CS 432: Databases

Assignment 3: Implementing a Web App using MySQL

Topic: Placement Management

Group Members

G1:
Pushpendra Pratap Singh (20110151)
Saatvik Rao (20110175)
Sahil Agrawal (20110178)
Sanskar Sharma (20110185)
Tanvi Dixit (20110212)

G2:
Dhyey Kumar Thummar (20110059)
Ksheer Sagar Agrawal (20110098)
Mihir Sutariya (20110208)
R Yeeshu Dhurandhar (20110152)
Shruhrid Banthia (20110198)
Utkarsh Mishra (20110218)

Responsibility of G1:

The code is submitted as per the instructions.

Responsibility of G2:

The code is submitted as per the instructions.

Responsibility of G1 & G2:

1. Create the database of the designed logical schema in your previous assignment number 2, enter dummy/imaginary values into it, and show these dummy entries on the web page.

The SQL database with dummy values is submitted to GitHub..

3a. Steps to run the Web App

We need three different emails due to the constraint that one person can not play the role of the above three roles. We will give access to the provided mails.

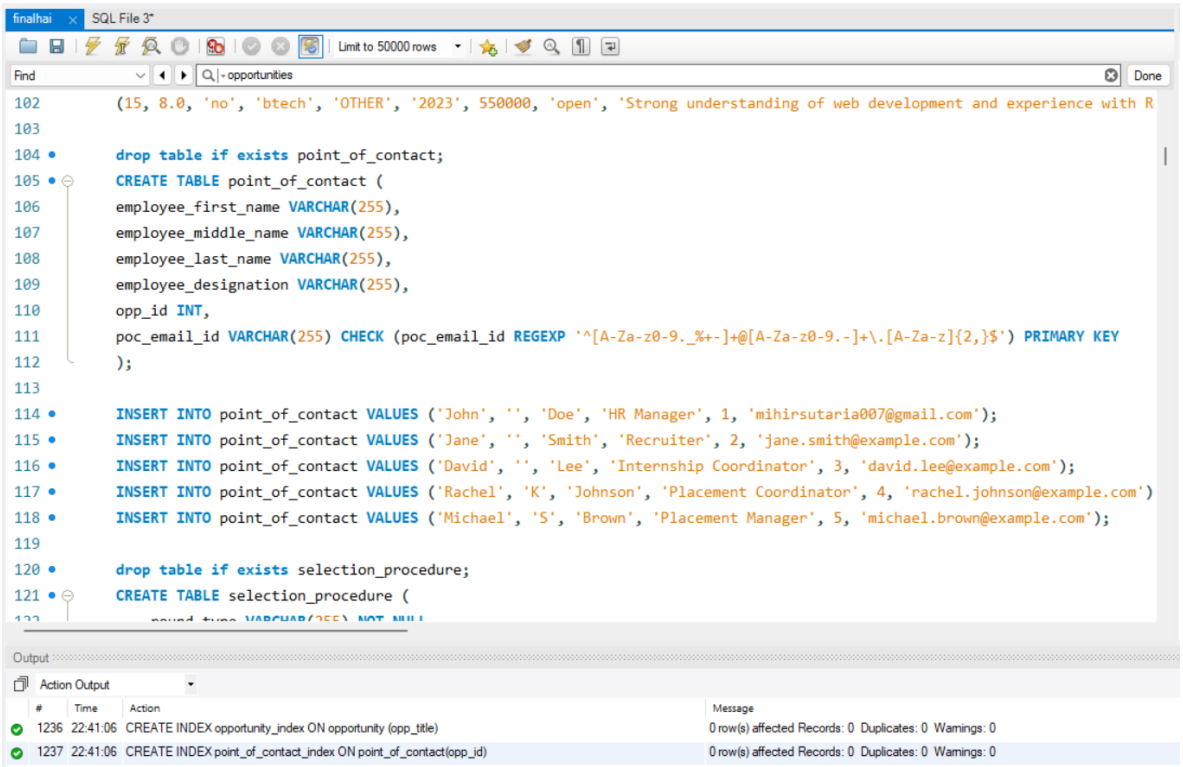
Step 1- Download the code from GitHub.

Step 2- Installing packages:
python3 -m venv env_flask
pip3 install flask

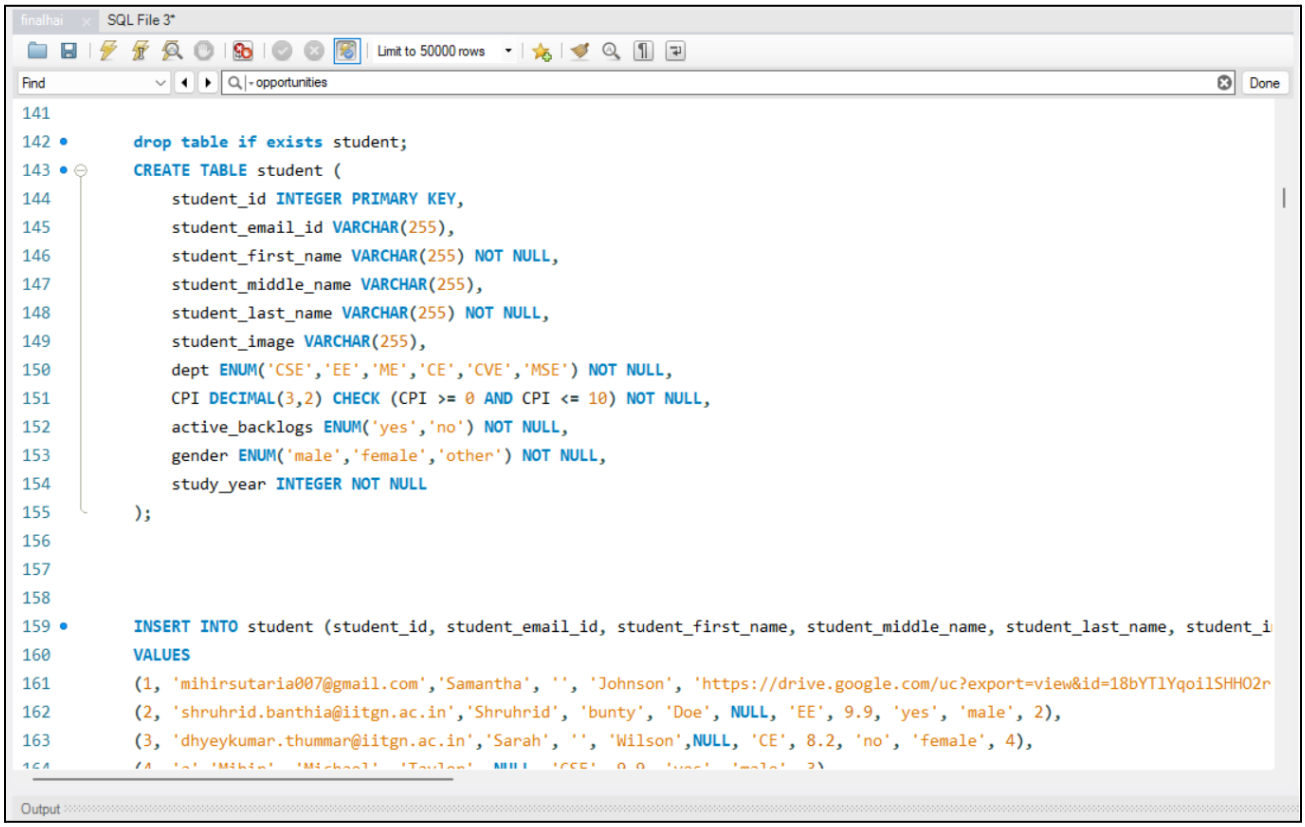
Step 3- To setup all the requirements for frontend as well as backend, write the command:
pip install -r requirements.txt

Step 4- To get access :

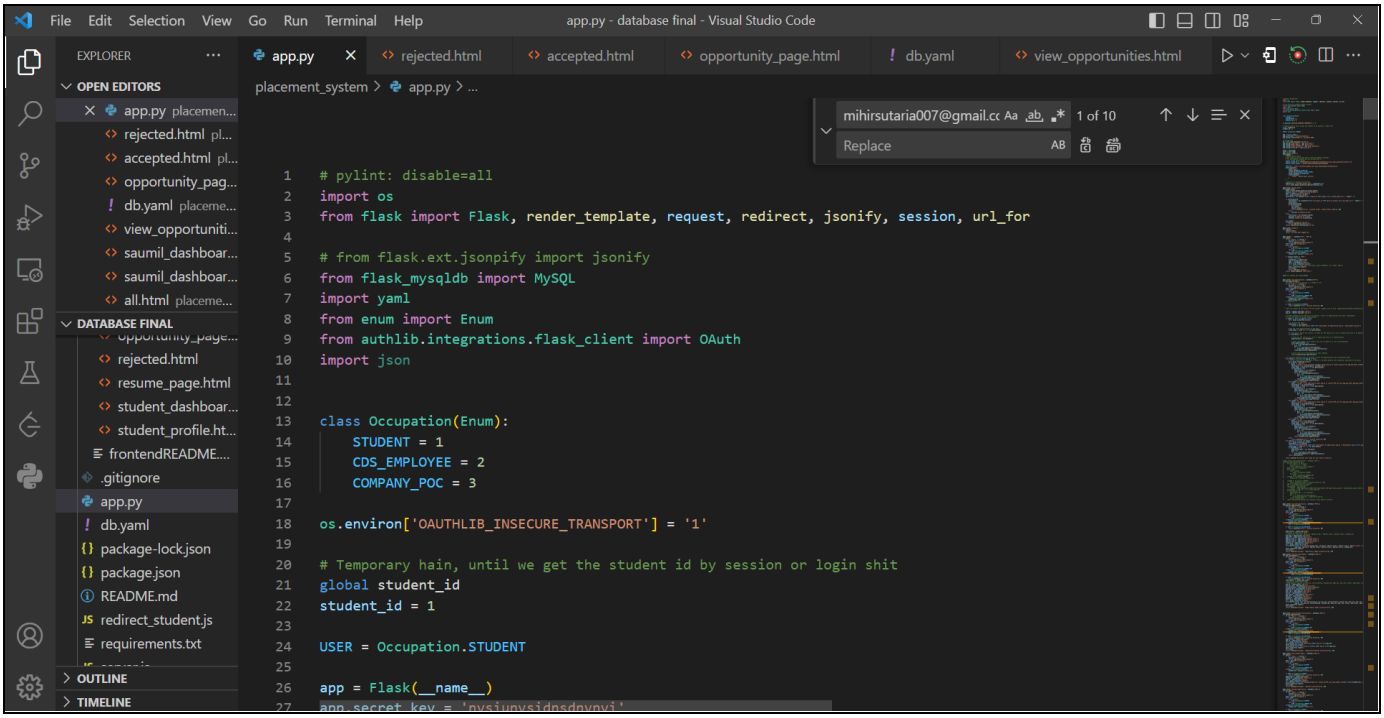
- For point of contact : add a point of contact details or replace the 'mihirsutaria007@gmail.com' gmail id (line 114) in the SQL file to get access to the point of contact page.



- For students : add your email id (and other student details) or replace the ‘mihirsutaria007@gmail.com’ gmail id (line 161) in the SQL file to get access to the student page.



- For CDS employee : Search ‘mihirsutaria007@gmail.com’ in the app.py file and replace all with your CDS employee email id.



If any problems, please contact us at <sutariya.mihirkumar@iitgn.ac.in>.

Run the whole SQL file in the workbench provided in the GitHub.

Run in terminal using the below code: python app.py

- ➢ To access the student view: http://localhost:5000/student
- ➢ To access the cds employee view: http://localhost:5000/cds
- ➢ To access the company poc (point of contact) view: http://localhost:5000/poc
- ➢ To logout : http://localhost:5000/logout

NOTE:

- To change view between student, cds employee and company poc, you have to logout from the previous view (you will have to change the email address accordingly) and then go to links mentioned above to log into the required view.
- You might face problem in running this in MacOS.

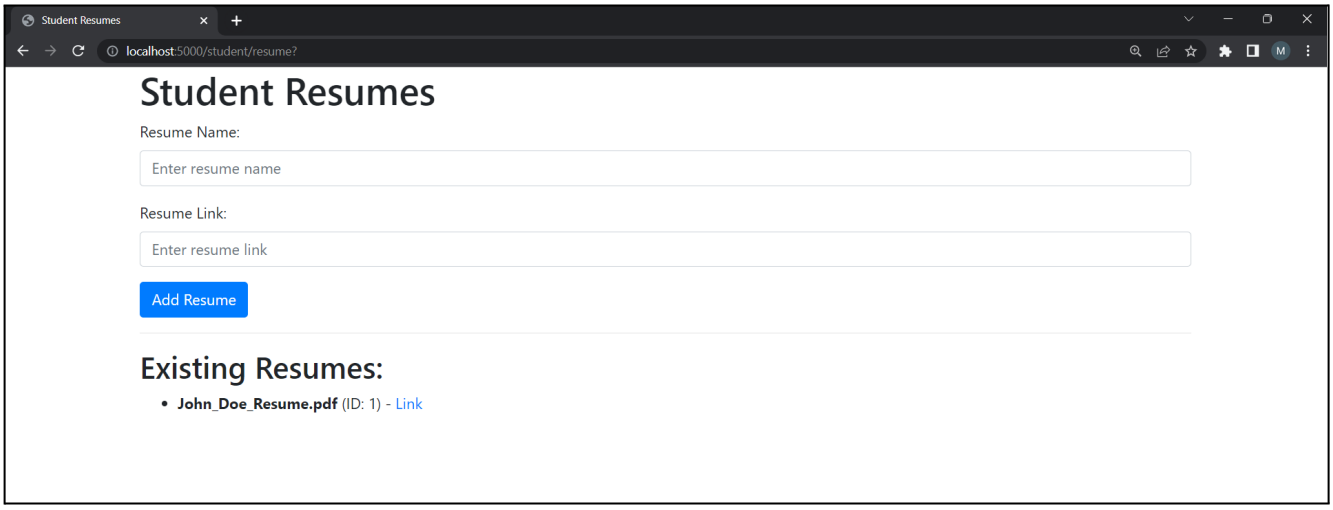
3b. Screenshots of successful execution of the dynamic operations

