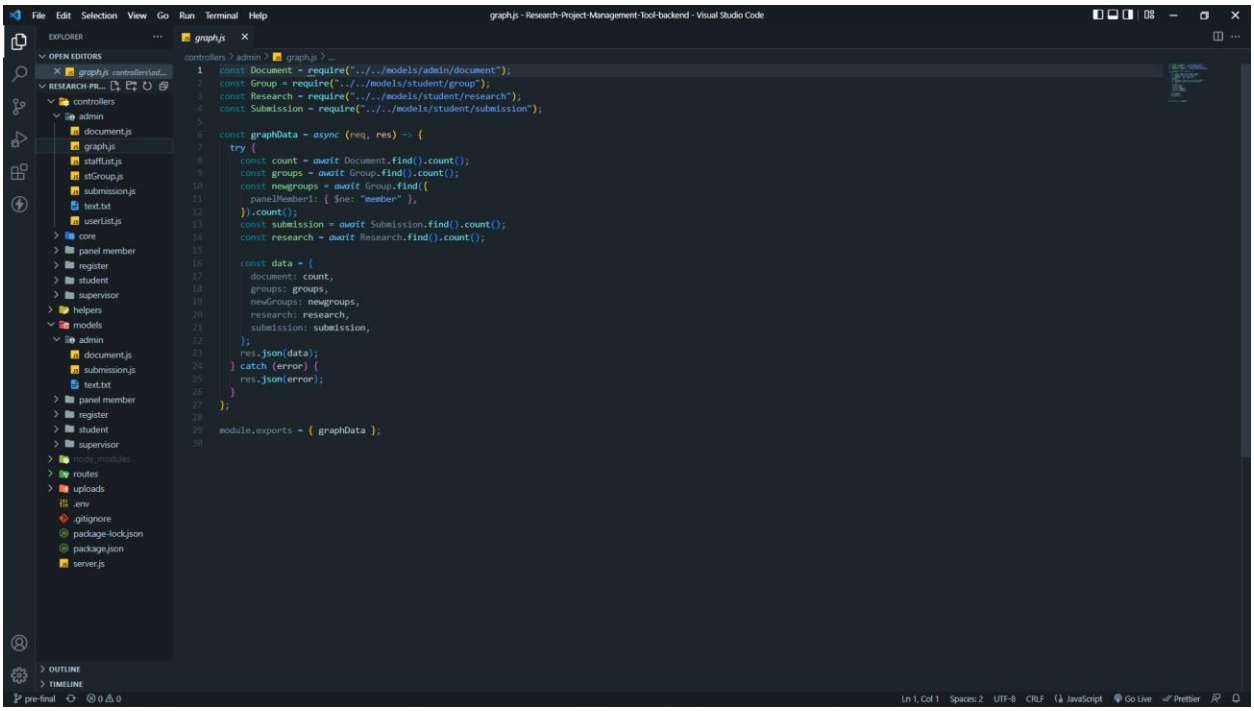


```
const Document = require("../../models/admin/document");
const cloudinary = require("../../helpers/admin/cloudinary");

const insertMarkingSchema = async (req, res) => {
  try {
    const result = await cloudinary.uploader.upload(req.file.path, {
      folder: "marking",
    });
    let document = new Document({
      name: req.body.name,
      desc: req.body.desc,
      studentAllowed: req.body.studentAllowed,
      staffAllowed: req.body.staffAllowed,
      docType: "marking",
      degree: req.body.degree,

      document: result.secure_url,
      cloudinary_id: result.public_id,
    });

    documentSaved = await document.save();

    res.json(documentSaved);
  } catch (error) {
    res.json(error);
  }
};

const insertPresentation = async (req, res) => {
  try {
    const result = await cloudinary.uploader.upload(req.file.path, {
      folder: "presentation",
    });
    let document = new Document({
      name: req.body.name,
      desc: req.body.desc,
      studentAllowed: req.body.studentAllowed,
      staffAllowed: req.body.staffAllowed,
      docType: "presentation",
      degree: req.body.degree,

      document: result.secure_url,
      cloudinary_id: result.public_id,
    });

    documentSaved = await document.save();
    res.json(documentSaved);
  } catch (error) {
    res.json(error);
  }
};

const insertDocument = async (req, res) => {
  try {
    const result = await cloudinary.uploader.upload(req.file.path, {
      folder: "document",
    });
    let document = new Document({
      name: req.body.name,
      desc: req.body.desc,
```

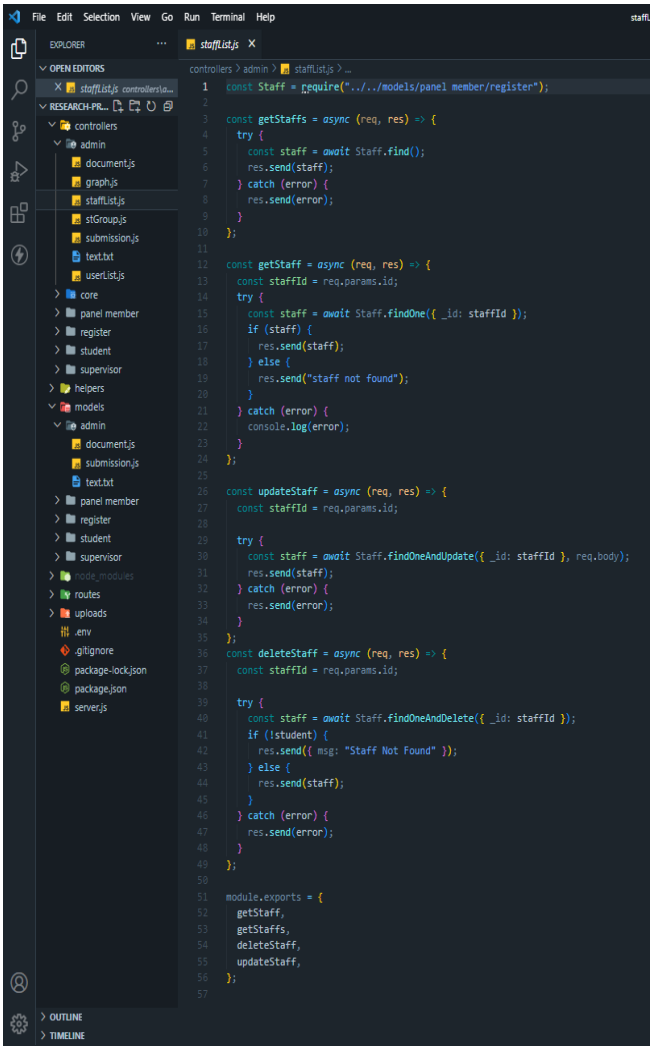


```
  /
};

const deleteDocument = async (req, res) => {
  try {
    let document = await Document.findById(req.params.id);
    await cloudinary.uploader.destroy(document.cloudinary_id);
    await document.remove();
    res.json(document);
  } catch (error) {
    res.json(error);
  }
};

const getDocument = async (req, res) => {
  const documentId = req.params.id;
  try {
    const document = await Document.findById({ _id: documentId });
    if (document) {
      res.json(document);
    } else {
      res.json("document not found");
    }
  } catch (error) {
    res.json(error);
  }
};

const getDocumentsST = async (req, res) => {
  try {
    const document = await Document.find({ studentAllowed: "true" });
    res.json(document);
  } catch (error) {
    res.json(error);
  }
};

const getDocumentAll = async (req, res) => {
  try {
    const document = await Document.find();
    res.json(document);
  } catch (error) {
    res.json(error);
  }
};

module.exports = {
  insertDocument,
  updateDocument,
  deleteDocument,
  getDocumentsST,
  getDocument,
  insertMarkingSchema,
  insertPresentation,
  getDocumentAll,
};
```

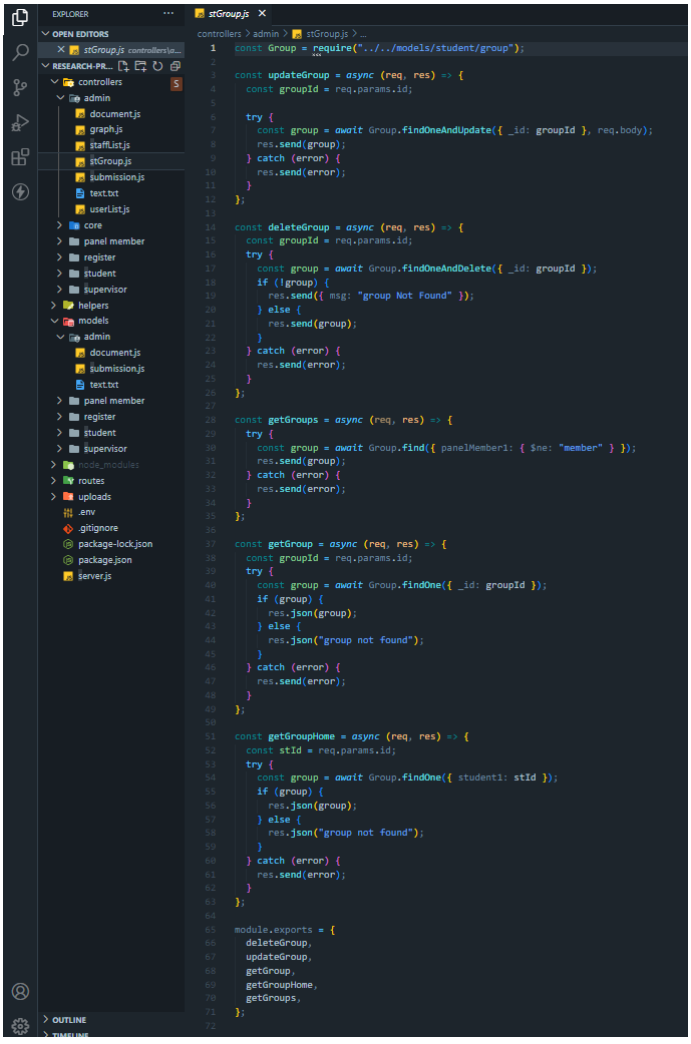

File Edit Selection View Go Run Terminal Help
submission.js - Research-Project-Management-Tool-backend - Visual Studio Code
EXPLORER
submission.js
OPEN EDITORS
submission.js models\ad...
RESEARCH-PR...
controllers
admin
document.js
graph.js
staffList.js
stGroup.js
submission.js
text.txt
userList.js
core
panel member
register
student
supervisor
helpers
models
admin
document.js
submission.js
text.txt
panel member
register
student
supervisor
node_modules
routes
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
models > admin > submission.js > ...
1 const mongoose = require("mongoose");
3 const submissionSchema = new mongoose.Schema({
4 name: {
5 type: "string",
6 required: [true, "please provide a submission name"],
7 },
8 desc: {
9 type: "string",
10 required: [true, "please provide a description"],
11 },
12 type: {
13 type: "string",
14 required: [true, "please provide a submission type "],
15 },
16 deadline: {
17 type: "string",
18 required: [true, "please provide a deadline"],
19 },
20 degree: {
21 type: "string",
22 required: [true, "please provide a degree"],
23 },
24 });
26 module.exports = new mongoose.model("submission", submissionSchema);
pre-final
Ln 1, Col 1 Spaces: 2 UTF-8 CRLF JavaScript Go Live Prettier



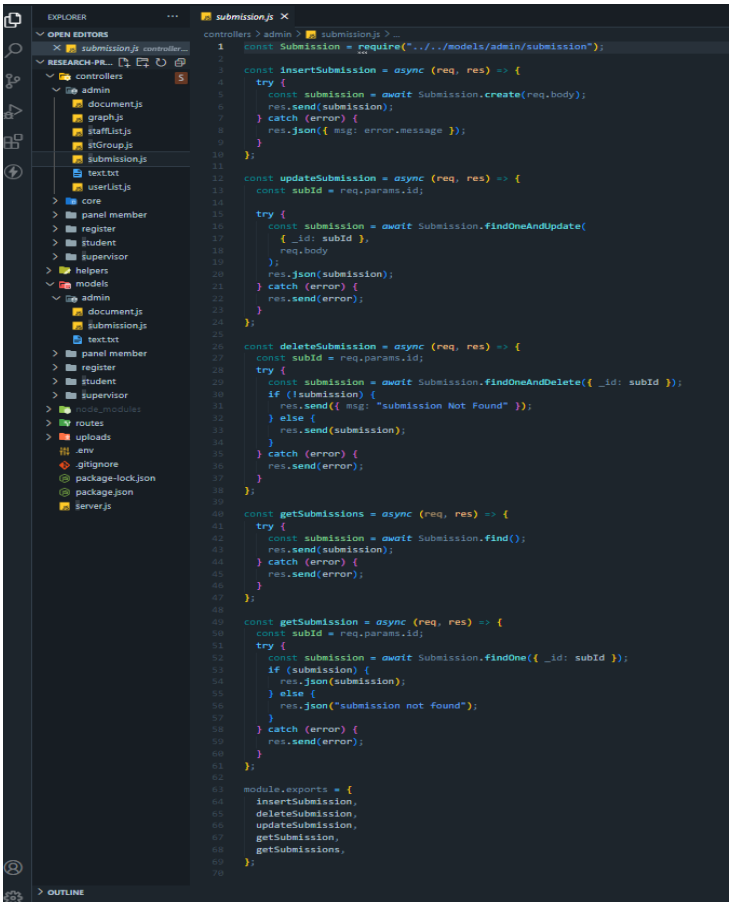
```
const Document = require("../../models/admin/document");
const Group = require("../../models/student/group");
const Research = require("../../models/student/research");
const Submission = require("../../models/student/submission");

const graphData = async (req, res) => {
  try {
    const count = await Document.find().count();
    const groups = await Group.find().count();
    const newgroups = await Group.find({
      panelMember1: { $ne: "member" },
    }).count();
    const submission = await Submission.find().count();
    const research = await Research.find().count();

    const data = {
      document: count,
      groups: groups,
      newGroups: newgroups,
      research: research,
      submission: submission,
    };
    res.json(data);
  } catch (error) {
    res.json(error);
  }
};

module.exports = { graphData };
```



```
const Staff = require("../../models/panel member/register");

const getStaffs = async (req, res) => {
  try {
    const staff = await Staff.find();
    res.send(staff);
  } catch (error) {
    res.send(error);
  }
};

const getStaff = async (req, res) => {
  const staffId = req.params.id;
  try {
    const staff = await Staff.findOne({ _id: staffId });
    if (staff) {
      res.send(staff);
    } else {
      res.send("staff not found");
    }
  } catch (error) {
    console.log(error);
  }
};

const updateStaff = async (req, res) => {
  const staffId = req.params.id;

  try {
    const staff = await Staff.findOneAndUpdate({ _id: staffId }, req.body);
    res.send(staff);
  } catch (error) {
    res.send(error);
  }
};
const deleteStaff = async (req, res) => {
  const staffId = req.params.id;

  try {
    const staff = await Staff.findOneAndDelete({ _id: staffId });
    if (!student) {
      res.send({ msg: "Staff Not Found" });
    } else {
      res.send(staff);
    }
  } catch (error) {
    res.send(error);
  }
};

module.exports = {
  getStaff,
  getStaffs,
  deleteStaff,
  updateStaff,
};
```

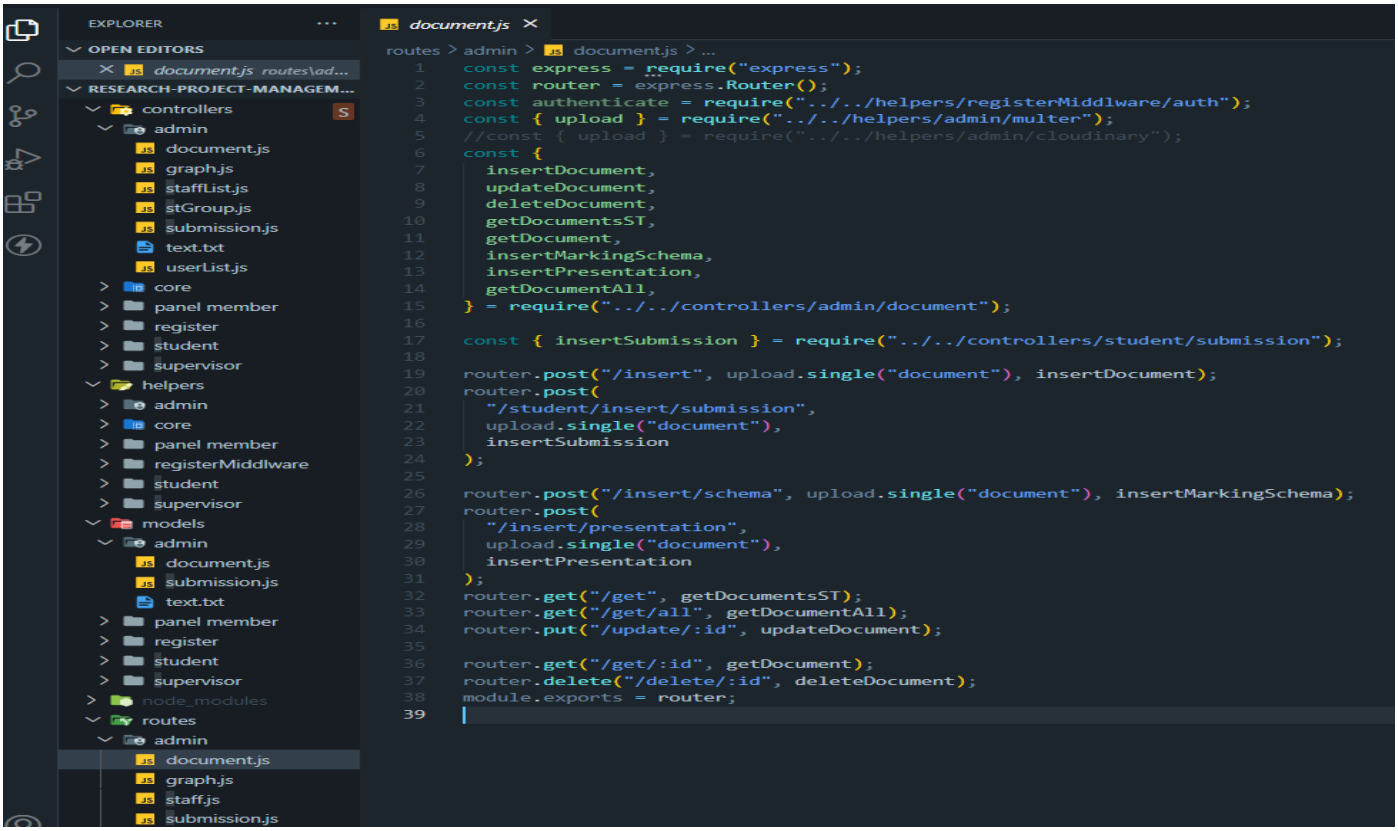


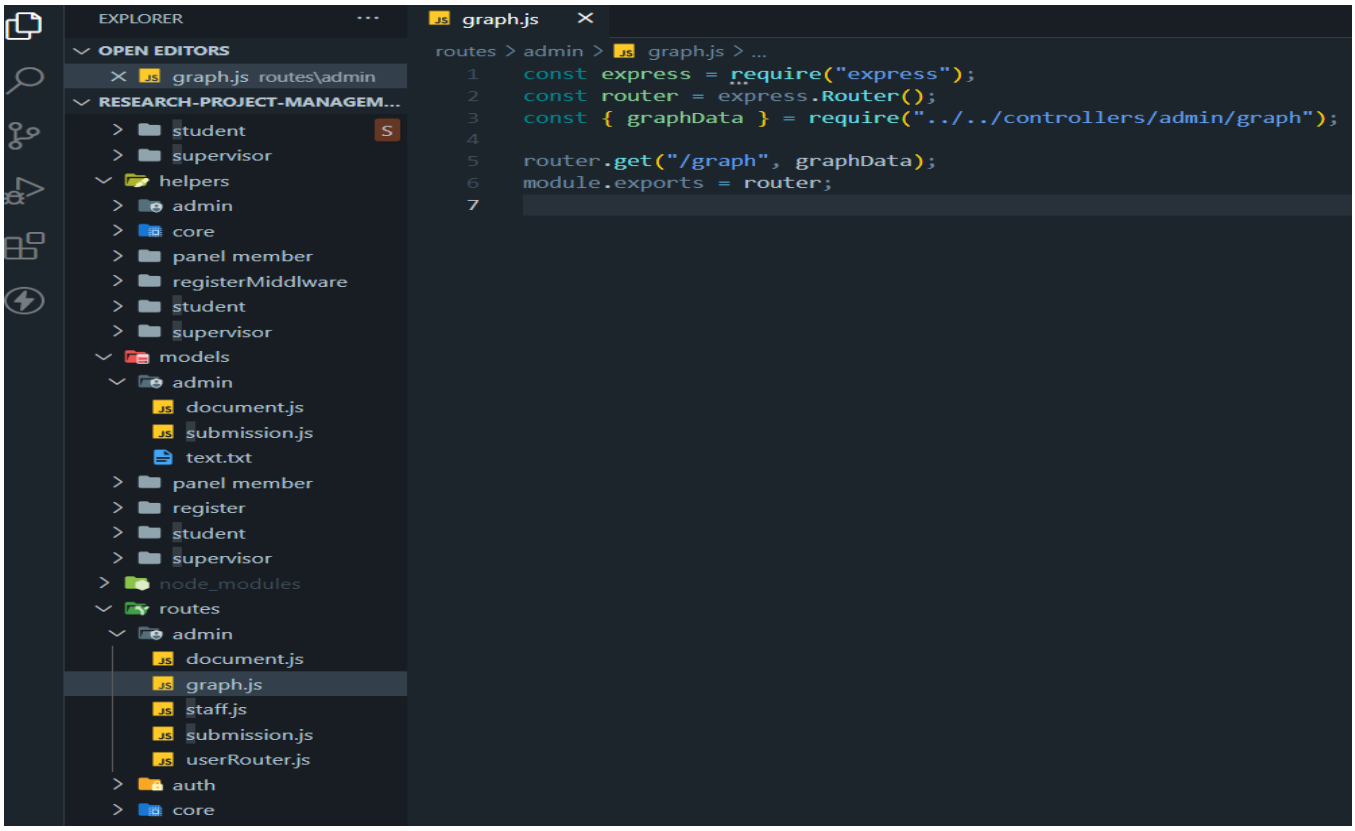
```
const Group = require("../../models/student/group");

const updateGroup = async (req, res) => {
  const groupId = req.params.id;

  try {
    const group = await Group.findOneAndUpdate({ _id: groupId }, req.body);
    res.send(group);
  } catch (error) {
    res.send(error);
  }
};

const deleteGroup = async (req, res) => {
  const groupId = req.params.id;
  try {
    const group = await Group.findOneAndDelete({ _id: groupId });
    if (!group) {
      res.send({ msg: "group Not Found" });
    } else {
      res.send(group);
    }
  } catch (error) {
    res.send(error);
  }
};

const getGroups = async (req, res) => {
  try {
    const group = await Group.find({ panelMember1: { $ne: "member" } });
    res.send(group);
  } catch (error) {
    res.send(error);
  }
};

const getGroup = async (req, res) => {
  const groupId = req.params.id;
  try {
    const group = await Group.findOne({ _id: groupId });
    if (group) {
      res.json(group);
    } else {
      res.json("group not found");
    }
  } catch (error) {
    res.send(error);
  }
};

const getGroupHome = async (req, res) => {
  const stId = req.params.id;
  try {
    const group = await Group.findOne({ student1: stId });
    if (group) {
      res.json(group);
    } else {
      res.json("group not found");
    }
  } catch (error) {
    res.send(error);
  }
};

module.exports = {
  deleteGroup,
  updateGroup,
  getGroup,
  getGroupHome,
  getGroups,
};
```

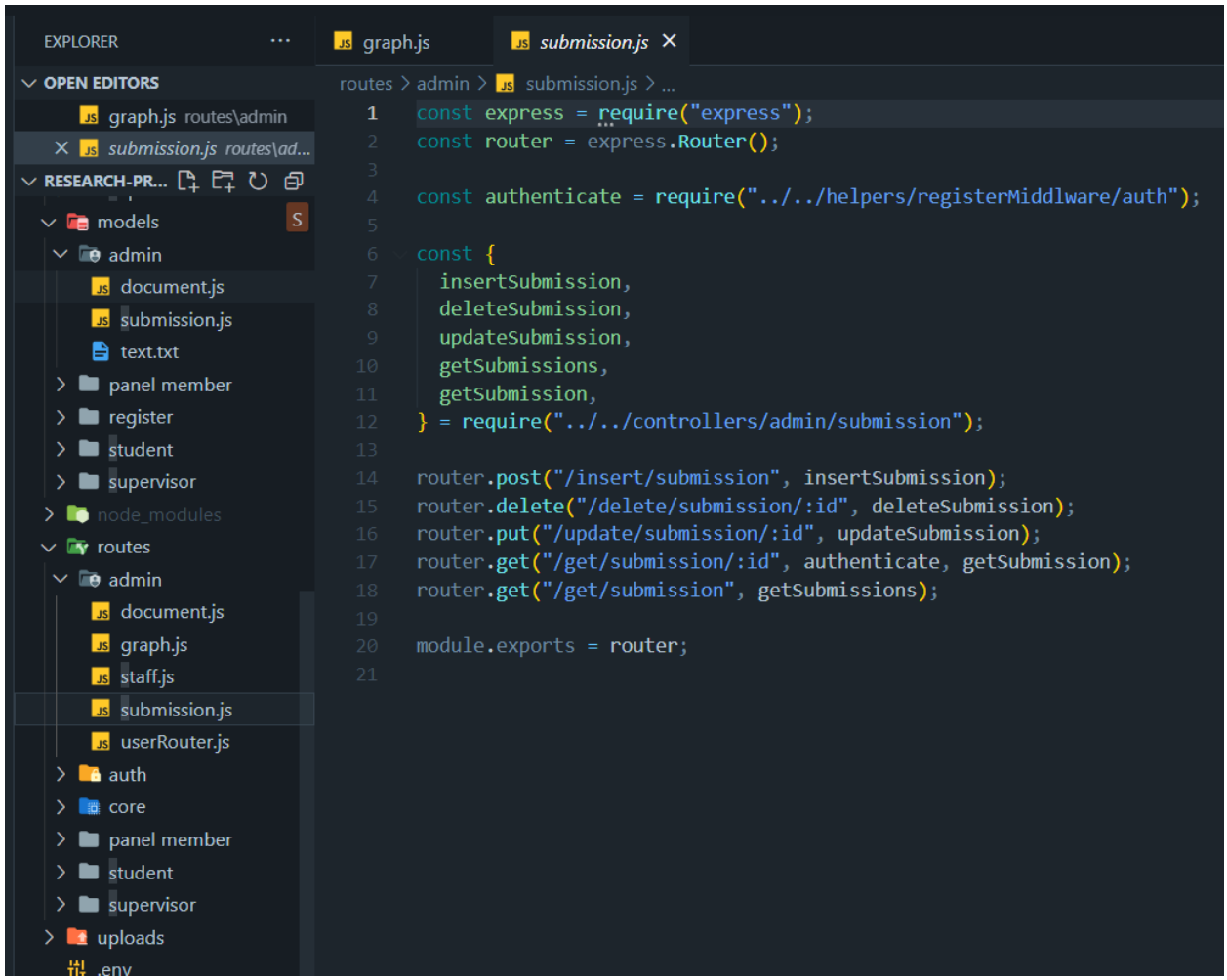


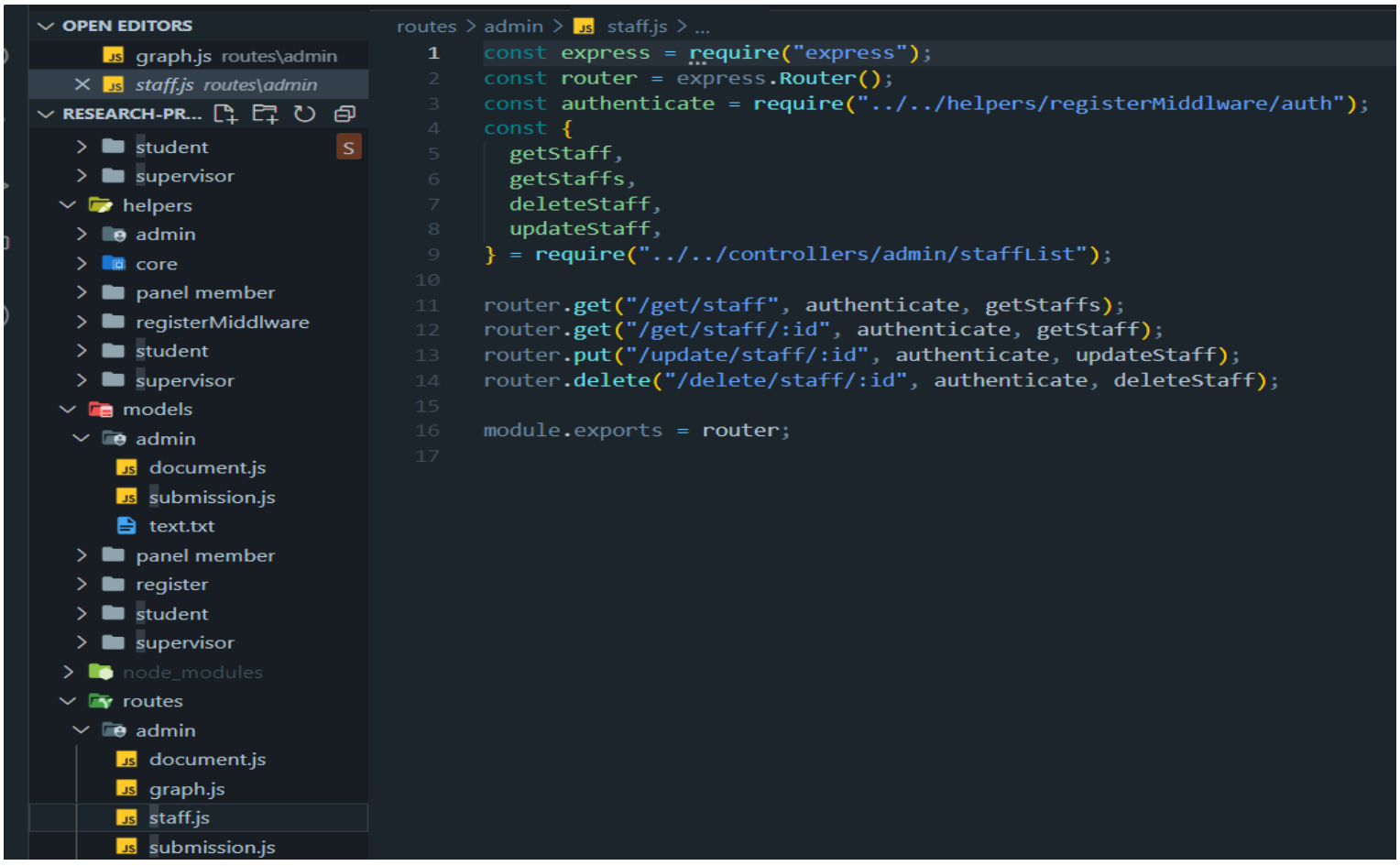
```
const Submission = require("../../models/admin/submission");

const insertSubmission = async (req, res) => {
  try {
    const submission = await Submission.create(req.body);
    res.send(submission);
  } catch (error) {
    res.json({ msg: error.message });
  }
};

const updateSubmission = async (req, res) => {
  const subId = req.params.id;

  try {
    const submission = await Submission.findOneAndUpdate(
      { _id: subId },
      req.body
    );
    res.json(submission);
  } catch (error) {
    res.send(error);
  }
};

const deleteSubmission = async (req, res) => {
  const subId = req.params.id;
  try {
    const submission = await Submission.findOneAndDelete({ _id: subId });
    if (!submission) {
      res.send({ msg: "submission Not Found" });
    } else {
      res.send(submission);
    }
  } catch (error) {
    res.send(error);
  }
};

const getSubmissions = async (req, res) => {
  try {
    const submission = await Submission.find();
    res.send(submission);
  } catch (error) {
    res.send(error);
  }
};

const getSubmission = async (req, res) => {
  const subId = req.params.id;
  try {
    const submission = await Submission.findOne({ _id: subId });
    if (submission) {
      res.json(submission);
    } else {
      res.json("submission not found");
    }
  } catch (error) {
    res.send(error);
  }
};

module.exports = {
  insertSubmission,
  deleteSubmission,
  updateSubmission,
  getSubmissions,
  getSubmission,
};
```

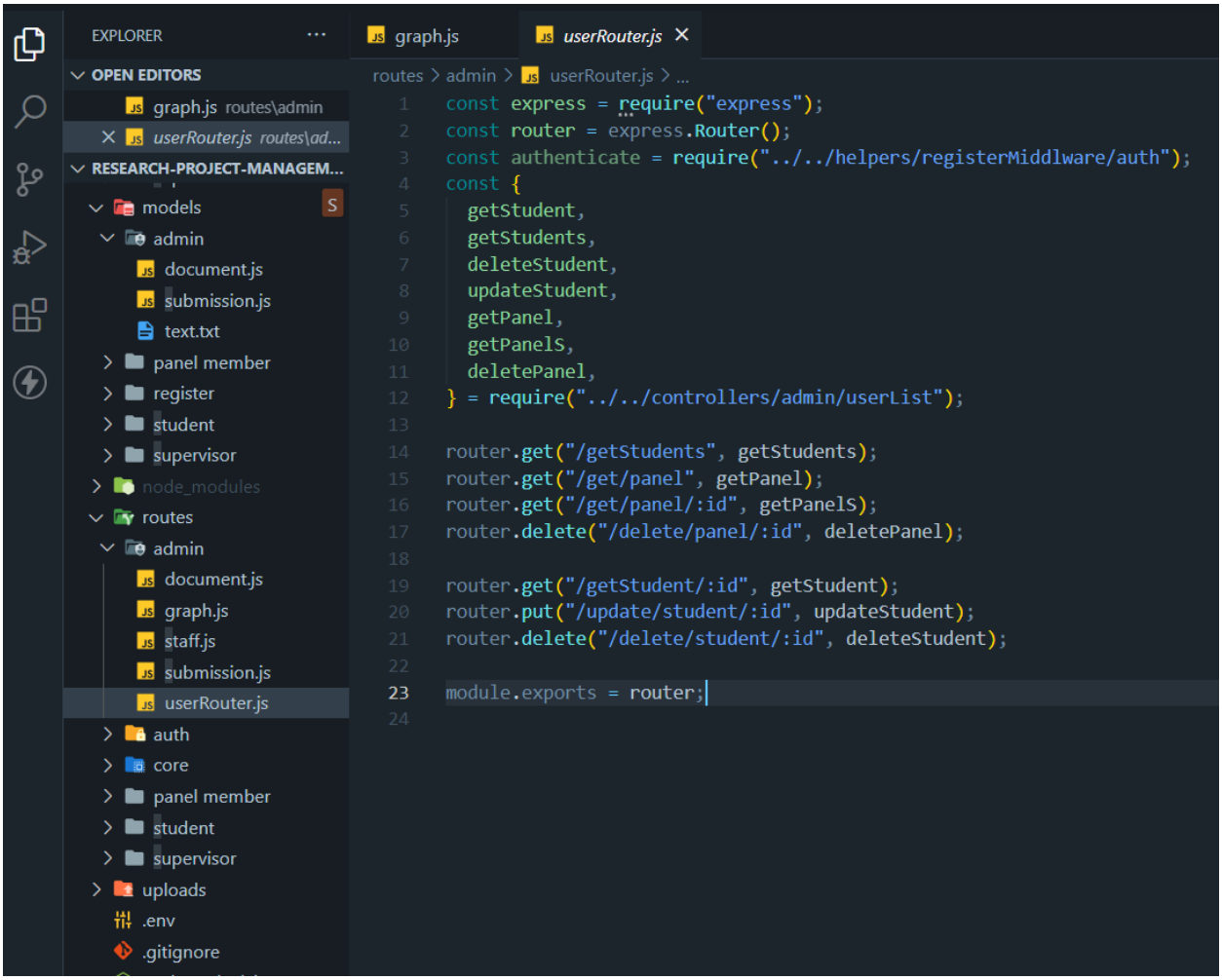
```
const User = require("../../models/register/register");
const Panel = require("../../models/panel member/panelMember");

const getStudents = async (req, res) => {
  try {
    const students = await User.find();
    res.json(students);
  } catch (error) {
    console.log(error);
  }
};

const getPanel = async (req, res) => {
  try {
    const panel = await Panel.find();
    res.json(panel);
  } catch (error) {
    console.log(error);
  }
};

const getPanelS = async (req, res) => {
  const userId = req.params.id;
  try {
    const panel = await Panel.findOne({ _id: userId });
    if (panel) {
      res.json(panel);
    } else {
      res.json("student not found");
    }
  } catch (error) {
    console.log(error);
  }
};

const getStudent = async (req, res) => {
  const userId = req.params.id;
  try {
    const student = await User.findOne({ _id: userId });
    if (student) {
      res.json(student);
    } else {
      res.json("student not found");
    }
  } catch (error) {
    console.log(error);
  }
};

const updateStudent = async (req, res) => {
  const studentId = req.params.id;

  try {
    const student = await User.findOneAndUpdate({ _id: studentId }, req.body);
    res.json(student);
  } catch (error) {
    res.send(error);
  }
};
const deleteStudent = async (req, res) => {
  const studentId = req.params.id;
  try {
    const student = await User.findOneAndDelete({ _id: studentId });
    if (!student) {
      res.json({ msg: "Student Not Found" });
    } else {
      res.json(student);
    }
  } catch (error) {
    res.send(error);
  }
};

const deletePanel = async (req, res) => {
  const panelId = req.params.id;
  try {
    const panel = await Panel.findOneAndDelete({ _id: panelId });
    if (!panel) {
      res.json({ msg: "Panel member Not Found" });
    } else {
      res.json(panel);
    }
  } catch (error) {
    res.send(error);
  }
};

module.exports = {
  getStudent,
  getStudents,
  deleteStudent,
  updateStudent,
  getPanel,
  getPanelS,

  deletePanel,
};
```


