

# Indian Institute of Technology, Gandhinagar



---

## Database for IITGN Maintenance

---

Authors:

Pulkit Gautam 21110169  
Manav Parmar 21110120  
Shubh Agarwal 21110205  
Gaurav Rawat 21110066  
Hrishikesh Birje 21110044  
Aman Singh 21110020  
Ishva Patel 21110082  
Atal Gupta 21110037

WebApp for Database

Databases

CS 432

*Under the guidance of*  
Prof. Mayank Singh

April 4, 2024

# Responsibility of G1

The front-end of the Web Application is meticulously developed using HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets), supplemented by Jinja2 templating for enhanced and easier formatting, and further bolstered by the inclusion of Bootstrap as a framework.

## Frameworks Used

- **HTML**

HTML, or Hypertext Markup Language, is the fundamental building block of the web, enabling the creation and structuring of web pages. Through a system of tags and elements, HTML allows for the insertion of text, images, links, and other content, forming the backbone of websites and web applications worldwide.

- **CSS**

CSS, or Cascading Style Sheets, is a stylesheet language used to style the appearance of web content. It allows for the customization of colors, fonts, layouts, and more, enabling developers and designers to create visually appealing and consistent user interfaces across different web pages, enhancing the user experience on the internet.

- **Bootstrap**

Bootstrap is a powerful, open-source front-end framework designed for responsive, mobile-first web development. It provides a vast array of pre-designed components, utilities, and plugins, making it easier to create aesthetically pleasing, consistent, and functional web pages and applications. Bootstrap facilitates rapid development by offering customizable templates for typography, forms, buttons, navigation, and other interface components, ensuring seamless adaptation to different screen sizes and devices. It has been implemented for designing our website to be more accessible and user-friendly.

- **Jinja2**

Jinja2 allows developers to write simpler, more readable code by inheriting HTML file structures, enabling the insertion of Python-like expressions directly into HTML files. Jinja2 is widely used for web development because it automates the generation of web pages from templates, making it easier to manage dynamic content.

## Login Page

A login page is implemented for quick and secure access to the database.

### Login Page

Please log in to access this page.

Email

Password

Figure 1: Login Page

## SignUp Page

A SignUp page is implemented for the new users to sign-up before they can access the WebApp.

## Sign Up Page

Name

Email

Password must be at least 8 characters long and contain at least one uppercase and one lowercase character.

Password

Figure 2: Signup Page

## Home Page

The preceding section showcases the website's homepage, serving as the central hub where users can undertake various actions.

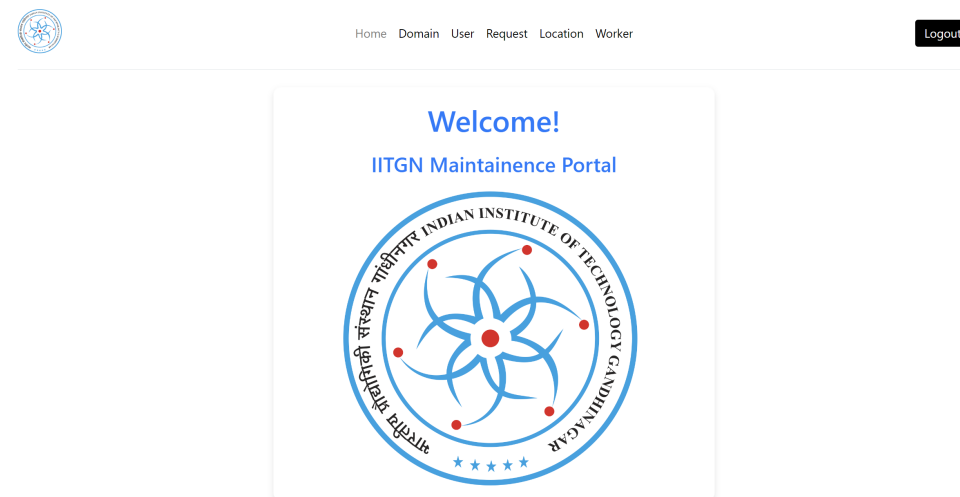



Figure 3: Home Page

## Request Page

This page displays the current added requests as well as provides an option for registering another request.



[Home](#)
[Domain](#)
[User](#)
[Request](#)
[Location](#)
[Worker](#)

Logout

## Request Table

Add Requests







Request ID	User ID	Domain ID	Location ID	Subject	Availability	Status	Description	Admin Comments	Image	Action
2	2	2	34	Fix Leaky Faucet, Please	Available on weekends and evenings	Pending	Leaky faucet in the kitchen needs repair	None		<a href="#">Update</a> <a href="#">Delete</a>
3	3	3	56	HVAC System Maintenance	Available on weekdays from 8am to 4pm	Pending	Require routine maintenance for HVAC system	None		<a href="#">Update</a> <a href="#">Delete</a>
4	4	4	78	Carpentry Work for Custom Furniture	Available on weekdays from 10am to 6pm	Pending	Need custom furniture built for the office	None		<a href="#">Update</a> <a href="#">Delete</a>
5	5	5	23	Interior Painting for Living Room	Available on weekends and evenings	Pending	Want to repaint the living room walls	None		<a href="#">Update</a> <a href="#">Delete</a>
6	6	6	45	Roof Repair Service Needed	Available on weekdays from 9am to 5pm	Pending	Roof has a leak and needs repair	None		<a href="#">Update</a> <a href="#">Delete</a>

Figure 4: Request Page

## Workers Page

This page displays the currently registered workers alongside the option to register another worker.



[Home](#)
[Domain](#)
[User](#)
[Request](#)
[Location](#)
[Worker](#)