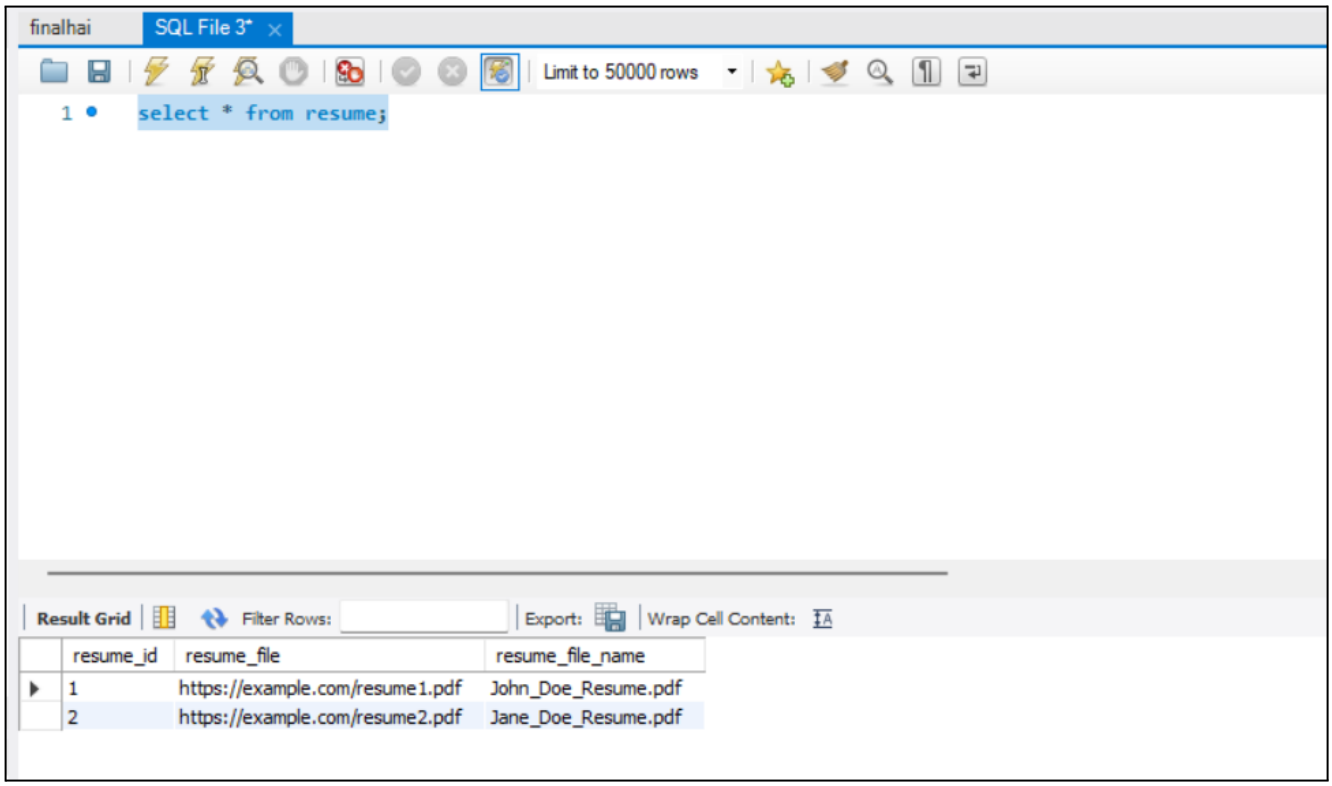
The web app supports the dynamic execution (the changes done by the user through the web app shall be reflected in the main database and web page also) of the following functions on your database:

A. **INSERT**

a. **Before** inserting the data

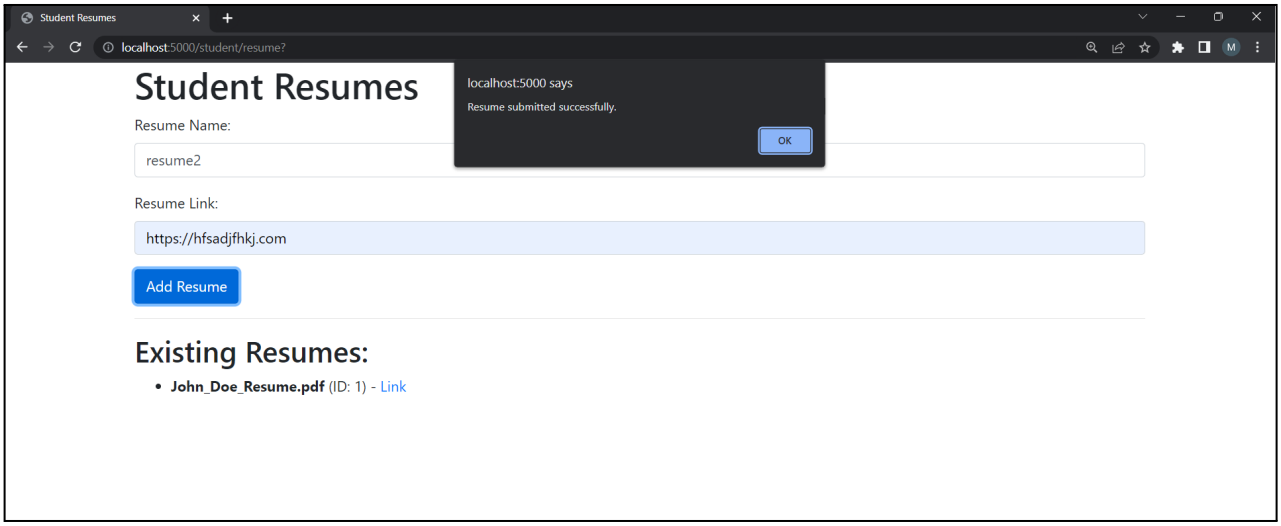
This is how the page looks before inserting a resume. We can see existing resumes which were uploaded earlier.





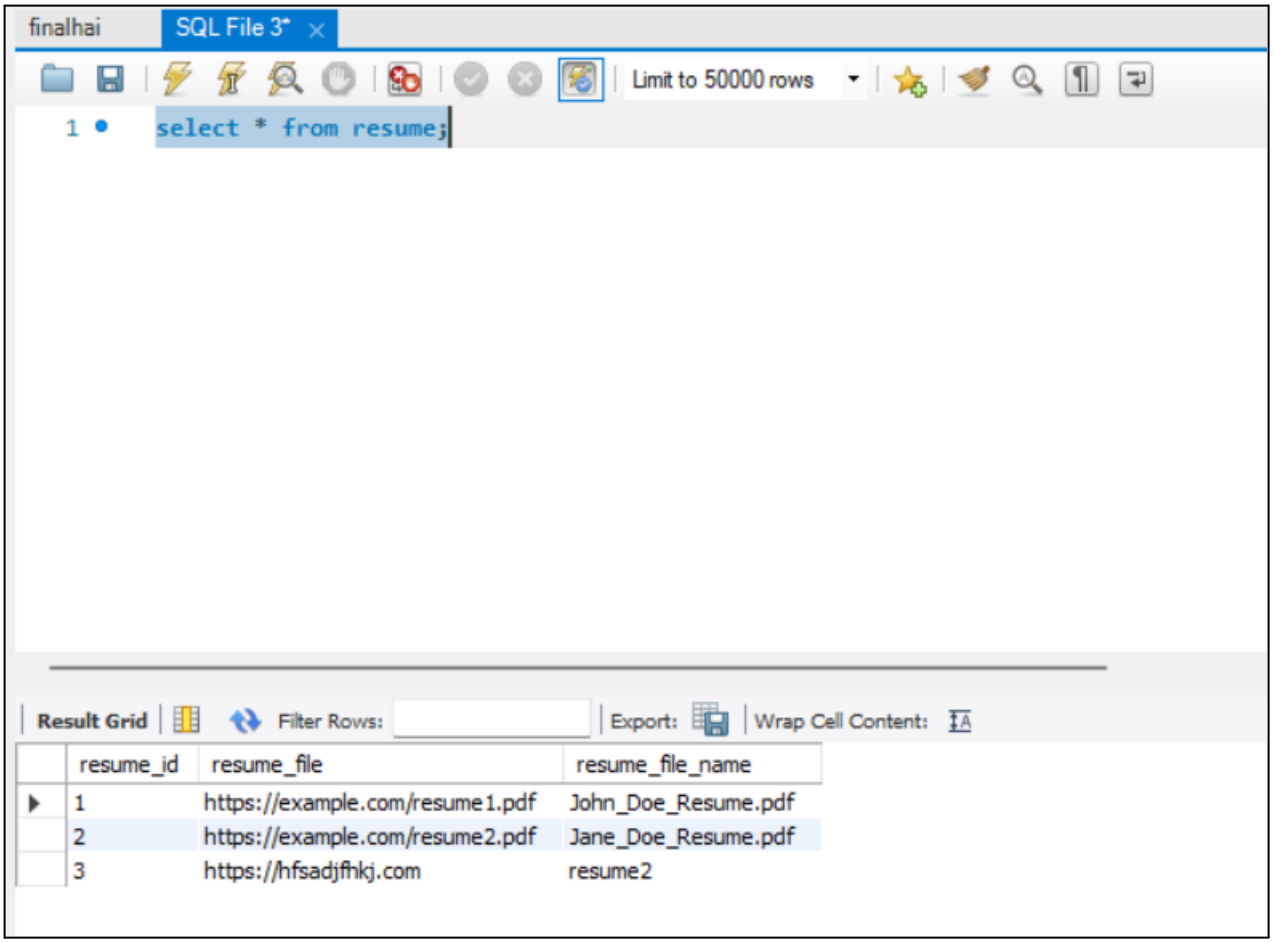
b. **Inserting**

Here, we are inserting a resume named ‘resume2’ on the page. A dialogue box appears which says that the resume is submitted successfully.



c. **After** insertion

Here, we have inserted ‘resume2’ and the changes can be seen in the SQL workbench.



Below, we can see on the page that ‘resume2’ has been uploaded and comes under existing resumes tab.

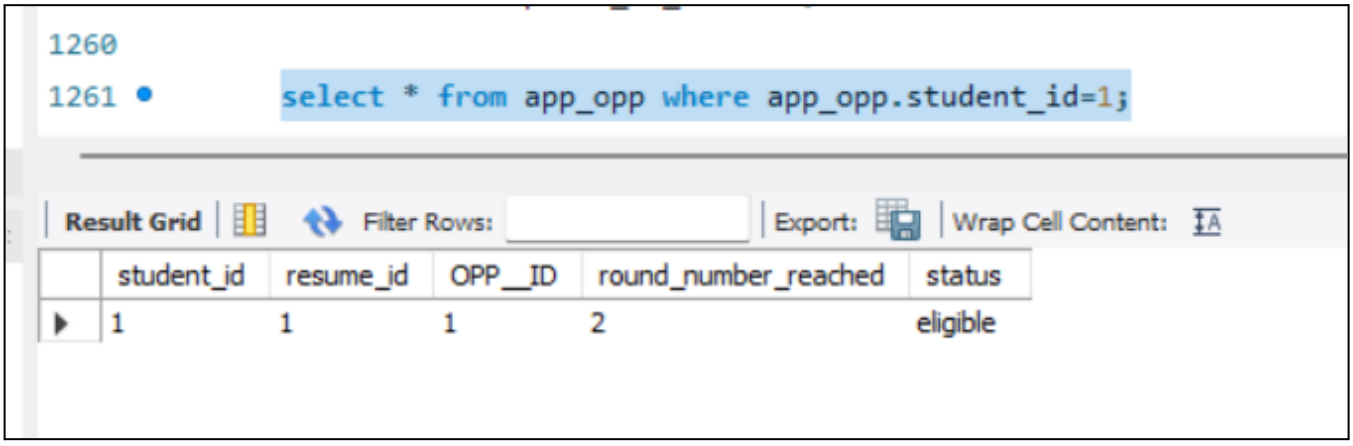
-

There are also other insertions we can do - add opportunity, add point of contact

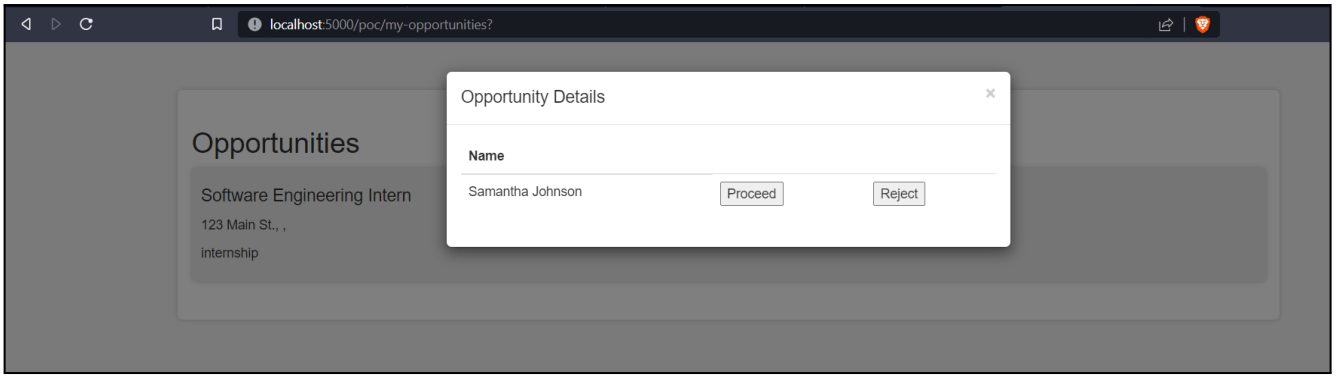
B. **UPDATE**

a. **Before** state

The current status is ‘eligible’ (in company poc view).

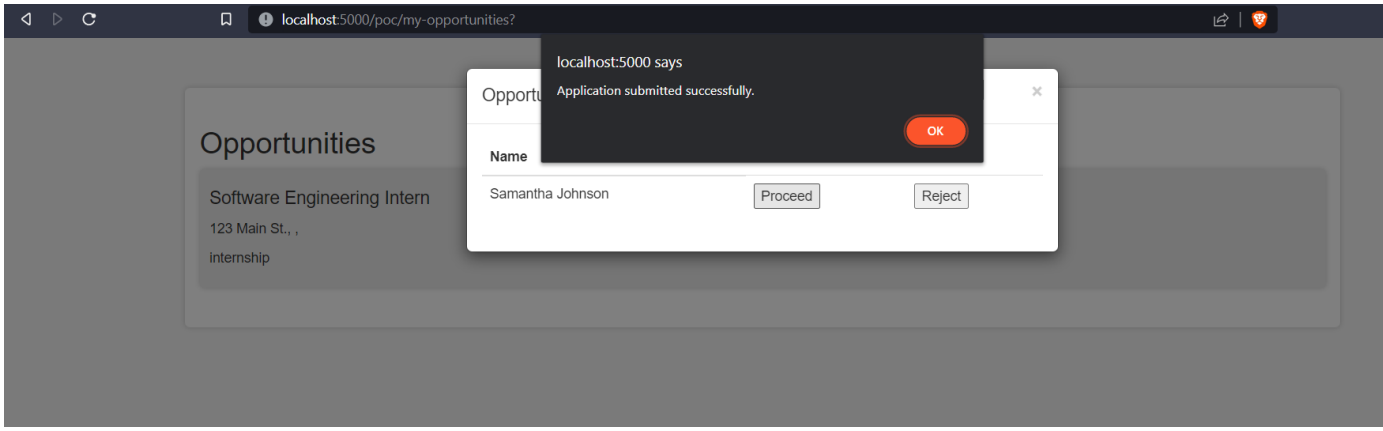


In company poc view, you can see options for the applicant.