EXPLORER
OPEN EDITORS
document.js routes\ad...
RESEARCH-PROJECT-MANAGEM...
controllers
admin
document.js
graph.js
staffList.js
stGroup.js
submission.js
text.txt
userList.js
core
panel member
register
student
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
document.js
submission.js
text.txt
panel member
register
student
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
document.js
routes > admin > document.js > ...
const express = require("express");
const router = express.Router();
const authenticate = require("../../helpers/registerMiddlware/auth");
const { upload } = require("../../helpers/admin/multer");
//const { upload } = require("../../helpers/admin/cloudinary");
const {
insertDocument,
updateDocument,
deleteDocument,
getDocumentsST,
getDocument,
insertMarkingSchema,
insertPresentation,
getDocumentAll,
} = require("../../controllers/admin/document");
const { insertSubmission } = require("../../controllers/student/submission");
router.post("/insert", upload.single("document"), insertDocument);
router.post(
"/student/insert/submission",
upload.single("document"),
insertSubmission
);
router.post("/insert/schema", upload.single("document"), insertMarkingSchema);
router.post(
"/insert/presentation",
upload.single("document"),
insertPresentation
);
router.get("/get", getDocumentsST);
router.get("/get/all", getDocumentAll);
router.put("/update/:id", updateDocument);
router.get("/get/:id", getDocument);
router.delete("/delete/:id", deleteDocument);
module.exports = router;

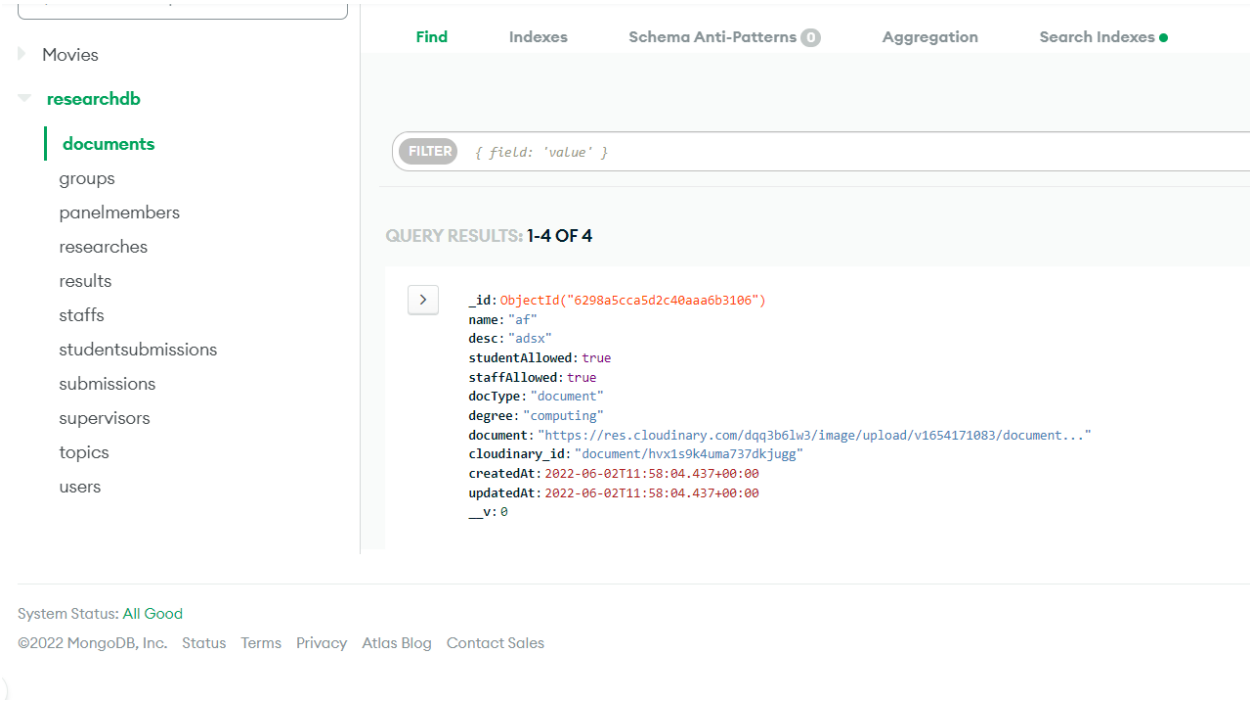

EXPLORER
OPEN EDITORS
graph.js routes\admin
RESEARCH-PROJECT-MANAGEM...
student
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
document.js
submission.js
text.txt
panel member
register
student
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
userRouter.js
auth
core
graph.js
routes > admin > graph.js > ...
const express = require("express");
const router = express.Router();
const { graphData } = require("../../controllers/admin/graph");
router.get("/graph", graphData);
module.exports = router;

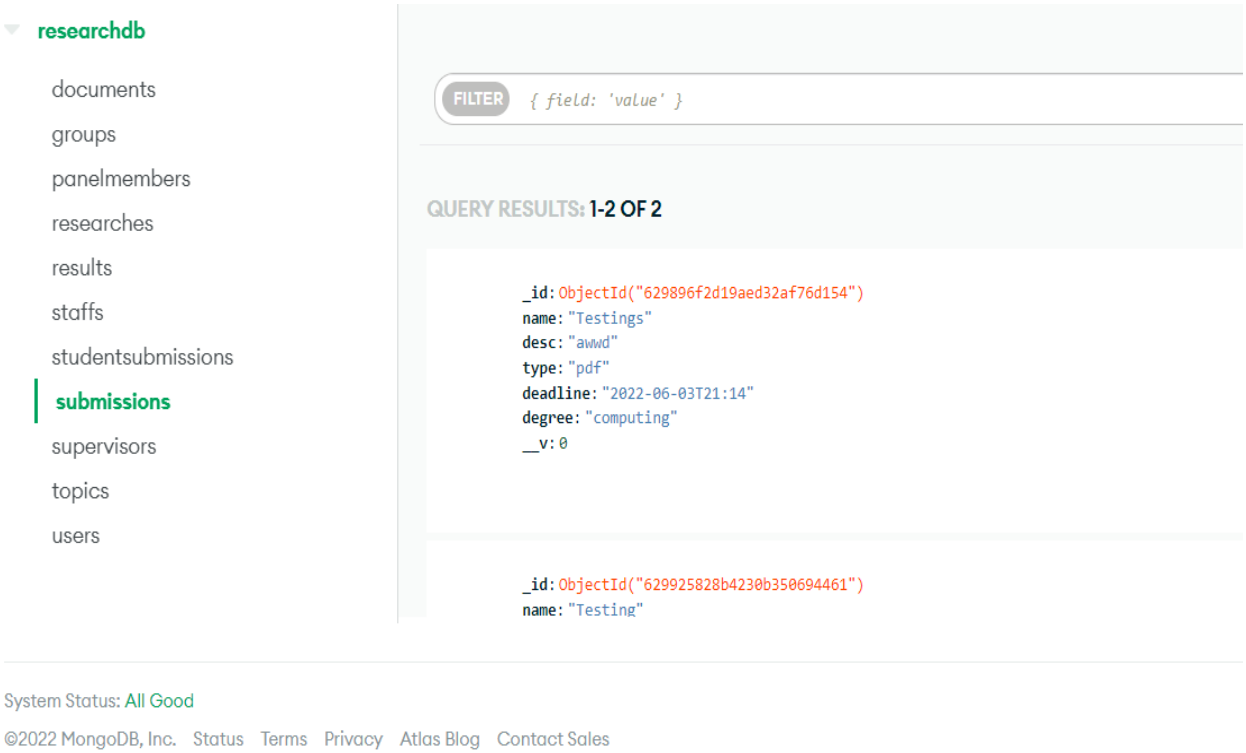

EXPLORER
OPEN EDITORS
graph.js routes\admin
submission.js routes\ad...
RESEARCH-PR...
models
admin
document.js
submission.js
text.txt
panel member
register
student
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
userRouter.js
auth
core
panel member
student
supervisor
uploads
.env
graph.js
submission.js
routes > admin > submission.js > ...
1 const express = require("express");
2 const router = express.Router();
3
4 const authenticate = require("../../helpers/registerMiddlware/auth");
5
6 const {
7 insertSubmission,
8 deleteSubmission,
9 updateSubmission,
10 getSubmissions,
11 getSubmission,
12 } = require("../../controllers/admin/submission");
13
14 router.post("/insert/submission", insertSubmission);
15 router.delete("/delete/submission/:id", deleteSubmission);
16 router.put("/update/submission/:id", updateSubmission);
17 router.get("/get/submission/:id", authenticate, getSubmission);
18 router.get("/get/submission", getSubmissions);
19
20 module.exports = router;
21


OPEN EDITORS
graph.js routes\admin
staff.js routes\admin
RESEARCH-PR...
student
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
document.js
submission.js
text.txt
panel member
register
student
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
routes > admin > staff.js > ...
1 const express = require("express");
2 const router = express.Router();
3 const authenticate = require("../../helpers/registerMiddlware/auth");
4 const {
5 getStaff,
6 getStaffs,
7 deleteStaff,
8 updateStaff,
9 } = require("../../controllers/admin/staffList");
10
11 router.get("/get/staff", authenticate, getStaffs);
12 router.get("/get/staff/:id", authenticate, getStaff);
13 router.put("/update/staff/:id", authenticate, updateStaff);
14 router.delete("/delete/staff/:id", authenticate, deleteStaff);
15
16 module.exports = router;
17


EXPLORER
OPEN EDITORS
graph.js routes\admin
userRouter.js routes\ad...
RESEARCH-PROJECT-MANAGEM...
models
admin
document.js
submission.js
text.txt
panel member
register
student
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
userRouter.js
auth
core
panel member
student
supervisor
uploads
.env
.gitignore
graph.js
userRouter.js
routes > admin > userRouter.js > ...
1 const express = require("express");
2 const router = express.Router();
3 const authenticate = require("../../helpers/registerMiddlware/auth");
4 const {
5 getStudent,
6 getStudents,
7 deleteStudent,
8 updateStudent,
9 getPanel,
10 getPanelS,
11 deletePanel,
12 } = require("../../controllers/admin/userList");
13
14 router.get("/getStudents", getStudents);
15 router.get("/get/panel", getPanel);
16 router.get("/get/panel/:id", getPanelS);
17 router.delete("/delete/panel/:id", deletePanel);
18
19 router.get("/getStudent/:id", getStudent);
20 router.put("/update/student/:id", updateStudent);
21 router.delete("/delete/student/:id", deleteStudent);
22
23 module.exports = router;
24

Screenshots of the MongoDB Schemas



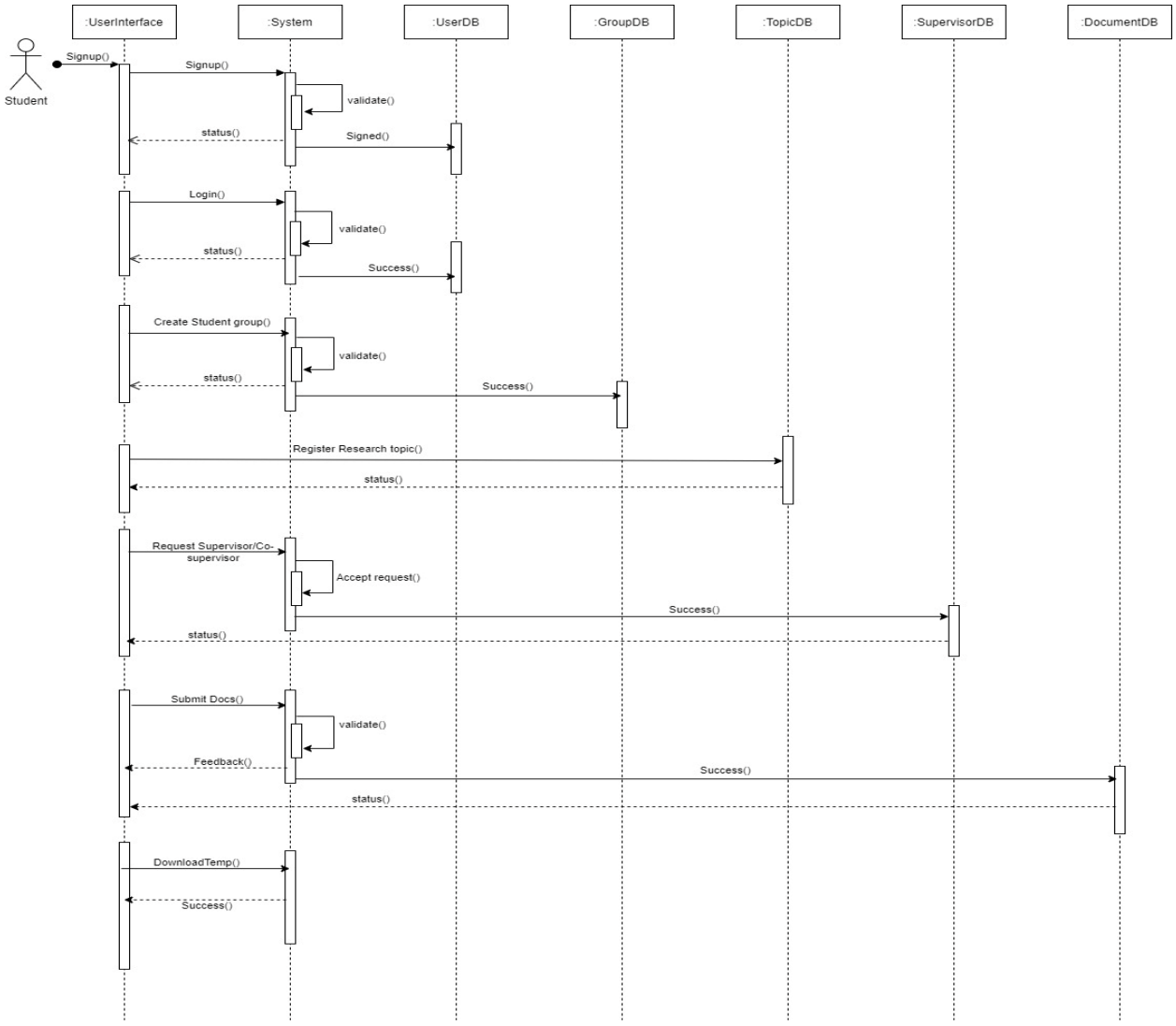


Component Description

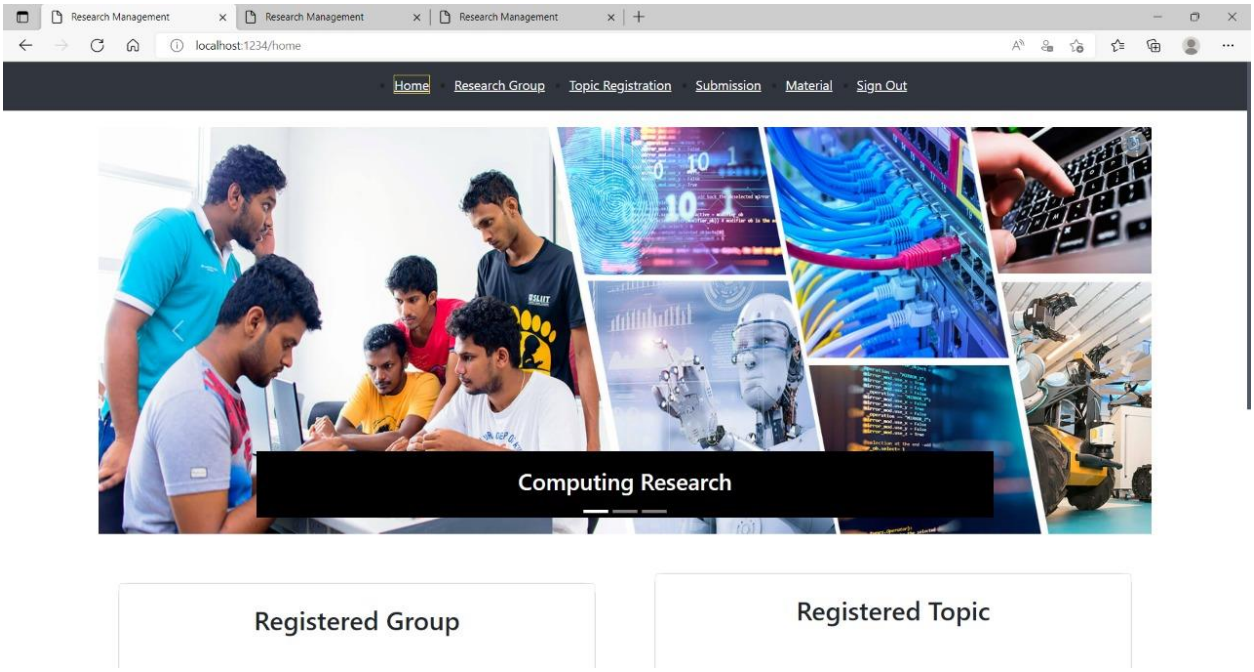
IT20134280 S. Ishalini

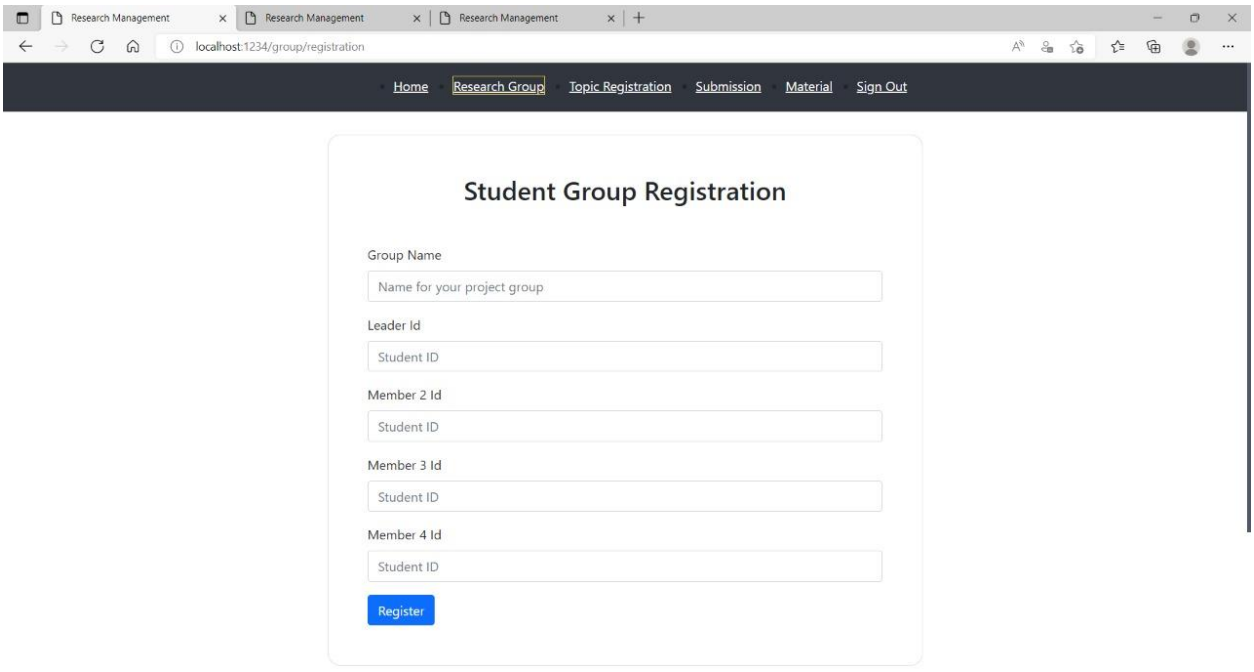
Student Role

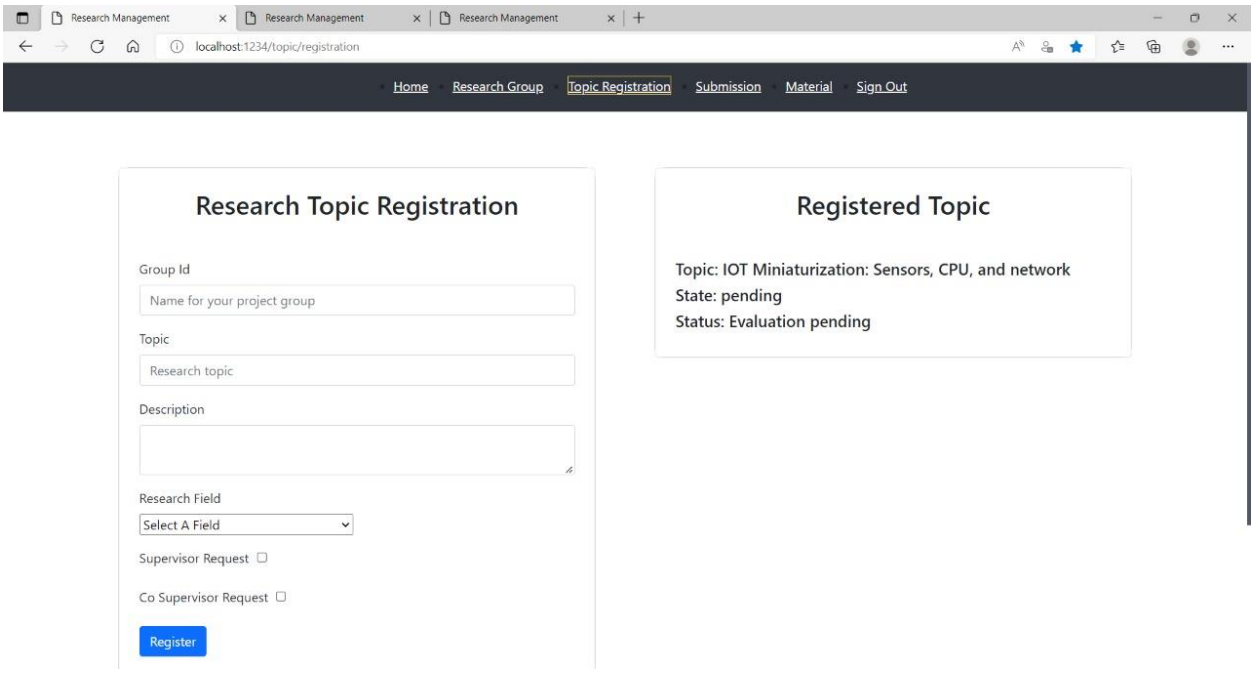
Description: In this system other main role is a student. Initially student should register and login to the system for access it. They can create a student group and register their research topic. They can request supervisor and co-supervisor from this system who are interested in their research area. Students can download templates which have been published by the administrator. After doing their course work, they can submit their document.

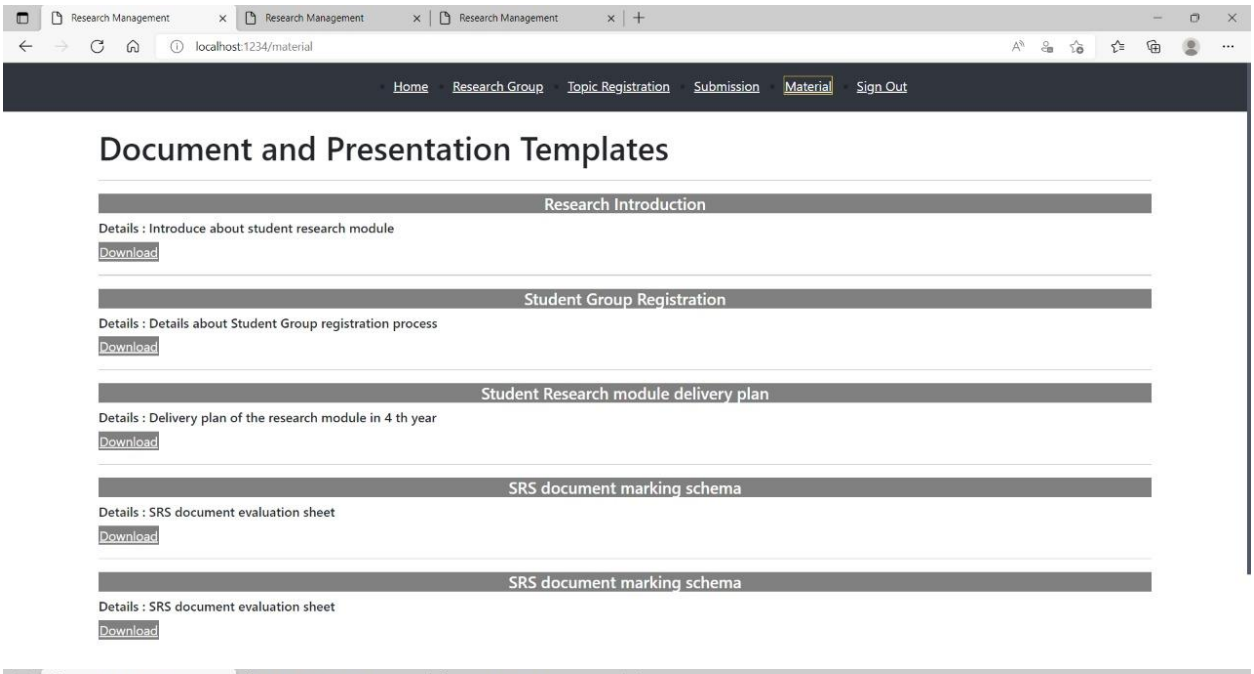


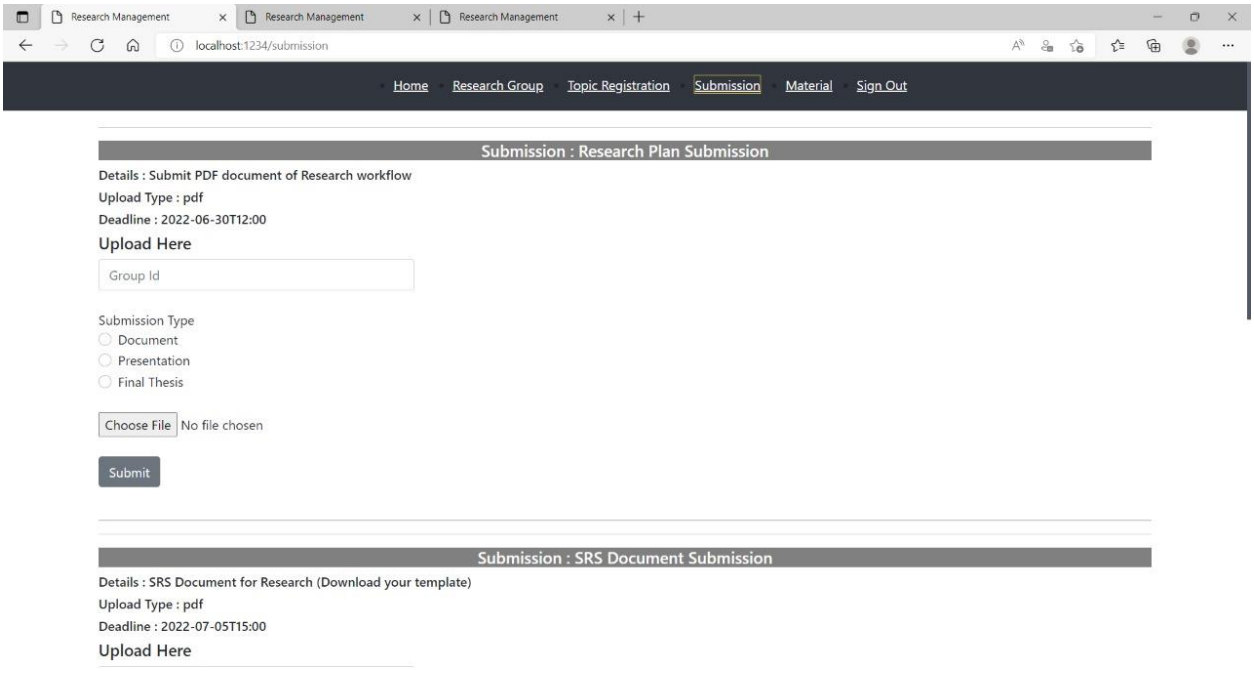
Screenshots of the Frontend

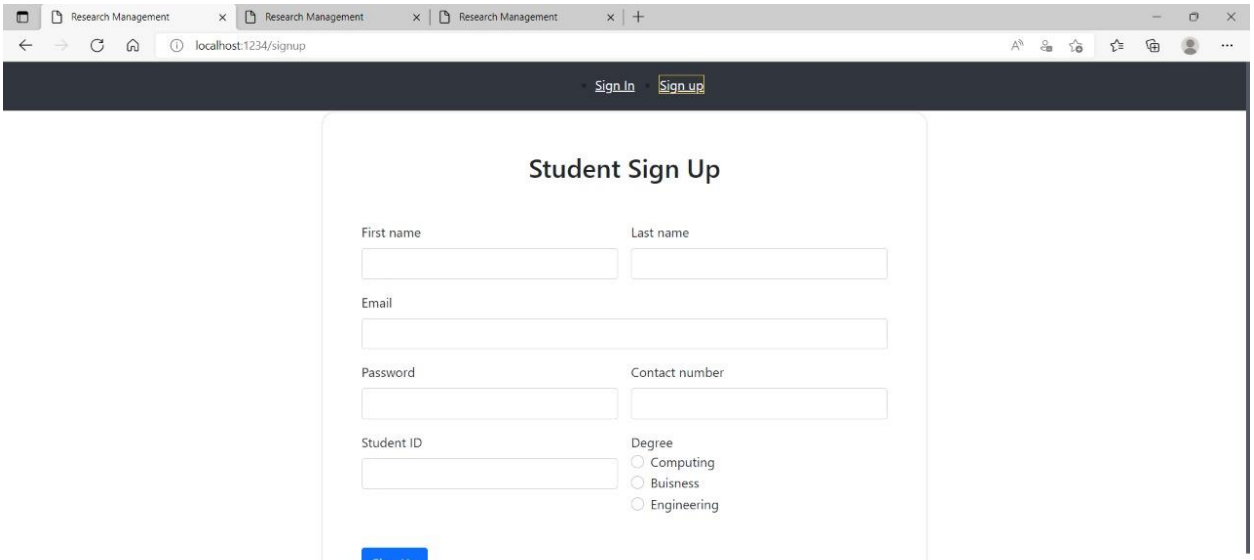


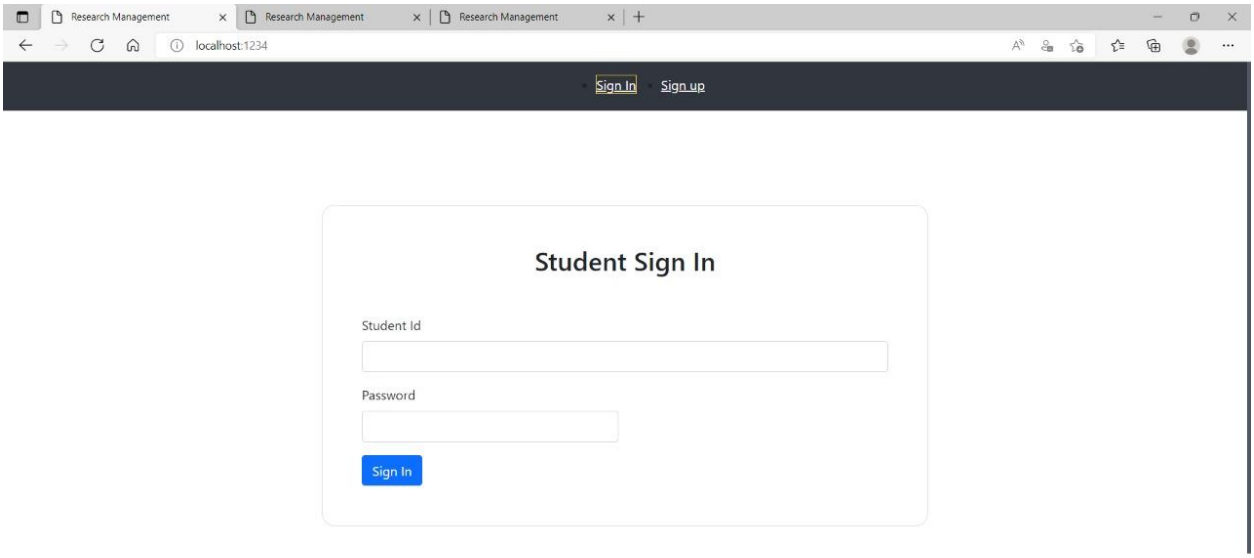



Research Management
Research Management
Research Management
localhost:1234/topic/registration
Home
Research Group
Topic Registration
Submission
Material
Sign Out
Research Topic Registration
Group Id
Name for your project group
Topic
Research topic
Description
Research Field
Select A Field
Supervisor Request
Co Supervisor Request
Register
Registered Topic
Topic: IOT Miniaturization: Sensors, CPU, and network
State: pending
Status: Evaluation pending