Logout

## Workers List

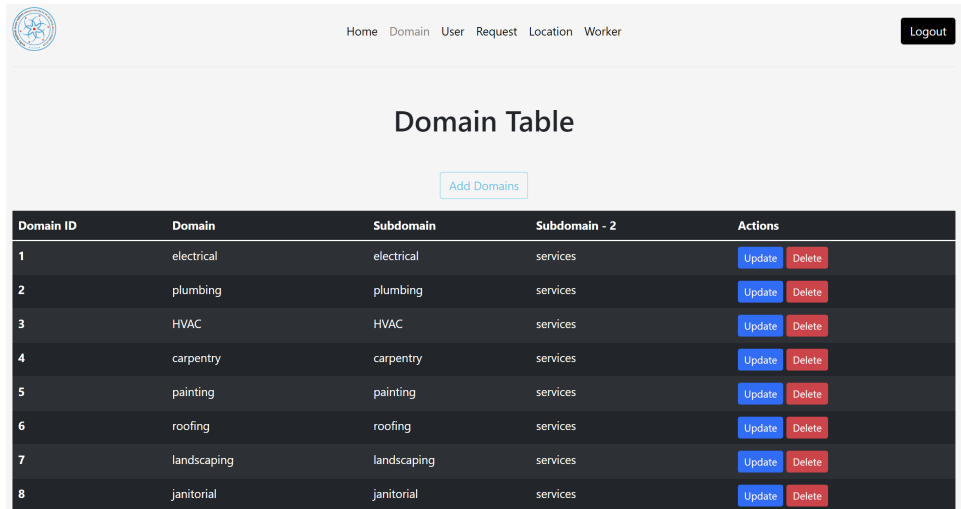
Add Workers

Worker ID	First Name	Middle Name	Last Name	Domain ID	Availability	Phone Number	Actions
2	Emma	Grace	John	2	Available on weekends and evenings	2345678901	<a href="#">Update</a> <a href="#">Delete</a>
3	Michael	David	Brown	3	Available on weekdays from 8am to 4pm	3456789012	<a href="#">Update</a> <a href="#">Delete</a>
4	Sophia	Elizabeth	Martinez	4	Available on weekdays from 10am to 6pm	4567890123	<a href="#">Update</a> <a href="#">Delete</a>
5	William	Christopher	Anderson	5	Available on weekends and evenings	5678901234	<a href="#">Update</a> <a href="#">Delete</a>
6	Isabella	Lauren	Taylor	6	Available on weekdays from 9am to 5pm	6789012345	<a href="#">Update</a> <a href="#">Delete</a>
7	James	Ryan	Thomas	7	Available on weekdays from 8am to 4pm	7890183456	<a href="#">Update</a> <a href="#">Delete</a>
8	Olivia	Avery	Lee	8	Available on weekends and evenings	8902234567	<a href="#">Update</a> <a href="#">Delete</a>
9	Benjamin	Kevin	White	9	Available on weekdays from 9am to 5pm	9012345678	<a href="#">Update</a> <a href="#">Delete</a>

Figure 5: Workers Page

## Domain Page

This page mentions all the domains and allows to enter another domain if needed.



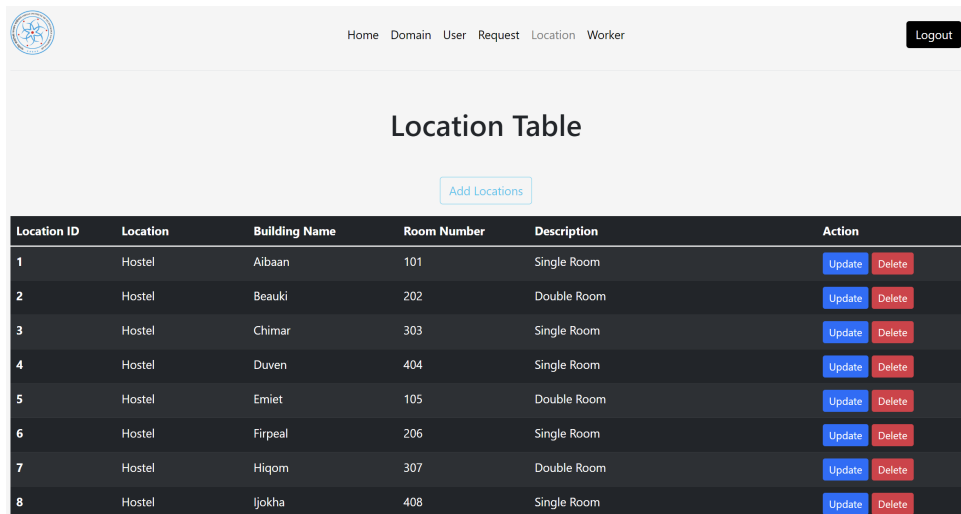
The screenshot shows the 'Domain Page' of a web application. At the top, there is a navigation bar with links: Home, Domain, User, Request, Location, Worker, and a Logout button. Below the navigation bar, the title 'Domain Table' is centered. Underneath the title is a button labeled 'Add Domains'. The main content is a table with the following structure:

Domain ID	Domain	Subdomain	Subdomain - 2	Actions
1	electrical	electrical	services	<a href="#">Update</a> <a href="#">Delete</a>
2	plumbing	plumbing	services	<a href="#">Update</a> <a href="#">Delete</a>
3	HVAC	HVAC	services	<a href="#">Update</a> <a href="#">Delete</a>
4	carpentry	carpentry	services	<a href="#">Update</a> <a href="#">Delete</a>
5	painting	painting	services	<a href="#">Update</a> <a href="#">Delete</a>
6	roofing	roofing	services	<a href="#">Update</a> <a href="#">Delete</a>
7	landscaping	landscaping	services	<a href="#">Update</a> <a href="#">Delete</a>
8	janitorial	janitorial	services	<a href="#">Update</a> <a href="#">Delete</a>

Figure 6: Domain Page

## Location Page

This page mentions all the currently registered locations and provides option to enter more.



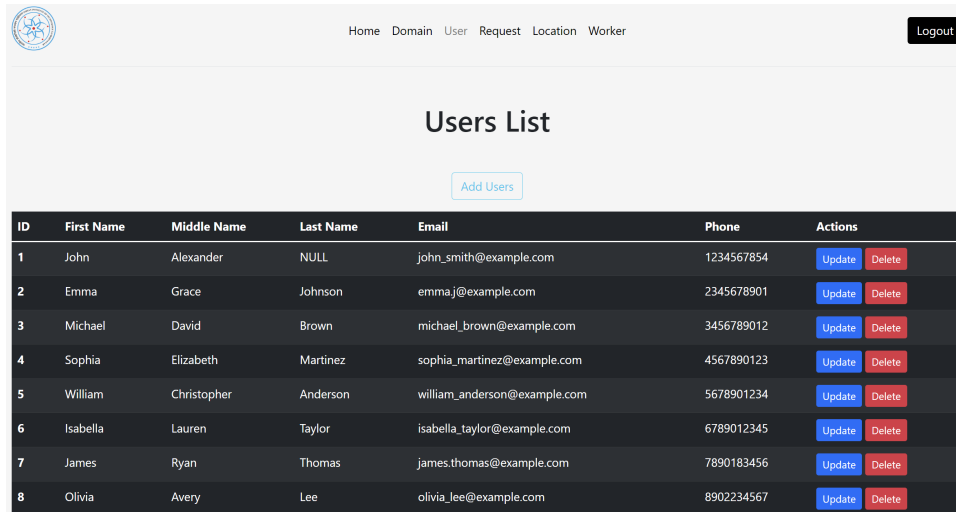
The screenshot shows the 'Location Page' of a web application. At the top, there is a navigation bar with links: Home, Domain, User, Request, Location, Worker, and a Logout button. Below the navigation bar, the title 'Location Table' is centered. Underneath the title is a button labeled 'Add Locations'. The main content is a table with the following structure:

Location ID	Location	Building Name	Room Number	Description	Action
1	Hostel	Aibaan	101	Single Room	<a href="#">Update</a> <a href="#">Delete</a>
2	Hostel	Beauki	202	Double Room	<a href="#">Update</a> <a href="#">Delete</a>
3	Hostel	Chimar	303	Single Room	<a href="#">Update</a> <a href="#">Delete</a>
4	Hostel	Duven	404	Single Room	<a href="#">Update</a> <a href="#">Delete</a>
5	Hostel	Emiet	105	Double Room	<a href="#">Update</a> <a href="#">Delete</a>
6	Hostel	Firpeal	206	Single Room	<a href="#">Update</a> <a href="#">Delete</a>
7	Hostel	Hiqom	307	Double Room	<a href="#">Update</a> <a href="#">Delete</a>
8	Hostel	Ijokha	408	Single Room	<a href="#">Update</a> <a href="#">Delete</a>

Figure 7: Location Page

## User Page

This page mentions all the registered users and option to add more.