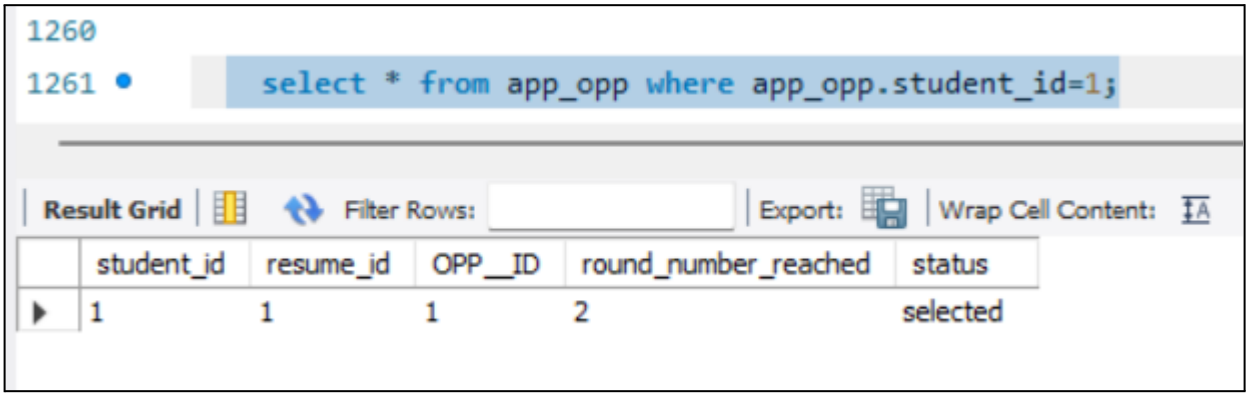
b. **Updating**

Clicking on ‘proceed’ (accepting student for next round).



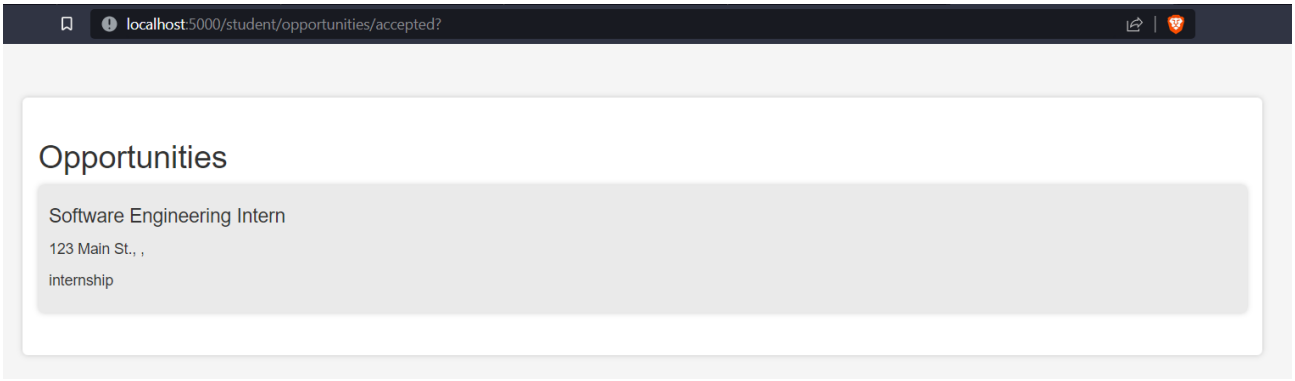
c. **After** updating

After selecting the student, the status updates to ‘selected’ and the round number updates to ‘2’.



	student_id	resume_id	OPP__ID	round_number_reached	status
▶	1	1	1	2	selected

This is the student view, in the URL you can see that this is the opportunity in which they are accepted.



C. **DELETE**

a. **Before** deleting

We will be deleting the last column of ‘Business Development Placement’ opportunity from the data.

finalhai | SQL File 3*

Limit to 50000 rows

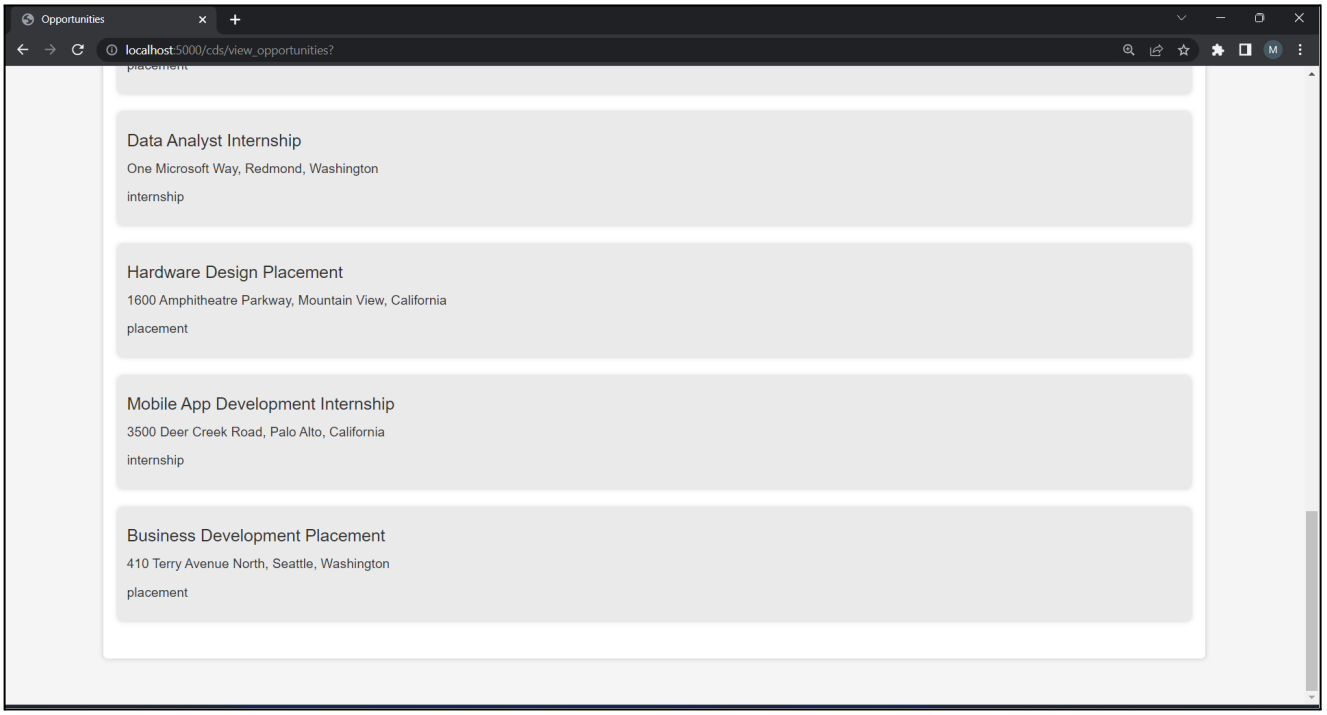
```
1 • select * from opportunity;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

opp_id	opp_type	opp_title	address_line_1	address_line_2	address_line_3	company_id
1	internship	Software Engineering Intern	123 Main St.			1
2	placement	Marketing Manager	456 Oak Ave.			2
3	internship	Data Science Intern	789 Elm St.			3
4	placement	Sales Representative	246 Maple Rd.			4
5	internship	Product Management Intern	135 Cedar Ln.			5
6	internship	Software Engineering Internship	1 Infinite Loop	Cupertino	California	6
7	placement	Hardware Engineer Placement	One Microsoft Way	Redmond	Washington	7
8	internship	Data Science Internship	1600 Amphitheatre Parkway	Mountain View	California	8
9	placement	Marketing Placement	3500 Deer Creek Road	Palo Alto	California	9
10	internship	Software Development Internship	410 Terry Avenue North	Seattle	Washington	1
11	placement	Product Manager Placement	1 Infinite Loop	Cupertino	California	2
12	internship	Data Analyst Internship	One Microsoft Way	Redmond	Washington	2
13	placement	Hardware Design Placement	1600 Amphitheatre Parkway	Mountain View	California	3
14	internship	Mobile App Development Internship	3500 Deer Creek Road	Palo Alto	California	4
15	placement	Business Development Placement	410 Terry Avenue North	Seattle	Washington	5

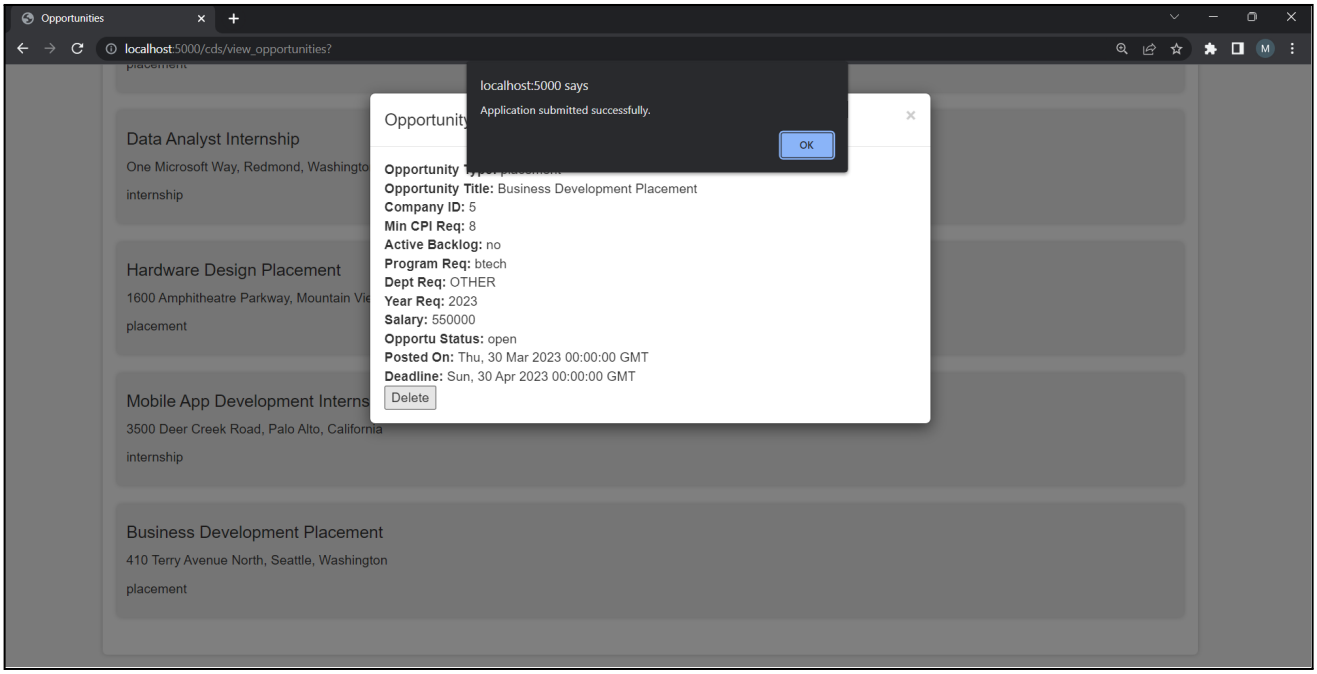
opportunity 4

The page looks something like this before deleting the opportunity from the data:



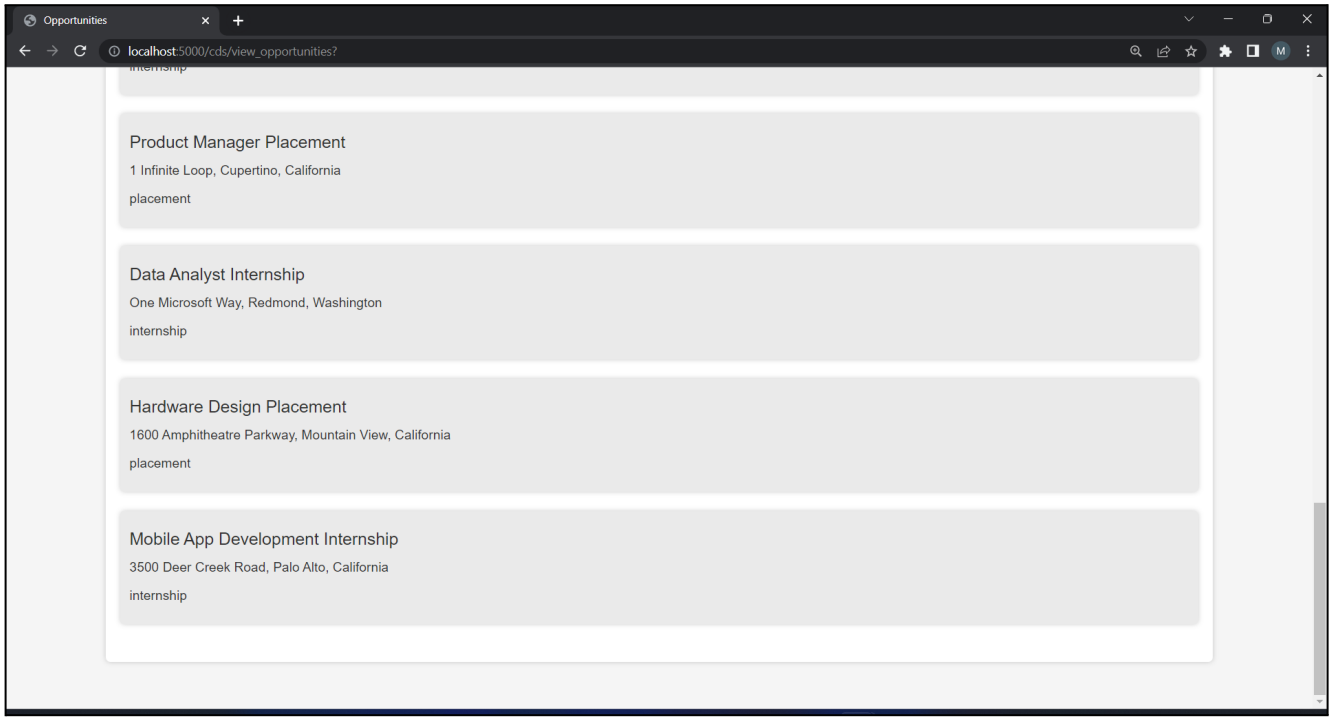
b. **Deleting**

Deleting the opportunity using ‘DELETE’ button.



c. **After** deletion

After deleting, the opportunity disappears from the ‘veiw opportunity’ page.



The opportunity also disappears from the data.

finalhai SQL File 3*

Limit to 50000 rows