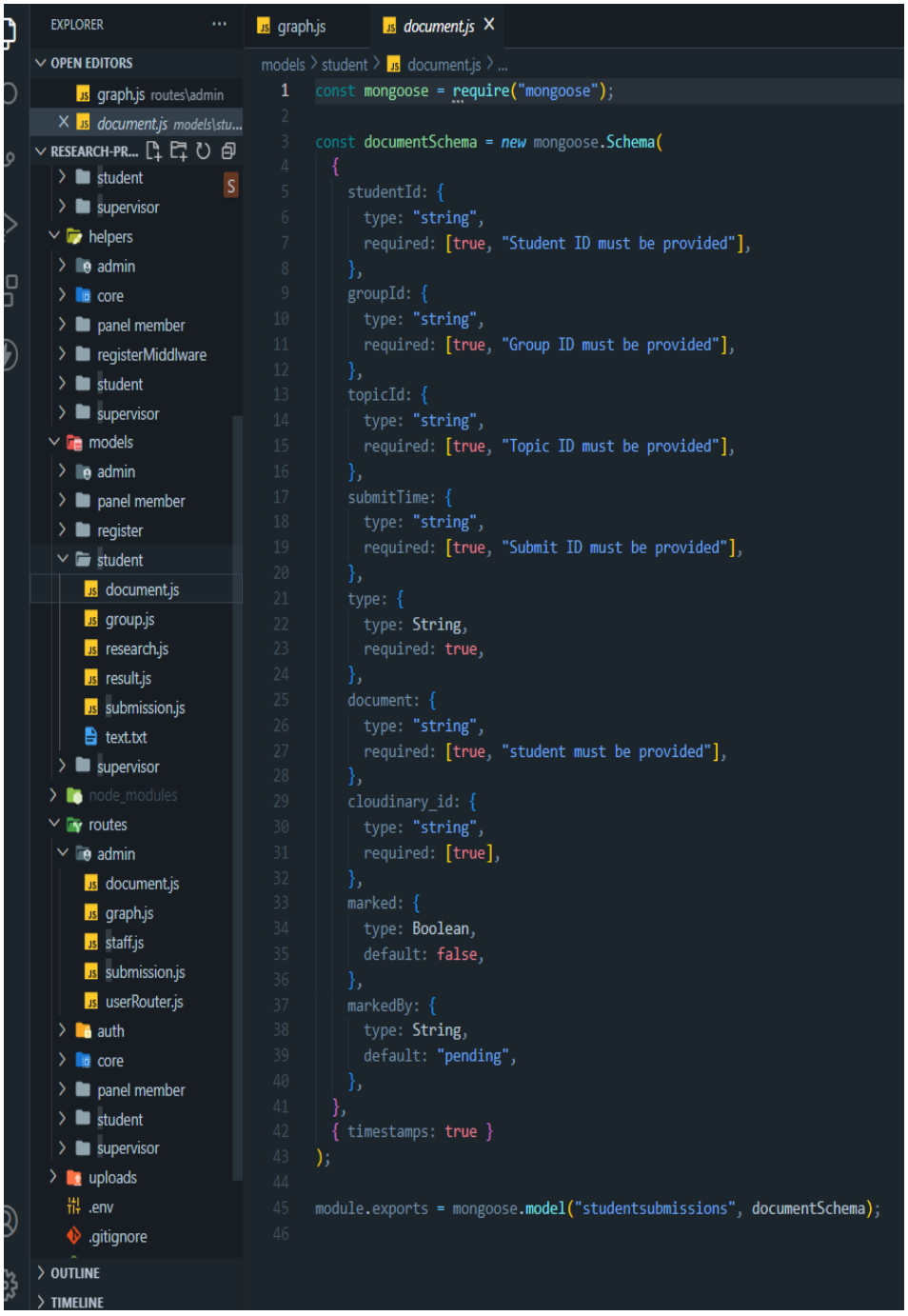

Research Management
Research Management
Research Management
localhost:1234/material
Home
Research Group
Topic Registration
Submission
Material
Sign Out
Document and Presentation Templates
Research Introduction
Details : Introduce about student research module
Download
Student Group Registration
Details : Details about Student Group registration process
Download
Student Research module delivery plan
Details : Delivery plan of the research module in 4 th year
Download
SRS document marking schema
Details : SRS document evaluation sheet
Download
SRS document marking schema
Details : SRS document evaluation sheet
Download

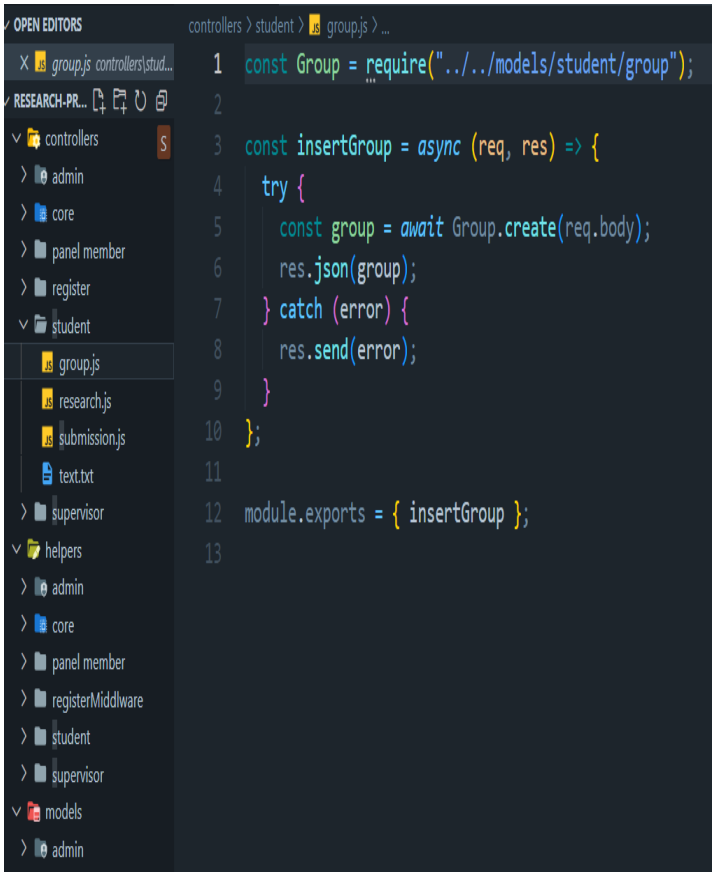

Research Management
Research Management
Research Management
localhost:1234/submission
Home
Research Group
Topic Registration
Submission
Material
Sign Out
Submission : Research Plan Submission
Details : Submit PDF document of Research workflow
Upload Type : pdf
Deadline : 2022-06-30T12:00
Upload Here
Group Id
Submission Type
Document
Presentation
Final Thesis
Choose File
No file chosen
Submit
Submission : SRS Document Submission
Details : SRS Document for Research (Download your template)
Upload Type : pdf
Deadline : 2022-07-05T15:00
Upload Here

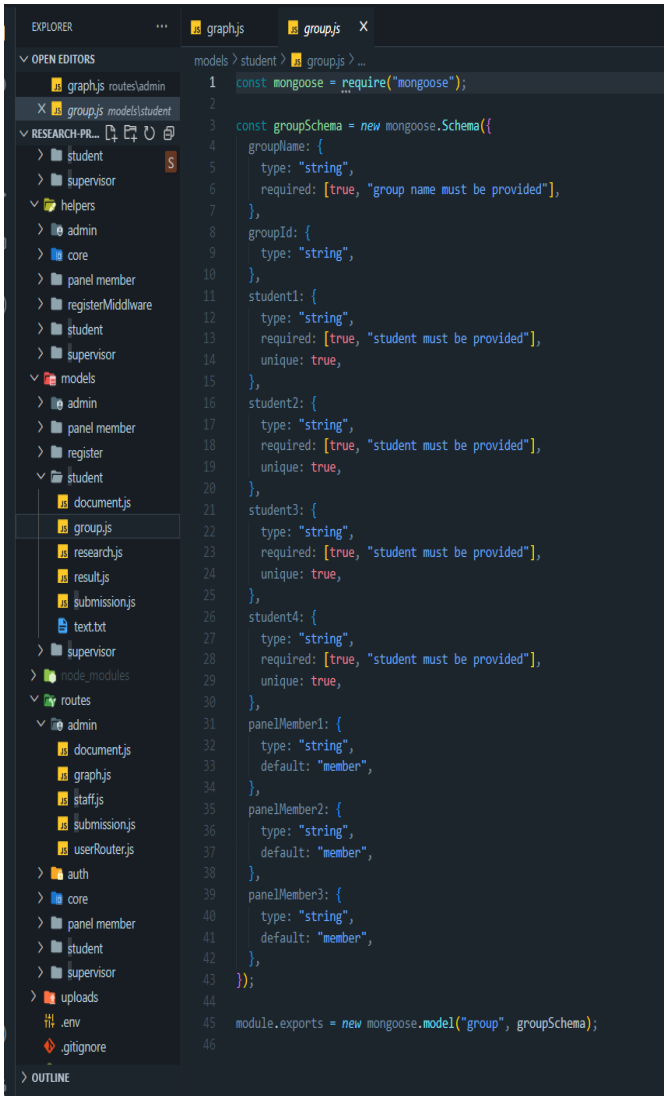

Research Management
Research Management
Research Management
localhost:1234/signup
Sign In
Sign up
Student Sign Up
First name
Last name
Email
Password
Contact number
Student ID
Degree
Computing
Buisness
Engineering
Sign Up


Research Management
Research Management
Research Management
localhost:1234
Sign In
Sign up
Student Sign In
Student Id
Password
Sign In

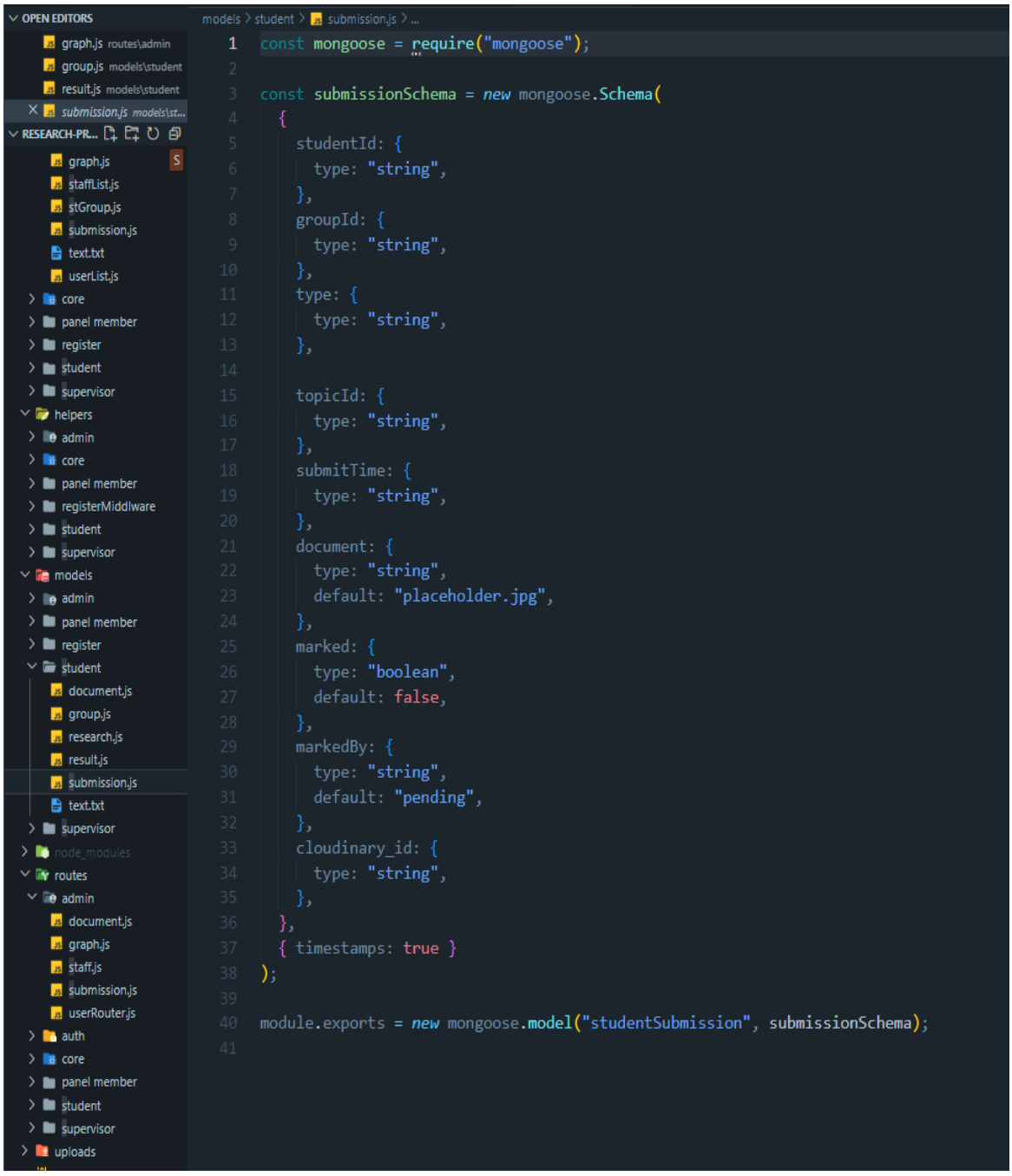
Screenshots/ Code Snippets of the Backend:

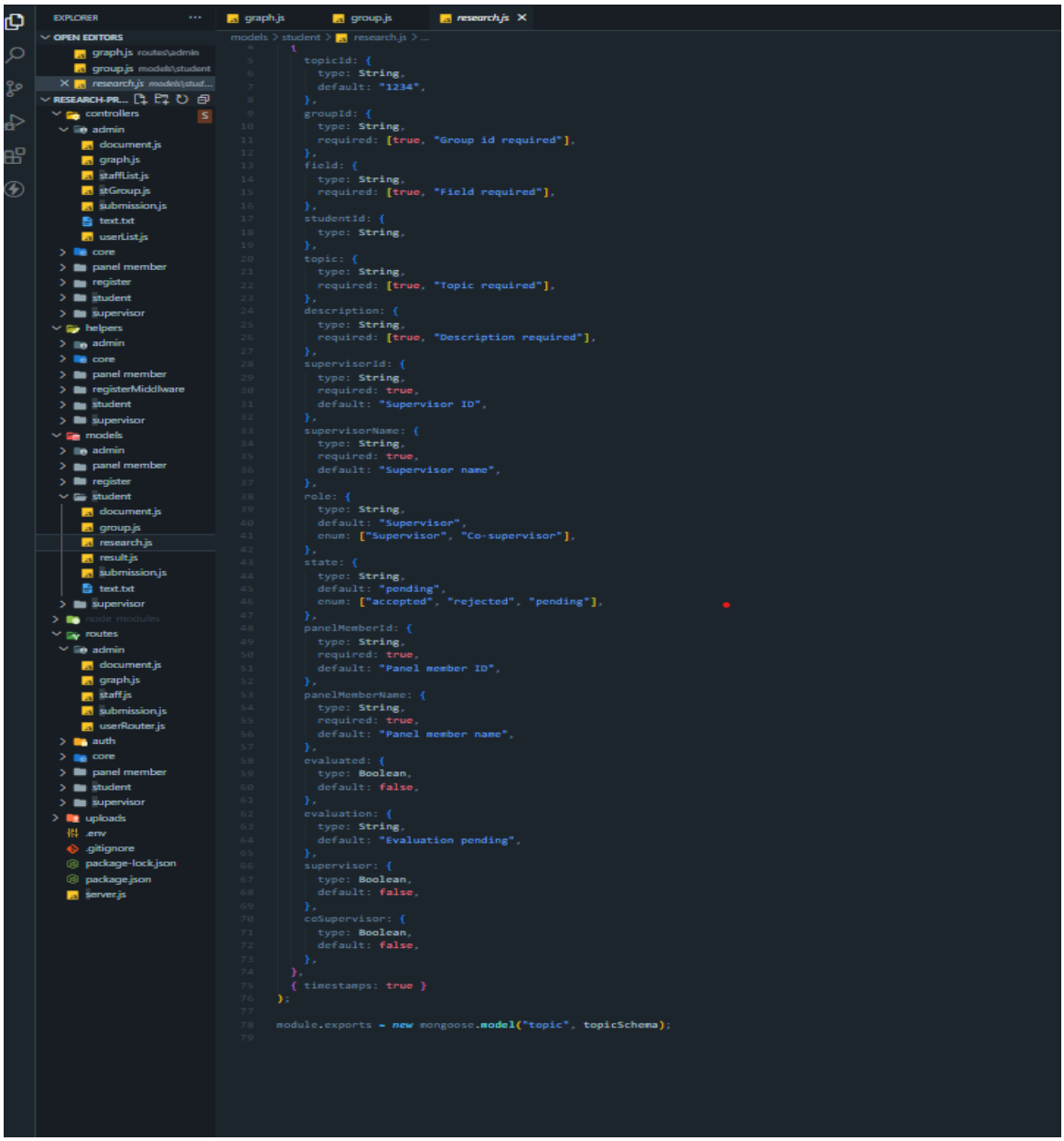

EXPLORER
OPEN EDITORS
graph.js routes\admin
document.js models\stu...
RESEARCH-PR...
student
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
panel member
register
student
document.js
group.js
research.js
result.js
submission.js
text.txt
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
userRouter.js
auth
core
panel member
student
supervisor
uploads
.env
.gitignore
OUTLINE
TIMELINE
graph.js
document.js
models > student > document.js > ...
1 const mongoose = require("mongoose");
2
3 const documentSchema = new mongoose.Schema(
4 {
5 studentId: {
6 type: "string",
7 required: [true, "Student ID must be provided"],
8 },
9 groupId: {
10 type: "string",
11 required: [true, "Group ID must be provided"],
12 },
13 topicId: {
14 type: "string",
15 required: [true, "Topic ID must be provided"],
16 },
17 submitTime: {
18 type: "string",
19 required: [true, "Submit ID must be provided"],
20 },
21 type: {
22 type: String,
23 required: true,
24 },
25 document: {
26 type: "string",
27 required: [true, "student must be provided"],
28 },
29 cloudinary_id: {
30 type: "string",
31 required: [true],
32 },
33 marked: {
34 type: Boolean,
35 default: false,
36 },
37 markedBy: {
38 type: String,
39 default: "pending",
40 },
41 },
42 { timestamps: true }
43);
44
45 module.exports = mongoose.model("studentsubmissions", documentSchema);
46

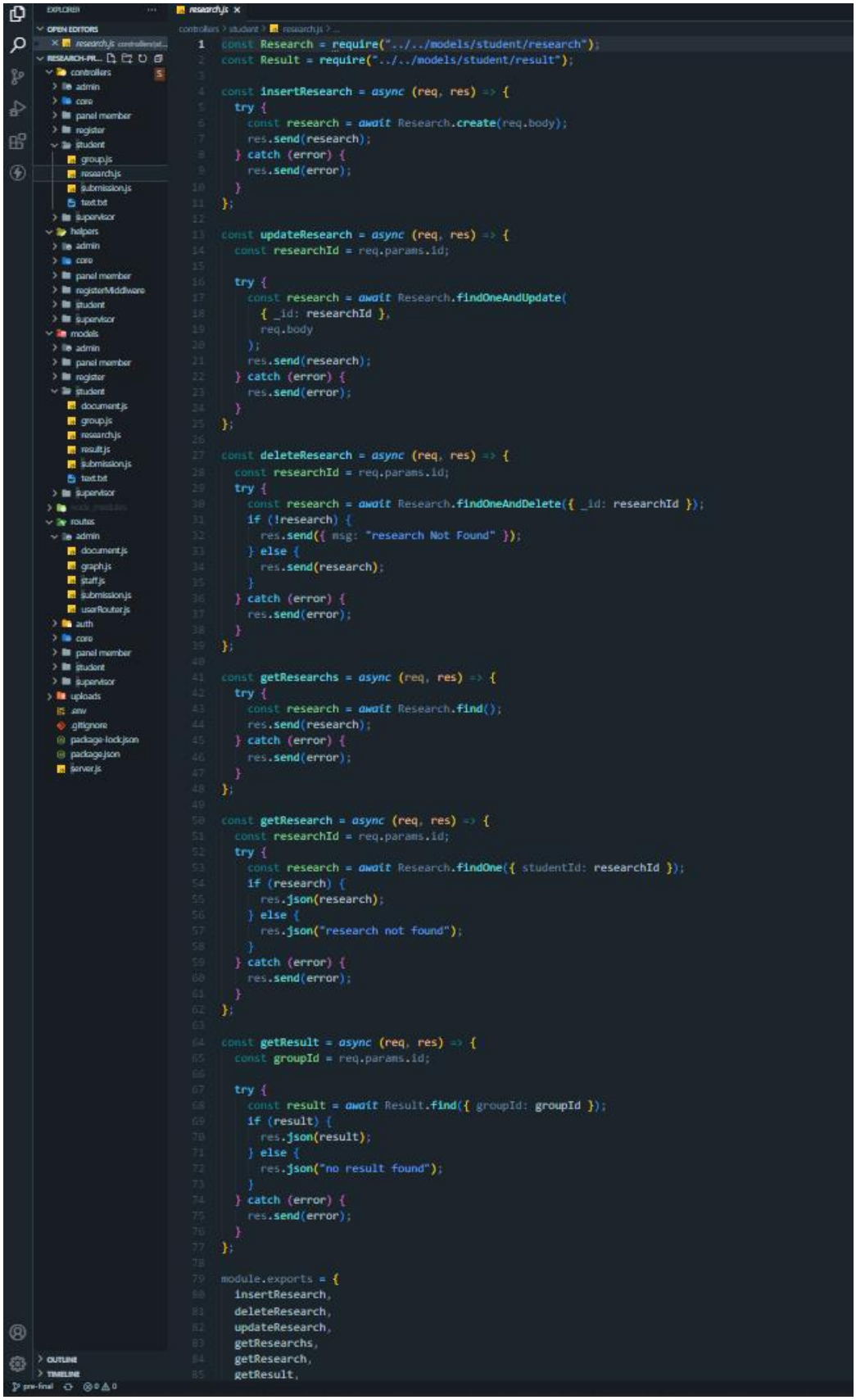

OPEN EDITORS
group.js controllers\stud...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
group.js
research.js
submission.js
text.txt
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
controllers > student > group.js > ...
1 const Group = require("../../models/student/group");
2
3 const insertGroup = async (req, res) => {
4 try {
5 const group = await Group.create(req.body);
6 res.json(group);
7 } catch (error) {
8 res.send(error);
9 }
10 };
11
12 module.exports = { insertGroup };
13


EXPLORER
graph.js
group.js
OPEN EDITORS
graph.js routes\admin
group.js models\student
RESEARCH-PR...
student
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
panel member
register
student
document.js
group.js
research.js
result.js
submission.js
text.txt
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
userRouter.js
auth
core
panel member
student
supervisor
uploads
.env
.gitignore
OUTLINE
models > student > group.js > ...
1 const mongoose = require("mongoose");
2
3 const groupSchema = new mongoose.Schema({
4 groupName: {
5 type: "string",
6 required: [true, "group name must be provided"],
7 },
8 groupId: {
9 type: "string",
10 },
11 student1: {
12 type: "string",
13 required: [true, "student must be provided"],
14 unique: true,
15 },
16 student2: {
17 type: "string",
18 required: [true, "student must be provided"],
19 unique: true,
20 },
21 student3: {
22 type: "string",
23 required: [true, "student must be provided"],
24 unique: true,
25 },
26 student4: {
27 type: "string",
28 required: [true, "student must be provided"],
29 unique: true,
30 },
31 panelMember1: {
32 type: "string",
33 default: "member",
34 },
35 panelMember2: {
36 type: "string",
37 default: "member",
38 },
39 panelMember3: {
40 type: "string",
41 default: "member",
42 },
43 });
44
45 module.exports = new mongoose.model("group", groupSchema);
46

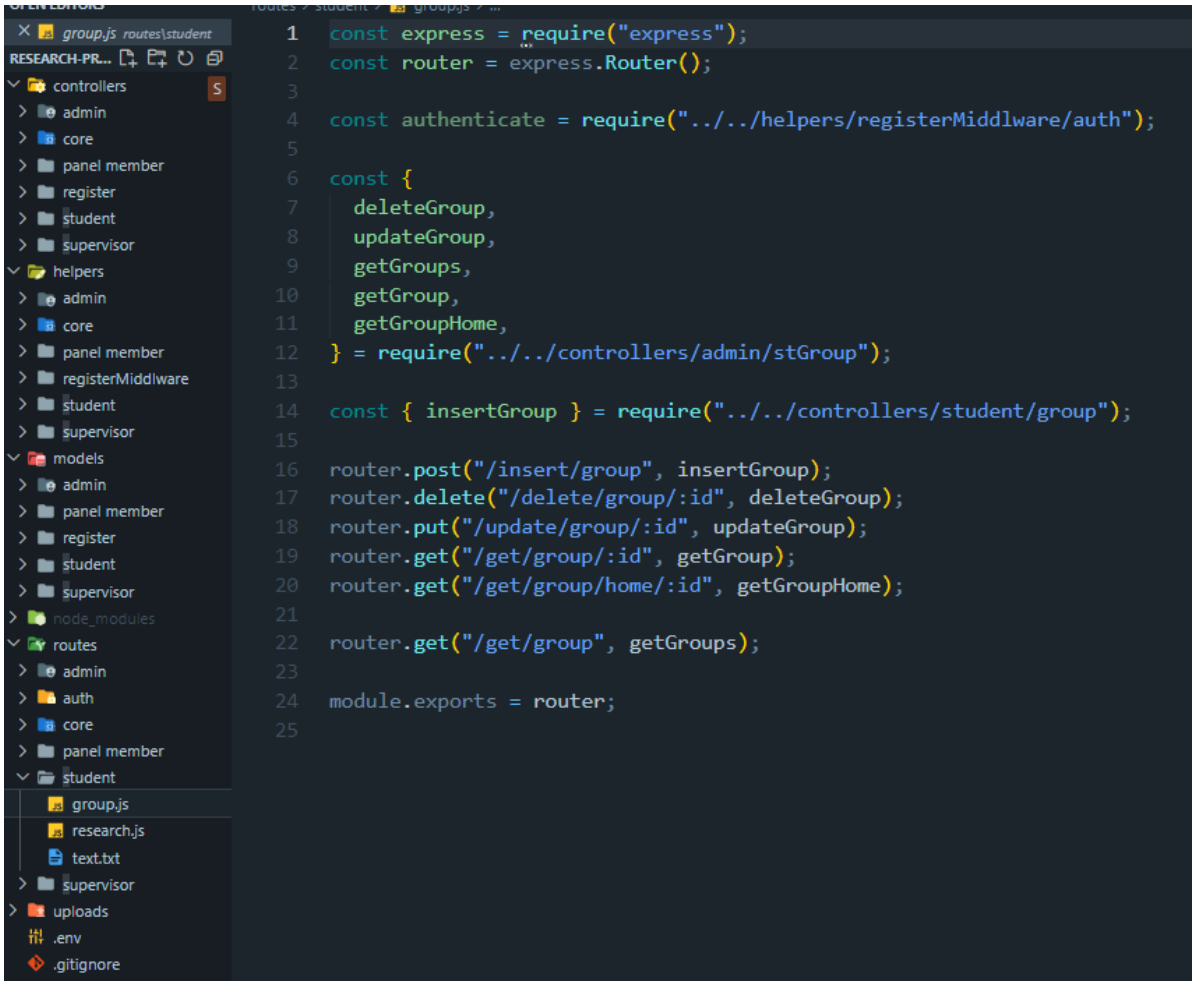

OPEN EDITORS 1 UNSAVED
graph.js routes\admin
group.js models\student
result.js models\student
RESEARCH-PROJECT-MANAGEM...
controllers
admin
document.js
graph.js
staffList.js
stGroup.js
submission.js
test.txt
userList.js
core
panel member
register
student
supervisor
helpers
admin
core
panel member
registerMiddleware
student
supervisor
models
admin
panel member
register
student
document.js
group.js
research.js
result.js
submission.js
test.txt
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
userRouter.js
auth
core
panel member
student
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
models > student > result.js > resultSchema
const mongoose = require("mongoose");
const resultSchema = new mongoose.Schema(
{
groupId: {
type: "string",
unique: true,
},
student1: {
studentId: {
type: String,
default: "No ID",
},
documentMarks: {
type: Number,
default: 0,
},
presentationMarks: {
type: Number,
default: 0,
},
finalThesisMarks: {
type: Number,
default: 0,
},
},
student2: {
studentId: {
type: String,
default: "No ID",
},
documentMarks: {
type: Number,
default: 0,
},
presentationMarks: {
type: Number,
default: 0,
},
finalThesisMarks: {
type: Number,
default: 0,
},
},
student3: {
studentId: {
type: String,
default: "No ID",
},
documentMarks: {
type: Number,
default: 0,
},
presentationMarks: {
type: Number,
default: 0,
},
finalThesisMarks: {
type: Number,
default: 0,
},
},
student4: {
studentId: {
type: String,
default: "No ID",
},
documentMarks: {
type: Number,
default: 0,
},
presentationMarks: {
type: Number,
default: 0,
},
finalThesisMarks: {
type: Number,
default: 0,
},
},
},
{ timestamps: true }
);

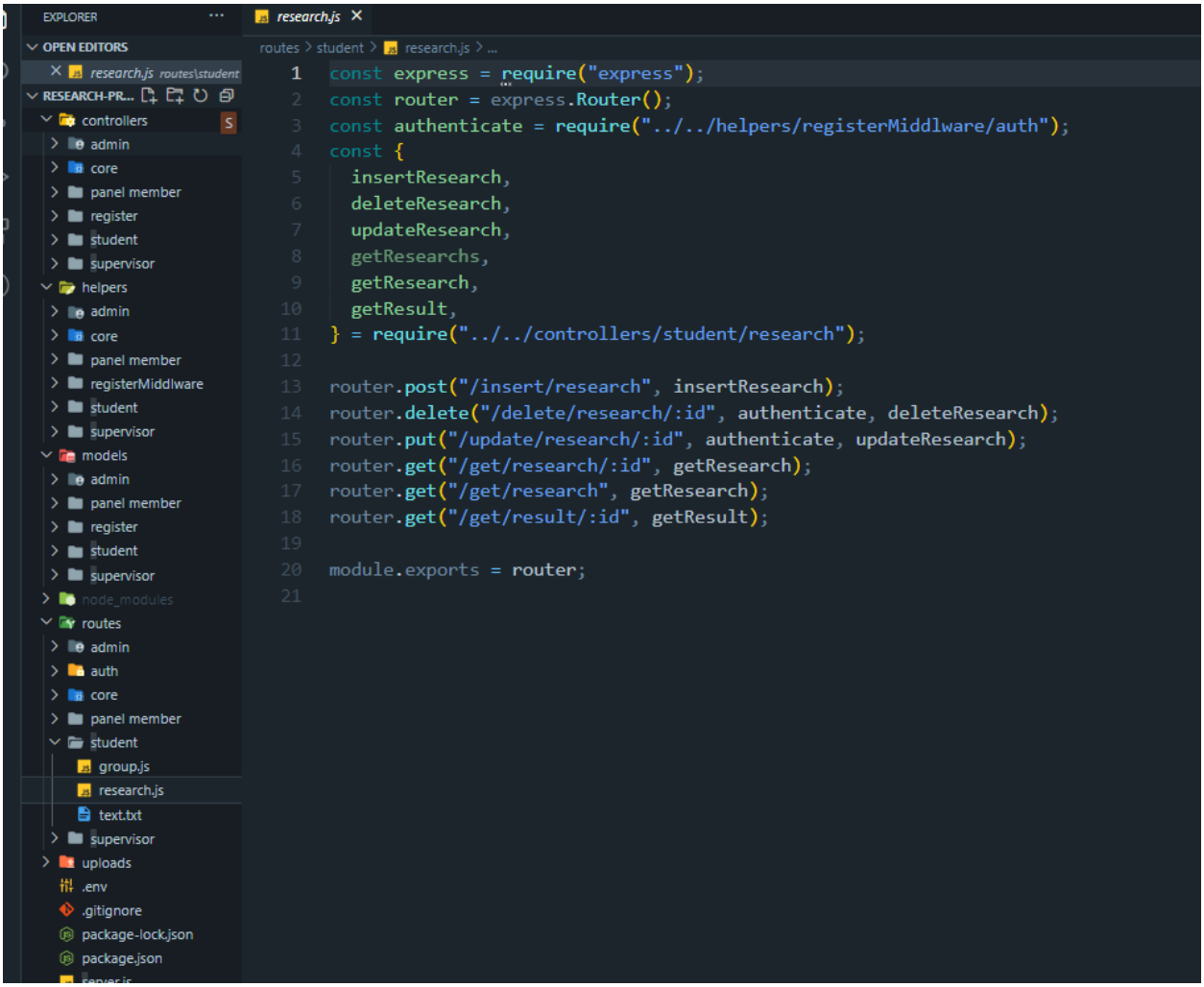

OPEN EDITORS
graph.js routes\admin
group.js models\student
result.js models\student
submission.js models\st...
RESEARCH-PR...
graph.js
staffList.js
stGroup.js
submission.js
text.txt
userList.js
core
panel member
register
student
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
panel member
register
student
document.js
group.js
research.js
result.js
submission.js
text.txt
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
userRouter.js
auth
core
panel member
student
supervisor
uploads
models > student > submission.js > ...
1 const mongoose = require("mongoose");
3 const submissionSchema = new mongoose.Schema(
4 {
5 studentId: {
6 type: "string",
7 },
8 groupId: {
9 type: "string",
10 },
11 type: {
12 type: "string",
13 },
14
15 topicId: {
16 type: "string",
17 },
18 submitTime: {
19 type: "string",
20 },
21 document: {
22 type: "string",
23 default: "placeholder.jpg",
24 },
25 marked: {
26 type: "boolean",
27 default: false,
28 },
29 markedBy: {
30 type: "string",
31 default: "pending",
32 },
33 cloudinary_id: {
34 type: "string",
35 },
36 },
37 { timestamps: true }
38);
39
40 module.exports = new mongoose.model("studentSubmission", submissionSchema);
41


EXPLORER
OPEN EDITORS
graph.js routes\admin
group.js models\student
research.js models\stud...
RESEARCH-PR...
controllers
admin
document.js
graph.js
staffList.js
stGroup.js
submission.js
text.txt
userList.js
core
panel member
register
student
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
panel member
register
student
document.js
group.js
research.js
result.js
submission.js
text.txt
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
userRouter.js
auth
core
panel member
student
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
graph.js
group.js
research.js
models > student > research.js > ...
{
topicId: {
type: String,
default: "1234",
},
groupId: {
type: String,
required: [true, "Group id required"],
},
field: {
type: String,
required: [true, "Field required"],
},
studentId: {
type: String,
},
topic: {
type: String,
required: [true, "Topic required"],
},
description: {
type: String,
required: [true, "Description required"],
},
supervisorId: {
type: String,
required: true,
default: "Supervisor ID",
},
supervisorName: {
type: String,
required: true,
default: "Supervisor name",
},
role: {
type: String,
default: "Supervisor",
enum: ["Supervisor", "Co-supervisor"],
},
state: {
type: String,
default: "pending",
enum: ["accepted", "rejected", "pending"],
},
panelMemberId: {
type: String,
required: true,
default: "Panel member ID",
},
panelMemberName: {
type: String,
required: true,
default: "Panel member name",
},
evaluated: {
type: Boolean,
default: false,
},
evaluation: {
type: String,
default: "Evaluation pending",
},
supervisor: {
type: Boolean,
default: false,
},
coSupervisor: {
type: Boolean,
default: false,
},
},
{ timestamps: true }
);
module.exports = new mongoose.model("topic", topicSchema);