ID	First Name	Middle Name	Last Name	Email	Phone	Actions
1	John	Alexander	NULL	john_smith@example.com	1234567854	<a href="#">Update</a> <a href="#">Delete</a>
2	Emma	Grace	Johnson	emma.j@example.com	2345678901	<a href="#">Update</a> <a href="#">Delete</a>
3	Michael	David	Brown	michael_brown@example.com	3456789012	<a href="#">Update</a> <a href="#">Delete</a>
4	Sophia	Elizabeth	Martinez	sophia_martinez@example.com	4567890123	<a href="#">Update</a> <a href="#">Delete</a>
5	William	Christopher	Anderson	william_anderson@example.com	5678901234	<a href="#">Update</a> <a href="#">Delete</a>
6	Isabella	Lauren	Taylor	isabella_taylor@example.com	6789012345	<a href="#">Update</a> <a href="#">Delete</a>
7	James	Ryan	Thomas	james.thomas@example.com	7890183456	<a href="#">Update</a> <a href="#">Delete</a>
8	Olivia	Avery	Lee	olivia_lee@example.com	8902234567	<a href="#">Update</a> <a href="#">Delete</a>

Figure 8: User Page

## Responsibility of G2

In developing the backend for our web application, we undertook the responsibility of architecting a robust system that manages various aspects crucial for its functionality. Leveraging the Flask framework alongside MySQL for database management, we orchestrated a backend solution capable of handling user authentication, domain and location management, request handling, as well as user and worker administration. By implementing a structured approach to backend development, we aimed to ensure seamless communication between the frontend and database while maintaining the integrity and security of user data. Through the use of Flask's modular design and extensive ecosystem of extensions, our backend system emerged as a versatile and scalable foundation for the web application.

## Backend Artitechture

The backend architecture follows a model-view-controller (MVC) pattern, where Flask serves as the controller handling incoming requests, querying the database, and rendering appropriate views. The model layer consists of SQLAlchemy ORM models representing database tables, facilitating object-oriented interactions with the database. Views are rendered using Jinja2 templates, allowing dynamic content generation based on data retrieved from the backend. Overall, this architecture ensures the separation of concerns, scalability, and maintainability of the backend system.

## Libraries Used

- **Flask:** Flask serves as the core framework for our web application, providing the foundational structure for request handling, routing, and view rendering.
- **Flask-Login:** Flask-Login is utilized for user session management, enabling authentication and authorization features.
- **Flask-MySQLdb:** This extension bridges Flask with MySQL, enabling seamless communication between the Flask application and the MySQL database.
- **Flask-SQLAlchemy:** Flask-SQLAlchemy integrates SQLAlchemy, an ORM tool, with Flask, simplifying database interactions by abstracting away SQL queries into Python objects.
- **mysqlclient:** mysqlclient is a MySQL database connector for Python, which Flask-MySQLdb utilizes internally.
- **SQLAlchemy:** SQLAlchemy is an SQL toolkit and ORM for Python, used by Flask-SQLAlchemy to interact with databases.
- **Werkzeug:** Werkzeug is a WSGI utility library for Python, providing low-level interfaces for HTTP request handling and response generation.

## API Endpoints Overview

Table 1: API Endpoints and Descriptions

API Endpoint	Method	Description
/auth/login	GET	Renders login page.
/auth/login	POST	Authenticates user.
/auth/signup	GET	Renders signup page.
/auth/signup	POST	Registers new user.
/auth/logout	GET	Logs out user.
/domains	GET	Retrieves all domains.
/domains/add	POST	Adds a new domain.
/domains/update/<id>	POST	Updates an existing domain.
/domains/delete/<id>	POST	Deletes an existing domain.
/locations	GET	Retrieves all locations.
/locations/add	POST	Adds a new location.
/locations/update/<id>	POST	Updates an existing location.
/locations/delete/<id>	POST	Deletes an existing location.
/requests	GET	Retrieves all requests.
/requests/add	POST	Adds a new request.
/requests/update/<id>	POST	Updates an existing request.
/requests/delete/<id>	POST	Deletes an existing request.
/users	GET	Retrieves all users.
/users/add	POST	Adds a new user.
/users/update/<id>	POST	Updates an existing user.
/users/delete/<id>	POST	Deletes an existing user.
/workers	GET	Retrieves all workers.
/workers/add	POST	Adds a new worker.
/workers/update/<id>	POST	Updates an existing worker.
/workers/delete/<id>	POST	Deletes an existing worker.

## Challenges with Implementing RENAME Operation

The RENAME operation, which allows renaming the name of a table or column, presents challenges in the context of our backend implementation. While the operation itself seems straightforward, its implementation introduces complexities due to dependencies on table and column names in other operations.

In our backend architecture, various operations such as querying, updating, and deleting data rely on specific table and column names. If these names are altered using the RENAME operation, it can potentially break the functionality of other operations that reference them. For instance, if a table or column name used in a query is renamed, the query would no longer retrieve the expected data, leading to errors or unintended behavior.