```
1 • select * from opportunity;
```

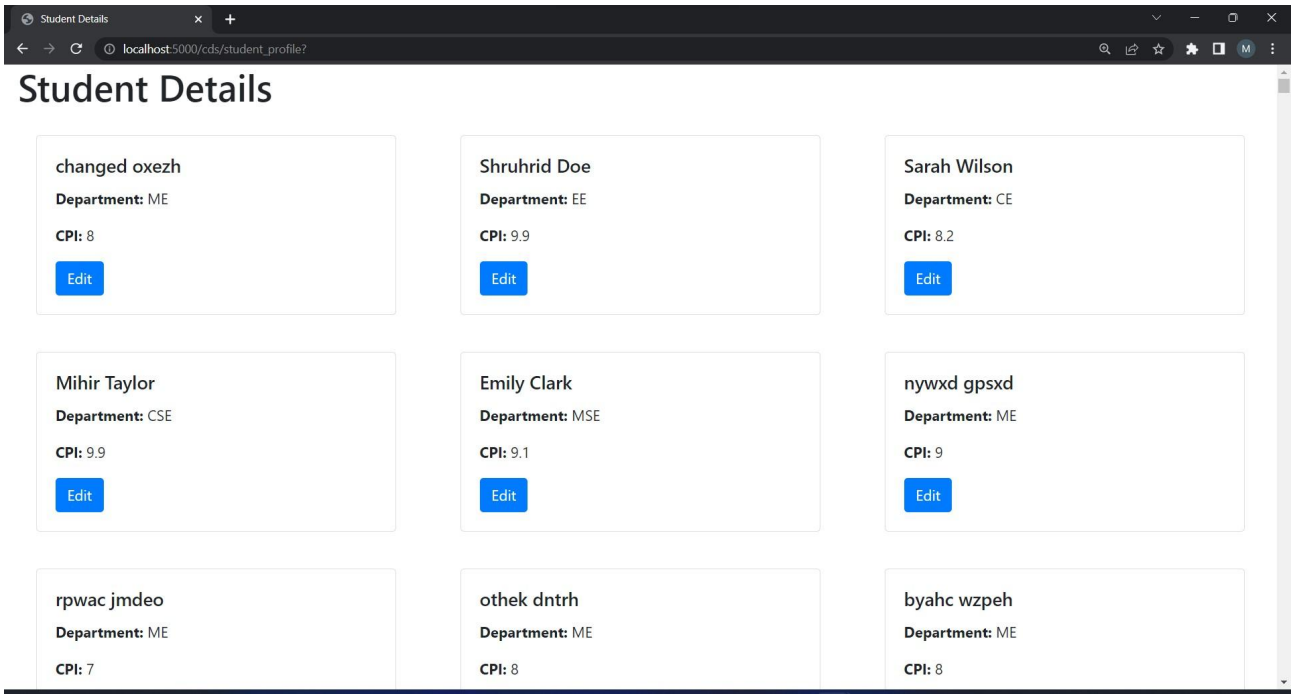
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

opp_id	opp_type	opp_title	address_line_1	address_line_2	address_line_3	company_id
1	internship	Software Engineering Intern	123 Main St.			1
2	placement	Marketing Manager	456 Oak Ave.			2
3	internship	Data Science Intern	789 Elm St.			3
4	placement	Sales Representative	246 Maple Rd.			4
5	internship	Product Management Intern	135 Cedar Ln.			5
6	internship	Software Engineering Internship	1 Infinite	1 Infinite Loop	Cupertino	California	6
7	placement	Hardware Engineer Placement	One Microsoft Way	Redmond	Washington	7
8	internship	Data Science Internship	1600 Amphitheatre Parkway	Mountain View	California	8
9	placement	Marketing Placement	3500 Deer Creek Road	Palo Alto	California	9
10	internship	Software Development Internship	410 Terry Avenue North	Seattle	Washington	1
11	placement	Product Manager Placement	1 Infinite Loop	Cupertino	California	2
12	internship	Data Analyst Internship	One Microsoft Way	Redmond	Washington	2
13	placement	Hardware Design Placement	1600 Amphitheatre Parkway	Mountain View	California	3