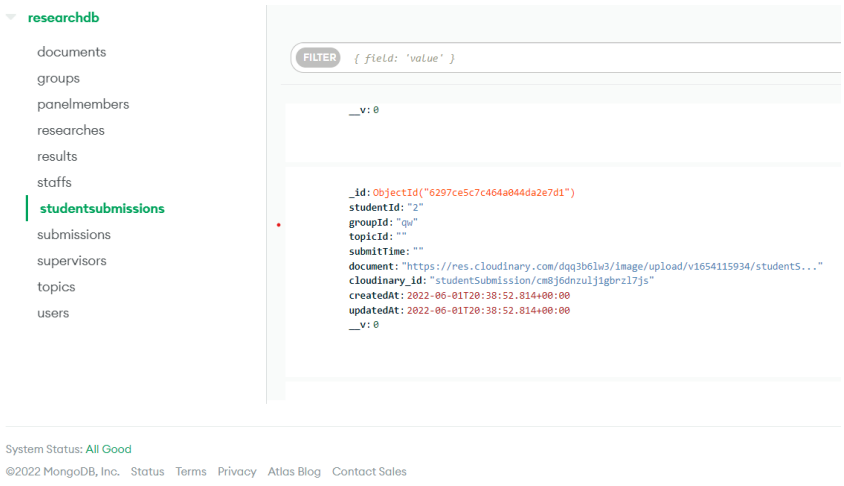

EXPLORER
OPEN EDITORS
RESEARCH-PR...
controllers
admin
core
panel member
register
student
group.js
research.js
submission.js
text.txt
supervisor
helpers
admin
core
panel member
registerMiddleware
student
supervisor
models
admin
panel member
register
student
document.js
group.js
research.js
result.js
submission.js
text.txt
supervisor
routes
admin
document.js
graph.js
staff.js
submission.js
userRouter.js
auth
core
panel member
student
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
pre-final
research.js
controllers > student > research.js > ...
const Research = require("../../models/student/research");
const Result = require("../../models/student/result");
const insertResearch = async (req, res) => {
try {
const research = await Research.create(req.body);
res.send(research);
} catch (error) {
res.send(error);
}
};
const updateResearch = async (req, res) => {
const researchId = req.params.id;
try {
const research = await Research.findOneAndUpdate(
{ _id: researchId },
req.body
);
res.send(research);
} catch (error) {
res.send(error);
}
};
const deleteResearch = async (req, res) => {
const researchId = req.params.id;
try {
const research = await Research.findOneAndDelete({ _id: researchId });
if (!research) {
res.send({ msg: "research Not Found" });
} else {
res.send(research);
}
} catch (error) {
res.send(error);
}
};
const getResearchs = async (req, res) => {
try {
const research = await Research.find();
res.send(research);
} catch (error) {
res.send(error);
}
};
const getResearch = async (req, res) => {
const researchId = req.params.id;
try {
const research = await Research.findOne({ studentId: researchId });
if (research) {
res.json(research);
} else {
res.json("research not found");
}
} catch (error) {
res.send(error);
}
};
const getResult = async (req, res) => {
const groupId = req.params.id;
try {
const result = await Result.find({ groupId: groupId });
if (result) {
res.json(result);
} else {
res.json("no result found");
}
} catch (error) {
res.send(error);
}
};
module.exports = {
insertResearch,
deleteResearch,
updateResearch,
getResearchs,
getResearch,
getResult,

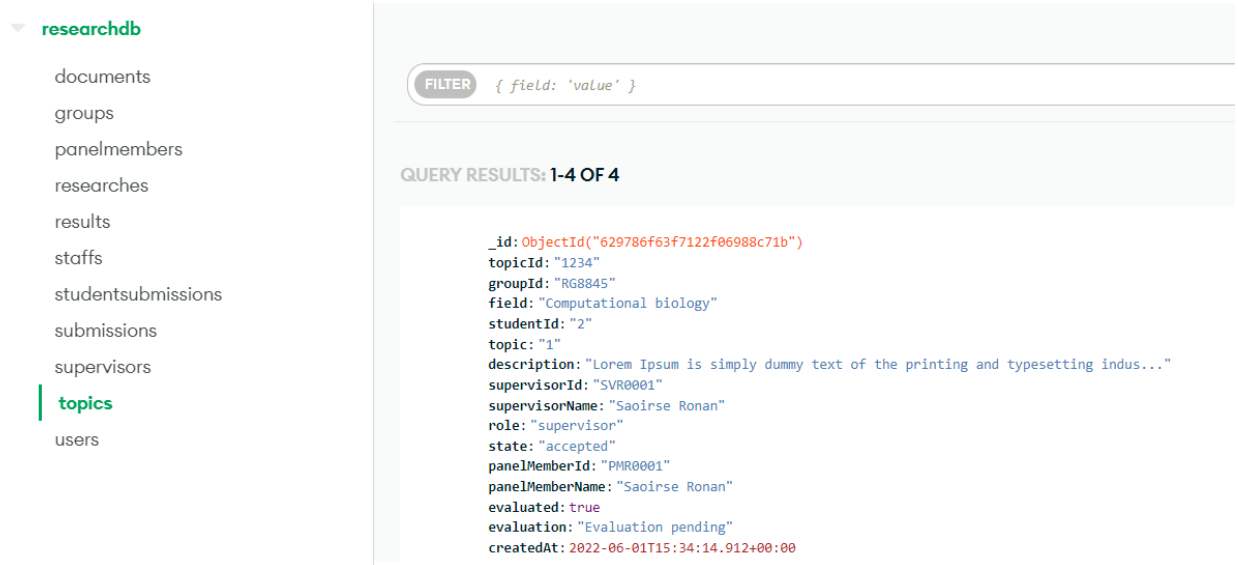

OPEN EDITORS
submission.js controller...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
group.js
research.js
submission.js
text.txt
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
panel member
register
student
document.js
group.js
research.js
result.js
submission.js
text.txt
supervisor
node_modules
routes
admin
document.js
graph.js
staff.js
submission.js
controllers > student > submission.js > ...
1 const Submission = require("../../models/student/submission");
2 const cloudinary = require("../../helpers/admin/cloudinary");
3
4 const insertSubmission = async (req, res) => {
5 try {
6 const result = await cloudinary.uploader.upload(req.file.path, {
7 folder: "studentSubmission",
8 });
9
10 let document = new Submission({
11 studentId: req.body.studentId,
12 groupId: req.body.groupId,
13 topicId: req.body.topicId,
14 type: req.body.type,
15 submitTime: req.body.submitTime,
16 document: result.secure_url,
17 cloudinary_id: result.public_id,
18 });
19
20 console.log(document);
21 documentSaved = await document.save();
22 res.json(documentSaved);
23 } catch (error) {
24 res.json(error);
25 }
26 };
27
28 module.exports = { insertSubmission };
29


group.js routes\student
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
helpers
admin
core
panel member
registerMiddlware
student
supervisor
models
admin
panel member
register
student
supervisor
node_modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
1 const express = require("express");
2 const router = express.Router();
3
4 const authenticate = require("../../helpers/registerMiddlware/auth");
5
6 const {
7 deleteGroup,
8 updateGroup,
9 getGroups,
10 getGroup,
11 getGroupHome,
12 } = require("../../controllers/admin/stGroup");
13
14 const { insertGroup } = require("../../controllers/student/group");
15
16 router.post("/insert/group", insertGroup);
17 router.delete("/delete/group/:id", deleteGroup);
18 router.put("/update/group/:id", updateGroup);
19 router.get("/get/group/:id", getGroup);
20 router.get("/get/group/home/:id", getGroupHome);
21
22 router.get("/get/group", getGroups);
23
24 module.exports = router;
25



Screenshots of the MongoDB Schemas



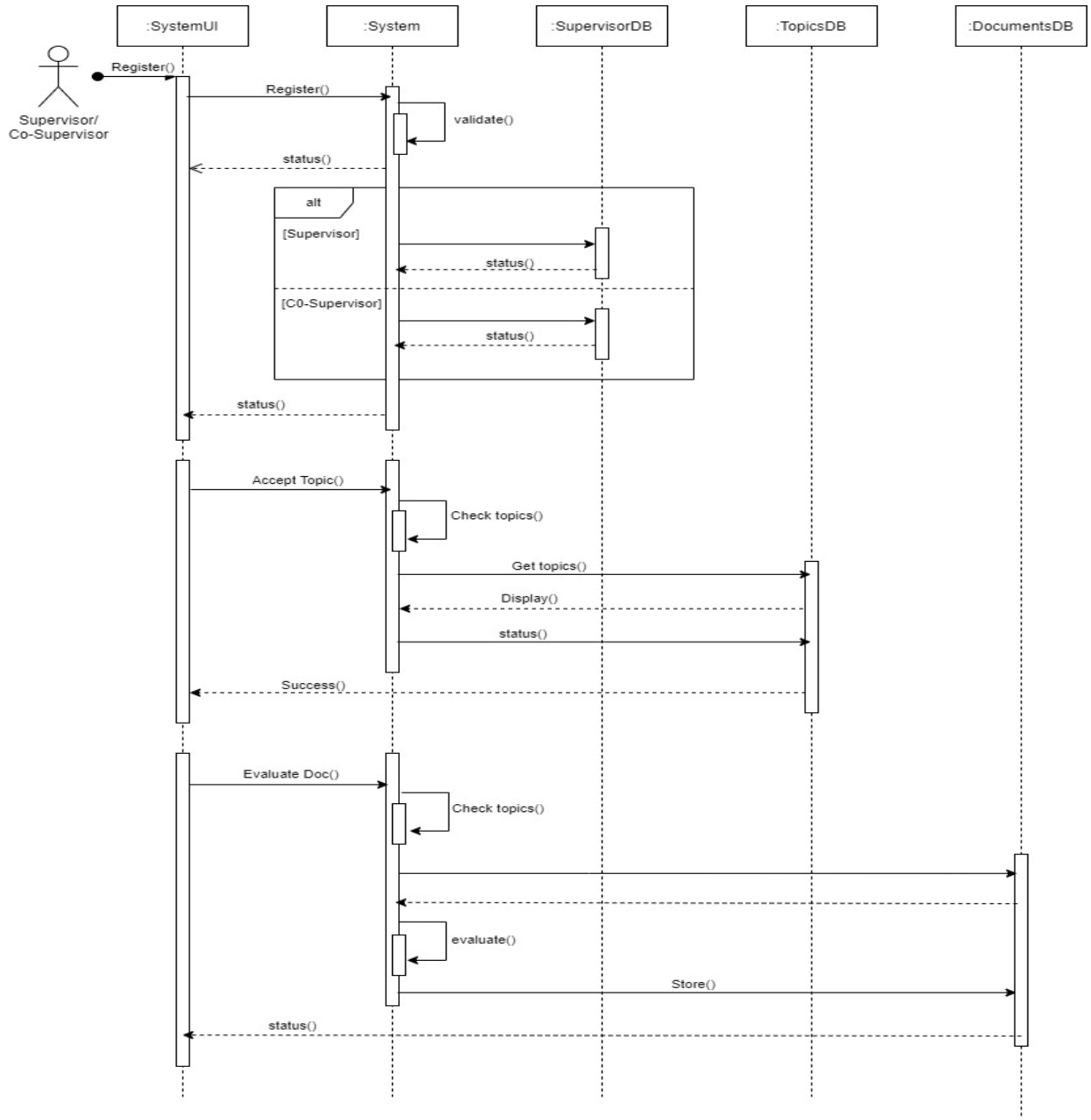


Component Description

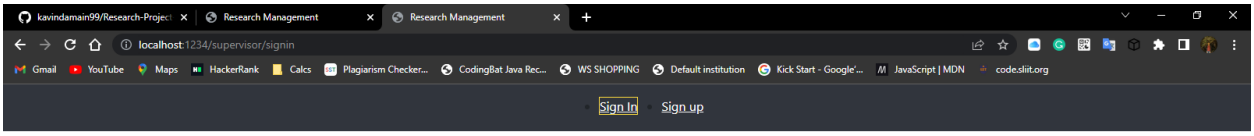
IT20227036 Moremada M.M.M.B

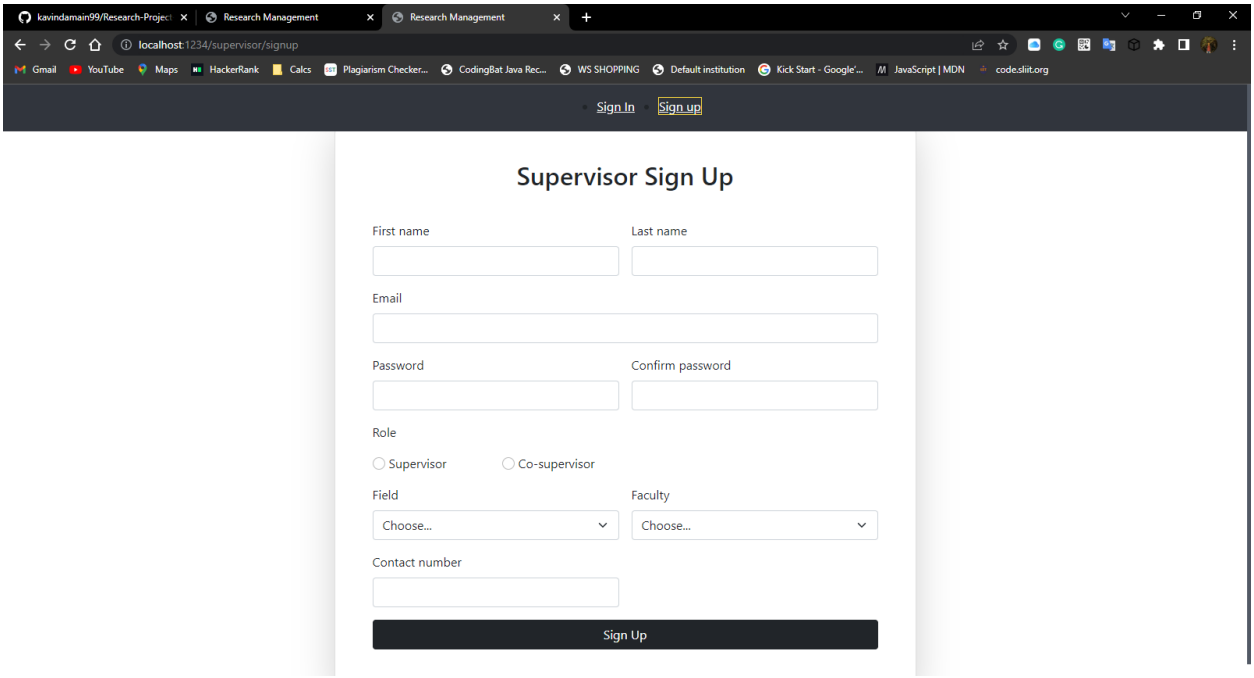
Supervisor / Co-supervisor Role

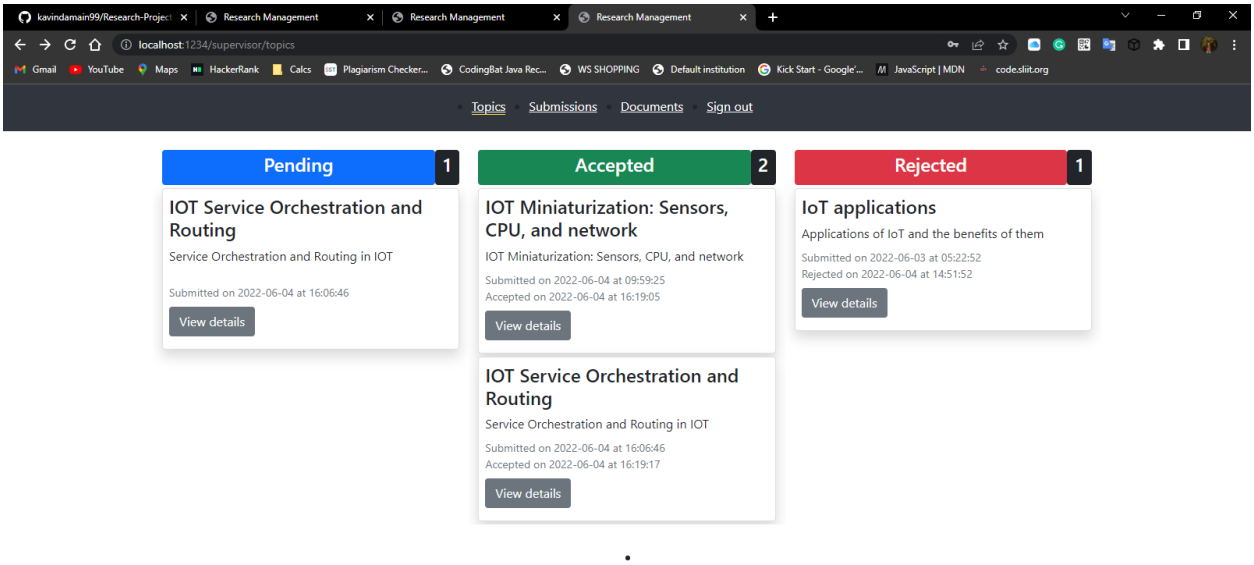
Description: Generally, supervisor and co-supervisor has same functions in the system. All should register and login to the system initially. The main task of supervisor and co-supervisor is accepting student's topics and give feedback for them and also communicate with them. They Should evaluate Documents submitted by groups using the provided marking scheme.

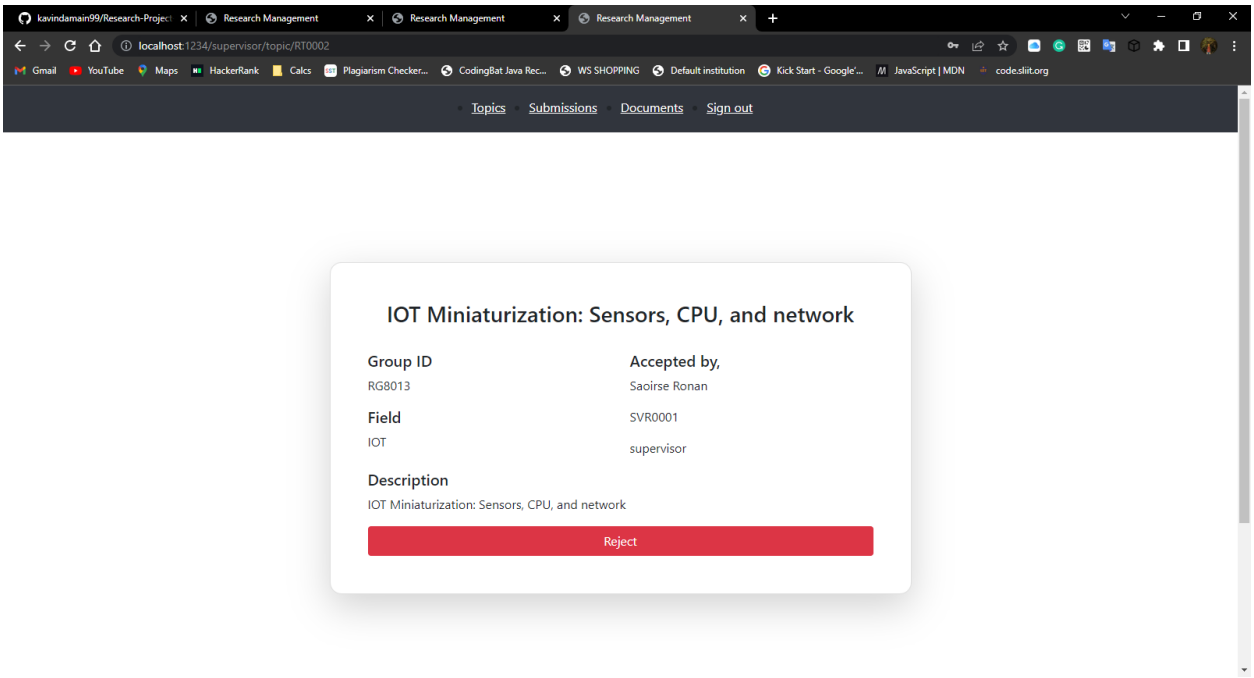


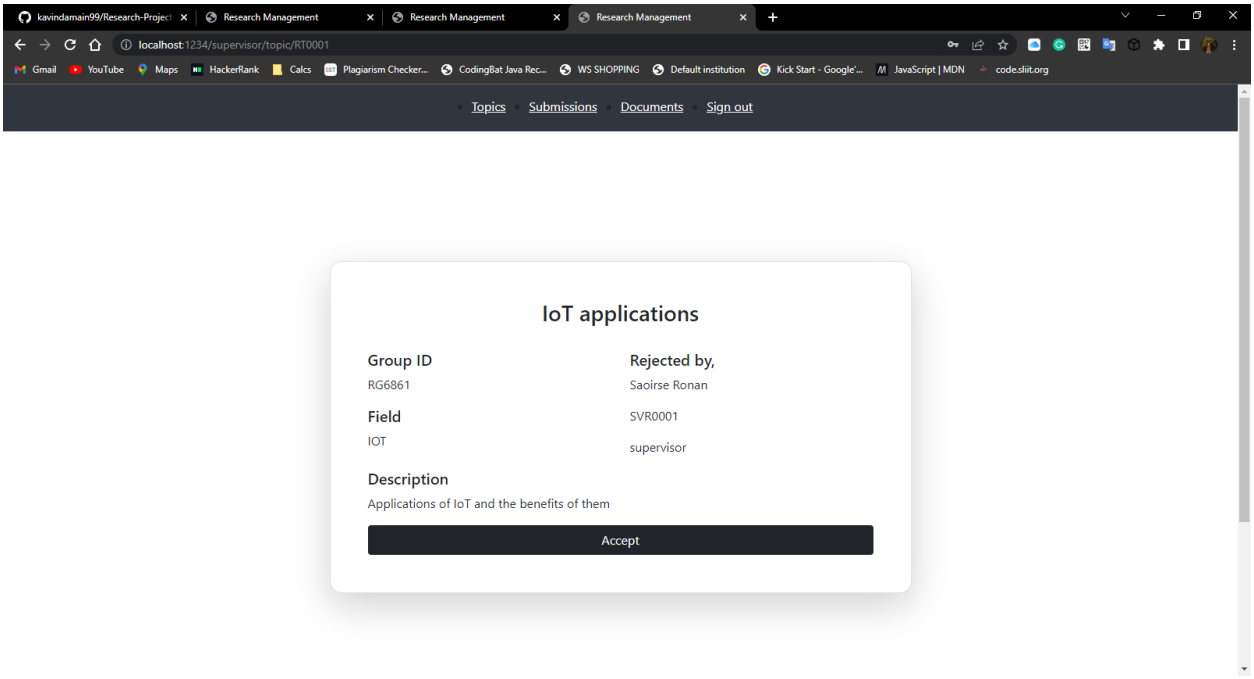
Screenshots of the Frontend

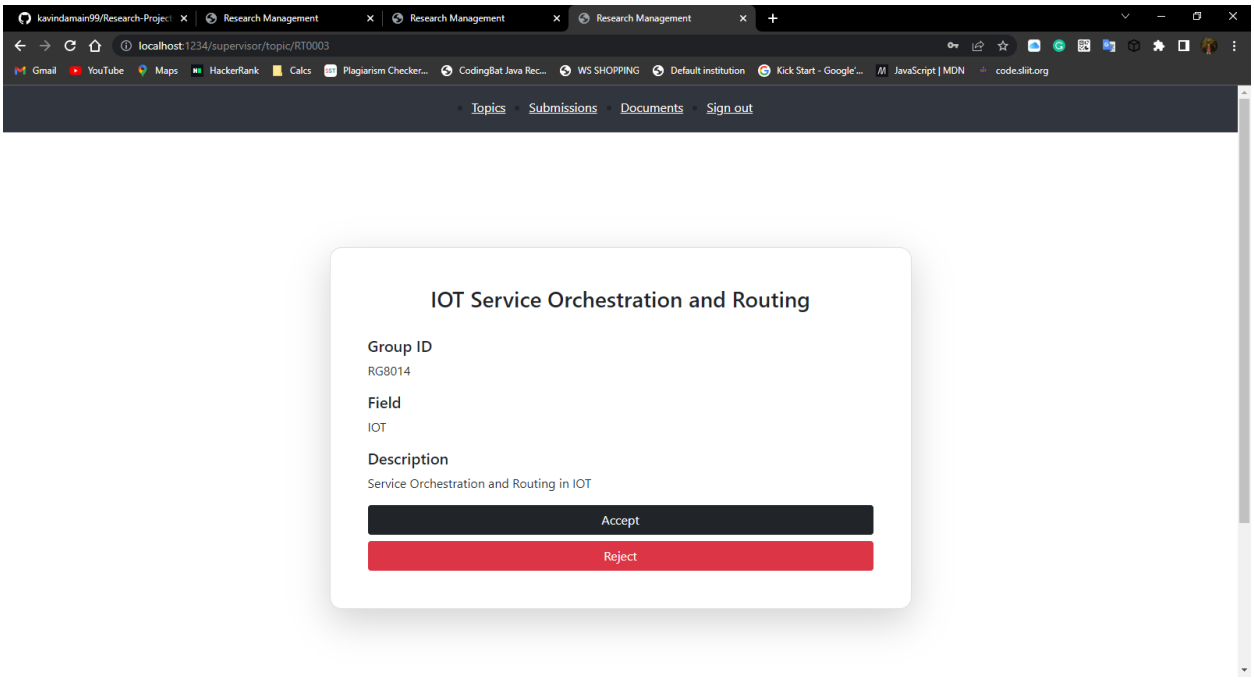


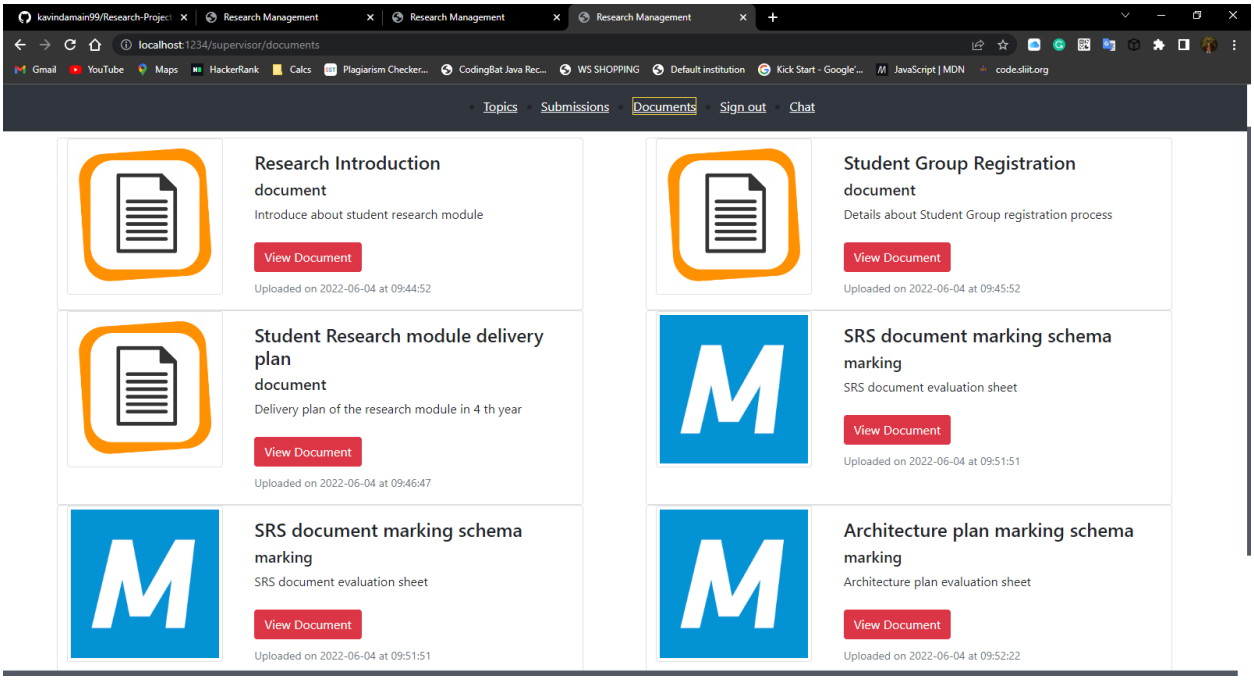


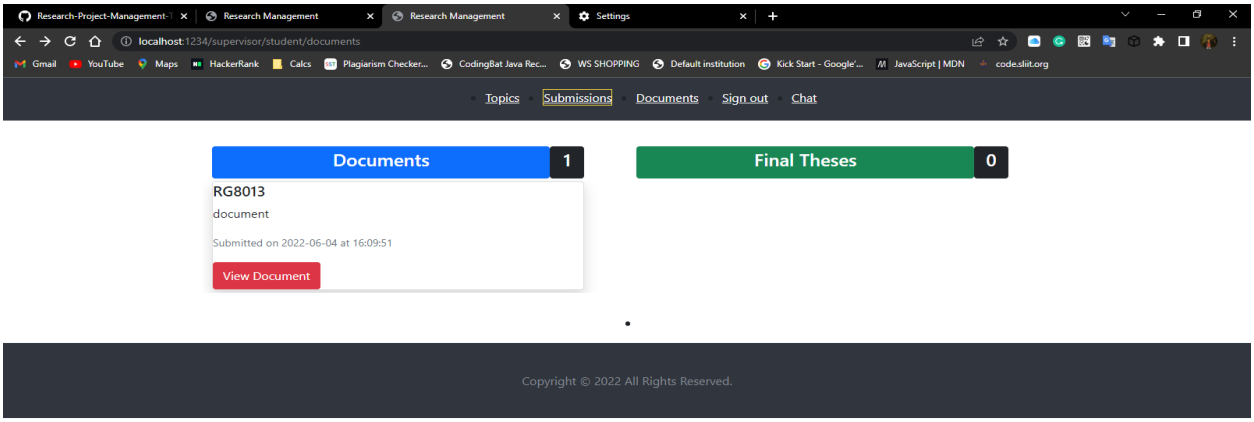

Topics
Submissions
Documents
Sign out
Pending 1
IOT Service Orchestration and Routing
Service Orchestration and Routing in IOT
Submitted on 2022-06-04 at 16:06:46
View details
Accepted 2
IOT Miniaturization: Sensors, CPU, and network
IOT Miniaturization: Sensors, CPU, and network
Submitted on 2022-06-04 at 09:59:25
Accepted on 2022-06-04 at 16:19:05
View details
IOT Service Orchestration and Routing
Service Orchestration and Routing in IOT
Submitted on 2022-06-04 at 16:06:46
Accepted on 2022-06-04 at 16:19:17
View details
Rejected 1
IoT applications
Applications of IoT and the benefits of them
Submitted on 2022-06-03 at 05:22:52
Rejected on 2022-06-04 at 14:51:52
View details
Copyright © 2022 All Rights Reserved.

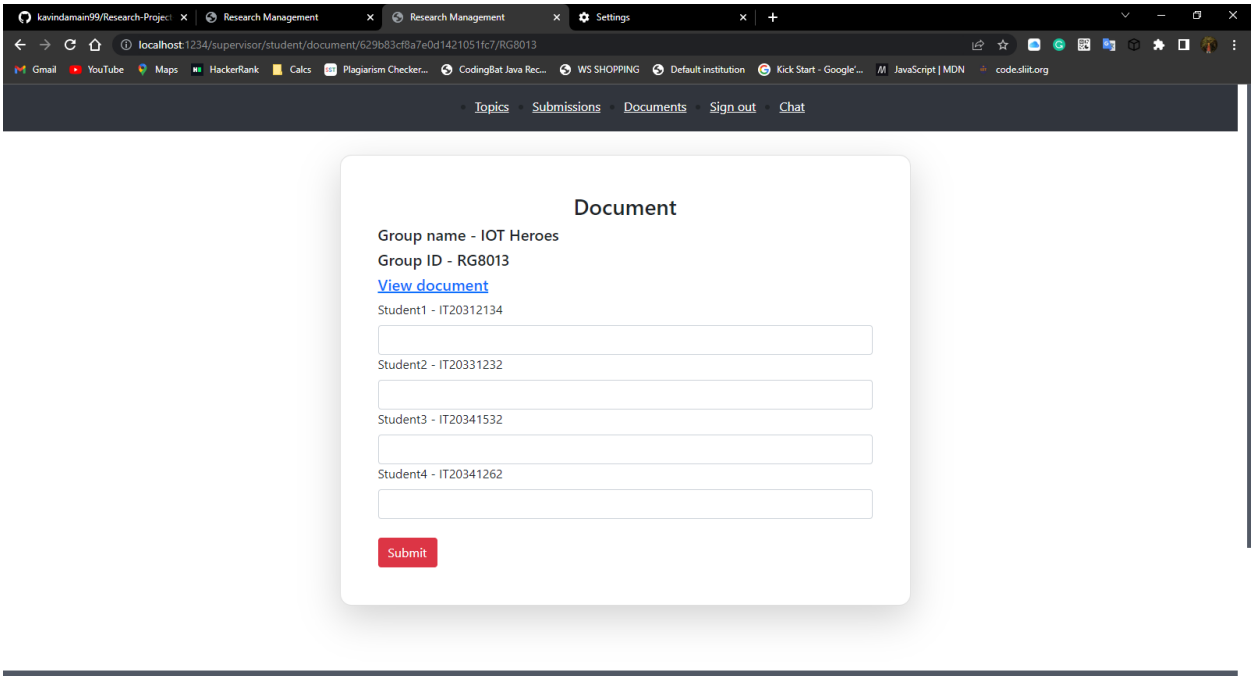

Topics
Submissions
Documents
Sign out
IOT Miniaturization: Sensors, CPU, and network
Group ID
RG8013
Field
IOT
Description
IOT Miniaturization: Sensors, CPU, and network
Accepted by,
Saoirse Ronan
SVR0001
supervisor
Reject

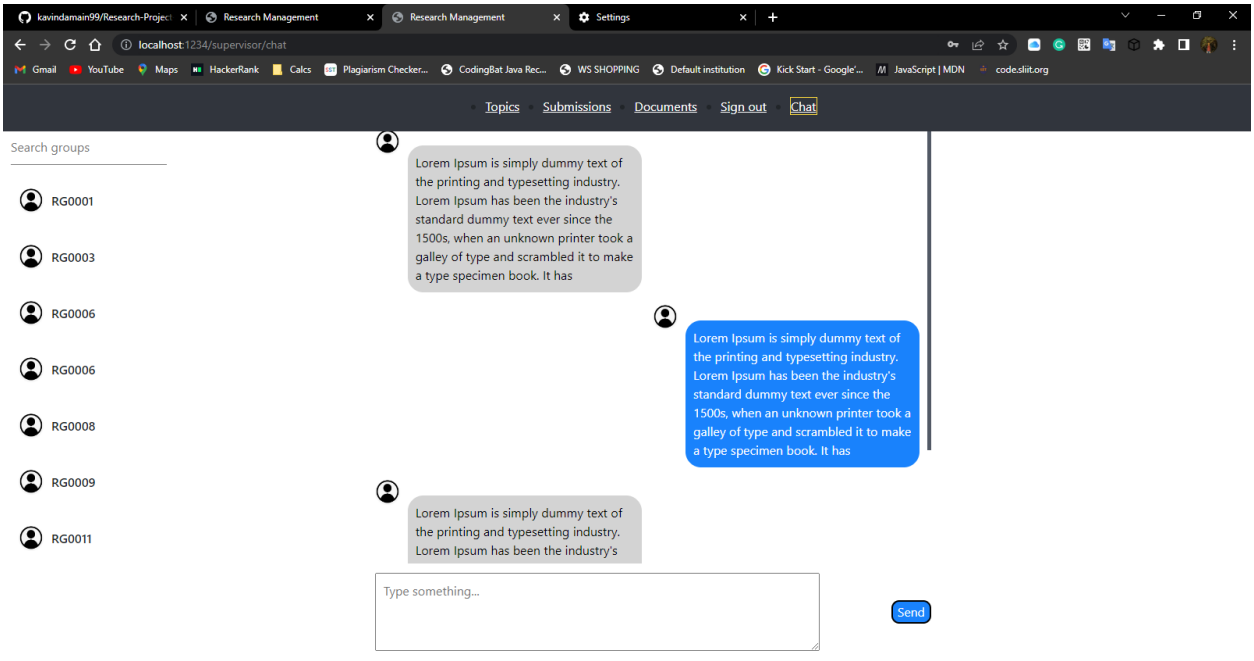

kavindamain99/Research-Project
Research Management
Research Management
Research Management
localhost:1234/supervisor/topic/RT0001
Gmail
YouTube
Maps
HackerRank
Calcs
Plagiarism Checker...
CodingBat Java Rec...
WS SHOPPING
Default institution
Kick Start - Google'...
JavaScript | MDN
code.sliit.org
Topics
Submissions
Documents
Sign out
IoT applications
Group ID
RG6861
Field
IOT
Description
Applications of IoT and the benefits of them
Rejected by,
Saoirse Ronan
SVR0001
supervisor
Accept


kavindamain99/Research-Project
Research Management
Research Management
Research Management
localhost:1234/supervisor/topic/RT0003
Gmail
YouTube
Maps
HackerRank
Calcs
Plagiarism Checker...
CodingBat Java Rec...
WS SHOPPING
Default institution
Kick Start - Google'...
JavaScript | MDN
code.sliit.org
Topics
Submissions
Documents
Sign out
IOT Service Orchestration and Routing
Group ID
RG8014
Field
IOT
Description
Service Orchestration and Routing in IOT
Accept
Reject