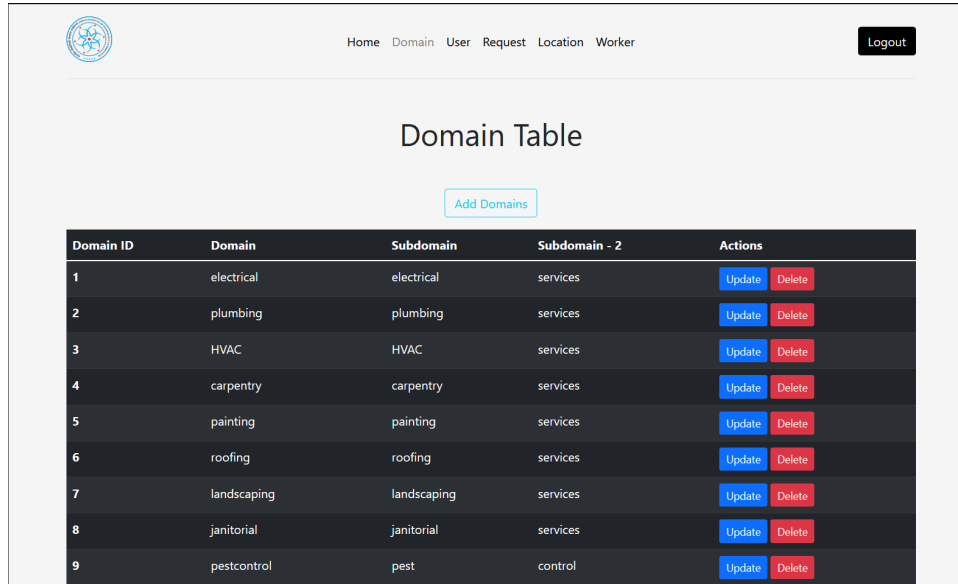
Moreover, ensuring consistency across the entire application becomes difficult when table or column names are subject to change. Developers would need to meticulously update all references to the renamed entities throughout the codebase, which is error-prone and time-consuming.

After discussing the matter with a Teaching Assistant (TA), it was concluded that integrating the RENAME operation into our backend system would require significant restructuring and validation to mitigate the risks associated with potential data inconsistencies and operational failures. Given the project constraints and prioritization of core functionalities, the decision was made to defer the implementation of the RENAME operation for future iterations, focusing instead on delivering a stable and reliable system that meets essential requirements.

# Responsibility of G1 and G2

## Display of Dummy Entries

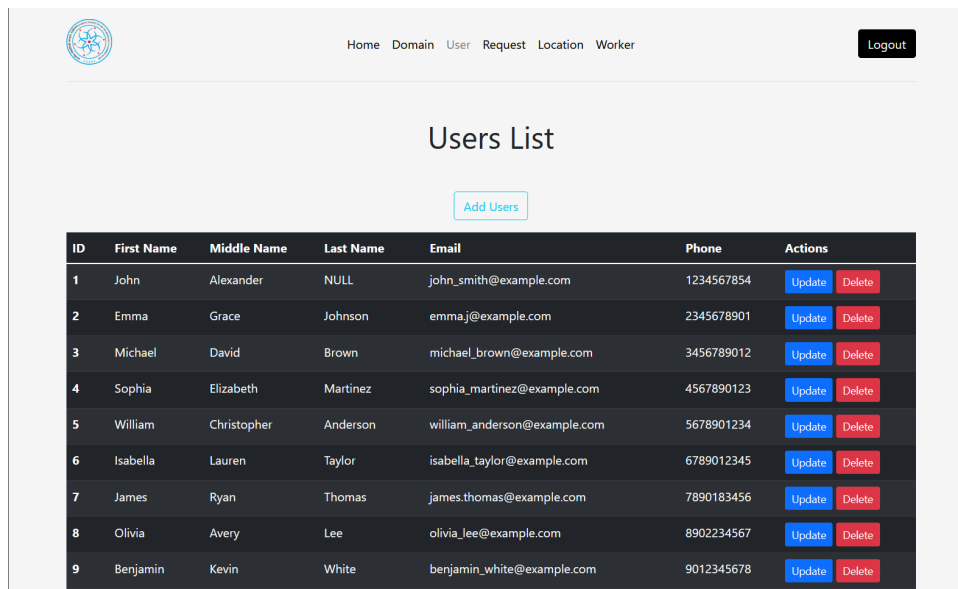
### Domain Table



Domain ID	Domain	Subdomain	Subdomain - 2	Actions
1	electrical	electrical	services	<a href="#">Update</a> <a href="#">Delete</a>
2	plumbing	plumbing	services	<a href="#">Update</a> <a href="#">Delete</a>
3	HVAC	HVAC	services	<a href="#">Update</a> <a href="#">Delete</a>
4	carpentry	carpentry	services	<a href="#">Update</a> <a href="#">Delete</a>
5	painting	painting	services	<a href="#">Update</a> <a href="#">Delete</a>
6	roofing	roofing	services	<a href="#">Update</a> <a href="#">Delete</a>
7	landscaping	landscaping	services	<a href="#">Update</a> <a href="#">Delete</a>
8	janitorial	janitorial	services	<a href="#">Update</a> <a href="#">Delete</a>
9	pestcontrol	pest	control	<a href="#">Update</a> <a href="#">Delete</a>

Figure 9: Domain Page dummy values

### User Table

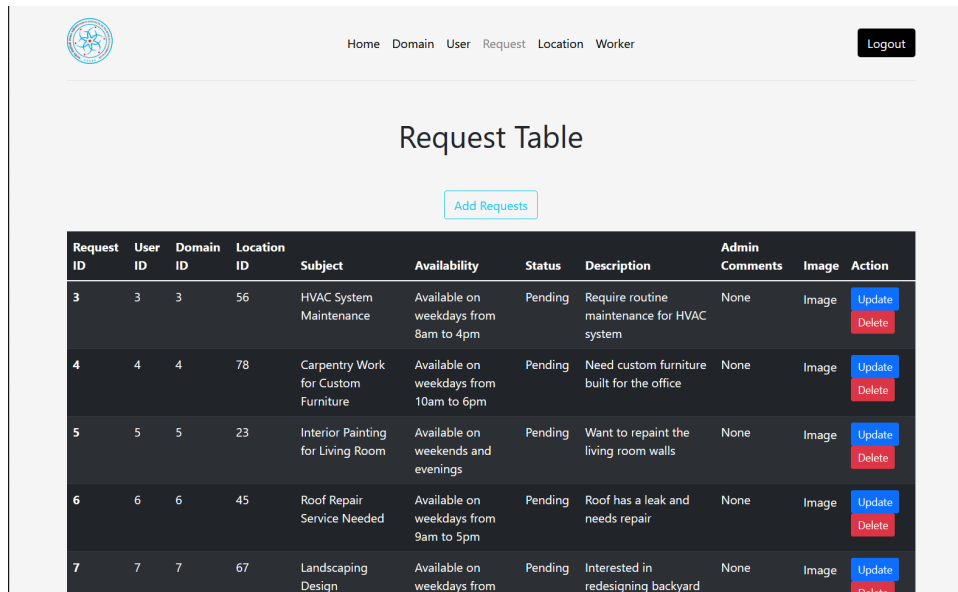


ID	First Name	Middle Name	Last Name	Email	Phone	Actions
1	John	Alexander	NULL	john_smith@example.com	1234567854	<a href="#">Update</a> <a href="#">Delete</a>
2	Emma	Grace	Johnson	emma.j@example.com	2345678901	<a href="#">Update</a> <a href="#">Delete</a>
3	Michael	David	Brown	michael_brown@example.com	3456789012	<a href="#">Update</a> <a href="#">Delete</a>
4	Sophia	Elizabeth	Martinez	sophia_martinez@example.com	4567890123	<a href="#">Update</a> <a href="#">Delete</a>
5	William	Christopher	Anderson	william_anderson@example.com	5678901234	<a href="#">Update</a> <a href="#">Delete</a>
6	Isabella	Lauren	Taylor	isabella_taylor@example.com	6789012345	<a href="#">Update</a> <a href="#">Delete</a>
7	James	Ryan	Thomas	james.thomas@example.com	7890183456	<a href="#">Update</a> <a href="#">Delete</a>
8	Olivia	Avery	Lee	olivia_lee@example.com	8902234567	<a href="#">Update</a> <a href="#">Delete</a>
9	Benjamin	Kevin	White	benjamin_white@example.com	9012345678	<a href="#">Update</a> <a href="#">Delete</a>