14	internship	Mobile App Development Internship	3500 Deer Creek Road	Palo Alto	California	4

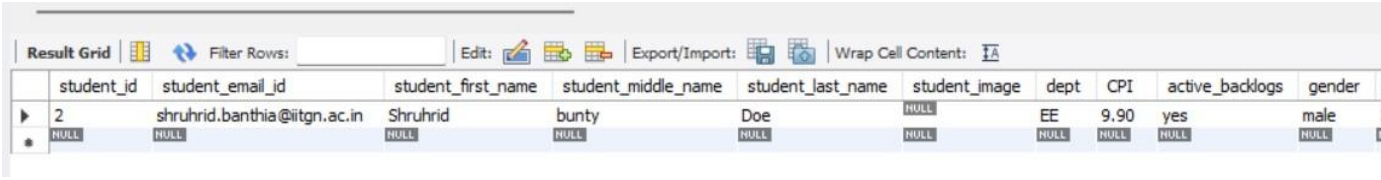
D. **RENAME**

a. **Before** renaming

CDS employee view, here we can rename student names and other details.



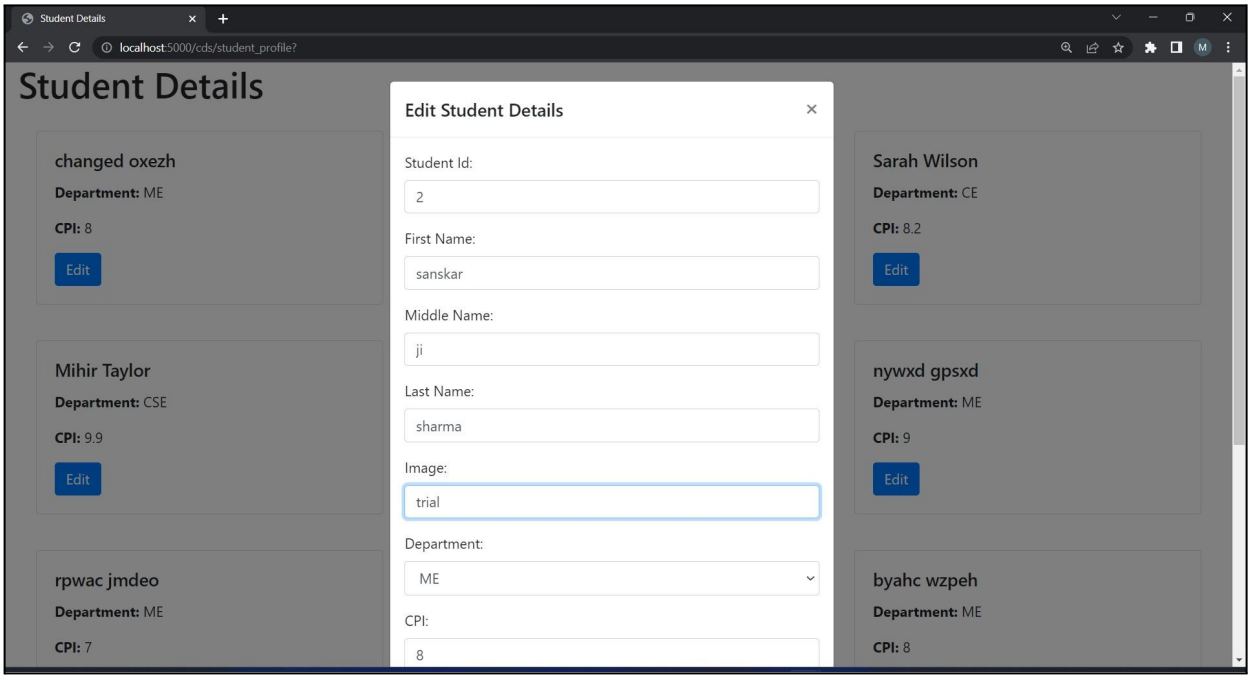
Data before renaming this student.



student_id	student_email_id	student_first_name	student_middle_name	student_last_name	student_image	dept	CPI	active_backlogs	gender
2	shruhrid.banthia@iitgn.ac.in	Shruhrid	bunty	Doe	NULL	EE	9.90	yes	male
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

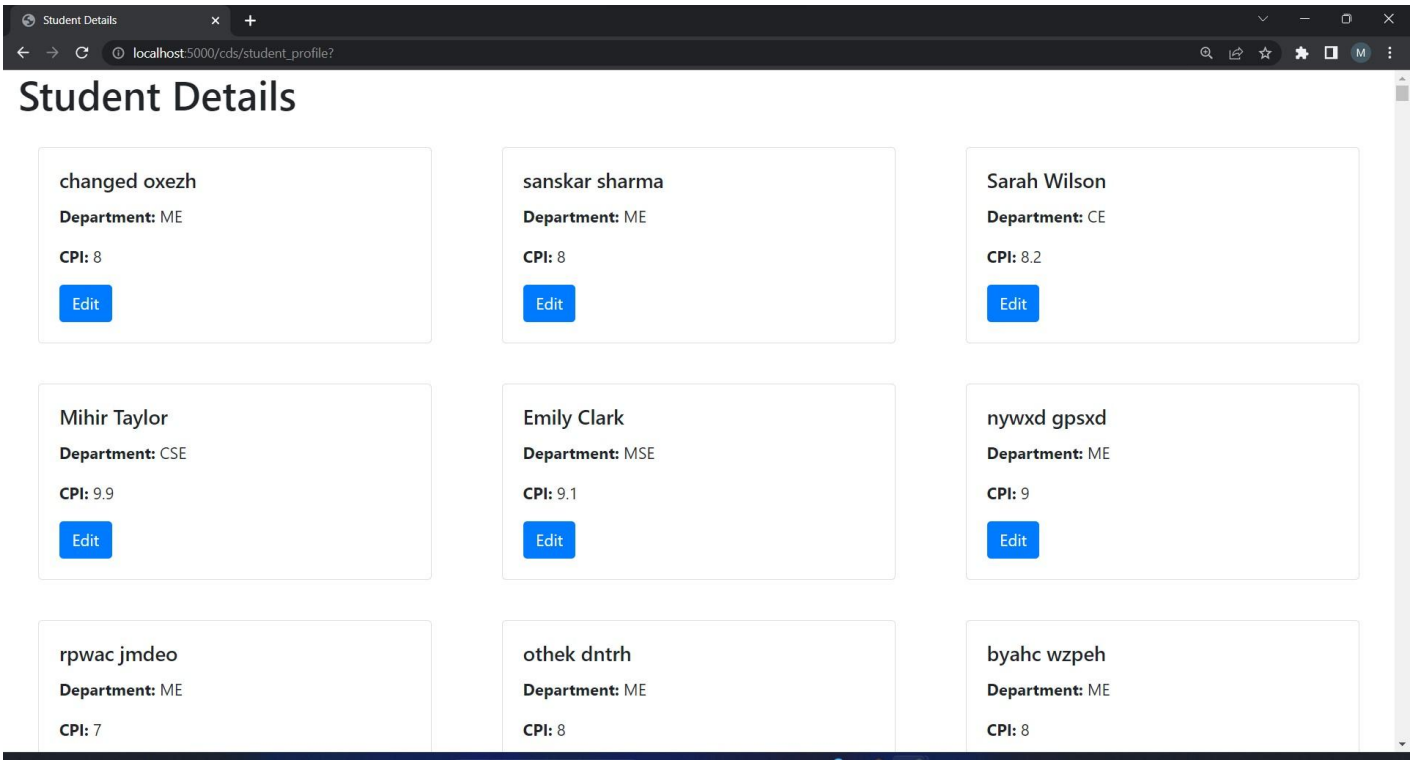
b. **Renaming**

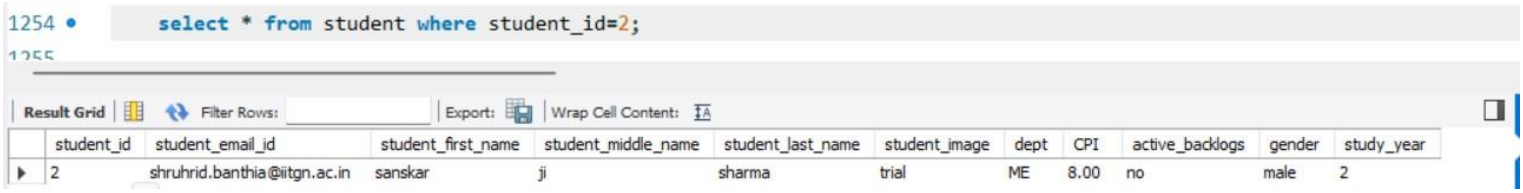
Process



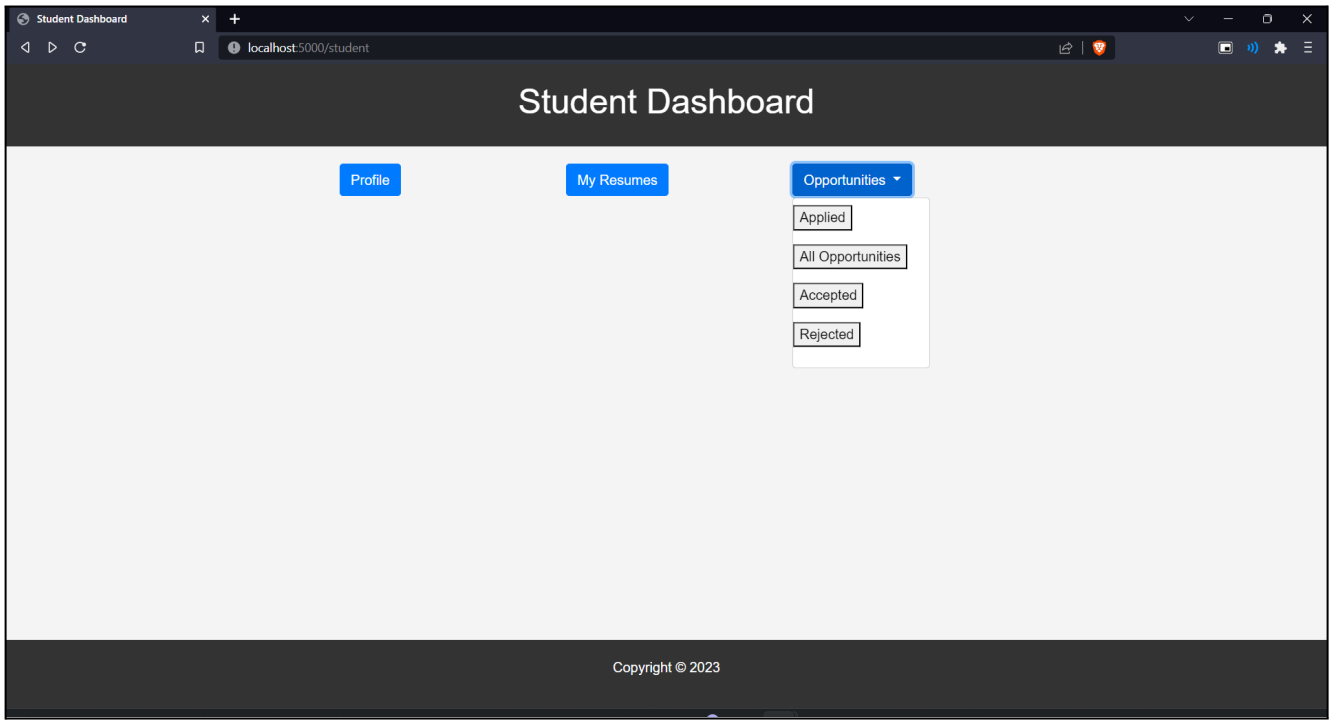
c. **After** state

Changed the named of the student .

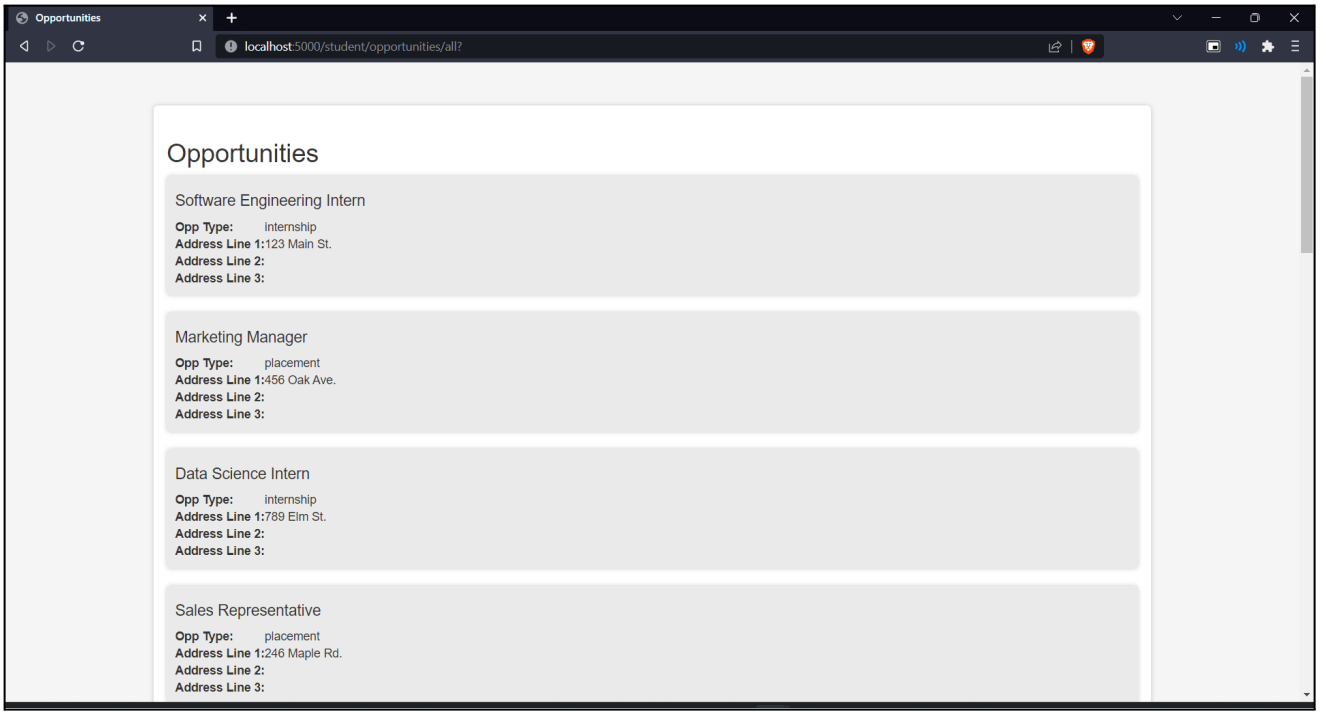




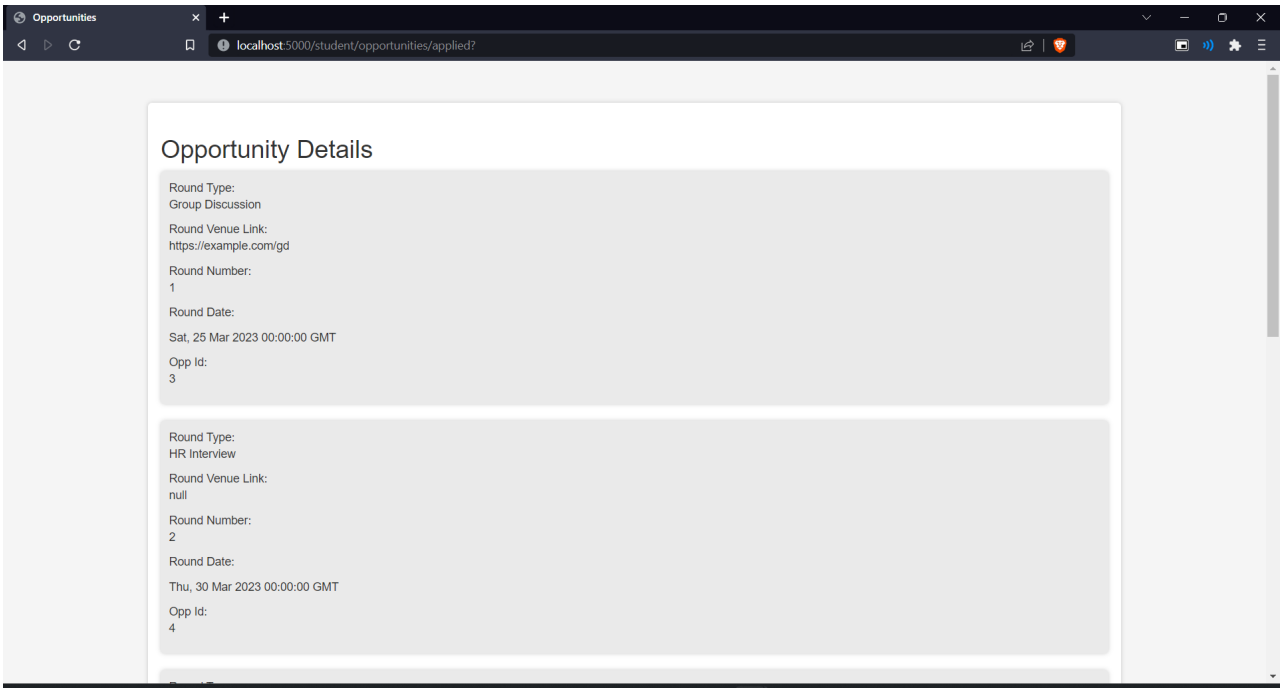
E. **WHERE** clause



On clicking ‘All Opportunities’ it shows all available opportunities.



The following are the applied opportunities.



On clicking ‘Applied’, it shows the ‘All Opportunities’ which we have applied to. Here we do not explicitly implement the where clause but the where clause is used at multiple instances in the backend for example:

```
query = "SELECT * FROM student WHERE student.student_id in (SELECT app_opp.student_id FROM app_opp WHERE app_opp.opp_id = %s);"
```

Here we can see the where clause being used in the code where we select the students with only the required id.

Components and views of whole Website

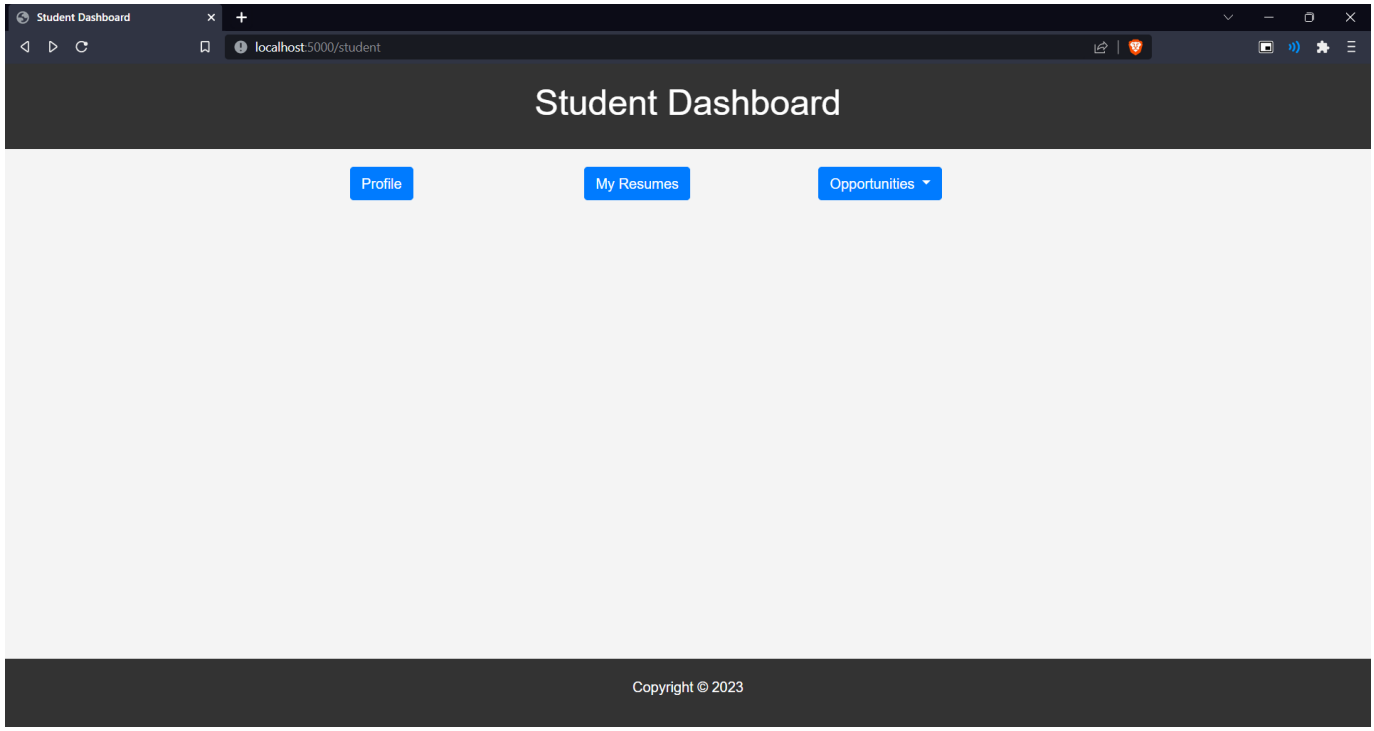
There is total of three views:

1. Student
    - Profilemo
    - My Resume
    - Opportunities
        - Applied
        - All Opportunities
        - Accepted
        - Rejected

2. CDS Employee (Admin)
3. Company Point of Contact

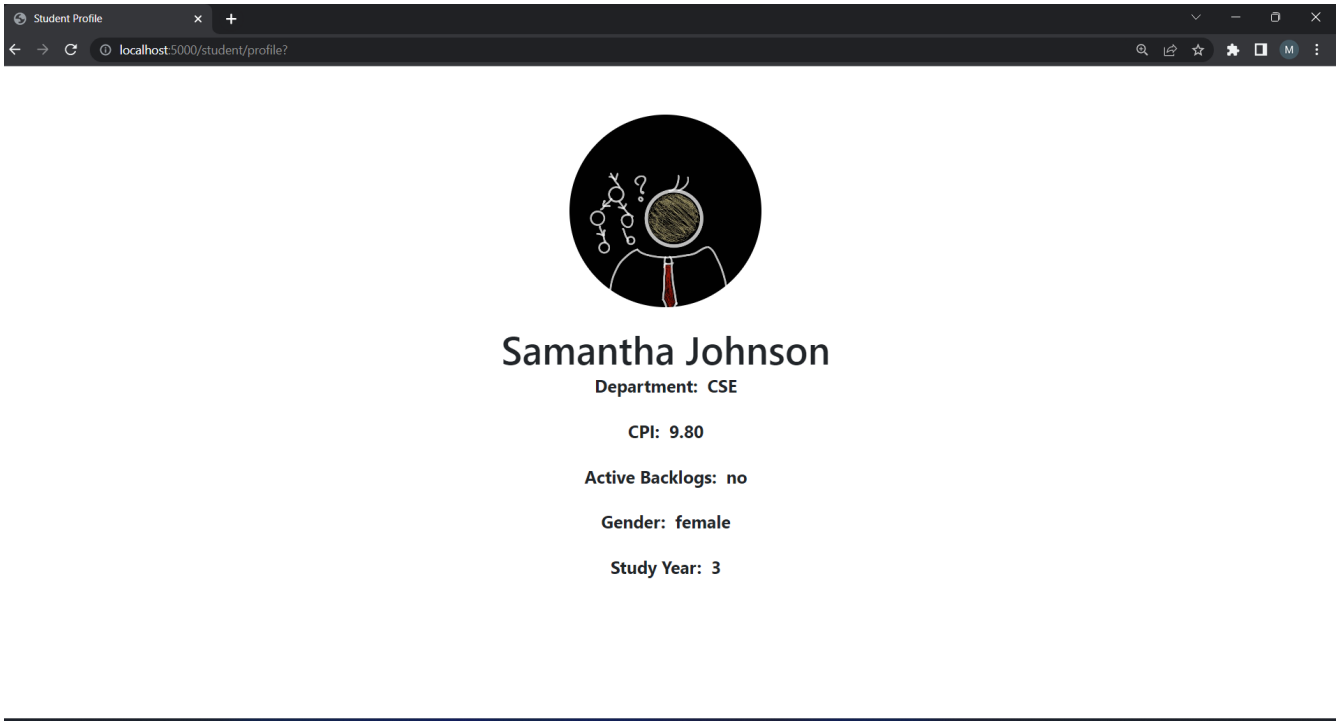
Student's View

- Dashboard for student



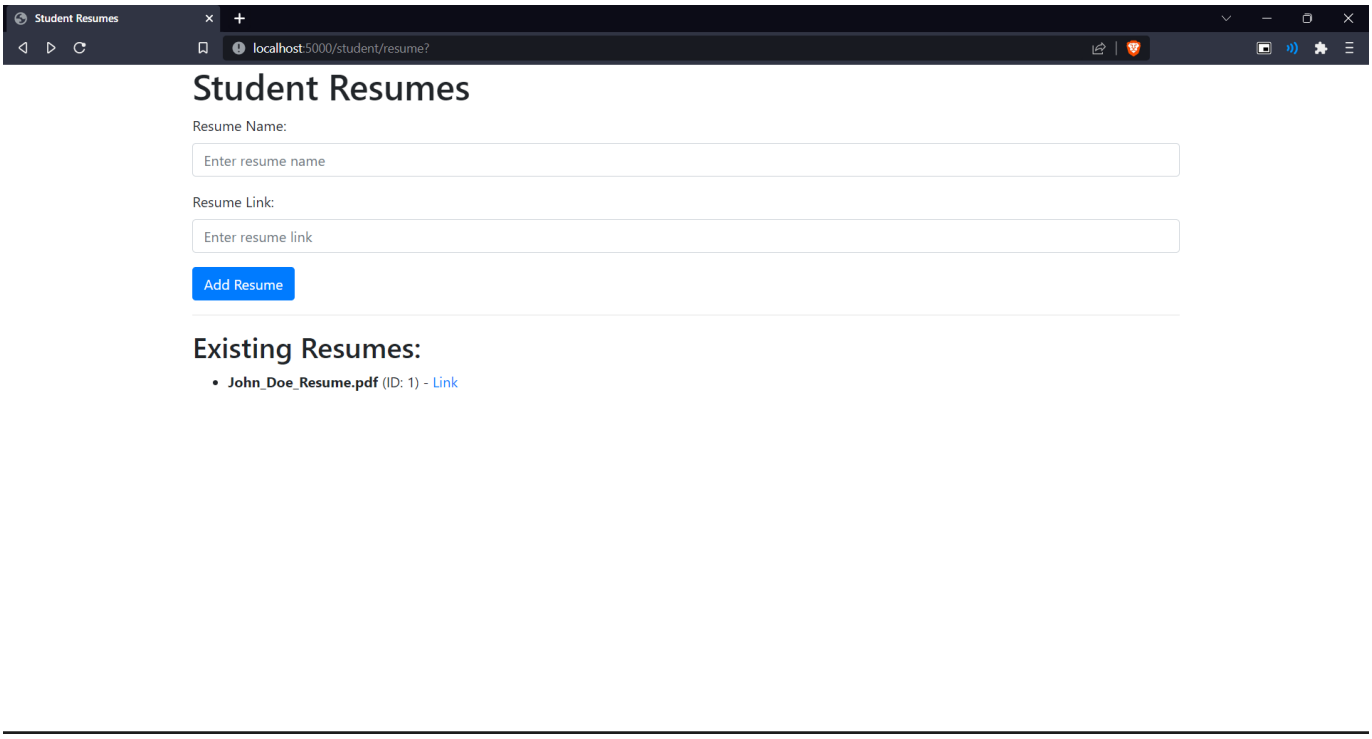
student/profile

- View of the student profile page

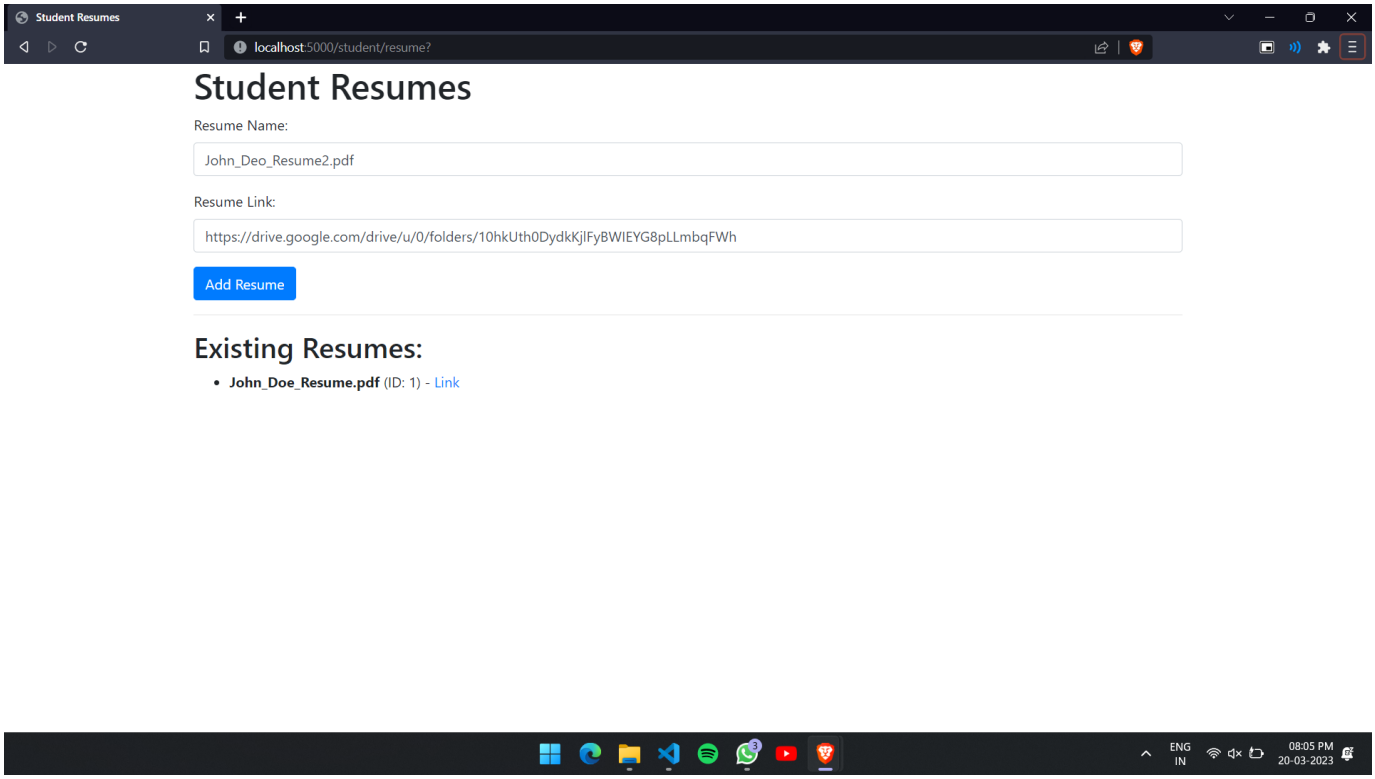


/student/resume?

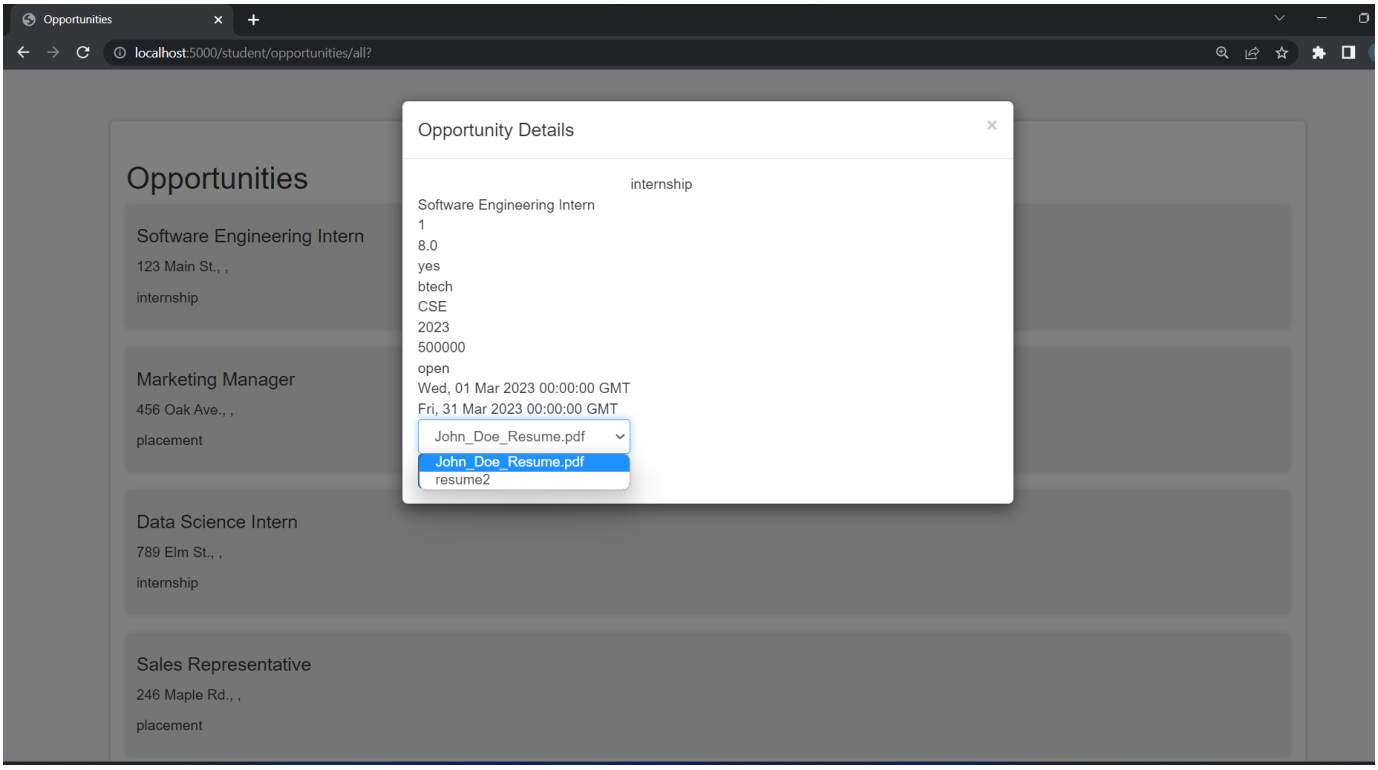
Here student can enter their resume and it will be saved in database with the name given by the student.



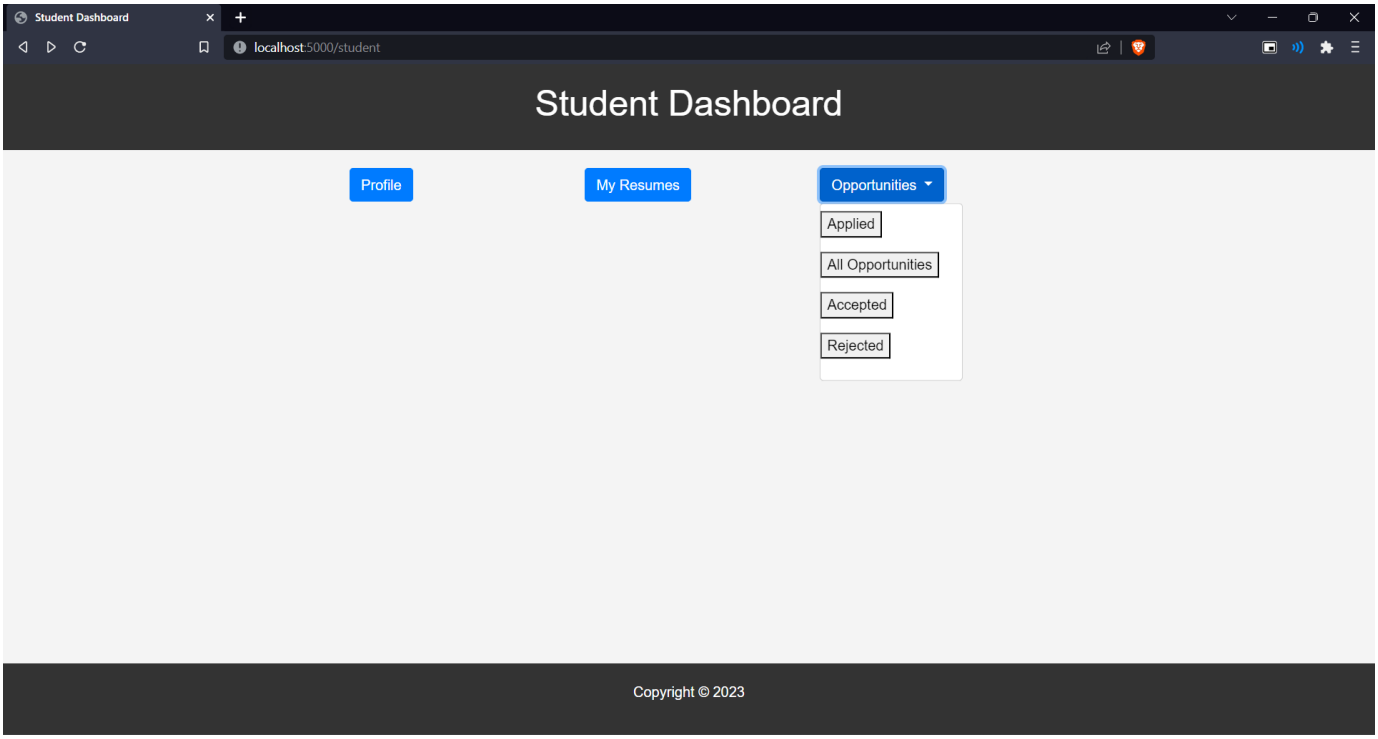
Adding new resume:



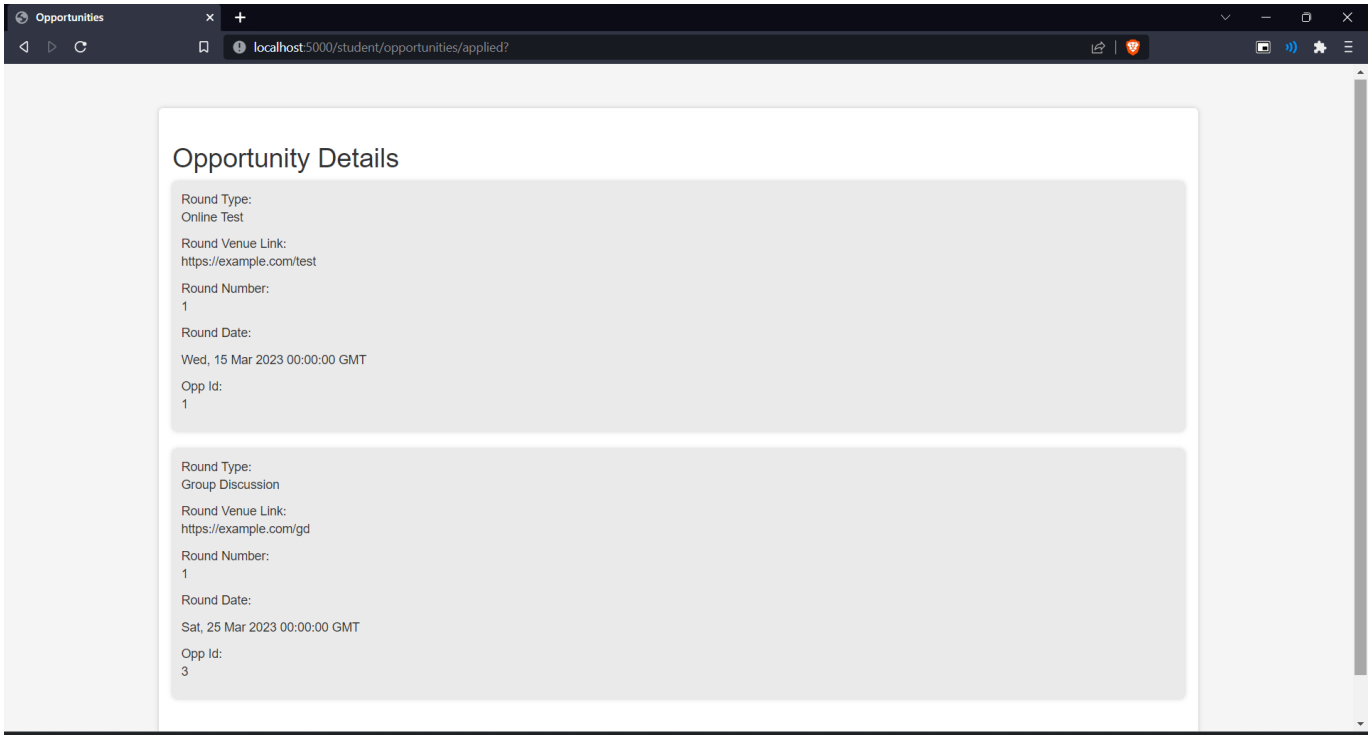
After inserting one resume:



There are four options in opportunities: Applied, All Opportunities, Accepted, Rejected

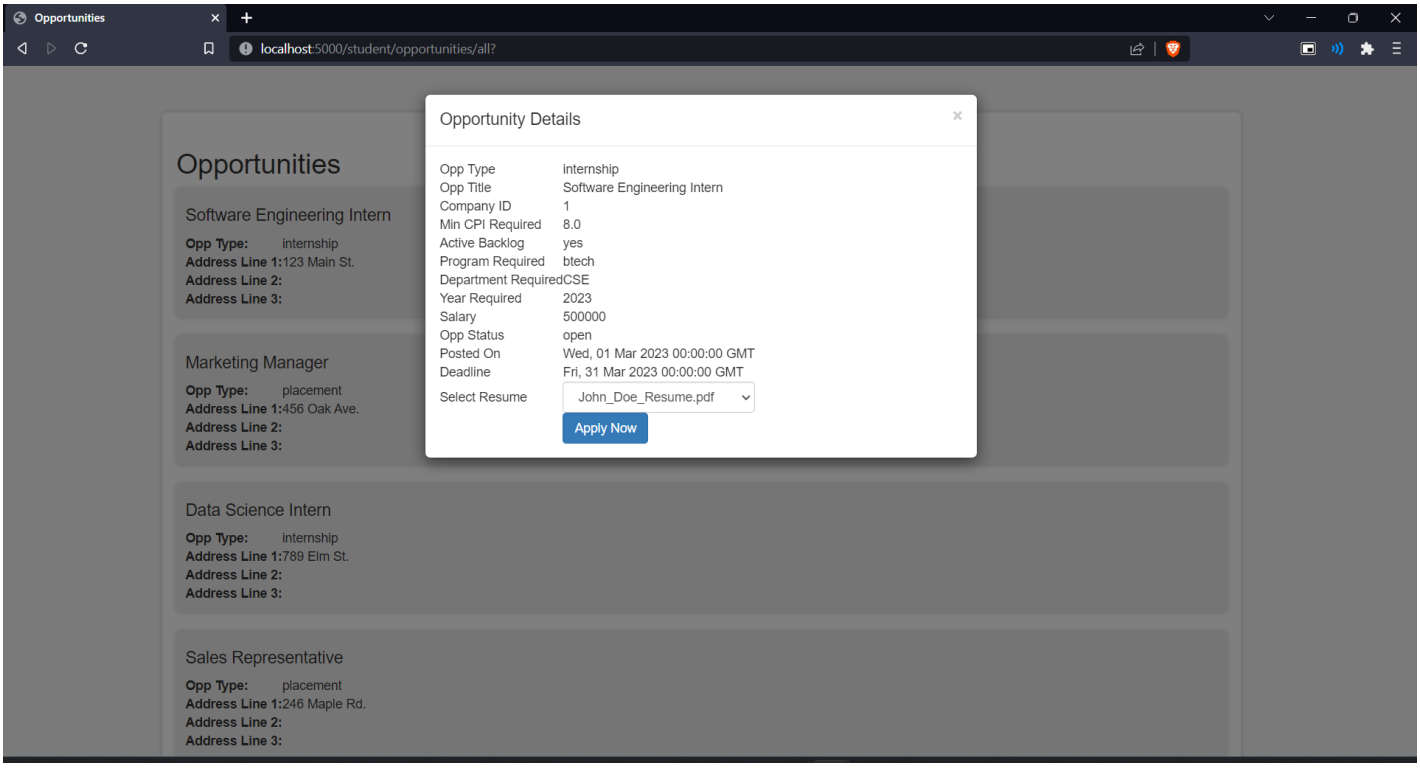


student/opportunities/applied
Here student can see all the opportunities on which student has applied.



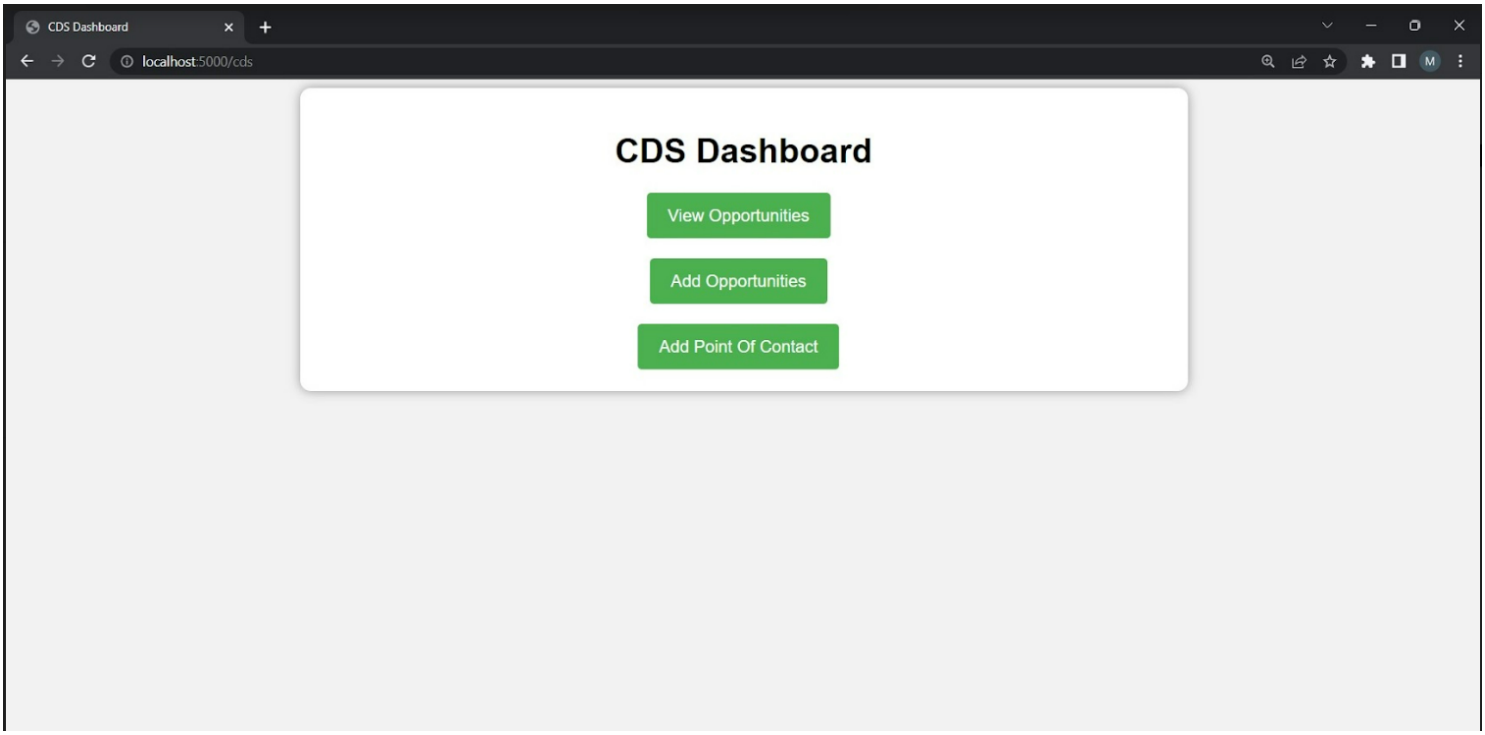
Here students can see all the opportunities on which they can apply.

Student can see the details of the opportunities available by clicking on it. It has the option of selecting the one of the uploaded resume and applying.

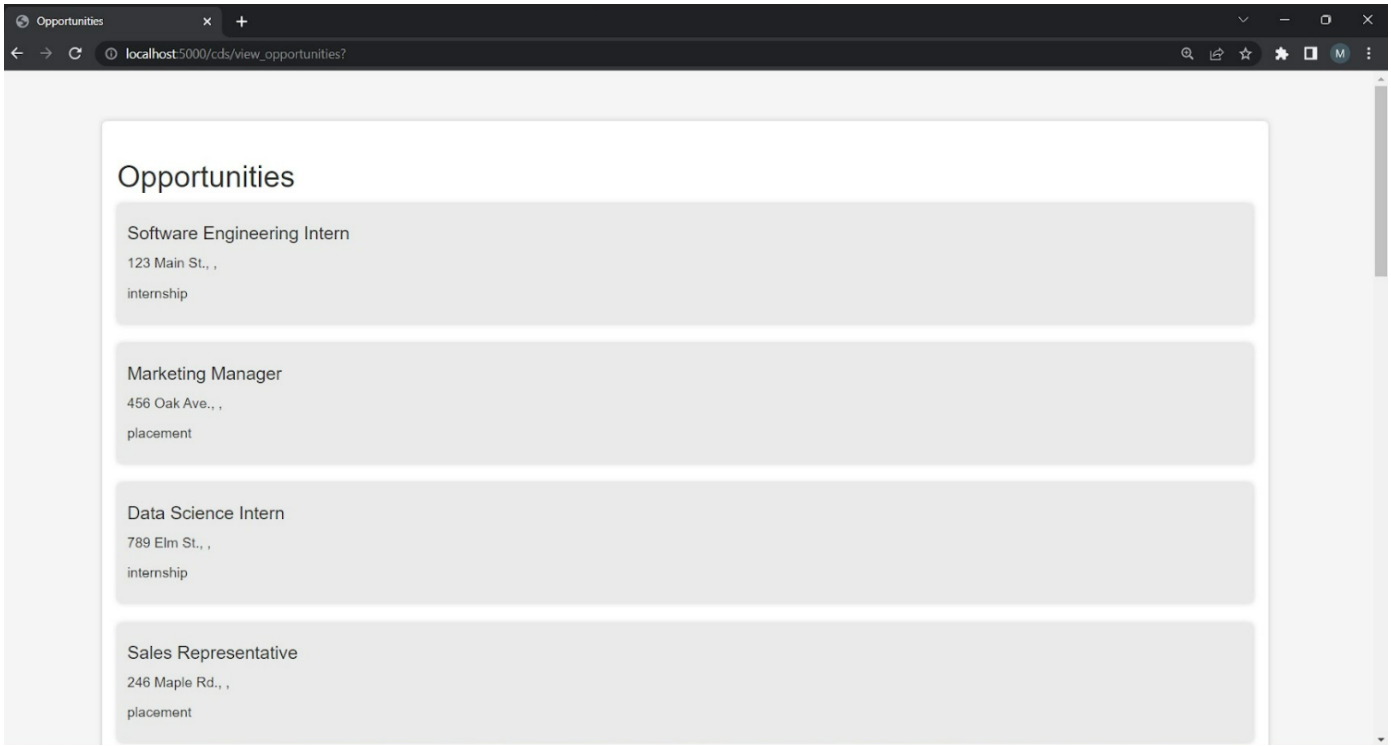