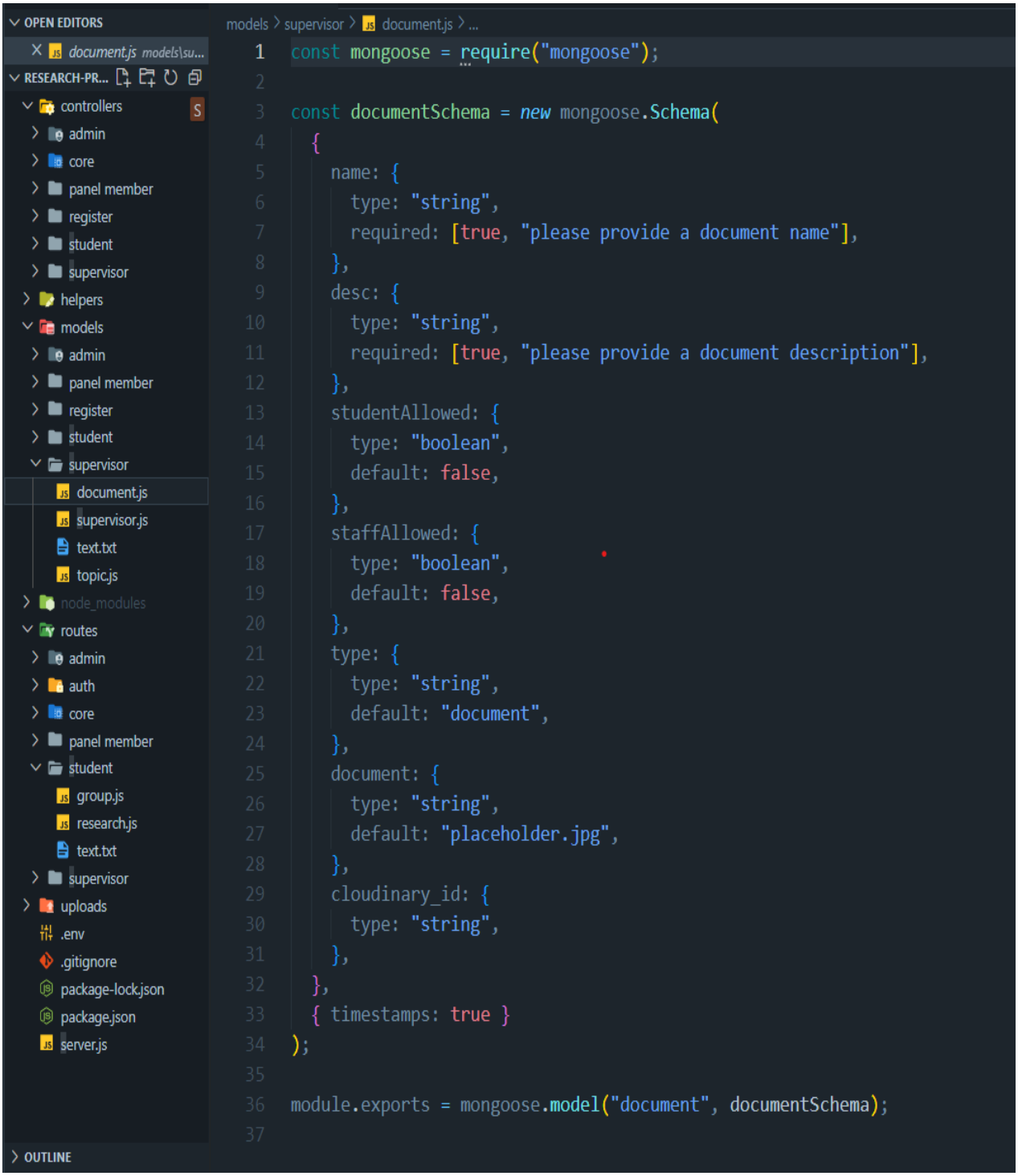

Topics
Submissions
Documents
Sign out
Chat
Research Introduction
document
Introduce about student research module
View Document
Uploaded on 2022-06-04 at 09:44:52
Student Group Registration
document
Details about Student Group registration process
View Document
Uploaded on 2022-06-04 at 09:45:52
Student Research module delivery plan
document
Delivery plan of the research module in 4 th year
View Document
Uploaded on 2022-06-04 at 09:46:47
SRS document marking schema
marking
SRS document evaluation sheet
View Document
Uploaded on 2022-06-04 at 09:51:51
SRS document marking schema
marking
SRS document evaluation sheet
View Document
Uploaded on 2022-06-04 at 09:51:51
Architecture plan marking schema
marking
Architecture plan evaluation sheet
View Document
Uploaded on 2022-06-04 at 09:52:22

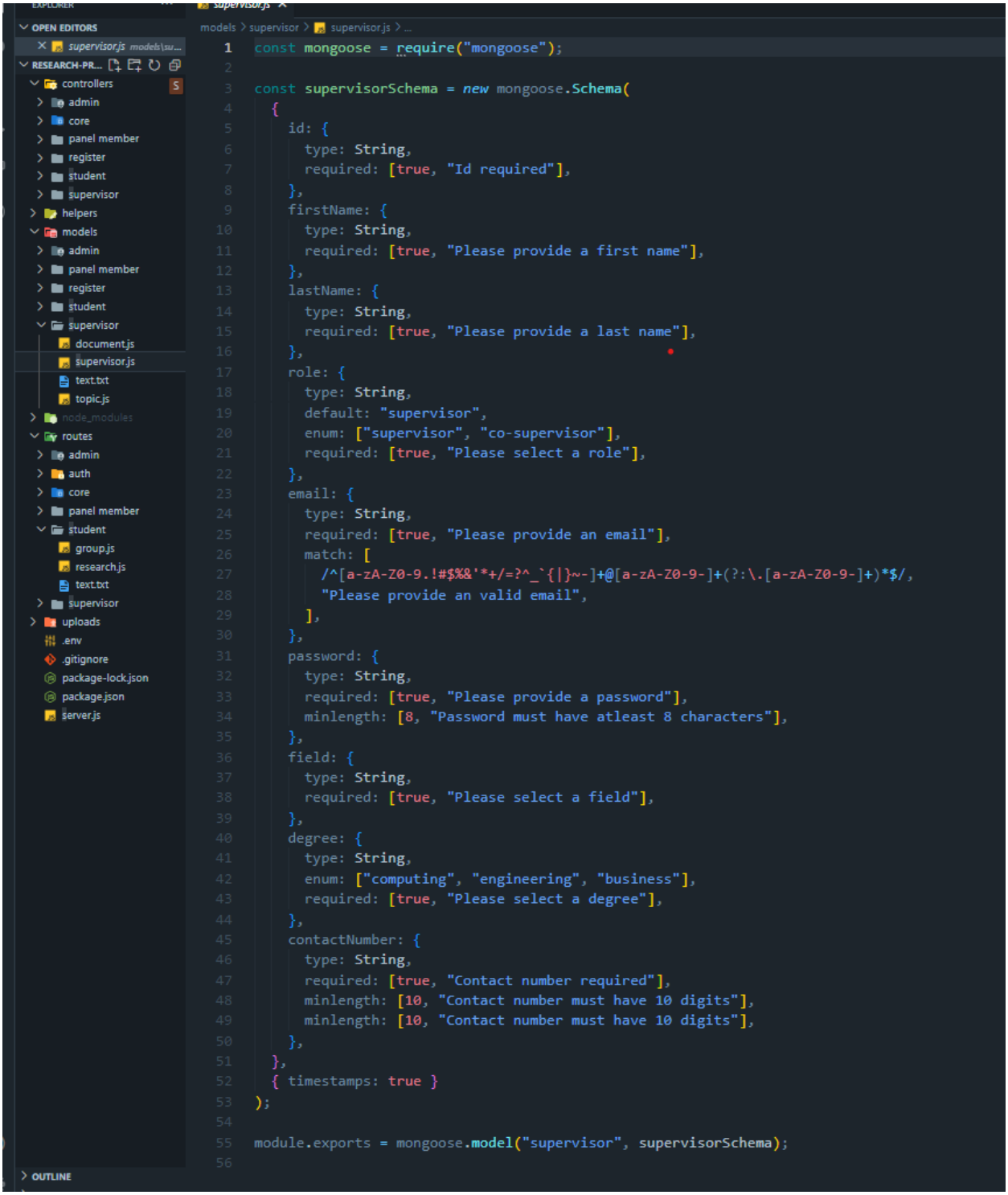

Topics
Submissions
Documents
Sign out
Chat
Documents 1
Final Theses 0
RG8013
document
Submitted on 2022-06-04 at 16:09:51
View Document
Copyright © 2022 All Rights Reserved.

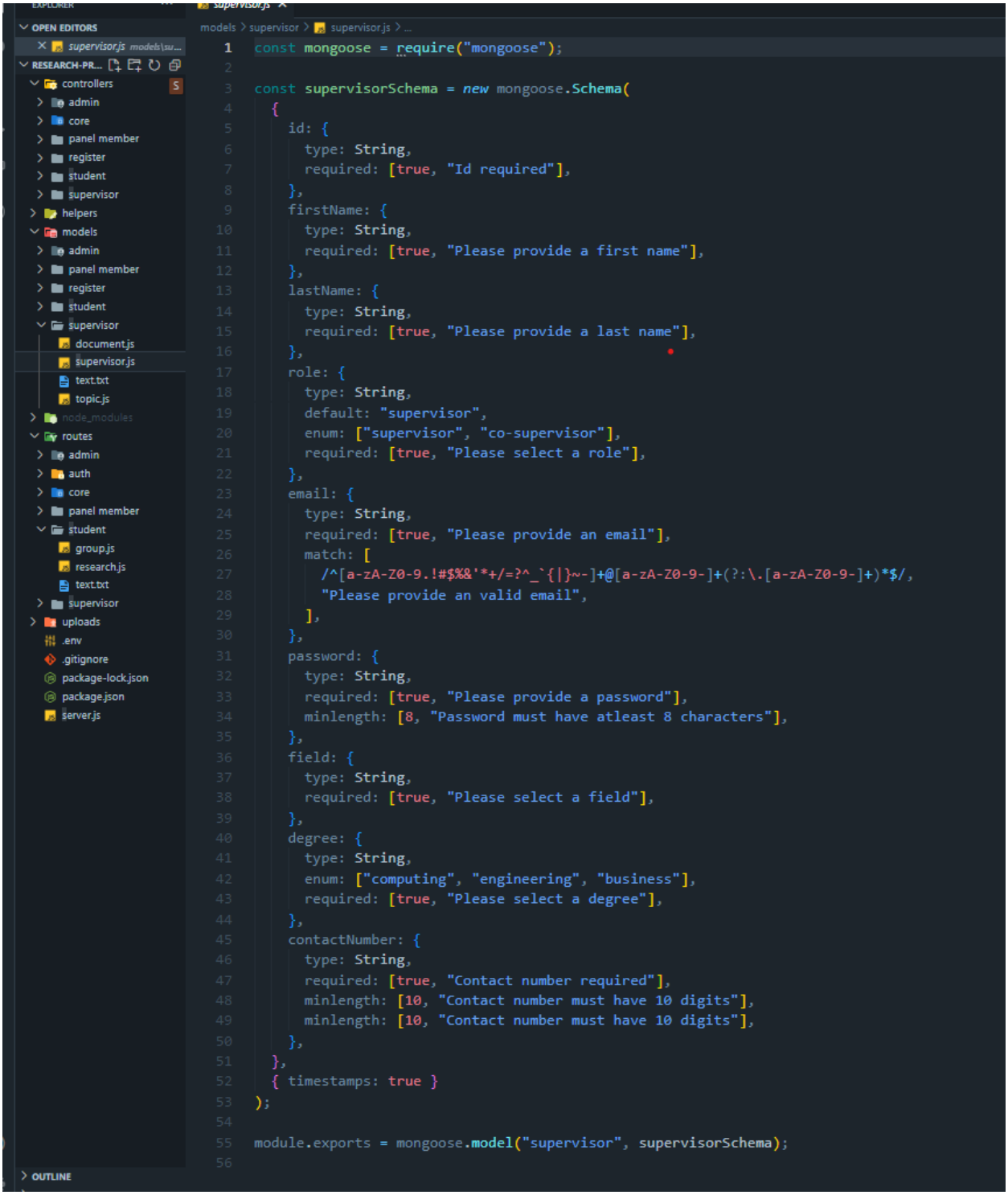

Topics
Submissions
Documents
Sign out
Chat
Document
Group name - IOT Heroes
Group ID - RG8013
View document
Student1 - IT20312134
Student2 - IT20331232
Student3 - IT20341532
Student4 - IT20341262
Submit

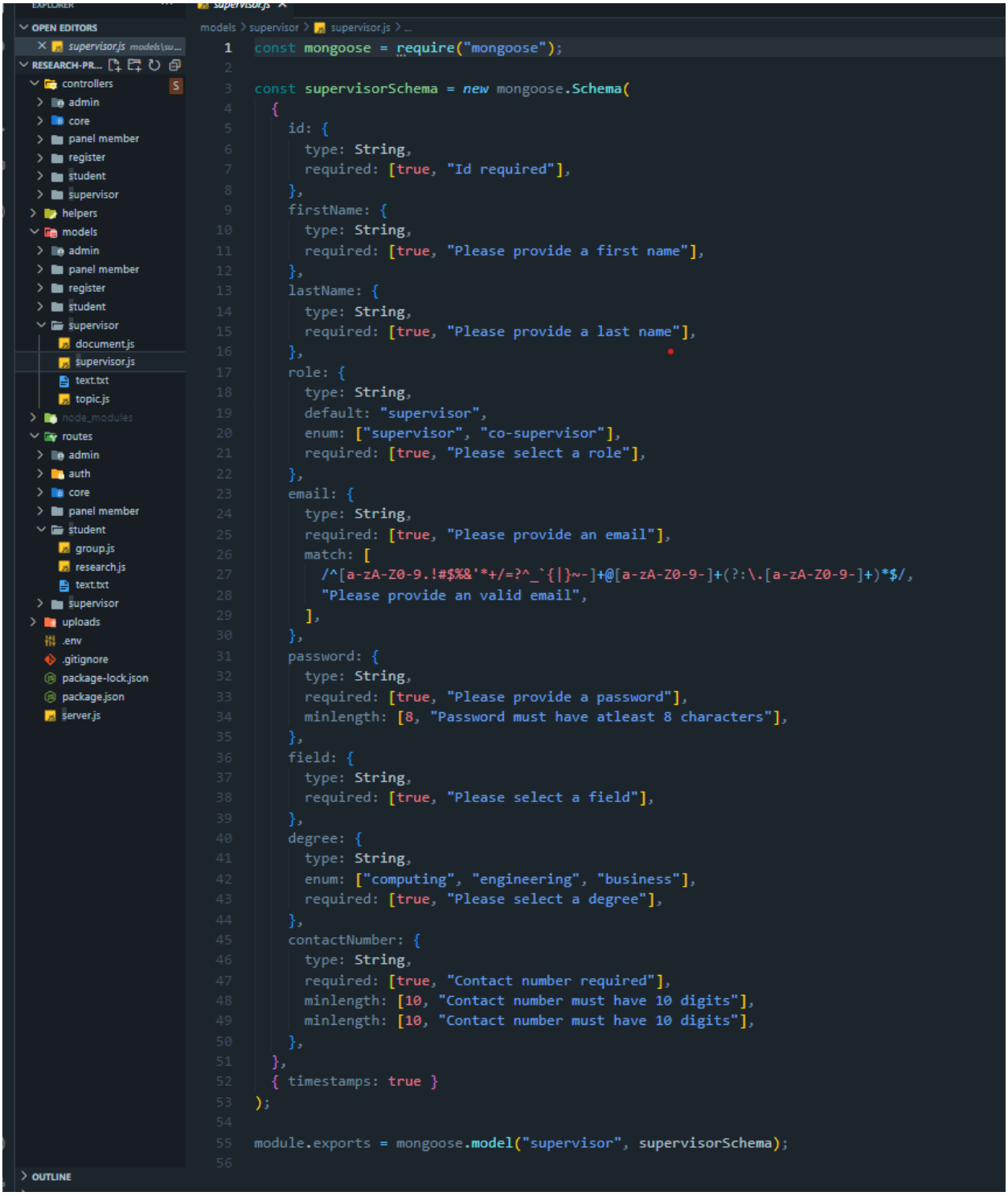

Topics
Submissions
Documents
Sign out
Chat
Search groups
RG0001
RG0003
RG0006
RG0006
RG0008
RG0009
RG0011
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's
Type something...
Send

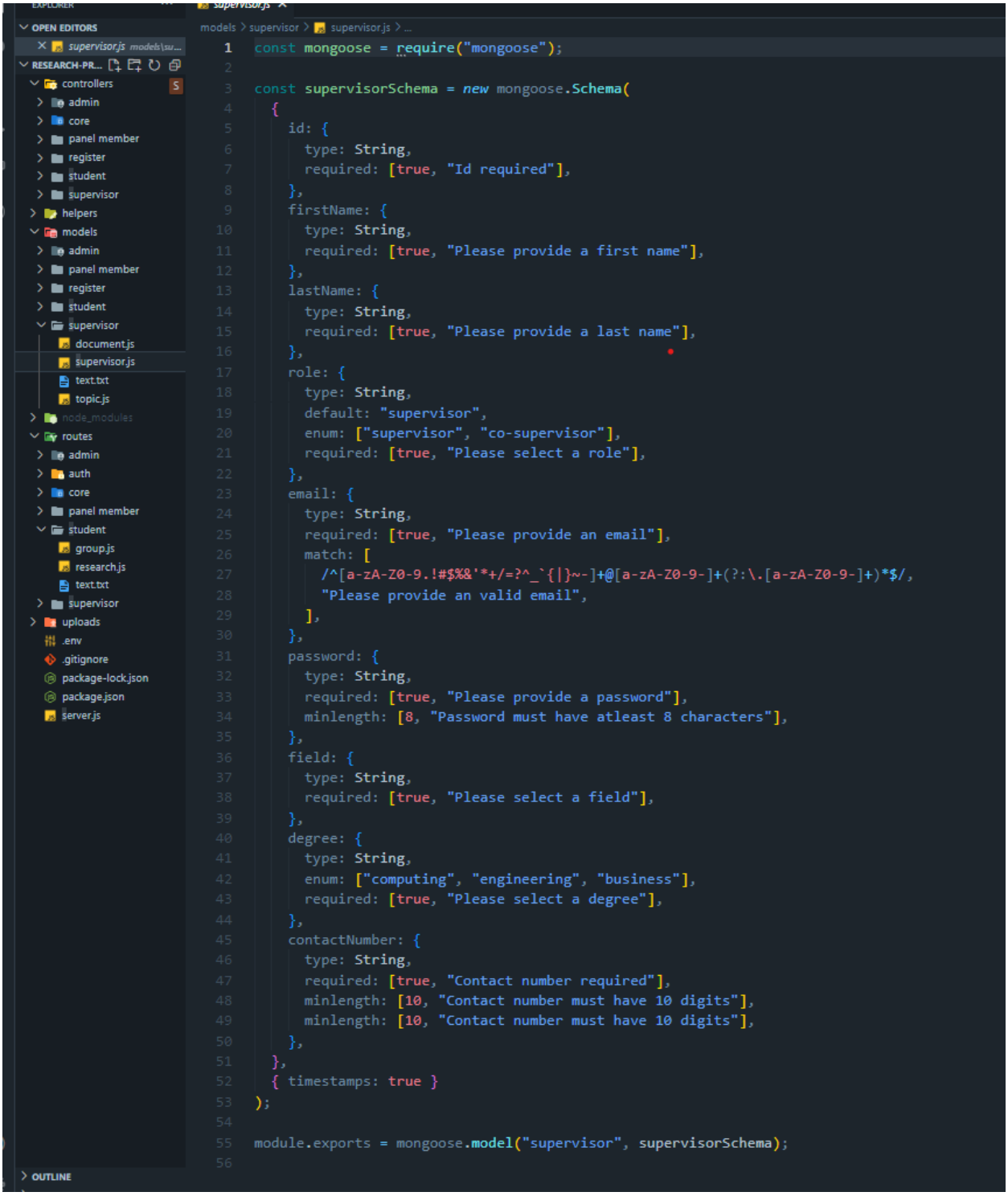
Screenshots/ Code Snippets of the Backend:

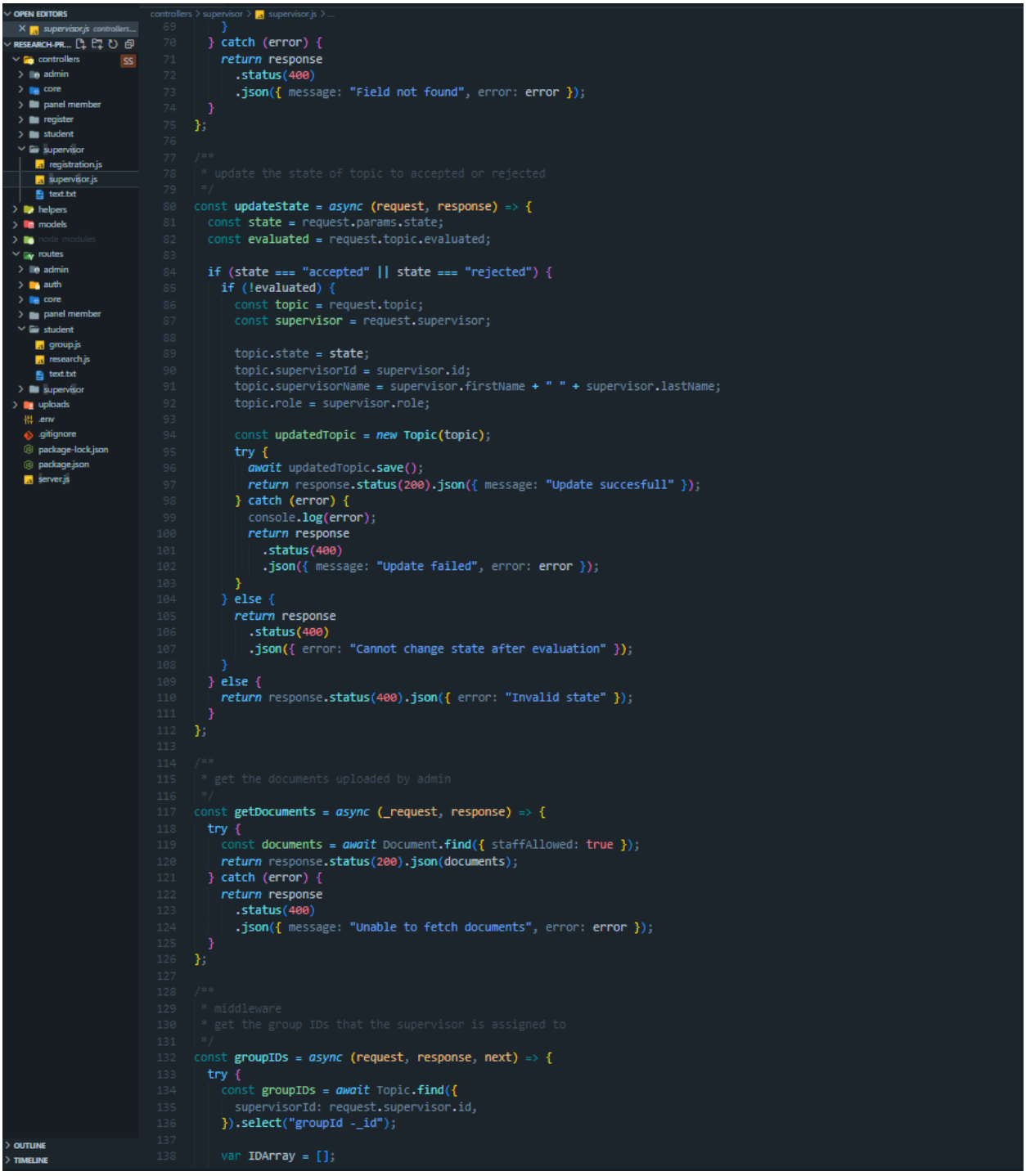


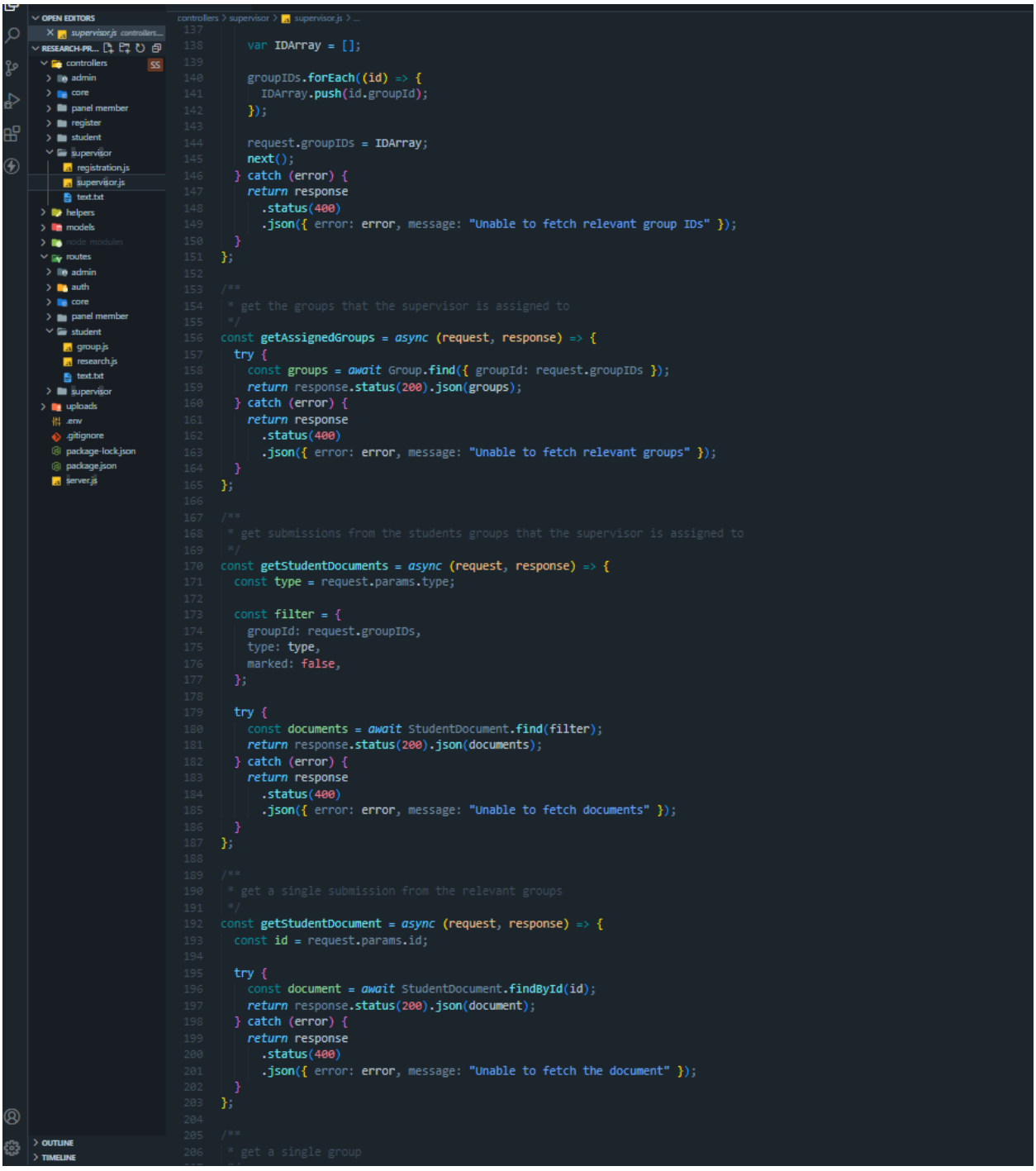

EXPLORER
OPEN EDITORS
supervisor.js models\su...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
helpers
models
admin
panel member
register
student
supervisor
document.js
supervisor.js
text.txt
topic.js
node_modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
supervisor.js
models > supervisor > supervisor.js > ...
1 const mongoose = require("mongoose");
2
3 const supervisorSchema = new mongoose.Schema(
4 {
5 id: {
6 type: String,
7 required: [true, "Id required"],
8 },
9 firstName: {
10 type: String,
11 required: [true, "Please provide a first name"],
12 },
13 lastName: {
14 type: String,
15 required: [true, "Please provide a last name"],
16 },
17 role: {
18 type: String,
19 default: "supervisor",
20 enum: ["supervisor", "co-supervisor"],
21 required: [true, "Please select a role"],
22 },
23 email: {
24 type: String,
25 required: [true, "Please provide an email"],
26 match: [
27 /^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)*$/,
28 "Please provide an valid email",
29],
30 },
31 password: {
32 type: String,
33 required: [true, "Please provide a password"],
34 minlength: [8, "Password must have atleast 8 characters"],
35 },
36 field: {
37 type: String,
38 required: [true, "Please select a field"],
39 },
40 degree: {
41 type: String,
42 enum: ["computing", "engineering", "business"],
43 required: [true, "Please select a degree"],
44 },
45 contactNumber: {
46 type: String,
47 required: [true, "Contact number required"],
48 minlength: [10, "Contact number must have 10 digits"],
49 minlength: [10, "Contact number must have 10 digits"],
50 },
51 },
52 { timestamps: true }
53);
54
55 module.exports = mongoose.model("supervisor", supervisorSchema);
56

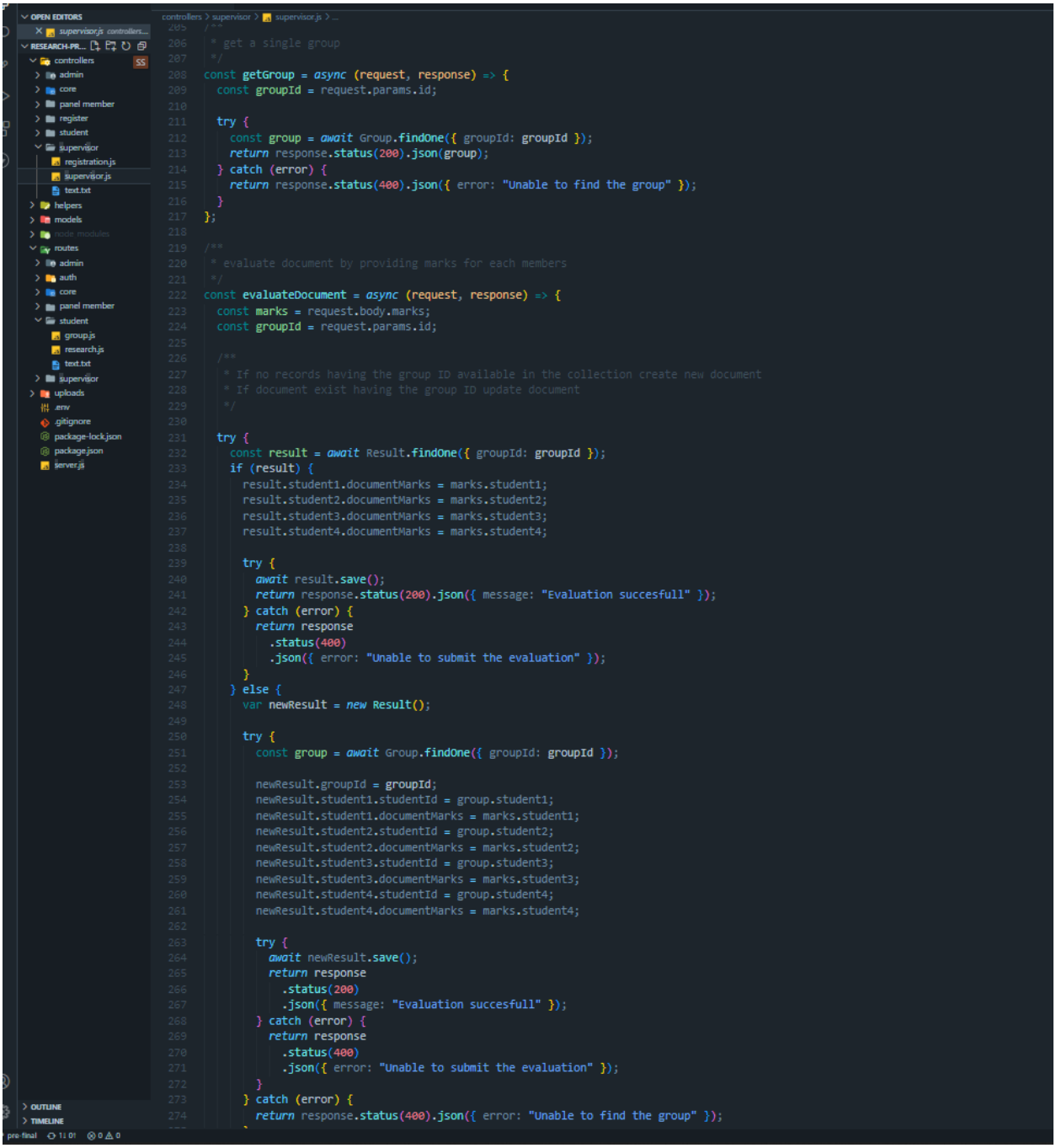

OPEN EDITORS
supervisor.js models\su...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
helpers
models
admin
panel member
register
student
supervisor
document.js
supervisor.js
text.txt
topic.js
node_modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
models > supervisor > supervisor.js > ...
const mongoose = require("mongoose");
const supervisorSchema = new mongoose.Schema(
{
id: {
type: String,
required: [true, "Id required"],
},
firstName: {
type: String,
required: [true, "Please provide a first name"],
},
lastName: {
type: String,
required: [true, "Please provide a last name"],
},
role: {
type: String,
default: "supervisor",
enum: ["supervisor", "co-supervisor"],
required: [true, "Please select a role"],
},
email: {
type: String,
required: [true, "Please provide an email"],
match: [
/^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)*$/,
"Please provide an valid email",
],
},
password: {
type: String,
required: [true, "Please provide a password"],
minlength: [8, "Password must have atleast 8 characters"],
},
field: {
type: String,
required: [true, "Please select a field"],
},
degree: {
type: String,
enum: ["computing", "engineering", "business"],
required: [true, "Please select a degree"],
},
contactNumber: {
type: String,
required: [true, "Contact number required"],
minlength: [10, "Contact number must have 10 digits"],
minlength: [10, "Contact number must have 10 digits"],
},
},
{ timestamps: true }
);
module.exports = mongoose.model("supervisor", supervisorSchema);