Figure 10: User Page dummy values



## Request Table

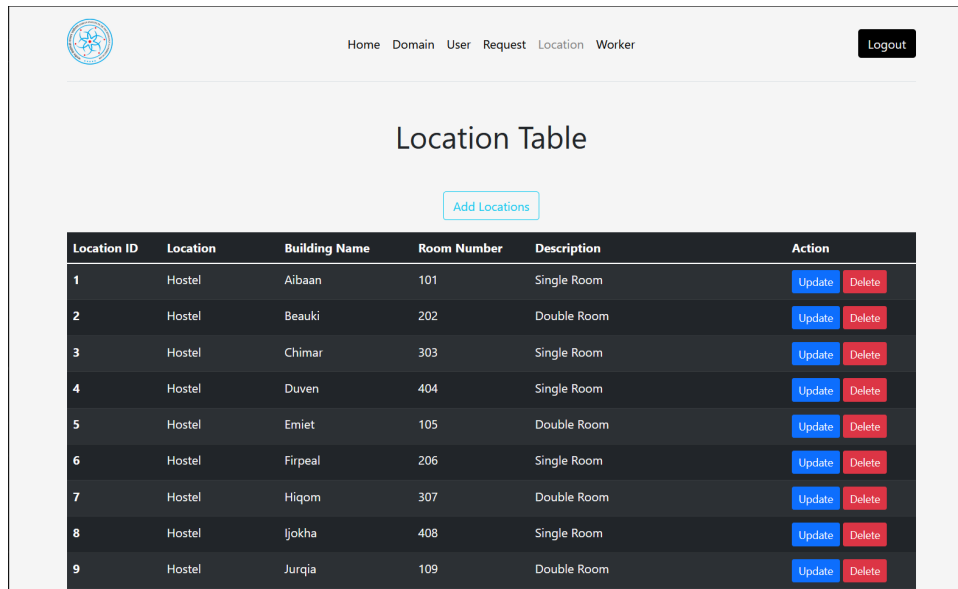


The image shows a web application interface for a 'Request Table'. At the top, there is a navigation bar with a logo on the left and links for 'Home', 'Domain', 'User', 'Request', 'Location', and 'Worker' in the center. A 'Logout' button is on the right. Below the navigation bar, the title 'Request Table' is centered. Underneath the title is a button labeled 'Add Requests'. The main content is a table with the following columns: Request ID, User ID, Domain ID, Location ID, Subject, Availability, Status, Description, Admin Comments, Image, and Action. The table contains five rows of dummy data.

Request ID	User ID	Domain ID	Location ID	Subject	Availability	Status	Description	Admin Comments	Image	Action
3	3	3	56	HVAC System Maintenance	Available on weekdays from 8am to 4pm	Pending	Require routine maintenance for HVAC system	None	Image	<a href="#">Update</a> <a href="#">Delete</a>
4	4	4	78	Carpentry Work for Custom Furniture	Available on weekdays from 10am to 6pm	Pending	Need custom furniture built for the office	None	Image	<a href="#">Update</a> <a href="#">Delete</a>
5	5	5	23	Interior Painting for Living Room	Available on weekends and evenings	Pending	Want to repaint the living room walls	None	Image	<a href="#">Update</a> <a href="#">Delete</a>
6	6	6	45	Roof Repair Service Needed	Available on weekdays from 9am to 5pm	Pending	Roof has a leak and needs repair	None	Image	<a href="#">Update</a> <a href="#">Delete</a>
7	7	7	67	Landscaping Design	Available on weekdays from	Pending	Interested in redesigning backyard	None	Image	<a href="#">Update</a> <a href="#">Delete</a>

Figure 11: Request Page dummy values

## Location Table

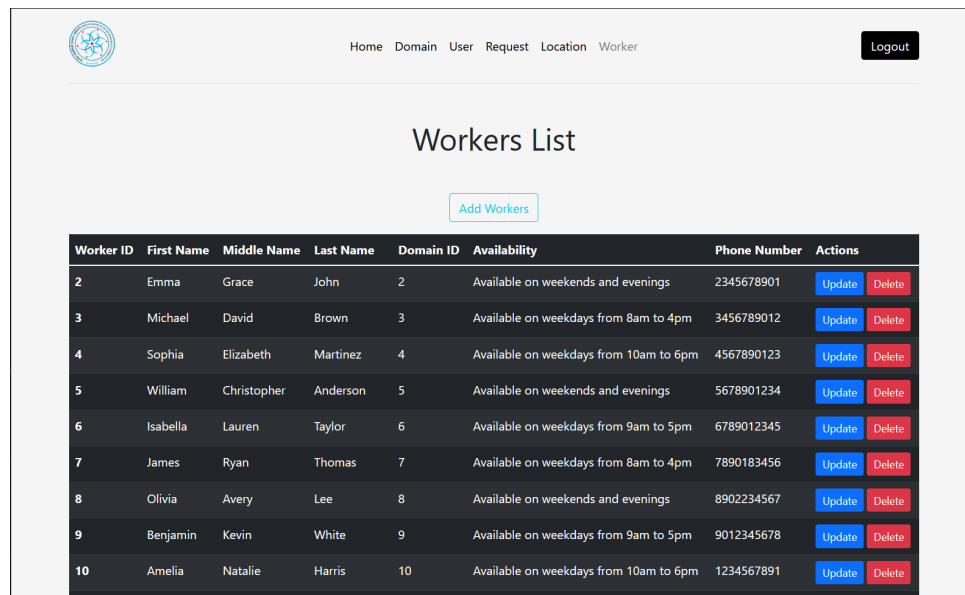


The image shows a web application interface for a 'Location Table'. At the top, there is a navigation bar with a logo on the left and links for 'Home', 'Domain', 'User', 'Request', 'Location', and 'Worker' in the center. A 'Logout' button is on the right. Below the navigation bar, the title 'Location Table' is centered. Underneath the title is a button labeled 'Add Locations'. The main content is a table with the following columns: Location ID, Location, Building Name, Room Number, Description, and Action. The table contains nine rows of dummy data.