CDS employee view:

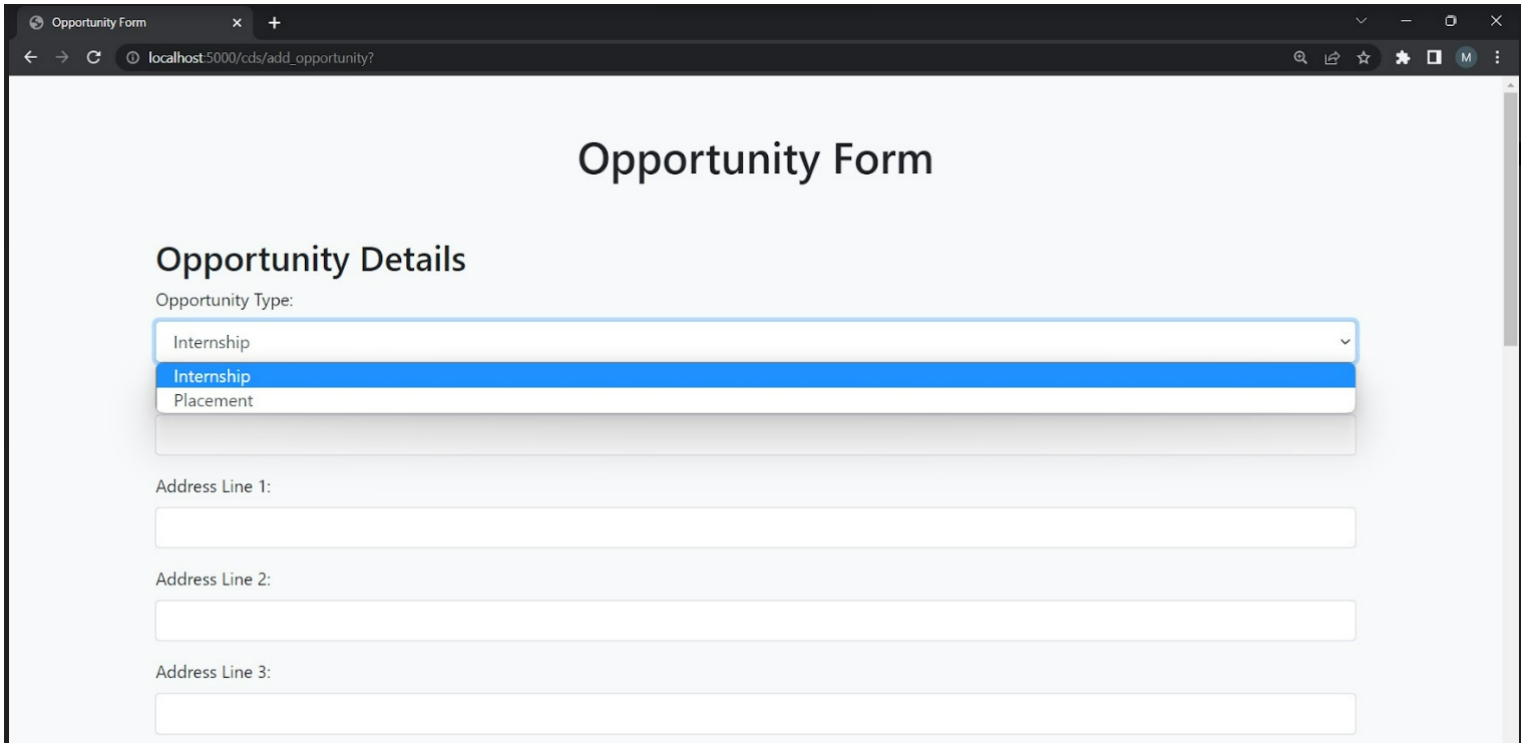
As soon as the employee of the CDS sign's in,, it opens up the below dashboard.



Upon clicking on view opportunities, the following page opens up, where we list down all opportunities available.



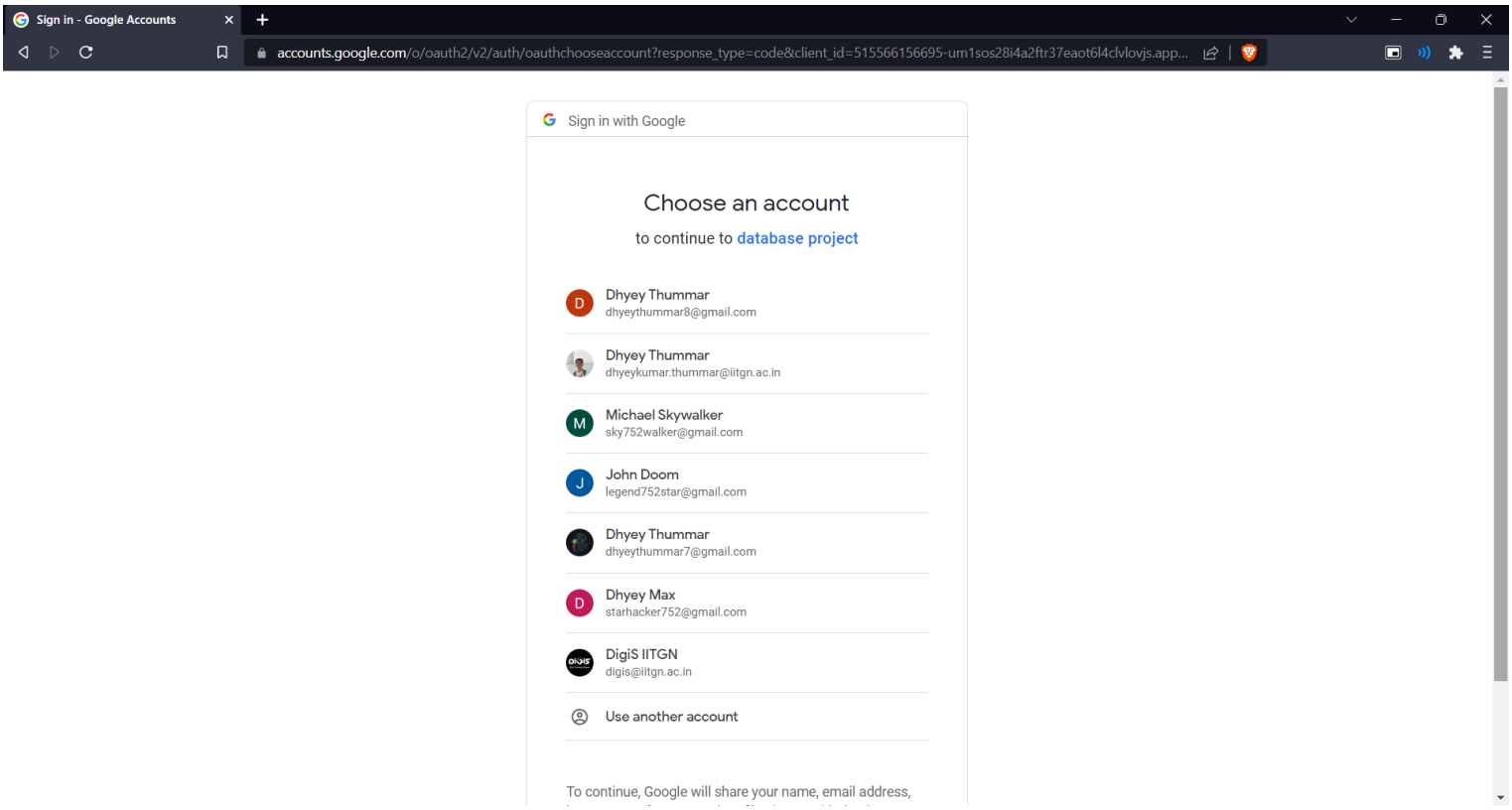
Upon clicking on add opportunities, the following form opens up where the cds employee can add an opportunity by filling the required details.



Upon clicking on add point of contact, the following form opens up such that the CDS employee can add POC by filling the below requirements.

Google Authentication

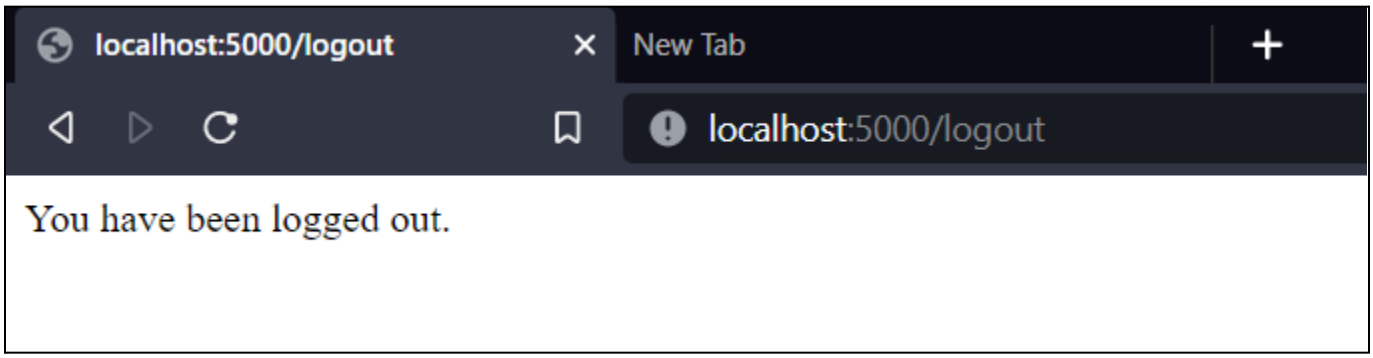
Any person can login only if their email id is available in the database.



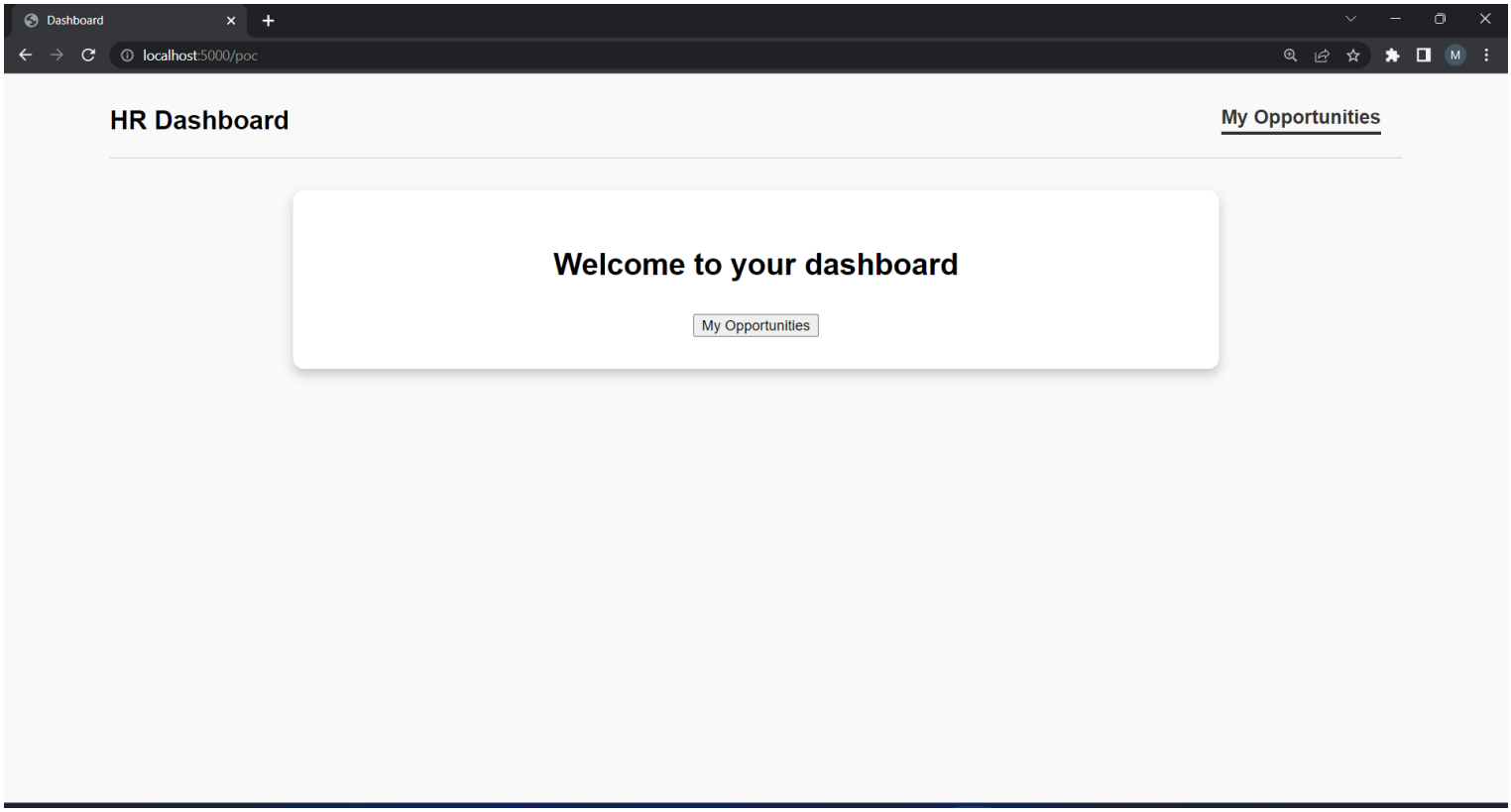
Profile
-
My Resumes

Opportunities

Logging out:



HR view:



Contributions:

G1:

Pushpendra Pratap Singh (20110151)

- Worked on the front end of CDS pages and POC pages.
- Worked on the integration of front end and back end. Worked on implementing successful requests in the frontend.
- Fixed the bugs in frontend..

Saatvik Rao (20110175)

- Wrote the code for front end of student pages
- Wrote the steps to run the Web App
- Added and explained screenshots of successful execution

Sahil Agrawal (20110178)

- Wrote the code for the front end of student pages.
- Wrote the code for the front end of cds pages.

Sanskar Sharma (20110185)

- Designed the frontend views
- Designed the HTML Pages and wrote the Javascript, CSS and JQuery code for most of the pages
- Integrated Frontend and Backend with Team