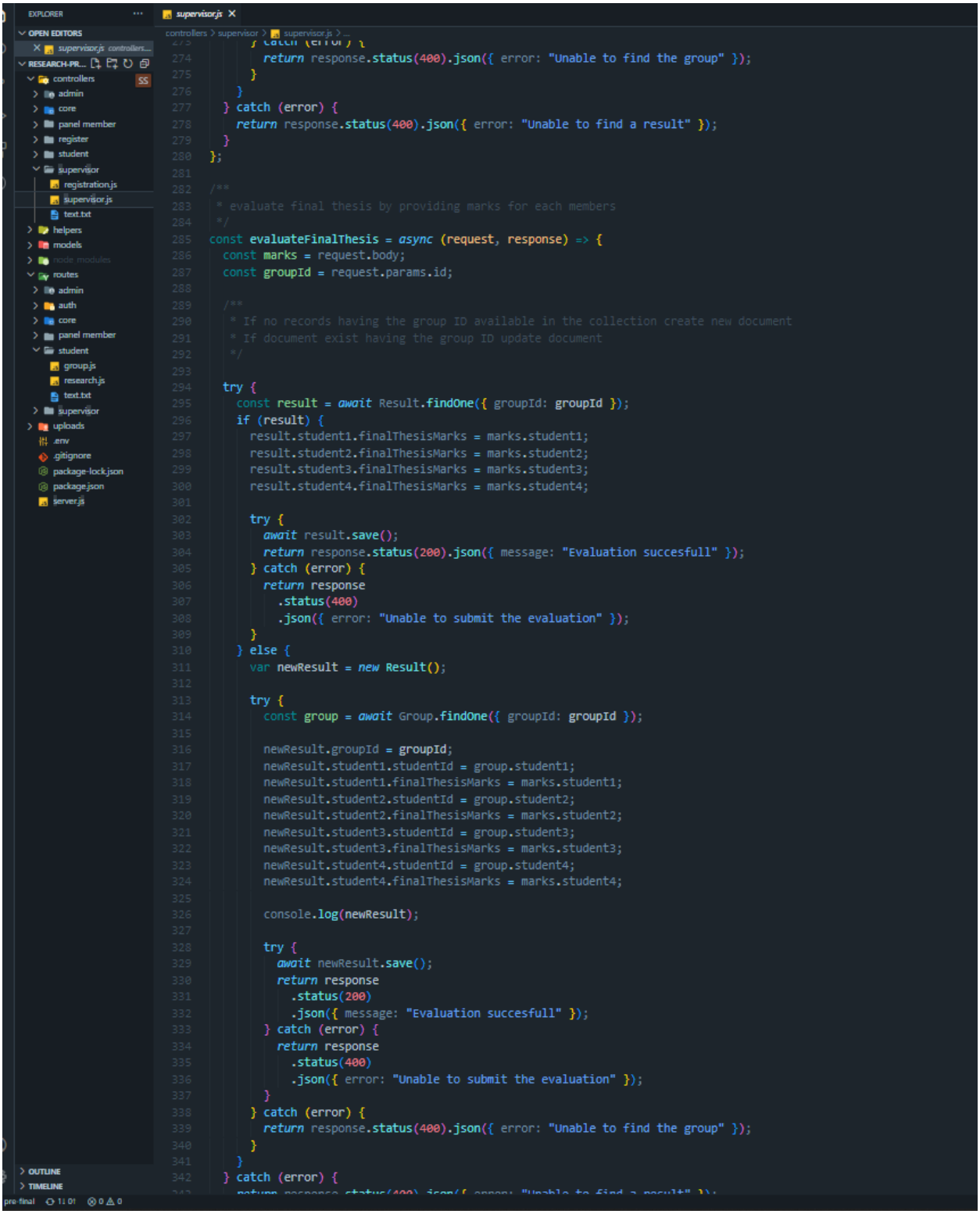

OPEN EDITORS
supervisor.js models\su...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
helpers
models
admin
panel member
register
student
supervisor
document.js
supervisor.js
text.txt
topic.js
node_modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
models > supervisor > supervisor.js > ...
const mongoose = require("mongoose");
const supervisorSchema = new mongoose.Schema(
{
id: {
type: String,
required: [true, "Id required"],
},
firstName: {
type: String,
required: [true, "Please provide a first name"],
},
lastName: {
type: String,
required: [true, "Please provide a last name"],
},
role: {
type: String,
default: "supervisor",
enum: ["supervisor", "co-supervisor"],
required: [true, "Please select a role"],
},
email: {
type: String,
required: [true, "Please provide an email"],
match: [
/^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)*$/,
"Please provide an valid email",
],
},
password: {
type: String,
required: [true, "Please provide a password"],
minlength: [8, "Password must have atleast 8 characters"],
},
field: {
type: String,
required: [true, "Please select a field"],
},
degree: {
type: String,
enum: ["computing", "engineering", "business"],
required: [true, "Please select a degree"],
},
contactNumber: {
type: String,
required: [true, "Contact number required"],
minlength: [10, "Contact number must have 10 digits"],
minlength: [10, "Contact number must have 10 digits"],
},
},
{ timestamps: true }
);
module.exports = mongoose.model("supervisor", supervisorSchema);

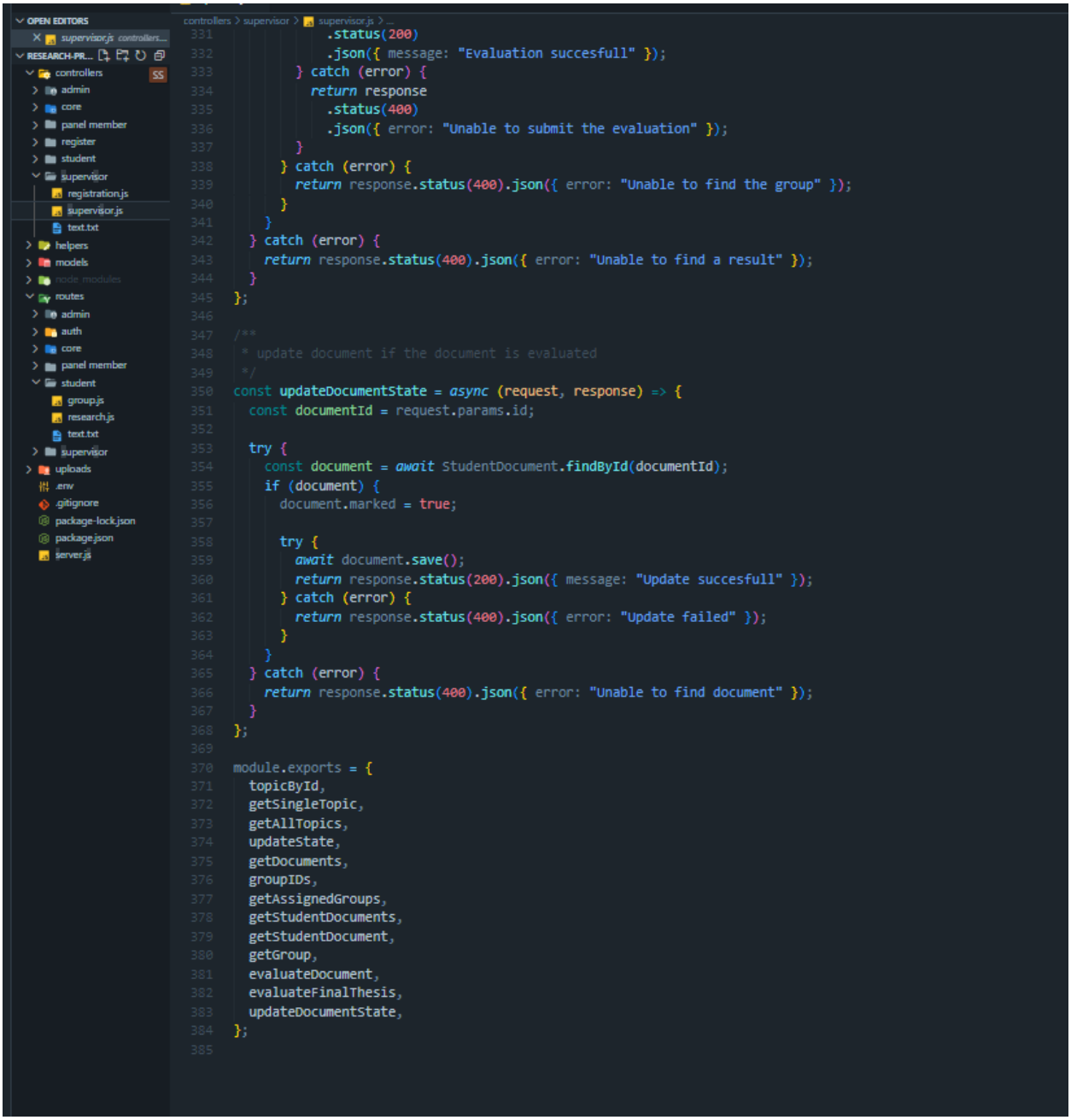

OPEN EDITORS
supervisor.js models\su...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
helpers
models
admin
panel member
register
student
supervisor
document.js
supervisor.js
text.txt
topic.js
node_modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
models > supervisor > supervisor.js > ...
const mongoose = require("mongoose");
const supervisorSchema = new mongoose.Schema(
{
id: {
type: String,
required: [true, "Id required"],
},
firstName: {
type: String,
required: [true, "Please provide a first name"],
},
lastName: {
type: String,
required: [true, "Please provide a last name"],
},
role: {
type: String,
default: "supervisor",
enum: ["supervisor", "co-supervisor"],
required: [true, "Please select a role"],
},
email: {
type: String,
required: [true, "Please provide an email"],
match: [
/^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)*$/,
"Please provide an valid email",
],
},
password: {
type: String,
required: [true, "Please provide a password"],
minlength: [8, "Password must have atleast 8 characters"],
},
field: {
type: String,
required: [true, "Please select a field"],
},
degree: {
type: String,
enum: ["computing", "engineering", "business"],
required: [true, "Please select a degree"],
},
contactNumber: {
type: String,
required: [true, "Contact number required"],
minlength: [10, "Contact number must have 10 digits"],
minlength: [10, "Contact number must have 10 digits"],
},
},
{ timestamps: true }
);
module.exports = mongoose.model("supervisor", supervisorSchema);

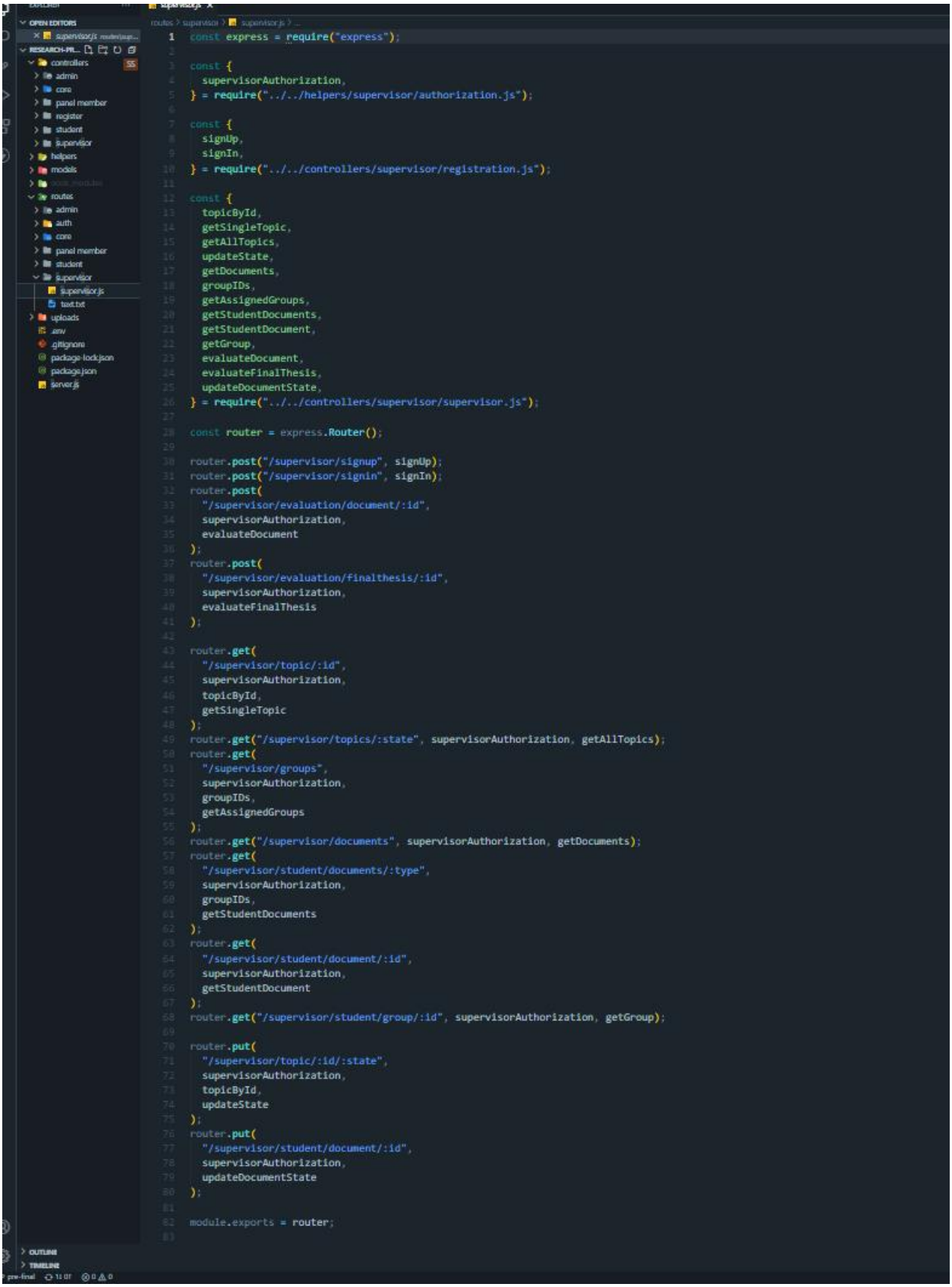

OPEN EDITORS
supervisor.js controllers...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
registration.js
supervisor.js
text.txt
helpers
models
node_modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
controllers > supervisor > supervisor.js > ...
}
} catch (error) {
return response
.status(400)
.json({ message: "Field not found", error: error });
}
};
/**
* update the state of topic to accepted or rejected
*/
const updateState = async (request, response) => {
const state = request.params.state;
const evaluated = request.topic.evaluated;
if (state === "accepted" || state === "rejected") {
if (!evaluated) {
const topic = request.topic;
const supervisor = request.supervisor;
topic.state = state;
topic.supervisorId = supervisor.id;
topic.supervisorName = supervisor.firstName + " " + supervisor.lastName;
topic.role = supervisor.role;
const updatedTopic = new Topic(topic);
try {
await updatedTopic.save();
return response.status(200).json({ message: "Update succesfull" });
} catch (error) {
console.log(error);
return response
.status(400)
.json({ message: "Update failed", error: error });
}
} else {
return response
.status(400)
.json({ error: "Cannot change state after evaluation" });
}
} else {
return response.status(400).json({ error: "Invalid state" });
}
};
/**
* get the documents uploaded by admin
*/
const getDocuments = async (_request, response) => {
try {
const documents = await Document.find({ staffAllowed: true });
return response.status(200).json(documents);
} catch (error) {
return response
.status(400)
.json({ message: "Unable to fetch documents", error: error });
}
};
/**
* middleware
* get the group IDs that the supervisor is assigned to
*/
const groupIDs = async (request, response, next) => {
try {
const groupIDs = await Topic.find({
supervisorId: request.supervisor.id,
}).select("groupId -_id");
var IDArray = [];


OPEN EDITORS
supervisor.js controllers...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
registration.js
supervisor.js
text.txt
helpers
models
node_modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
controllers > supervisor > supervisor.js > ...
var IDArray = [];
groupIDs.forEach((id) => {
IDArray.push(id.groupId);
});
request.groupIDs = IDArray;
next();
} catch (error) {
return response
.status(400)
.json({ error: error, message: "Unable to fetch relevant group IDs" });
}
};
/**
* get the groups that the supervisor is assigned to
*/
const getAssignedGroups = async (request, response) => {
try {
const groups = await Group.find({ groupId: request.groupIDs });
return response.status(200).json(groups);
} catch (error) {
return response
.status(400)
.json({ error: error, message: "Unable to fetch relevant groups" });
}
};
/**
* get submissions from the students groups that the supervisor is assigned to
*/
const getStudentDocuments = async (request, response) => {
const type = request.params.type;
const filter = {
groupId: request.groupIDs,
type: type,
marked: false,
};
try {
const documents = await StudentDocument.find(filter);
return response.status(200).json(documents);
} catch (error) {
return response
.status(400)
.json({ error: error, message: "Unable to fetch documents" });
}
};
/**
* get a single submission from the relevant groups
*/
const getStudentDocument = async (request, response) => {
const id = request.params.id;
try {
const document = await StudentDocument.findById(id);
return response.status(200).json(document);
} catch (error) {
return response
.status(400)
.json({ error: error, message: "Unable to fetch the document" });
}
};
/**
* get a single group

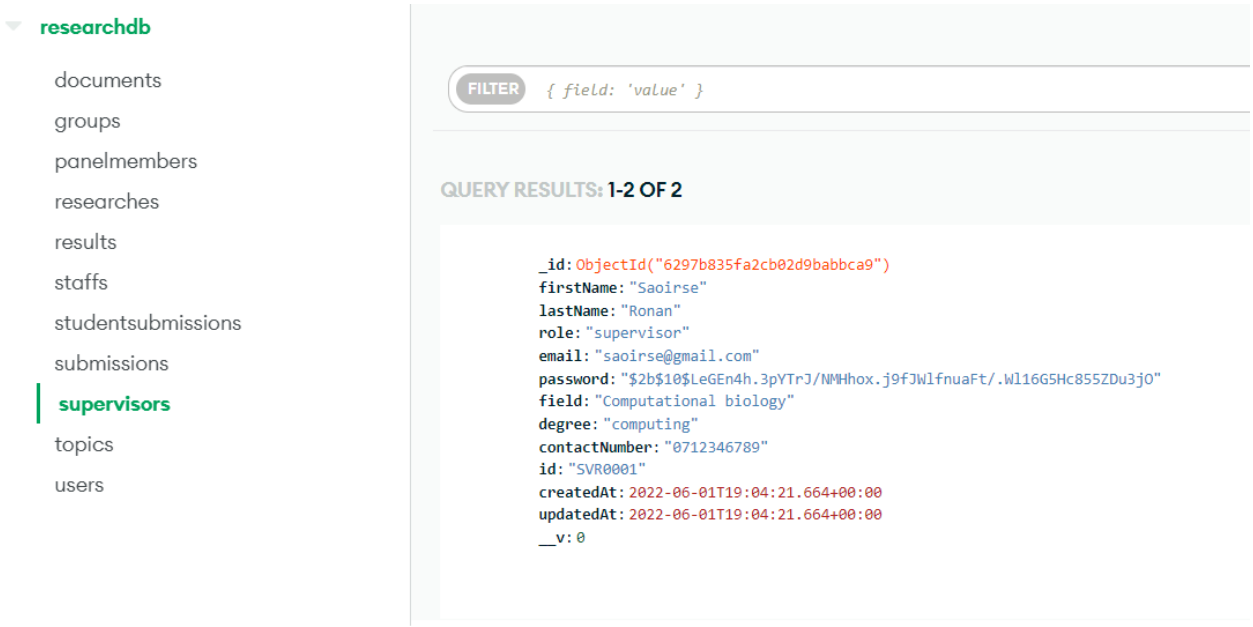

OPEN EDITORS
supervisor.js controllers...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
registration.js
supervisor.js
text.txt
helpers
models
node_modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
pre-final
controllers > supervisor > supervisor.js > ...
/**
* get a single group
*/
const getGroup = async (request, response) => {
const groupId = request.params.id;
try {
const group = await Group.findOne({ groupId: groupId });
return response.status(200).json(group);
} catch (error) {
return response.status(400).json({ error: "Unable to find the group" });
}
};
/**
* evaluate document by providing marks for each members
*/
const evaluateDocument = async (request, response) => {
const marks = request.body.marks;
const groupId = request.params.id;
/**
* If no records having the group ID available in the collection create new document
* If document exist having the group ID update document
*/
try {
const result = await Result.findOne({ groupId: groupId });
if (result) {
result.student1.documentMarks = marks.student1;
result.student2.documentMarks = marks.student2;
result.student3.documentMarks = marks.student3;
result.student4.documentMarks = marks.student4;
try {
await result.save();
return response.status(200).json({ message: "Evaluation succesfull" });
} catch (error) {
return response
.status(400)
.json({ error: "Unable to submit the evaluation" });
}
} else {
var newResult = new Result();
try {
const group = await Group.findOne({ groupId: groupId });
newResult.groupId = groupId;
newResult.student1.studentId = group.student1;
newResult.student1.documentMarks = marks.student1;
newResult.student2.studentId = group.student2;
newResult.student2.documentMarks = marks.student2;
newResult.student3.studentId = group.student3;
newResult.student3.documentMarks = marks.student3;
newResult.student4.studentId = group.student4;
newResult.student4.documentMarks = marks.student4;
try {
await newResult.save();
return response
.status(200)
.json({ message: "Evaluation succesfull" });
} catch (error) {
return response
.status(400)
.json({ error: "Unable to submit the evaluation" });
}
} catch (error) {
return response.status(400).json({ error: "Unable to find the group" });

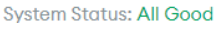

EXPLORER
supervisor.js
OPEN EDITORS
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
registration.js
supervisor.js
text.txt
helpers
models
node_modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
pre-final
controllers > supervisor > supervisor.js > ...
return response.status(400).json({ error: "Unable to find the group" });
} catch (error) {
return response.status(400).json({ error: "Unable to find a result" });
};
/**
* evaluate final thesis by providing marks for each members
*/
const evaluateFinalThesis = async (request, response) => {
const marks = request.body;
const groupId = request.params.id;
/**
* If no records having the group ID available in the collection create new document
* If document exist having the group ID update document
*/
try {
const result = await Result.findOne({ groupId: groupId });
if (result) {
result.student1.finalThesisMarks = marks.student1;
result.student2.finalThesisMarks = marks.student2;
result.student3.finalThesisMarks = marks.student3;
result.student4.finalThesisMarks = marks.student4;
try {
await result.save();
return response.status(200).json({ message: "Evaluation succesfull" });
} catch (error) {
return response
.status(400)
.json({ error: "Unable to submit the evaluation" });
}
} else {
var newResult = new Result();
try {
const group = await Group.findOne({ groupId: groupId });
newResult.groupId = groupId;
newResult.student1.studentId = group.student1;
newResult.student1.finalThesisMarks = marks.student1;
newResult.student2.studentId = group.student2;
newResult.student2.finalThesisMarks = marks.student2;
newResult.student3.studentId = group.student3;
newResult.student3.finalThesisMarks = marks.student3;
newResult.student4.studentId = group.student4;
newResult.student4.finalThesisMarks = marks.student4;
console.log(newResult);
try {
await newResult.save();
return response
.status(200)
.json({ message: "Evaluation succesfull" });
} catch (error) {
return response
.status(400)
.json({ error: "Unable to submit the evaluation" });
}
} catch (error) {
return response.status(400).json({ error: "Unable to find the group" });
}
}
} catch (error) {


OPEN EDITORS
supervisor.js controllers...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
registration.js
supervisor.js
text.txt
helpers
models
node modules
routes
admin
auth
core
panel member
student
group.js
research.js
text.txt
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
controllers > supervisor > supervisor.js > ...
.status(200)
.json({ message: "Evaluation succesfull" });
} catch (error) {
return response
.status(400)
.json({ error: "Unable to submit the evaluation" });
}
} catch (error) {
return response.status(400).json({ error: "Unable to find the group" });
}
}
} catch (error) {
return response.status(400).json({ error: "Unable to find a result" });
}
};
/**
* update document if the document is evaluated
*/
const updateDocumentState = async (request, response) => {
const documentId = request.params.id;
try {
const document = await StudentDocument.findById(documentId);
if (document) {
document.marked = true;
try {
await document.save();
return response.status(200).json({ message: "Update succesfull" });
} catch (error) {
return response.status(400).json({ error: "Update failed" });
}
}
} catch (error) {
return response.status(400).json({ error: "Unable to find document" });
}
};
module.exports = {
topicById,
getSingleTopic,
getAllTopics,
updateState,
getDocuments,
groupIDs,
getAssignedGroups,
getStudentDocuments,
getStudentDocument,
getGroup,
evaluateDocument,
evaluateFinalThesis,
updateDocumentState,
};


OPEN EDITORS
supervisor.js routes\sup...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
helpers
models
node_modules
routes
admin
auth
core
panel member
student
supervisor
supervisor.js
test.txt
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
routes > supervisor > supervisor.js > ...
const express = require("express");
const {
supervisorAuthorization,
} = require("../../helpers/supervisor/authorization.js");
const {
signUp,
signIn,
} = require("../../controllers/supervisor/registration.js");
const {
topicById,
getSingleTopic,
getAllTopics,
updateState,
getDocuments,
groupIDs,
getAssignedGroups,
getStudentDocuments,
getStudentDocument,
getGroup,
evaluateDocument,
evaluateFinalThesis,
updateDocumentState,
} = require("../../controllers/supervisor/supervisor.js");
const router = express.Router();
router.post("/supervisor/signup", signUp);
router.post("/supervisor/signin", signIn);
router.post(
"/supervisor/evaluation/document/:id",
supervisorAuthorization,
evaluateDocument
);
router.post(
"/supervisor/evaluation/finalthesis/:id",
supervisorAuthorization,
evaluateFinalThesis
);
router.get(
"/supervisor/topic/:id",
supervisorAuthorization,
topicById,
getSingleTopic
);
router.get("/supervisor/topics/:state", supervisorAuthorization, getAllTopics);
router.get(
"/supervisor/groups",
supervisorAuthorization,
groupIDs,
getAssignedGroups
);
router.get("/supervisor/documents", supervisorAuthorization, getDocuments);
router.get(
"/supervisor/student/documents/:type",
supervisorAuthorization,
groupIDs,
getStudentDocuments
);
router.get(
"/supervisor/student/document/:id",
supervisorAuthorization,
getStudentDocument
);
router.get("/supervisor/student/group/:id", supervisorAuthorization, getGroup);
router.put(
"/supervisor/topic/:id/:state",
supervisorAuthorization,
topicById,
updateState
);
router.put(
"/supervisor/student/document/:id",
supervisorAuthorization,
updateDocumentState
);
module.exports = router;

Screenshots of the MongoDB Schemas



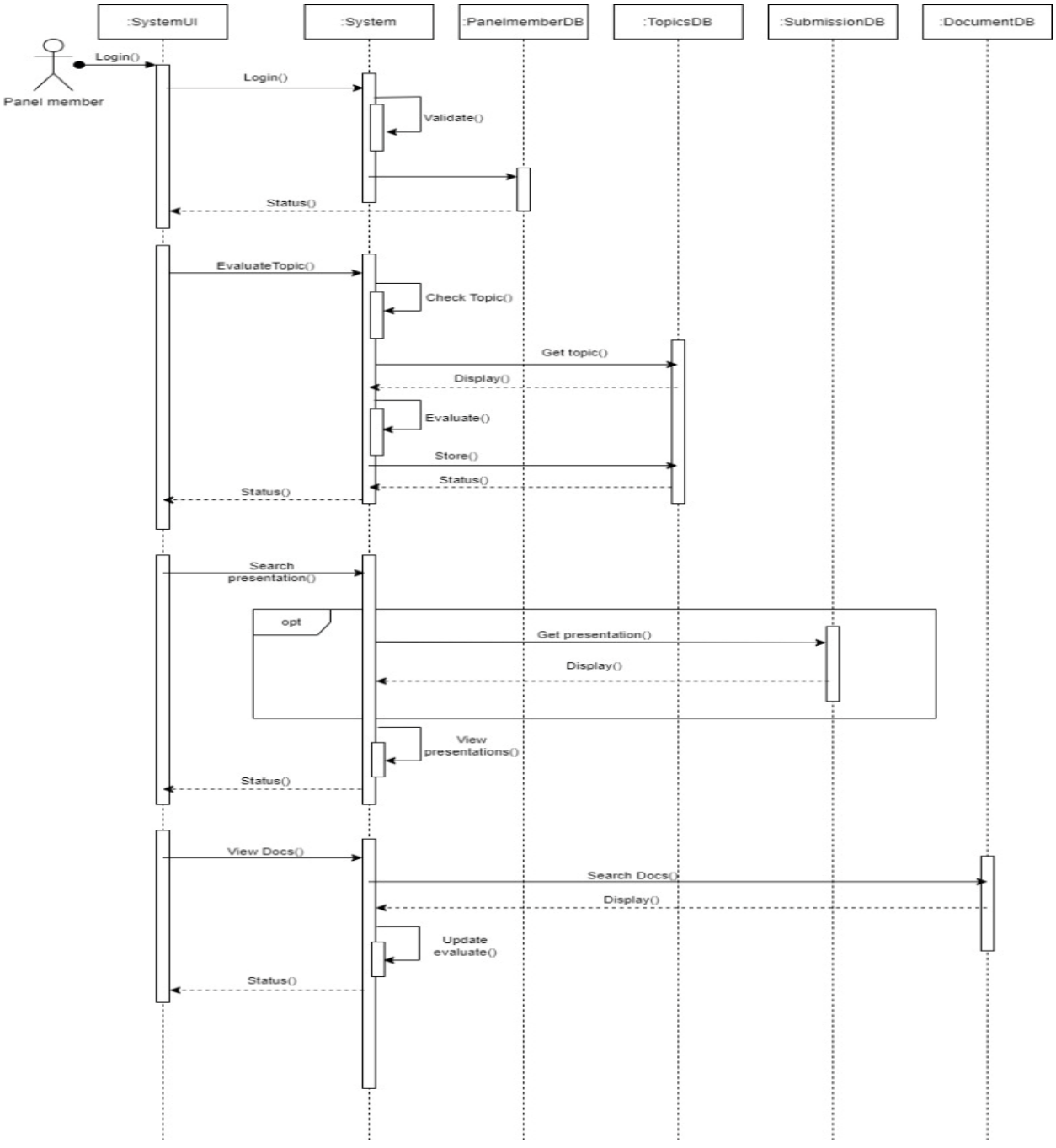


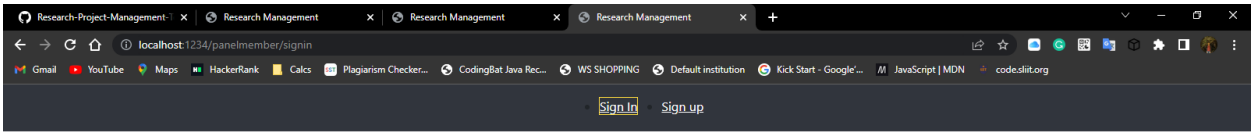
Component Description

IT20236564 Kumarathunga M.A.M.T

Panel Member Role

Description: Panel member add by the administrator. They have several tasks in this system and their main task is to evaluated presentations. They have evaluated student's topics also and evaluate student's presentation according to the provided marking scheme.



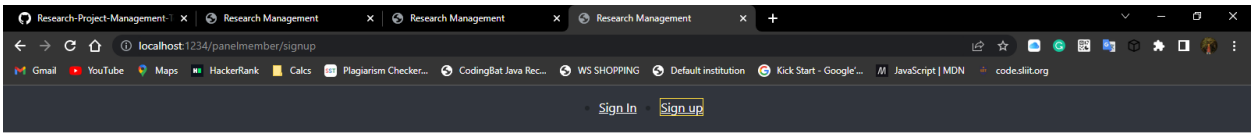


Panel Member Sign In

Email

Password

Sign In



Panel Member Sign Up

First name

Last name

Email

Password

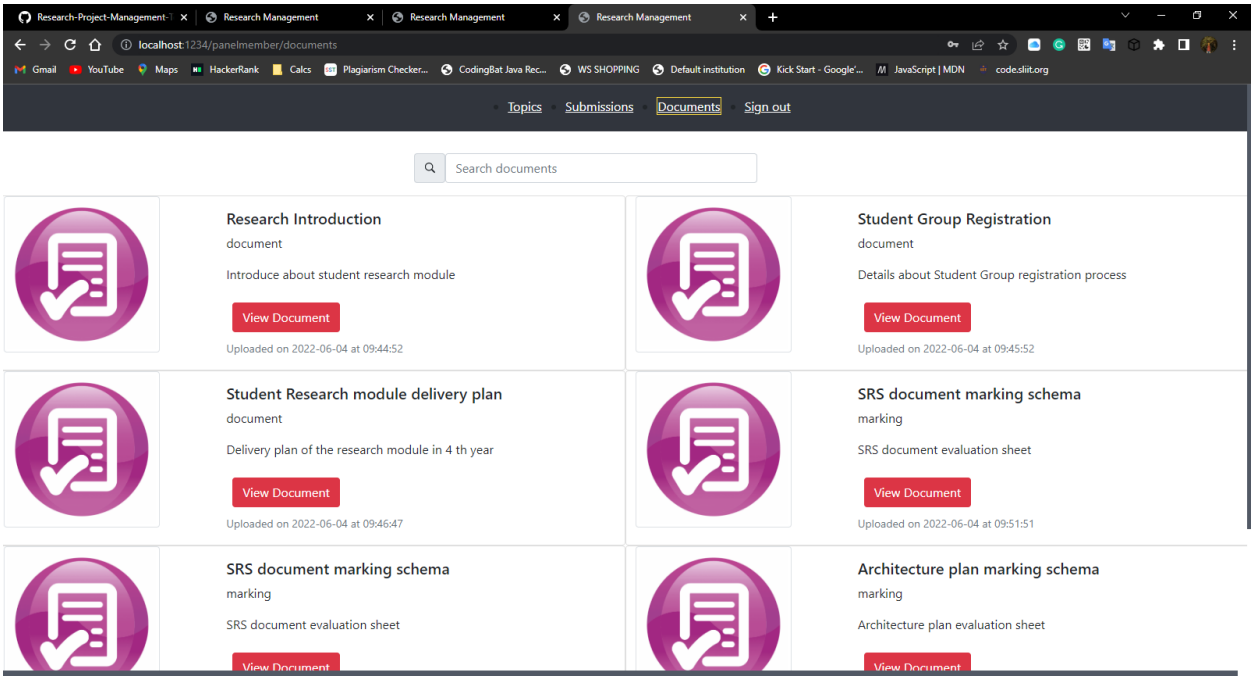
Confirm password

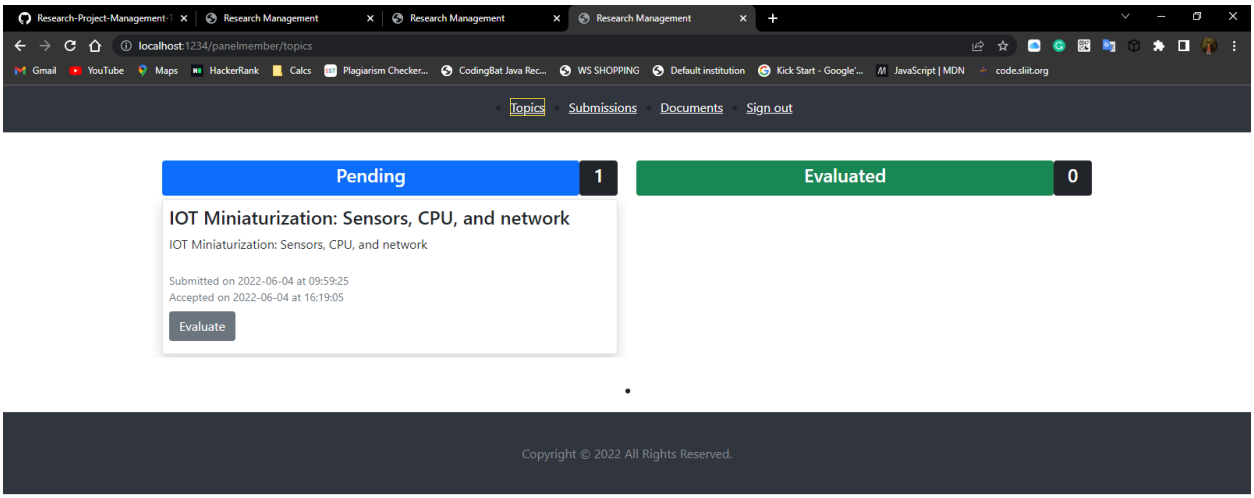
Faculty

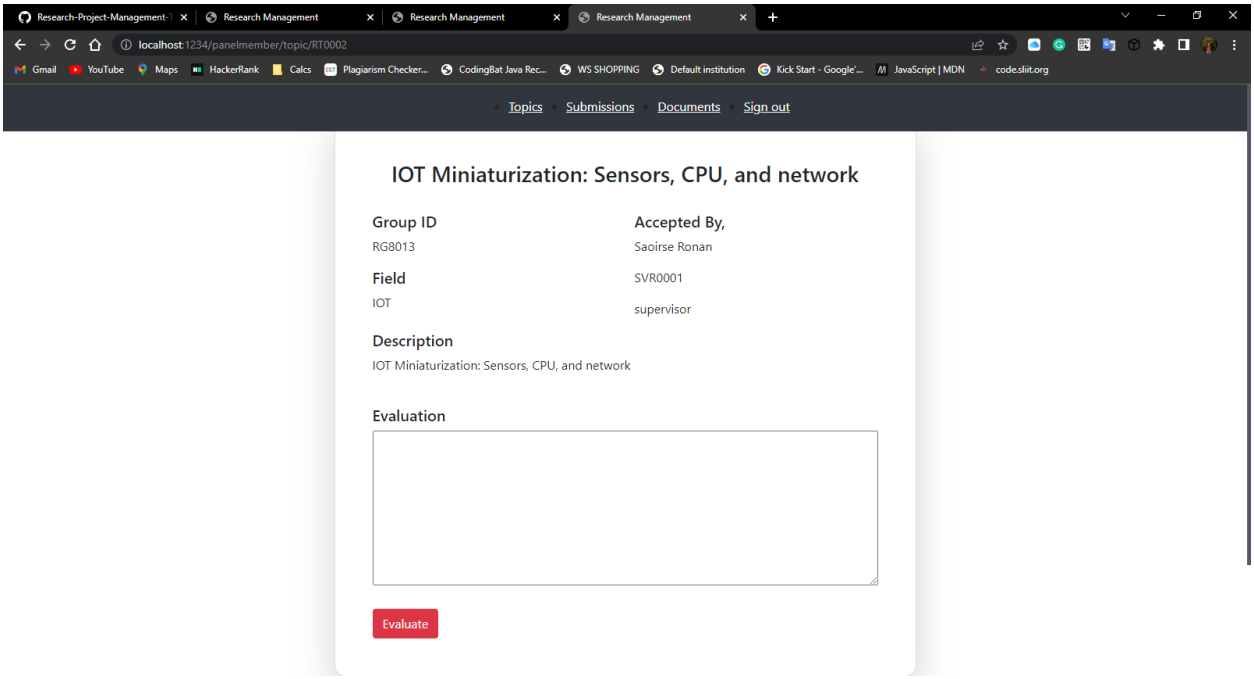
Choose...