Location ID	Location	Building Name	Room Number	Description	Action
1	Hostel	Aibaan	101	Single Room	<a href="#">Update</a> <a href="#">Delete</a>
2	Hostel	Beauki	202	Double Room	<a href="#">Update</a> <a href="#">Delete</a>
3	Hostel	Chimar	303	Single Room	<a href="#">Update</a> <a href="#">Delete</a>
4	Hostel	Duven	404	Single Room	<a href="#">Update</a> <a href="#">Delete</a>
5	Hostel	Emiet	105	Double Room	<a href="#">Update</a> <a href="#">Delete</a>
6	Hostel	Firpeal	206	Single Room	<a href="#">Update</a> <a href="#">Delete</a>
7	Hostel	Hiqom	307	Double Room	<a href="#">Update</a> <a href="#">Delete</a>
8	Hostel	Ijokha	408	Single Room	<a href="#">Update</a> <a href="#">Delete</a>
9	Hostel	Jurqia	109	Double Room	<a href="#">Update</a> <a href="#">Delete</a>

Figure 12: Location Page dummy values

## Worker Table

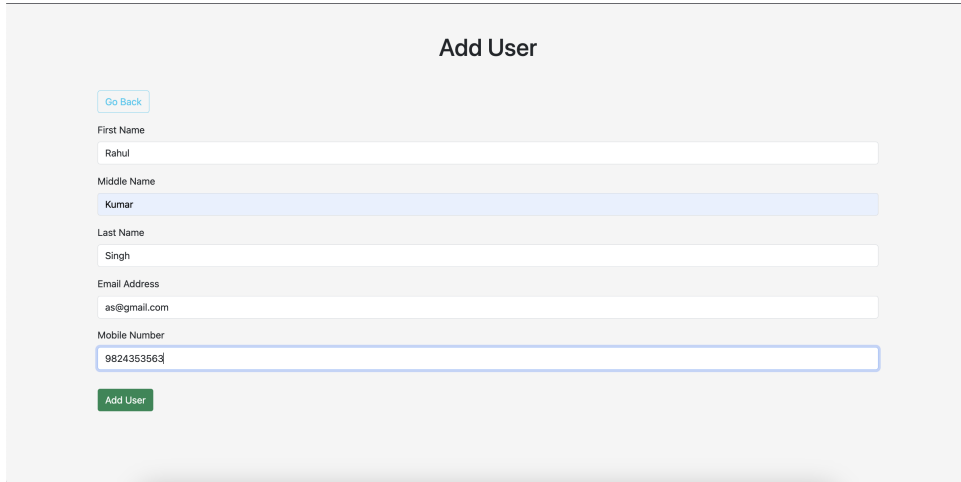


Worker ID	First Name	Middle Name	Last Name	Domain ID	Availability	Phone Number	Actions
2	Emma	Grace	John	2	Available on weekends and evenings	2345678901	<a href="#">Update</a> <a href="#">Delete</a>
3	Michael	David	Brown	3	Available on weekdays from 8am to 4pm	3456789012	<a href="#">Update</a> <a href="#">Delete</a>
4	Sophia	Elizabeth	Martinez	4	Available on weekdays from 10am to 6pm	4567890123	<a href="#">Update</a> <a href="#">Delete</a>
5	William	Christopher	Anderson	5	Available on weekends and evenings	5678901234	<a href="#">Update</a> <a href="#">Delete</a>
6	Isabella	Lauren	Taylor	6	Available on weekdays from 9am to 5pm	6789012345	<a href="#">Update</a> <a href="#">Delete</a>
7	James	Ryan	Thomas	7	Available on weekdays from 8am to 4pm	7890183456	<a href="#">Update</a> <a href="#">Delete</a>
8	Olivia	Avery	Lee	8	Available on weekends and evenings	8902234567	<a href="#">Update</a> <a href="#">Delete</a>
9	Benjamin	Kevin	White	9	Available on weekdays from 9am to 5pm	9012345678	<a href="#">Update</a> <a href="#">Delete</a>
10	Amelia	Natalie	Harris	10	Available on weekdays from 10am to 6pm	1234567891	<a href="#">Update</a> <a href="#">Delete</a>

Figure 13: Worker Page dummy values

## Operations in User List

### INSERT operation



The image shows a web form titled "Add User". At the top left is a "Go Back" button. Below it are input fields for "First Name" (containing "Rahul"), "Middle Name" (containing "Kumar"), "Last Name" (containing "Singh"), "Email Address" (containing "as@gmail.com"), and "Mobile Number" (containing "9824353563"). At the bottom is an "Add User" button.

Figure 14: Insert Operation on User Table

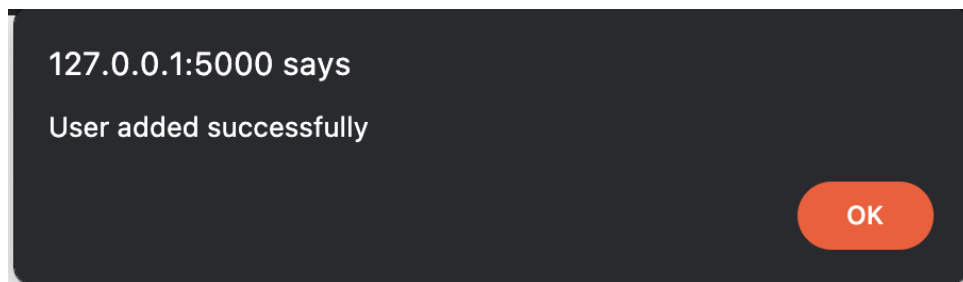


Figure 15: Added Successfully Dialog Box

147	Shubh			shb@gmail.com	7043236104	<a href="#">Update</a>	<a href="#">Delete</a>
149	Narendra	Damodardas	Modi	nm@gmail.com	9876543210	<a href="#">Update</a>	<a href="#">Delete</a>
151	Rahul	Kumar	Singh	as@gmail.com	9824353563	<a href="#">Update</a>	<a href="#">Delete</a>

Figure 16: Changes reflected on WebApp

147	Shubh			shb@gmail.com	7043236104	
149	Narendra	Damodardas	Modi	nm@gmail.com	9876543210	
151	Rahul	Kumar	Singh	as@gmail.com	9824353563	
NULL	NULL	NULL	NULL	NULL	NULL	