- Described all the GET and POST requests that are needed for the application to operate

Tanvi Dixit (20110212)

- Worked on the frontend of the web app
- Created and assisted HTML codes of web pages for all views of the web application
- Checked for any inconsistencies in the cardinalities while running of the application
- Writing and formatting the report

G2:

Dhyeykumar Thummar (20110059)

- Handled POST requests (for student, CDS employee and poc) in app.py.
- Wrote all the required SQL queries for the post requests.
- Helped in debugging the code.

Ksheer Sagar Agrawal (20110098)

- Started the repository & templates for the front & backend.
- Wrote code for backend, wrote different sql queries.
- Checked flask, mysql connection & all necessary pip installs.

Mihir Sutariya (20110208)

- Wrote code for backend, wrote different sql queries. Wrote code for google authentication.
- Gave all the user’s unique occupation.
- Fixed bugs and review code pushed on github.

R Yeeshu Dhurandhar (20110152)

- Contributed for get requests and SQL commands for company POC and CDS Employee required for different requests
- Wrote the setup requirements and steps to run the Web App.
- Contributed to the documentation

Shruhrid Banthia (20110198)

- Wrote code for backend, wrote different sql queries.
- Integrated the frontend and backend components. Fixed bugs and review code pushed on github
- Contributed for get requests and SQL commands for student required for different requests

Utkarsh Mishra (20110218)

- Populated the SQL database with required data following the constraints mentioned in the previous assignment
- Helped in documentation