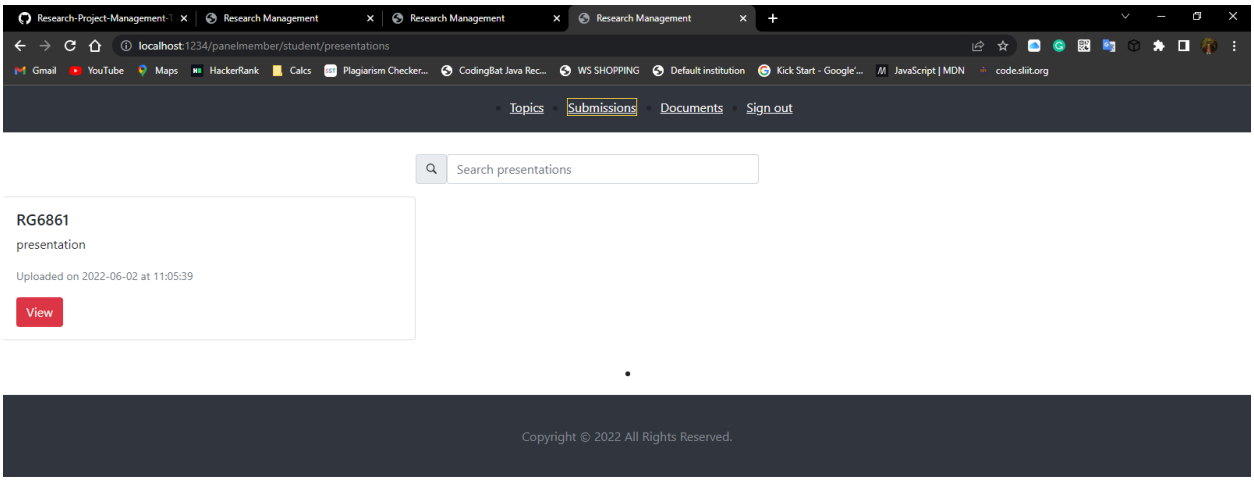
Contact number

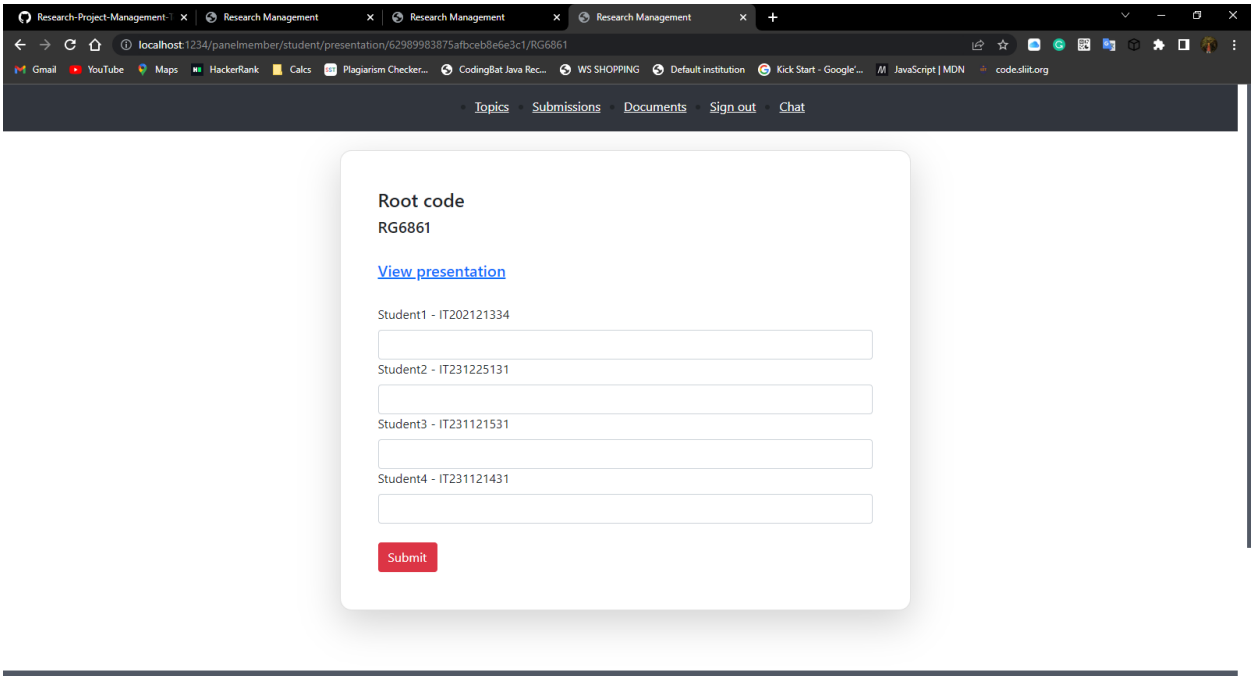
Sign Up


Research-Project-Management-T
Research Management
Research Management
Research Management
localhost:1234/panelmember/documents
Gmail
YouTube
Maps
HackerRank
Calcs
Plagiarism Checker...
CodingBat Java Rec...
WS SHOPPING
Default institution
Kick Start - Google'...
JavaScript | MDN
code.sliit.org
Topics
Submissions
Documents
Sign out
Search documents
Research Introduction
document
Introduce about student research module
View Document
Uploaded on 2022-06-04 at 09:44:52
Student Group Registration
document
Details about Student Group registration process
View Document
Uploaded on 2022-06-04 at 09:45:52
Student Research module delivery plan
document
Delivery plan of the research module in 4 th year
View Document
Uploaded on 2022-06-04 at 09:46:47
SRS document marking schema
marking
SRS document evaluation sheet
View Document
Uploaded on 2022-06-04 at 09:51:51
SRS document marking schema
marking
SRS document evaluation sheet
Architecture plan marking schema
marking
Architecture plan evaluation sheet

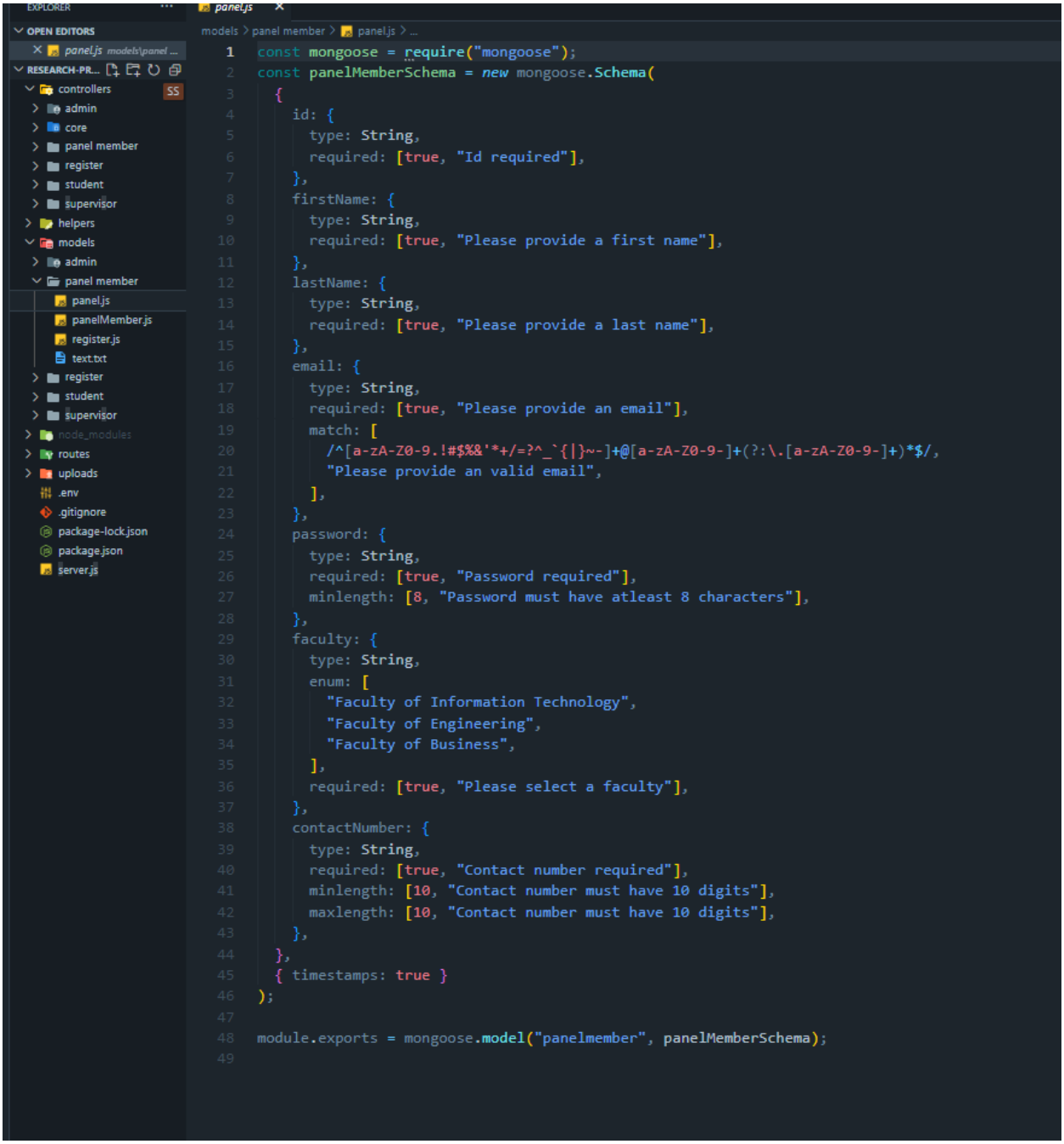

Research-Project-Management-T
Research Management
Research Management
Research Management
localhost:1234/panelmember/topics
Gmail
YouTube
Maps
HackerRank
Calcs
Plagiarism Checker...
CodingBat Java Rec...
WS SHOPPING
Default institution
Kick Start - Google'...
JavaScript | MDN
code.sliit.org
Topics
Submissions
Documents
Sign out
Pending
1
Evaluated
0
IOT Miniaturization: Sensors, CPU, and network
IOT Miniaturization: Sensors, CPU, and network
Submitted on 2022-06-04 at 09:59:25
Accepted on 2022-06-04 at 16:19:05
Evaluate
Copyright © 2022 All Rights Reserved.

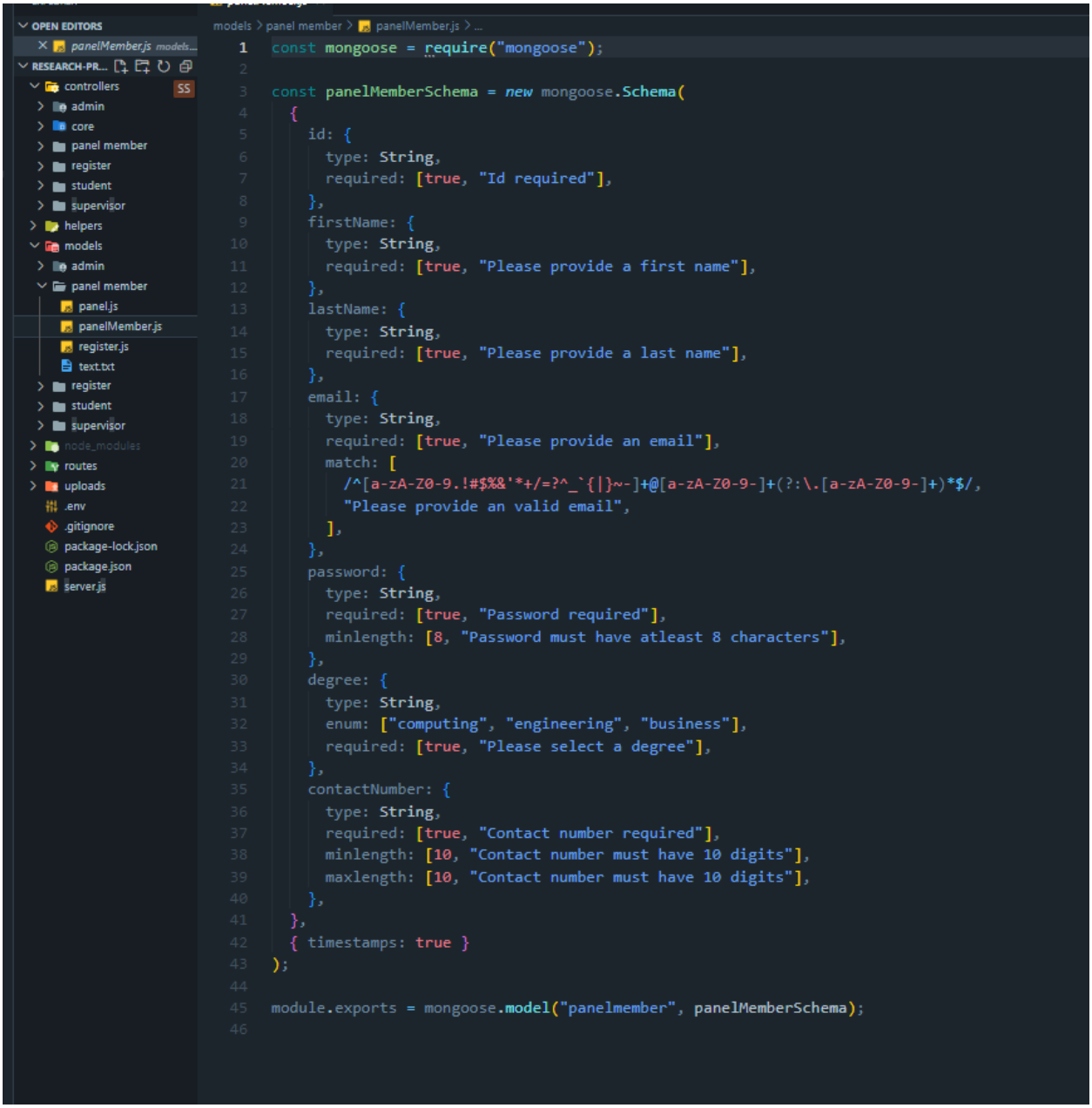

Research-Project-Management-T
Research Management
Research Management
Research Management
localhost:1234/panelmember/topic/RT0002
Gmail
YouTube
Maps
HackerRank
Calcs
Plagiarism Checker...
CodingBat Java Rec...
WS SHOPPING
Default institution
Kick Start - Google'...
JavaScript | MDN
code.sliit.org
Topics
Submissions
Documents
Sign out
IOT Miniaturization: Sensors, CPU, and network
Group ID
RG8013
Accepted By,
Saoirse Ronan
Field
IOT
SVR0001
supervisor
Description
IOT Miniaturization: Sensors, CPU, and network
Evaluation
Evaluate

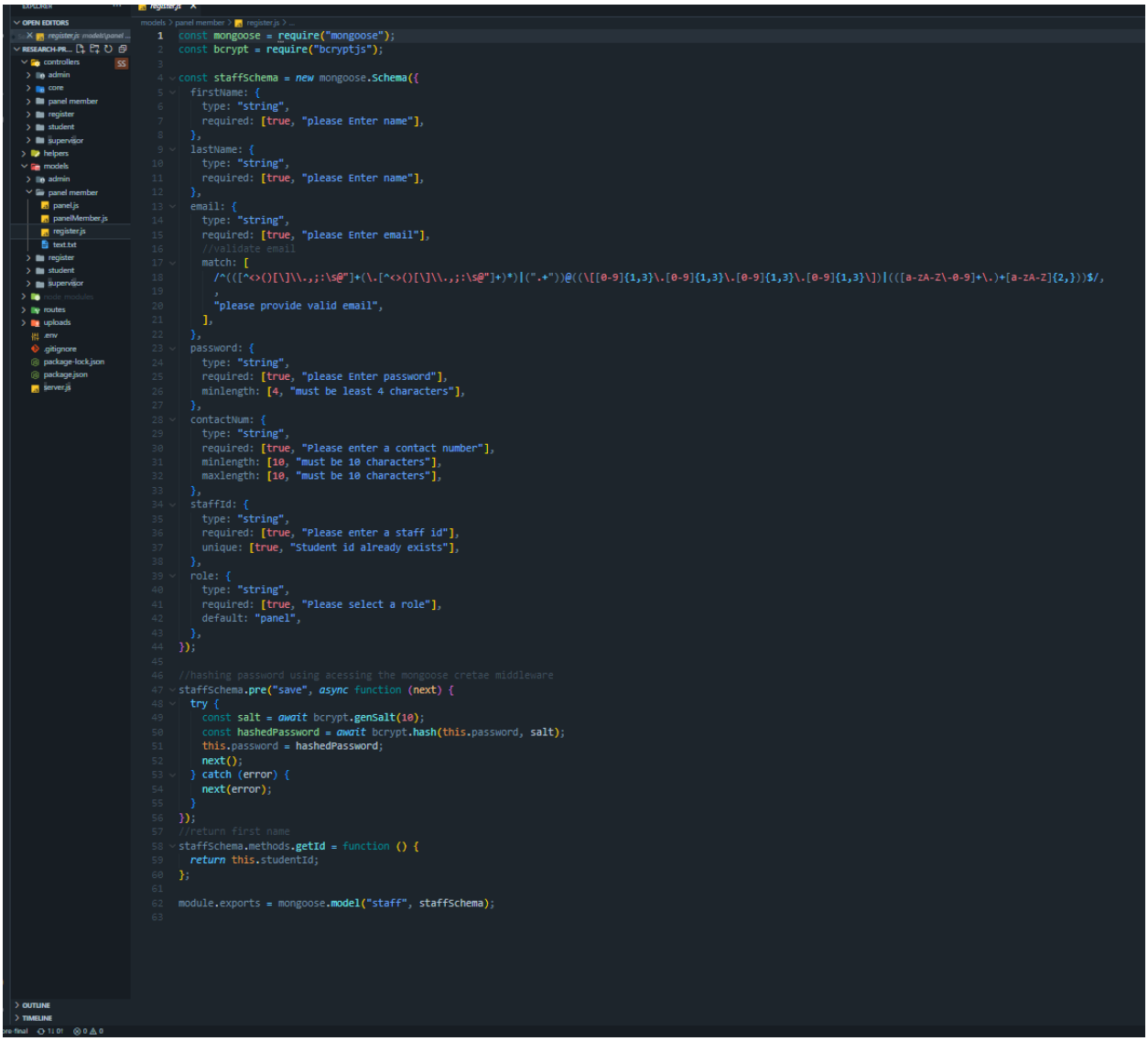

Research-Project-Management-T
Research Management
Research Management
Research Management
localhost:1234/panelmember/student/presentations
Gmail
YouTube
Maps
HackerRank
Calcs
Plagiarism Checker...
CodingBat Java Rec...
WS SHOPPING
Default institution
Kick Start - Google'...
JavaScript | MDN
code.sliit.org
Topics
Submissions
Documents
Sign out
Search presentations
RG6861
presentation
Uploaded on 2022-06-02 at 11:05:39
View
Copyright © 2022 All Rights Reserved.


Research-Project-Management-1
Research Management
Research Management
Research Management
localhost:1234/panelmember/student/presentation/62989983875afbceb8e6e3c1/RG6861
Gmail
YouTube
Maps
HackerRank
Calcs
Plagiarism Checker...
CodingBat Java Rec...
WS SHOPPING
Default institution
Kick Start - Google'...
JavaScript | MDN
code.sliit.org
Topics
Submissions
Documents
Sign out
Chat
Root code
RG6861
View presentation
Student1 - IT202121334
Student2 - IT231225131
Student3 - IT231121531
Student4 - IT231121431
Submit

Screenshots/ Code Snippets of the Backend:

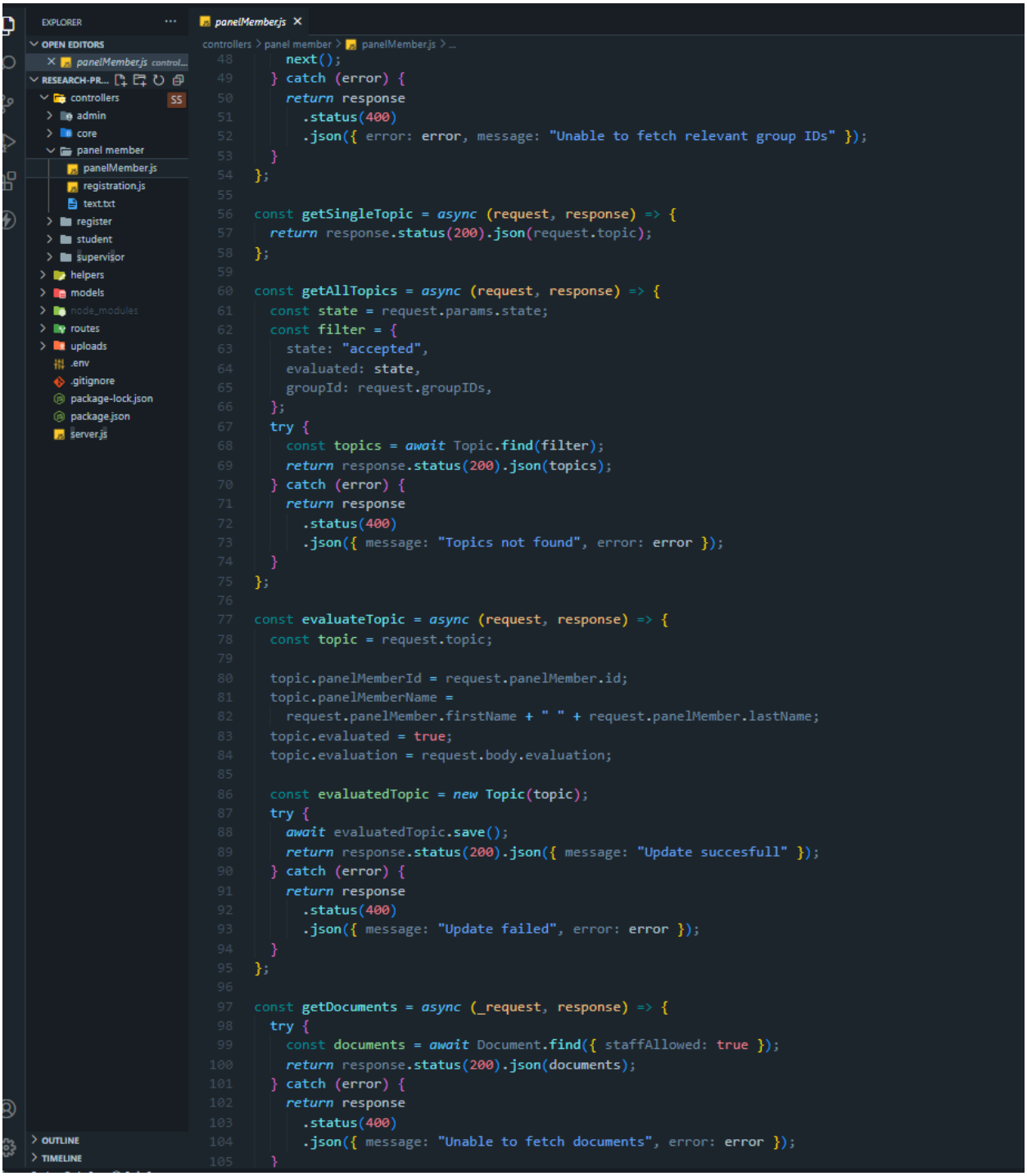



OPEN EDITORS
panelMember.js models...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
helpers
models
admin
panel member
panel.js
panelMember.js
register.js
text.txt
register
student
supervisor
node_modules
routes
uploads
.env
.gitignore
package-lock.json
package.json
server.js
models > panel member > panelMember.js > ...
const mongoose = require("mongoose");
const panelMemberSchema = new mongoose.Schema(
{
id: {
type: String,
required: [true, "Id required"],
},
firstName: {
type: String,
required: [true, "Please provide a first name"],
},
lastName: {
type: String,
required: [true, "Please provide a last name"],
},
email: {
type: String,
required: [true, "Please provide an email"],
match: [
/^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)*$/,
"Please provide an valid email",
],
},
password: {
type: String,
required: [true, "Password required"],
minlength: [8, "Password must have atleast 8 characters"],
},
degree: {
type: String,
enum: ["computing", "engineering", "business"],
required: [true, "Please select a degree"],
},
contactNumber: {
type: String,
required: [true, "Contact number required"],
minlength: [10, "Contact number must have 10 digits"],
maxlength: [10, "Contact number must have 10 digits"],
},
},
{ timestamps: true }
);
module.exports = mongoose.model("panelmember", panelMemberSchema);

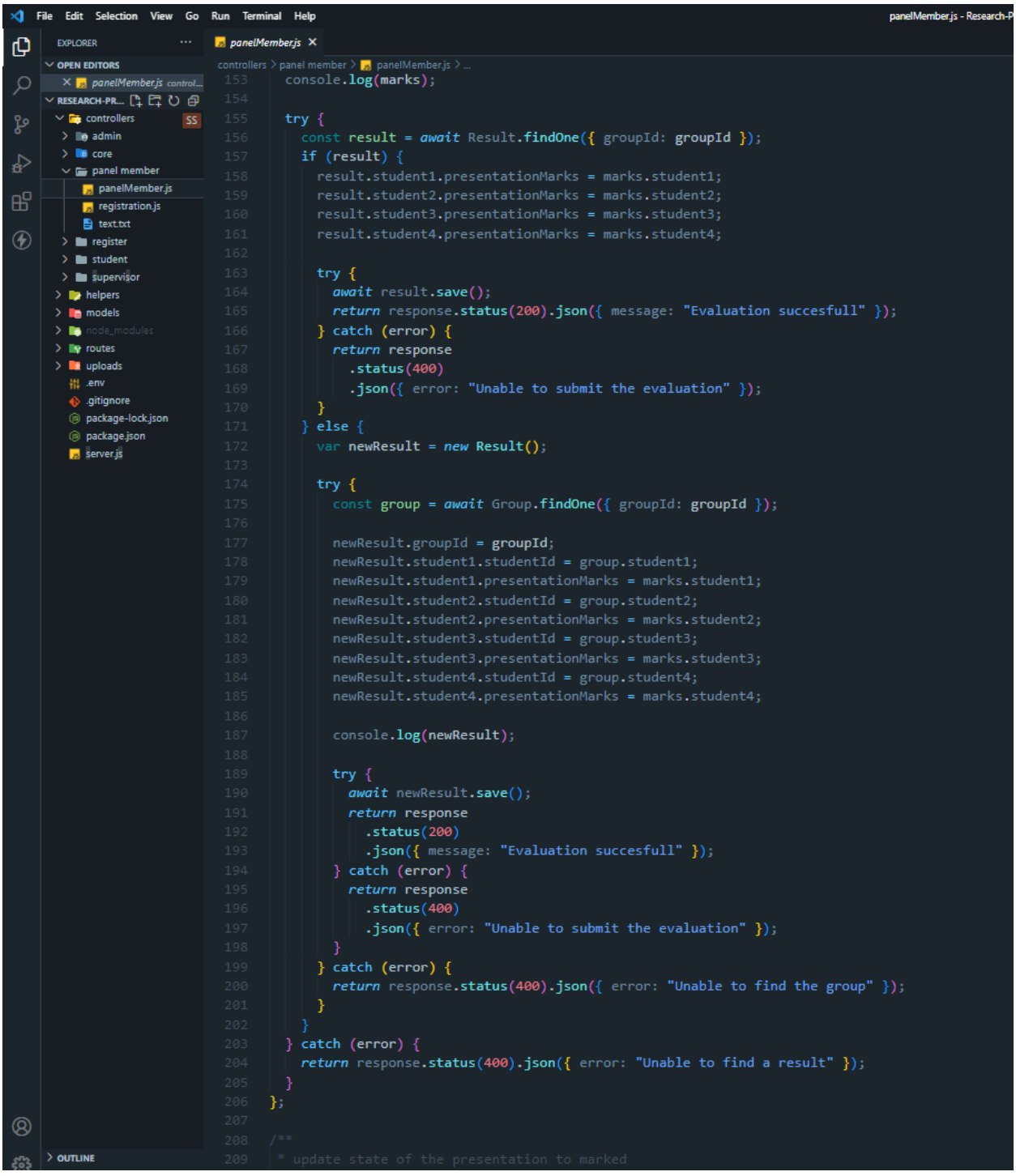


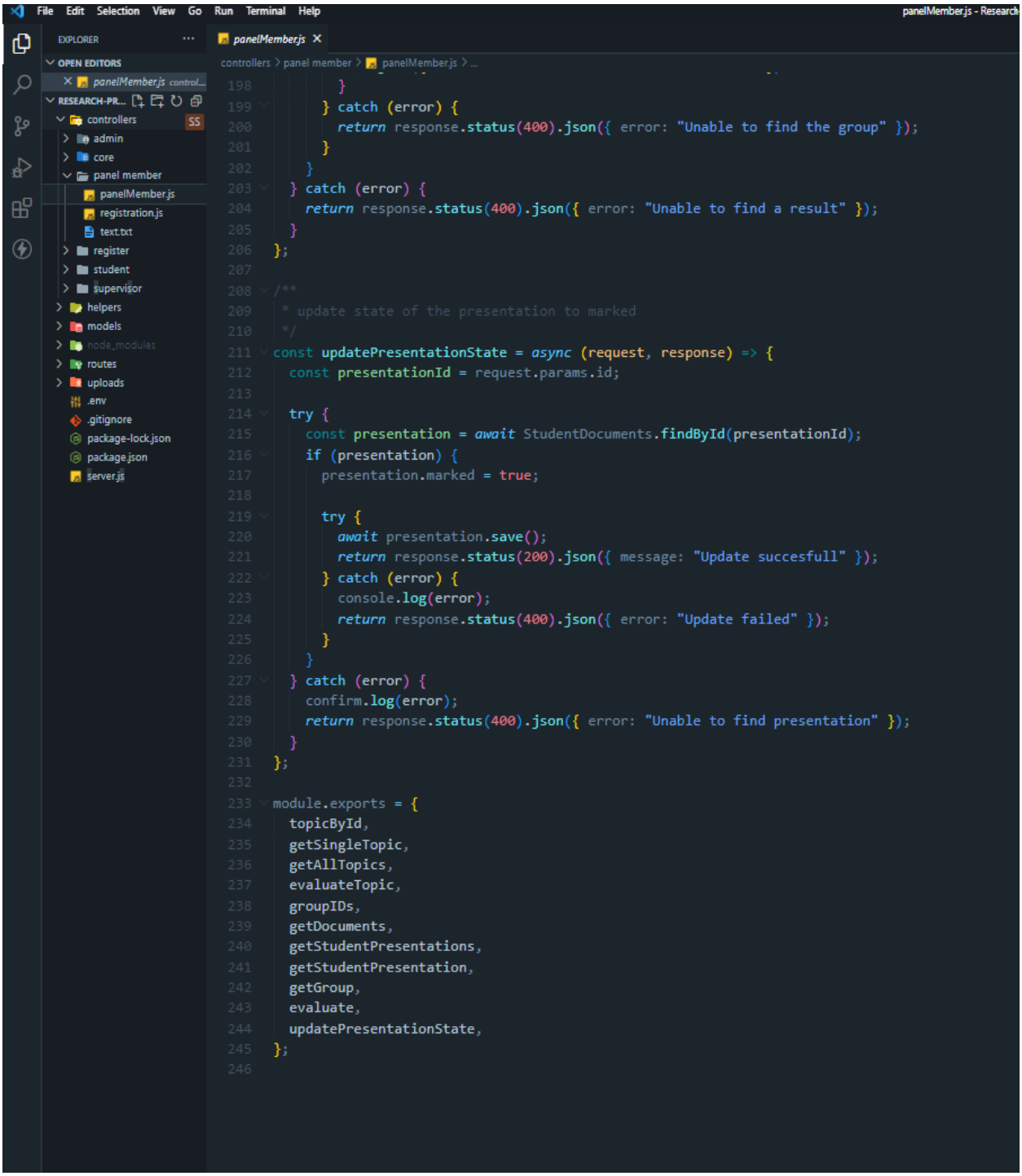
```
const mongoose = require("mongoose");
const bcrypt = require("bcryptjs");

const staffSchema = new mongoose.Schema({
  firstName: {
    type: "string",
    required: [true, "please Enter name"],
  },
  lastName: {
    type: "string",
    required: [true, "please Enter name"],
  },
  email: {
    type: "string",
    required: [true, "please Enter email"],
    //validate email
    match: [
      /^(([^<>()[\]\\.,;:\s@"]+(\.[^<>()[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/,
      ,
      "please provide valid email",
    ],
  },
  password: {
    type: "string",
    required: [true, "please Enter password"],
    minlength: [4, "must be least 4 characters"],
  },
  contactNum: {
    type: "string",
    required: [true, "Please enter a contact number"],
    minlength: [10, "must be 10 characters"],
    maxlength: [10, "must be 10 characters"],
  },
  staffId: {
    type: "string",
    required: [true, "Please enter a staff id"],
    unique: [true, "Student id already exists"],
  },
  role: {
    type: "string",
    required: [true, "Please select a role"],
    default: "panel",
  },
});

//hashing password using acessing the mongoose cretae middleware
staffSchema.pre("save", async function (next) {
  try {
    const salt = await bcrypt.genSalt(10);
    const hashedPassword = await bcrypt.hash(this.password, salt);
    this.password = hashedPassword;
    next();
  } catch (error) {
    next(error);
  }
});
//return first name
staffSchema.methods.getId = function () {
  return this.studentId;
};

module.exports = mongoose.model("staff", staffSchema);