Figure 17: Changes reflected in Database

## DELETE operations

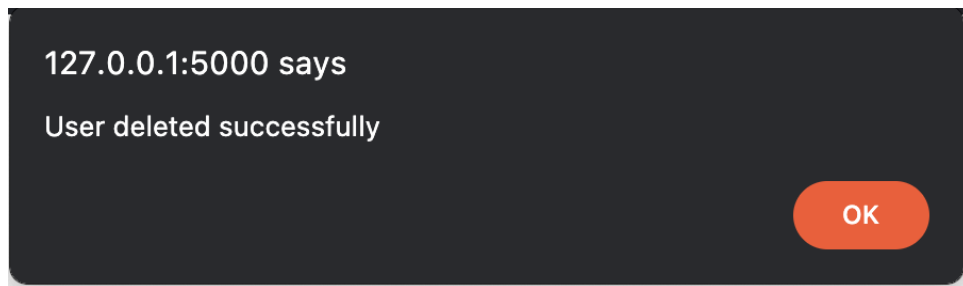


Figure 18: Deleting the value

## UPDATE operations

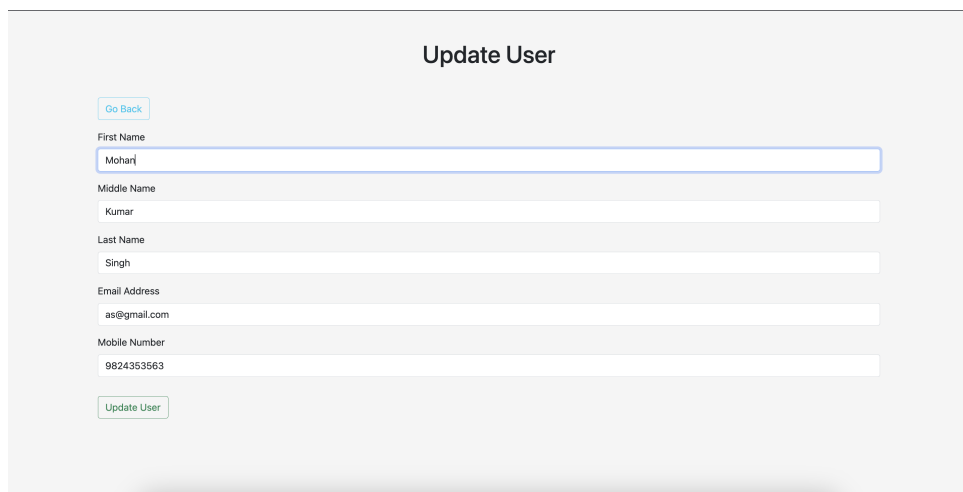


Figure 19: Updating the value

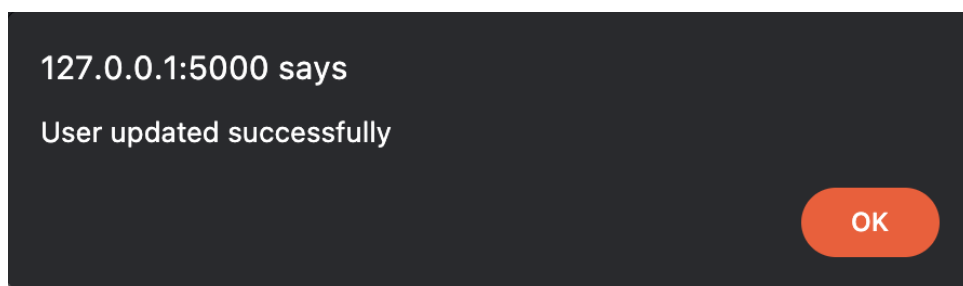


Figure 20: Updated successfully dialog box

147	Shubh			shb@gmail.com	7043236104	<a href="#">Update</a>	<a href="#">Delete</a>
149	Narendra	Damodardas	Modi	nm@gmail.com	9876543210	<a href="#">Update</a>	<a href="#">Delete</a>
151	Mohan	Kumar	Singh	as@gmail.com	9824353563	<a href="#">Update</a>	<a href="#">Delete</a>

Figure 21: Update reflected on the webApp

147	Shubh			shb@gmail.com	7043236104	
149	Narendra	Damodardas	Modi	nm@gmail.com	9876543210	
151	Mohan	Kumar	Singh	as@gmail.com	9824353563	
NULL	NULL	NULL	NULL	NULL	NULL	

Figure 22: Update reflected on the database

## Operations in Request List

### INSERT operation

**Add Request**

[Go Back](#)

User ID: 34

Domain ID: 2

Location ID: 3

Subject: AC leakage

Availability: Available on weekends

Status: Pending

Description: none

Admin Comments: none

Image: Choose file No file chosen

[Add request](#)

Figure 23: Insert Operation on Request Table

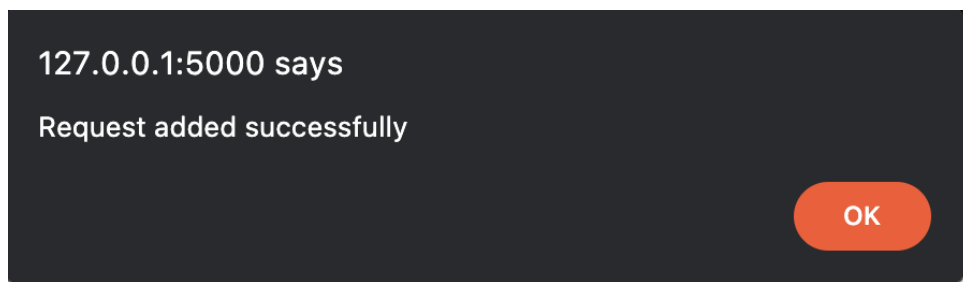


Figure 24: Added Successfully Dialog Box

28	28	8	32	Janitorial Services for Medical Facility	Available on weekends and evenings	Pending	Medical facility requires specialized cleaning services	None		<a href="#">Update</a> <a href="#">Delete</a>
29	29	9	54	Pest Control for Bed Bug Infestation	Available on weekdays from 9am to 5pm	Pending	Bed bug infestation in the bedroom needs treatment	None		<a href="#">Update</a> <a href="#">Delete</a>
34	34	2	3	AC leakage	Available on weekends	Pending	none	none		<a href="#">Update</a> <a href="#">Delete</a>

Figure 25: updated webpage

28	28	8	32	Janitorial Services for Medical Facility	Available on weekends and evenings	Pending	Medical facility requires specialized cleaning
29	29	9	54	Pest Control for Bed Bug Infestation	Available on weekdays from 9am to 5pm	Pending	Bed bug infestation in the bedroom needs
34	34	2	3	AC leakage	Available on weekends	Pending	none
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 26: updated database