```

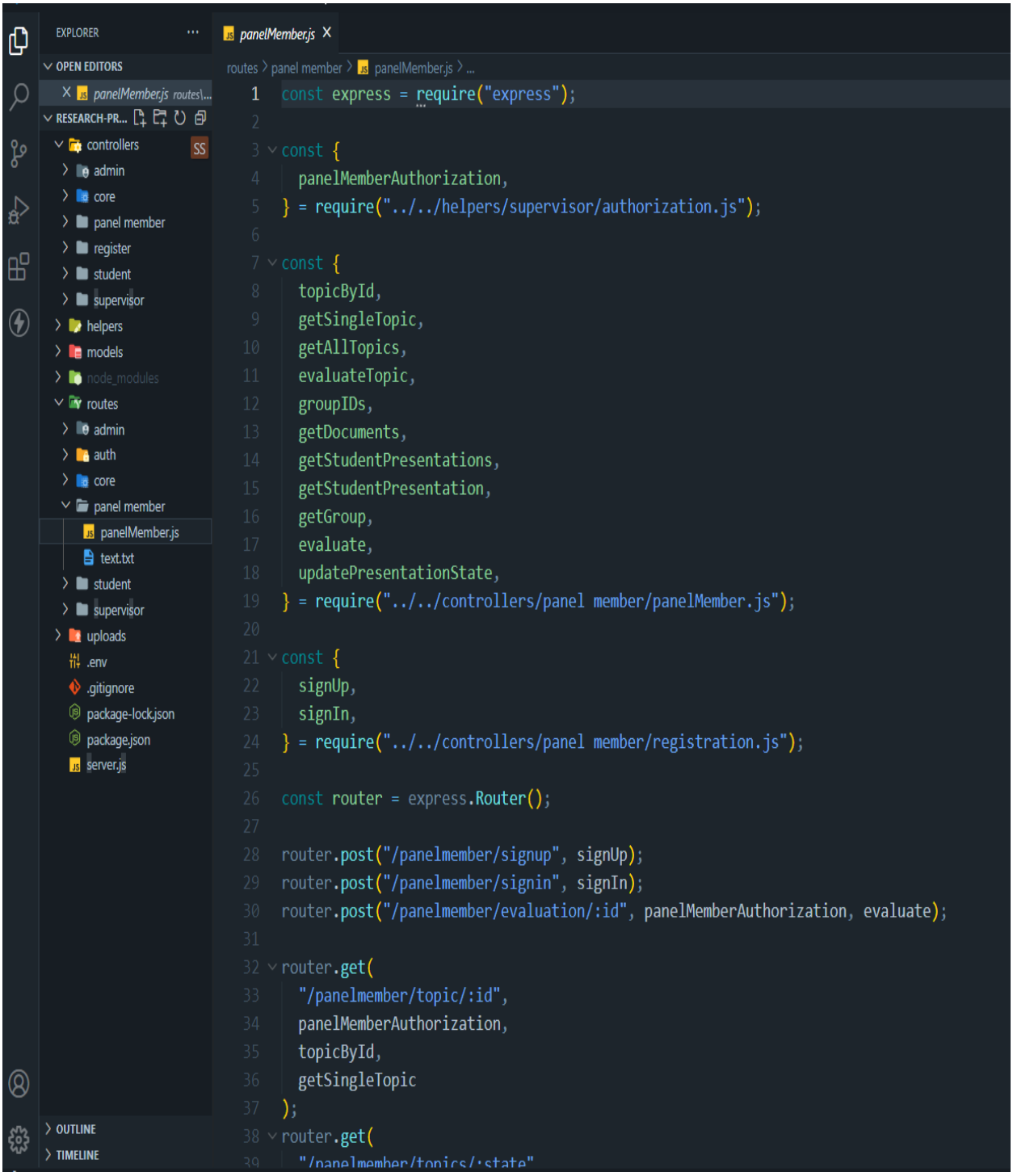

OPEN EDITORS
panelMember.js controll...
RESEARCH-PR...
controllers
admin
core
panel member
panelMember.js
registration.js
text.txt
register
student
supervisor
helpers
models
node_modules
routes
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
controllers > panel member > panelMember.js > ...
1 const Topic = require("../../models/student/research");
2 const Group = require("../../models/student/group.js");
3 const StudentDocuments = require("../../models/student/document.js");
4 const Result = require("../../models/student/result.js");
5 const Document = require("../../models/admin/document");
6
7 const topicById = async (request, response, next) => {
8 const topicId = request.params.id;
9 const filter = {
10 topicId: topicId,
11 state: "accepted",
12 };
13 try {
14 const topic = await Topic.findOne(filter);
15 if (topic) {
16 request.topic = topic;
17 next();
18 } else {
19 return response
20 .status(400)
21 .json({ message: "Topic could not be found", error: error });
22 }
23 } catch (error) {
24 return response
25 .status(400)
26 .json({ message: "Topic could not be found", error: error });
27 }
28 };
29
30 const groupIDs = async (request, response, next) => {
31 const panelMemberId = request.panelMember.id;
32
33 try {
34 const groupIDs = await Group.find({
35 $or: [
36 { panelMember1: panelMemberId },
37 { panelMember2: panelMemberId },
38 { panelMember3: panelMemberId },
39],
40 }).select("groupId -_id");
41 var IDArray = [];
42
43 groupIDs.forEach((id) => {
44 IDArray.push(id.groupId);
45 });
46
47 request.groupIDs = IDArray;
48 next();
49 } catch (error) {
50 return response
51 .status(400)
52 .json({ error: error, message: "Unable to fetch relevant group IDs" });
53 }
54 };
55
56 const getSingleTopic = async (request, response) => {
57 return response.status(200).json(request.topic);

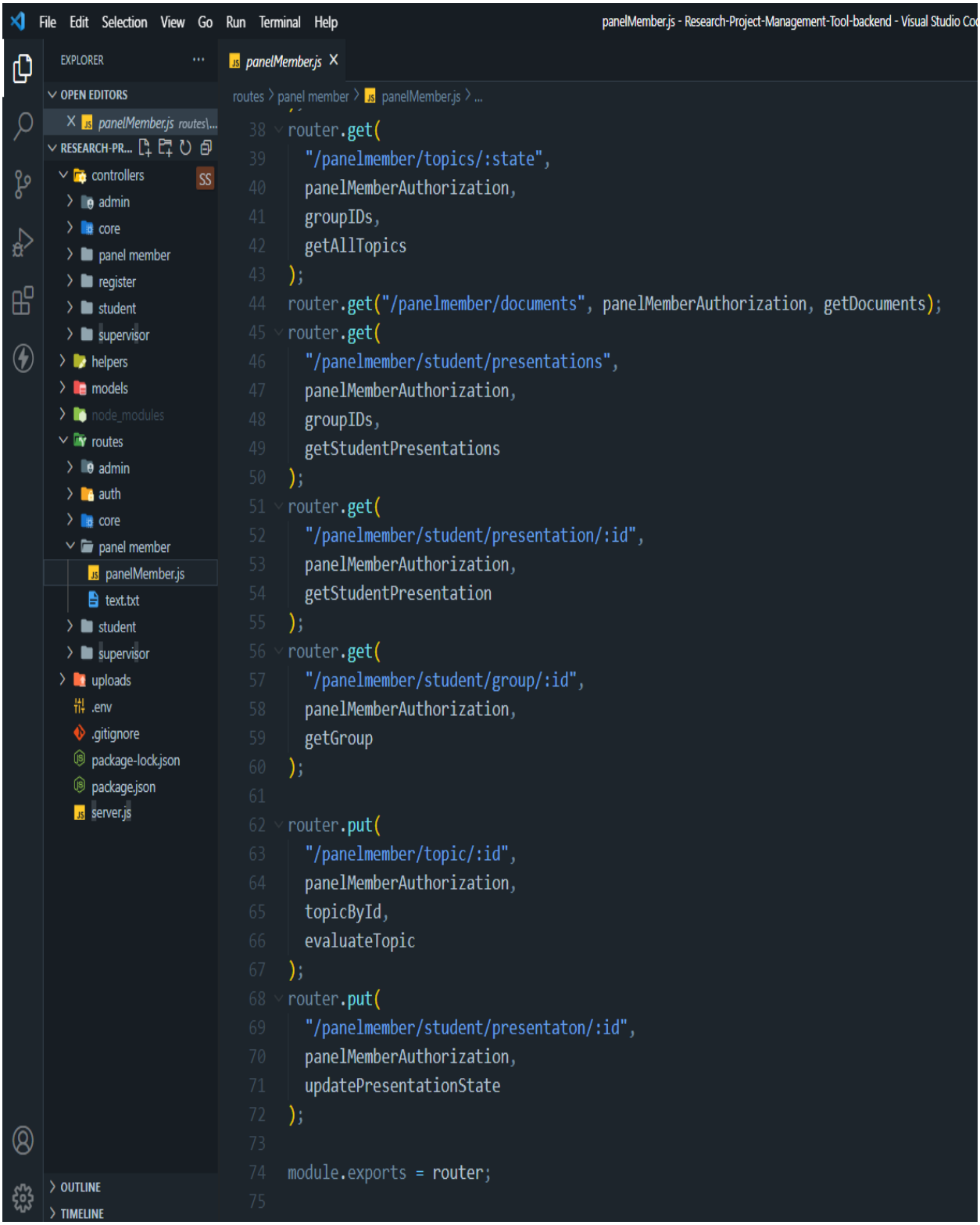

EXPLORER
OPEN EDITORS
panelMember.js controll...
RESEARCH-PR...
controllers
admin
core
panel member
panelMember.js
registration.js
text.txt
register
student
supervisor
helpers
models
node_modules
routes
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
panelMember.js
controllers > panel member > panelMember.js > ...
48 next();
49 } catch (error) {
50 return response
51 .status(400)
52 .json({ error: error, message: "Unable to fetch relevant group IDs" });
53 }
54 };
55
56 const getSingleTopic = async (request, response) => {
57 return response.status(200).json(request.topic);
58 };
59
60 const getAllTopics = async (request, response) => {
61 const state = request.params.state;
62 const filter = {
63 state: "accepted",
64 evaluated: state,
65 groupId: request.groupIDs,
66 };
67 try {
68 const topics = await Topic.find(filter);
69 return response.status(200).json(topics);
70 } catch (error) {
71 return response
72 .status(400)
73 .json({ message: "Topics not found", error: error });
74 }
75 };
76
77 const evaluateTopic = async (request, response) => {
78 const topic = request.topic;
79
80 topic.panelMemberId = request.panelMember.id;
81 topic.panelMemberName =
82 request.panelMember.firstName + " " + request.panelMember.lastName;
83 topic.evaluated = true;
84 topic.evaluation = request.body.evaluation;
85
86 const evaluatedTopic = new Topic(topic);
87 try {
88 await evaluatedTopic.save();
89 return response.status(200).json({ message: "Update succesfull" });
90 } catch (error) {
91 return response
92 .status(400)
93 .json({ message: "Update failed", error: error });
94 }
95 };
96
97 const getDocuments = async (_request, response) => {
98 try {
99 const documents = await Document.find({ staffAllowed: true });
100 return response.status(200).json(documents);
101 } catch (error) {
102 return response
103 .status(400)
104 .json({ message: "Unable to fetch documents", error: error });
105 }


File Edit Selection View Go Run Terminal Help
panelMember.js - Research-Pro
EXPLORER
panelMember.js
OPEN EDITORS
panelMember.js controll...
RESEARCH-PR...
controllers
admin
core
panel member
panelMember.js
registration.js
text.txt
register
student
supervisor
helpers
models
node_modules
routes
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
pre-final
controllers > panel member > panelMember.js > ...
try {
const documents = await Document.find({ staffAllowed: true });
return response.status(200).json(documents);
} catch (error) {
return response
.status(400)
.json({ message: "Unable to fetch documents", error: error });
}
};
const getStudentPresentations = async (request, response) => {
const filter = {
groupId: request.groupIDs,
type: "presentation",
marked: false,
};
try {
const presentations = await StudentDocuments.find(filter);
return response.status(200).json(presentations);
} catch (error) {
return response
.status(400)
.json({ error: error, message: "Unable to fetch presentations" });
}
};
const getStudentPresentation = async (request, response) => {
const presentationId = request.params.id;
try {
const presentation = await StudentDocuments.findById(presentationId);
return response.status(200).json(presentation);
} catch (error) {
return response
.status(400)
.json({ error: error, message: "Unable to the fetch presentation" });
}
};
const getGroup = async (request, response) => {
const groupId = request.params.id;
try {
const group = await Group.findOne({ groupId: groupId });
return response.status(200).json(group);
} catch (error) {
return response.status(400).json({ error: "Unable to find the group" });
}
};
const evaluate = async (request, response) => {
const marks = request.body;
const groupId = request.params.id;
console.log(marks);

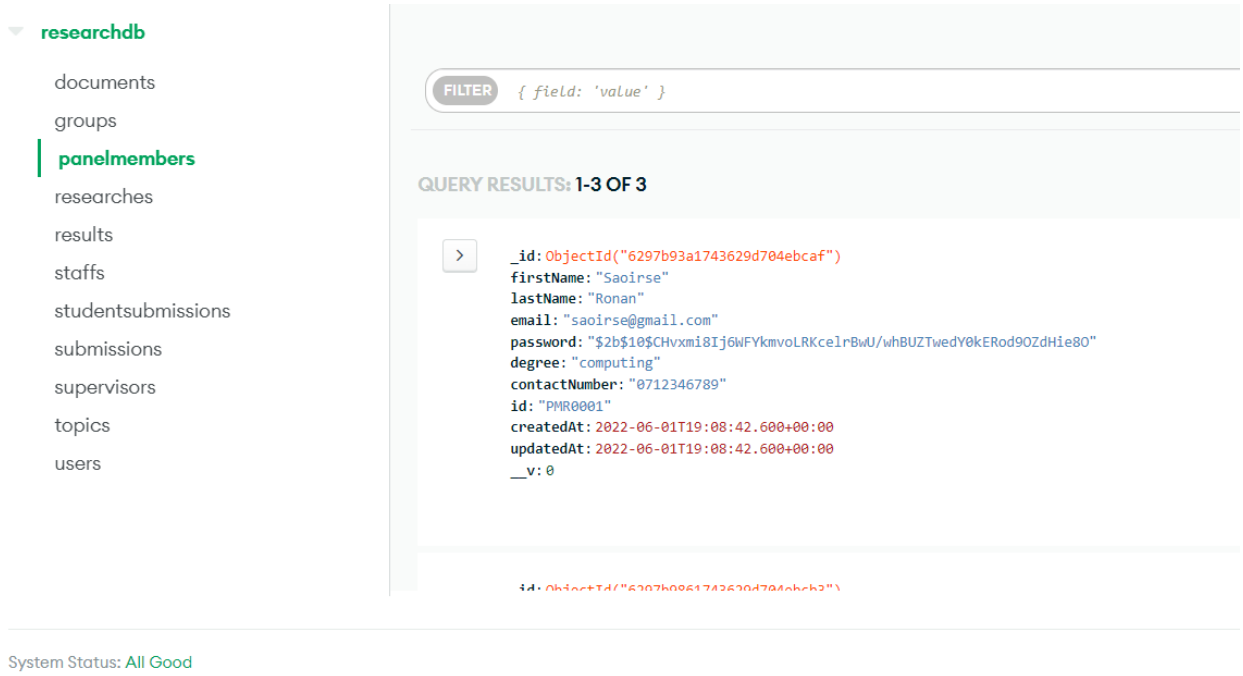

File Edit Selection View Go Run Terminal Help
panelMember.js - Research-P
EXPLORER
OPEN EDITORS
panelMember.js controll...
RESEARCH-PR...
controllers
admin
core
panel member
panelMember.js
registration.js
text.txt
register
student
supervisor
helpers
models
node_modules
routes
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
panelMember.js
controllers > panel member > panelMember.js > ...
153 console.log(marks);
154
155 try {
156 const result = await Result.findOne({ groupId: groupId });
157 if (result) {
158 result.student1.presentationMarks = marks.student1;
159 result.student2.presentationMarks = marks.student2;
160 result.student3.presentationMarks = marks.student3;
161 result.student4.presentationMarks = marks.student4;
162
163 try {
164 await result.save();
165 return response.status(200).json({ message: "Evaluation succesfull" });
166 } catch (error) {
167 return response
168 .status(400)
169 .json({ error: "Unable to submit the evaluation" });
170 }
171 } else {
172 var newResult = new Result();
173
174 try {
175 const group = await Group.findOne({ groupId: groupId });
176
177 newResult.groupId = groupId;
178 newResult.student1.studentId = group.student1;
179 newResult.student1.presentationMarks = marks.student1;
180 newResult.student2.studentId = group.student2;
181 newResult.student2.presentationMarks = marks.student2;
182 newResult.student3.studentId = group.student3;
183 newResult.student3.presentationMarks = marks.student3;
184 newResult.student4.studentId = group.student4;
185 newResult.student4.presentationMarks = marks.student4;
186
187 console.log(newResult);
188
189 try {
190 await newResult.save();
191 return response
192 .status(200)
193 .json({ message: "Evaluation succesfull" });
194 } catch (error) {
195 return response
196 .status(400)
197 .json({ error: "Unable to submit the evaluation" });
198 }
199 } catch (error) {
200 return response.status(400).json({ error: "Unable to find the group" });
201 }
202 }
203 } catch (error) {
204 return response.status(400).json({ error: "Unable to find a result" });
205 }
206 };
207
208 /**
209 * update state of the presentation to marked


File Edit Selection View Go Run Terminal Help
panelMember.js - Research
EXPLORER
OPEN EDITORS
panelMember.js controll...
RESEARCH-PR...
controllers
admin
core
panel member
panelMember.js
registration.js
text.txt
register
student
supervisor
helpers
models
node_modules
routes
uploads
.env
.gitignore
package-lock.json
package.json
server.js
panelMember.js
controllers > panel member > panelMember.js > ...
198 }
199 } catch (error) {
200 return response.status(400).json({ error: "Unable to find the group" });
201 }
202 }
203 } catch (error) {
204 return response.status(400).json({ error: "Unable to find a result" });
205 }
206 };
207
208 /**
209 * update state of the presentation to marked
210 */
211 const updatePresentationState = async (request, response) => {
212 const presentationId = request.params.id;
213
214 try {
215 const presentation = await StudentDocuments.findById(presentationId);
216 if (presentation) {
217 presentation.marked = true;
218
219 try {
220 await presentation.save();
221 return response.status(200).json({ message: "Update succesfull" });
222 } catch (error) {
223 console.log(error);
224 return response.status(400).json({ error: "Update failed" });
225 }
226 }
227 } catch (error) {
228 confirm.log(error);
229 return response.status(400).json({ error: "Unable to find presentation" });
230 }
231 };
232
233 module.exports = {
234 topicById,
235 getSingleTopic,
236 getAllTopics,
237 evaluateTopic,
238 groupIDs,
239 getDocuments,
240 getStudentPresentations,
241 getStudentPresentation,
242 getGroup,
243 evaluate,
244 updatePresentationState,
245 };
246

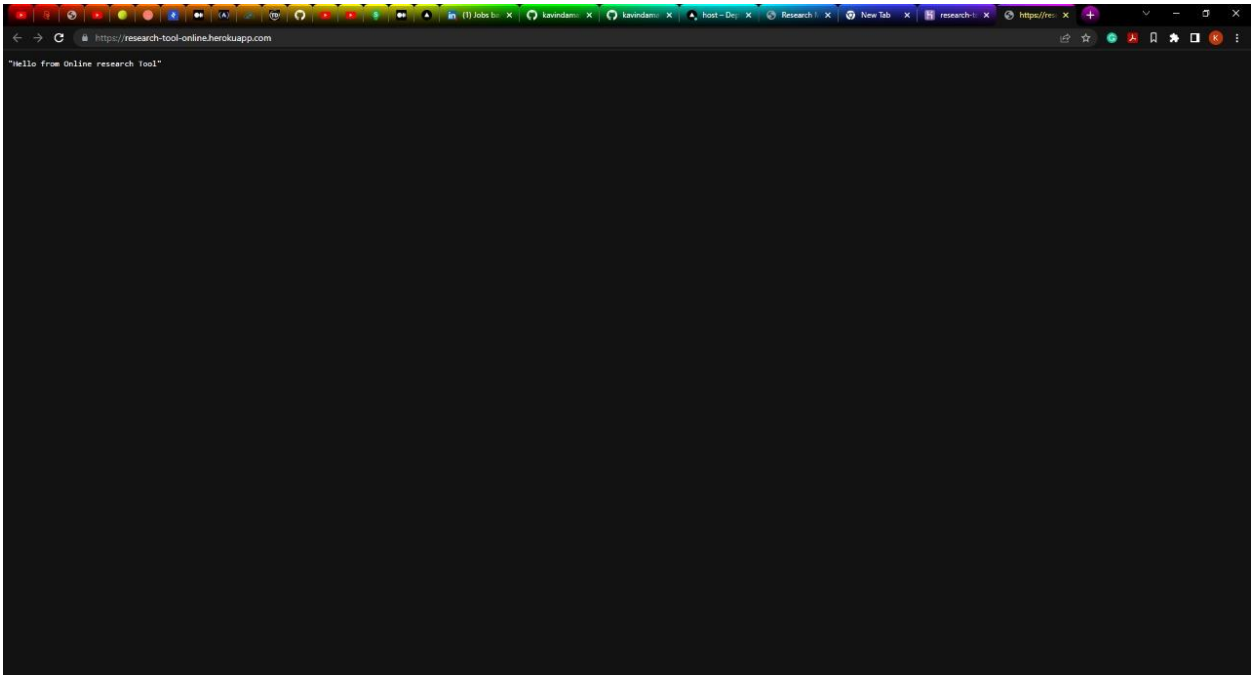

EXPLORER
panelMember.js
OPEN EDITORS
panelMember.js routes\...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
helpers
models
node_modules
routes
admin
auth
core
panel member
panelMember.js
text.txt
student
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
routes > panel member > panelMember.js > ...
1 const express = require("express");
2
3 const {
4 panelMemberAuthorization,
5 } = require("../../helpers/supervisor/authorization.js");
6
7 const {
8 topicById,
9 getSingleTopic,
10 getAllTopics,
11 evaluateTopic,
12 groupIDs,
13 getDocuments,
14 getStudentPresentations,
15 getStudentPresentation,
16 getGroup,
17 evaluate,
18 updatePresentationState,
19 } = require("../../controllers/panel member/panelMember.js");
20
21 const {
22 signUp,
23 signIn,
24 } = require("../../controllers/panel member/registration.js");
25
26 const router = express.Router();
27
28 router.post("/panelmember/signup", signUp);
29 router.post("/panelmember/signin", signIn);
30 router.post("/panelmember/evaluation/:id", panelMemberAuthorization, evaluate);
31
32 router.get(
33 "/panelmember/topic/:id",
34 panelMemberAuthorization,
35 topicById,
36 getSingleTopic
37);
38 router.get(

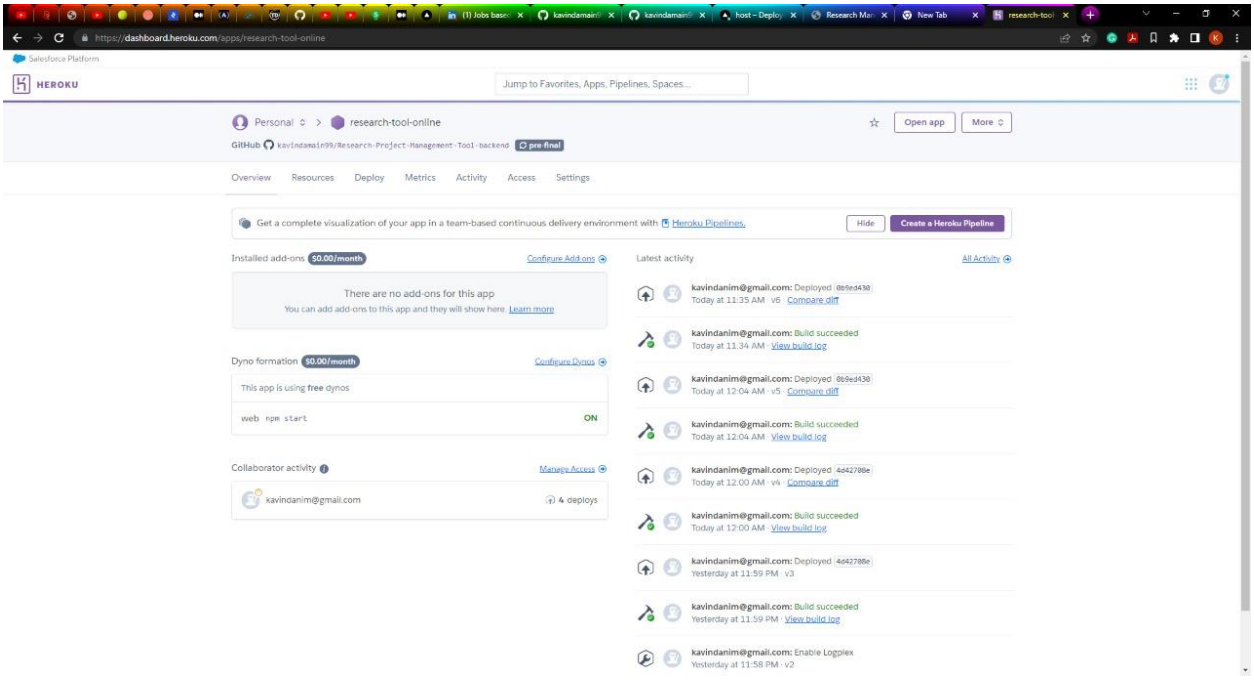

File Edit Selection View Go Run Terminal Help
panelMember.js - Research-Project-Management-Tool-backend - Visual Studio Co
EXPLORER
panelMember.js
OPEN EDITORS
panelMember.js routes\...
RESEARCH-PR...
controllers
admin
core
panel member
register
student
supervisor
helpers
models
node_modules
routes
admin
auth
core
panel member
panelMember.js
text.txt
student
supervisor
uploads
.env
.gitignore
package-lock.json
package.json
server.js
OUTLINE
TIMELINE
routes > panel member > panelMember.js > ...
router.get(
"/panelmember/topics/:state",
panelMemberAuthorization,
groupIDs,
getAllTopics
);
router.get("/panelmember/documents", panelMemberAuthorization, getDocuments);
router.get(
"/panelmember/student/presentations",
panelMemberAuthorization,
groupIDs,
getStudentPresentations
);
router.get(
"/panelmember/student/presentation/:id",
panelMemberAuthorization,
getStudentPresentation
);
router.get(
"/panelmember/student/group/:id",
panelMemberAuthorization,
getGroup
);
router.put(
"/panelmember/topic/:id",
panelMemberAuthorization,
topicById,
evaluateTopic
);
router.put(
"/panelmember/student/presentaton/:id",
panelMemberAuthorization,
updatePresentationState
);
module.exports = router;

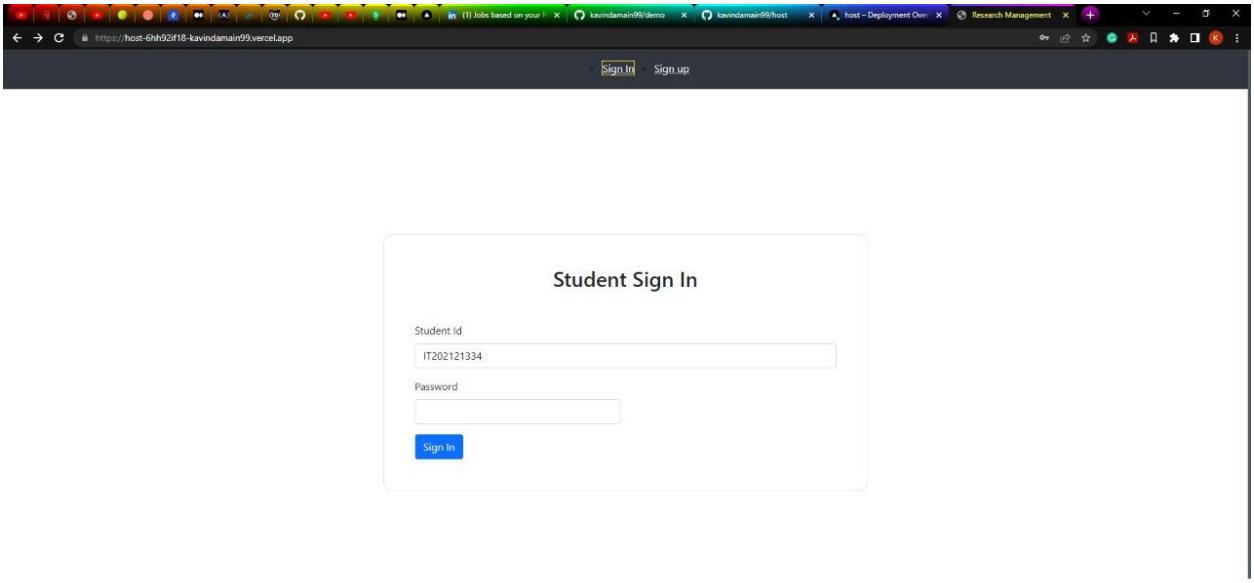
Screenshots of the MongoDB Schemas

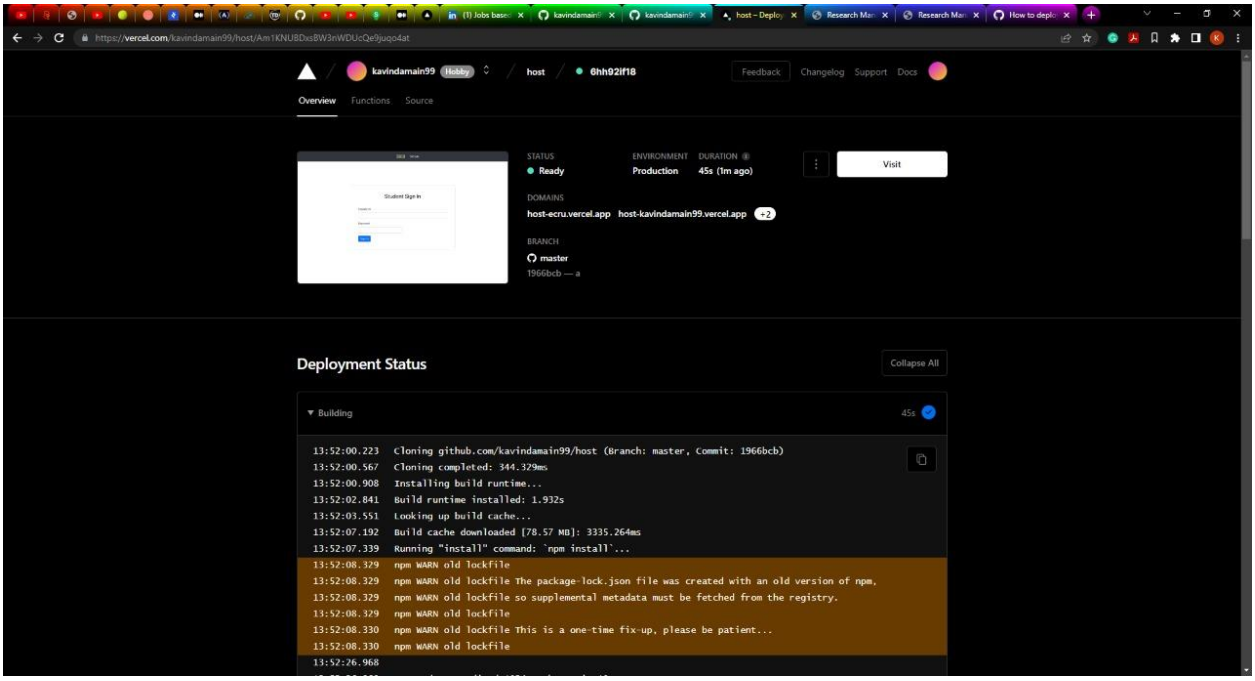


Proof of Hosting (Backend and Frontend)






https://host-6hh92if18-kavindamain99.vercel.app
Sign In
Sign up
Student Sign In
Student Id
IT202121334
Password
Sign In


https://vercel.com/kavindamain99/host/Am1KNU8Dxs8W3nWDUcQe9juqo4at
kavindamain99
Hobby
host
6hh92if18
Feedback
Changelog
Support
Docs
Overview
Functions
Source
STATUS
Ready
ENVIRONMENT
Production
DURATION
45s (1m ago)
Visit
DOMAINS
host-ecru.vercel.app
host-kavindamain99.vercel.app
+2
BRANCH
master
1966bcb — a
Deployment Status
Collapse All
Building
45s
13:52:00.223 Cloning github.com/kavindamain99/host (Branch: master, Commit: 1966bcb)
13:52:00.567 Cloning completed: 344.329ms
13:52:00.908 Installing build runtime...
13:52:02.841 Build runtime installed: 1.932s
13:52:03.551 Looking up build cache...
13:52:07.192 Build cache downloaded [78.57 MB]: 3335.264ms
13:52:07.339 Running "install" command: `npm install`...
13:52:08.329 npm WARN old lockfile
13:52:08.329 npm WARN old lockfile The package-lock.json file was created with an old version of npm,
13:52:08.329 npm WARN old lockfile so supplemental metadata must be fetched from the registry.
13:52:08.329 npm WARN old lockfile
13:52:08.330 npm WARN old lockfile This is a one-time fix-up, please be patient...
13:52:08.330 npm WARN old lockfile
13:52:26.968

Test Cases

Member ID : IT20226046	
Member Name: Kavinda M.A.I.N	
Testing Function 1: Insert a submission	
Test case ID : 01	**Test designed by:** IT20226046
Test priority (High/Medium/Low): Medium	Test executed by: Kumarathunga M.A.M.T
Test Description: Admin can decide the submission type of research paper.	

Test steps	Test data	Expected result	Actual result	Status
Login as admin	Email, password	Display admin page	Display admin page	Pass
Go to the dashboard	N/A	Display dashboard	Display dashboard	Pass
Select the submission	N/A	Display submission insert option	Display submission insert option	Pass
Insert a submission	N/A	Ask submission type	Ask submission type	Pass
Add submission type	Pdf, doc	Successfully added	Successfully added	Pass

Member ID : IT20134280	
Member Name: S. Ishalini	
Testing Function 1: Insert a submission	
Test case ID : 01	**Test designed by:** IT20134280
Test priority (High/Medium/Low): Medium	Test executed by: Kumarathunga M.A.M.T
Test Description: Student can register their student group.	

Test steps	Test data	Expected result	Actual result	Status
Register to the System	First name, last name, email, password, phone number	Display the login page	Display the login page	Pass
Login	Student ID, password	Display student page	Display student page	Pass
Select Topic Registration	Group Id, topic, description, research field	Successfully Submitted and display the submitted details	Successfully Submitted and display the submitted details	Pass

Member ID : IT20227036	
Member Name: Moremad M.M.M.B	
Testing Function 1:	
Test case ID : 01	**Test designed by:** IT20227036
Test priority (High/Medium/Low): Medium	Test executed by: Kumarathunga M.A.M.T
Test Description: Supervisor can be accept the student's research topic	

Test steps	Test data	Expected result	Actual result	Status
Register to the System	First name, last name, email, password, phone number	Display the login page	Display the login page	Pass
Login	Student ID, password	Display student page	Display student page	Pass
Select Topic	N / A	Display pending topics, accepted topics and rejected topics	Display pending topics, accepted topics and rejected topics	Pass
Select pending Topics	N / A	Display about the topic	Display about the topic	Pass
Select Accept Button	N / A	Successful message and display at the topic pages as a accepted topic	Successful message and display at the topic pages as a accepted topic	Pass

Member ID : IT20236564	
Member Name: Kumarathunga M.A.M.T	
Testing Function 1: Insert a submission	
Test case ID : 01	**Test designed by:** IT20235453
Test priority (High/Medium/Low): Medium	Test executed by: Kavinda M.A,I.N
Test Description: Panel member can see the students' final submissions and presentations.	

Test steps	Test data	Expected result	Actual result	Status
Login	Student ID, password	Display student page	Display student page	Pass
Go to Documents page	N/A	Display all the documents which were submitted by the student groups.	Display all the documents which were submitted by the student groups.	Pass
Select View Documents	N / A	Display Document's details	Display Document's details	Pass