## DELETE operations

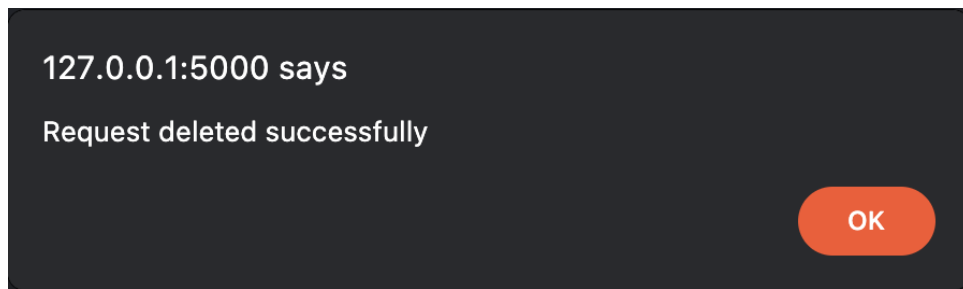


Figure 27: Deleted Successfully Dialog Box

## UPDATE operations

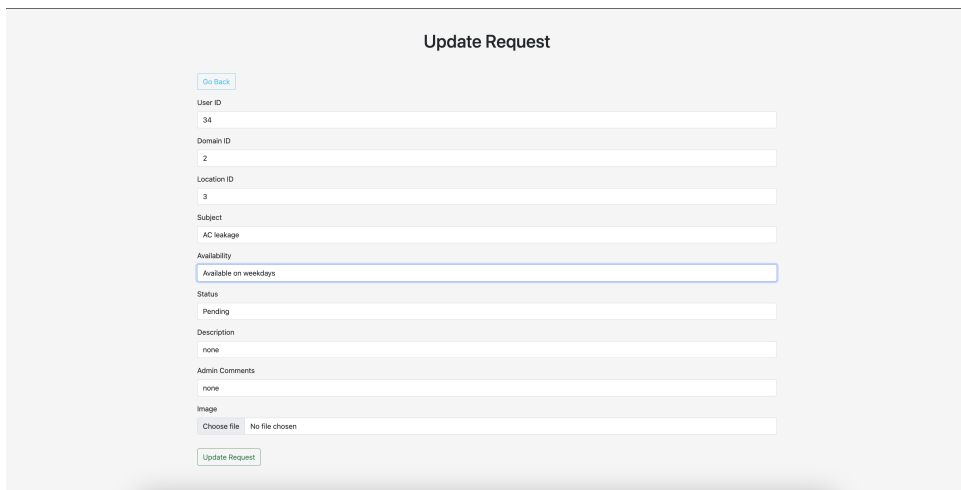
A light gray form titled "Update Request" at the top center. At the top left of the form is a blue "Go Back" button. The form contains several input fields: "User ID" with the value "34", "Domain ID" with "2", "Location ID" with "3", "Subject" with "AC leakage", and "Availability" with "Available on weekdays". Below these are "Status" (Pending), "Description" (none), and "Admin Comments" (none). At the bottom is an "Image" section with a "Choose file" button and the text "No file chosen". A green "Update Request" button is located at the bottom center of the form.

Figure 28: Updating the value

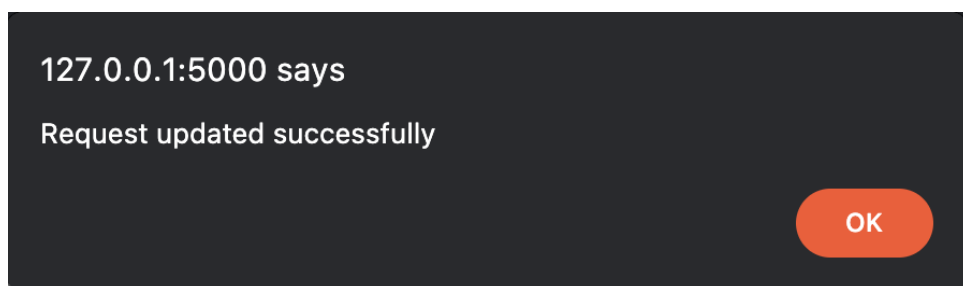


Figure 29: Updated successfully dialog box




28	28	8	32	Janitorial Services for Medical Facility	Available on weekends and evenings	Pending	Medical facility requires specialized cleaning services	None	 Image	<a href="#">Update</a> <a href="#">Delete</a>
29	29	9	54	Pest Control for Bed Bug Infestation	Available on weekdays from 9am to 5pm	Pending	Bed bug infestation in the bedroom needs treatment	None	 Image	<a href="#">Update</a> <a href="#">Delete</a>
34	34	2	3	AC leakage	Available on weekdays	Pending	none	none	 Image	<a href="#">Update</a> <a href="#">Delete</a>

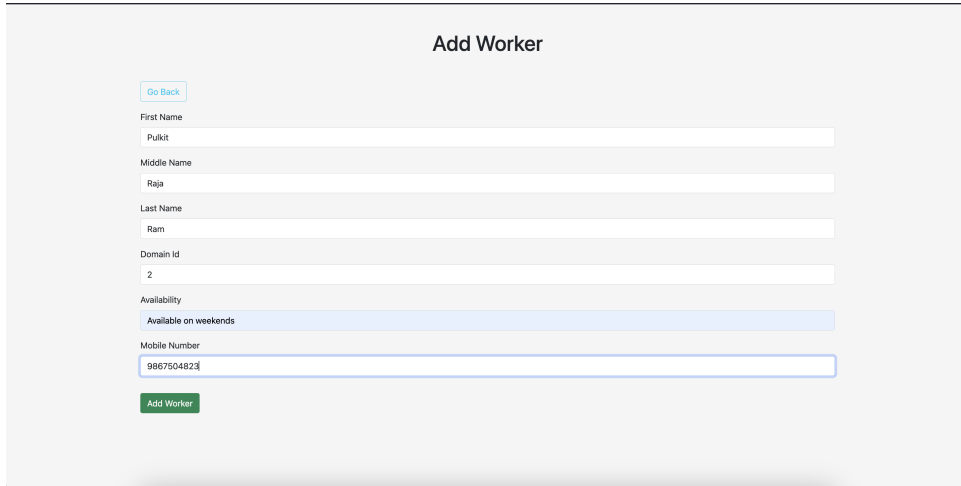
Figure 30: Updated webpage

28	28	8	32	Janitorial Services for Medical Facility	Available on weekends and evenings	Pending	Medical facility requires specialized cleaning
29	29	9	54	Pest Control for Bed Bug Infestation	Available on weekdays from 9am to 5pm	Pending	Bed bug infestation in the bedroom needs
34	34	2	3	AC leakage	Available on weekdays	Pending	none
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 31: Updated database

## Operations in Worker List

### INSERT operation



**Add Worker**

[Go Back](#)

First Name  
Pulkit

Middle Name  
Raja

Last Name  
Ram

Domain Id  
2

Availability  
Available on weekends

Mobile Number  
9867504823

[Add Worker](#)

Figure 32: Insert Operation on Worker Table

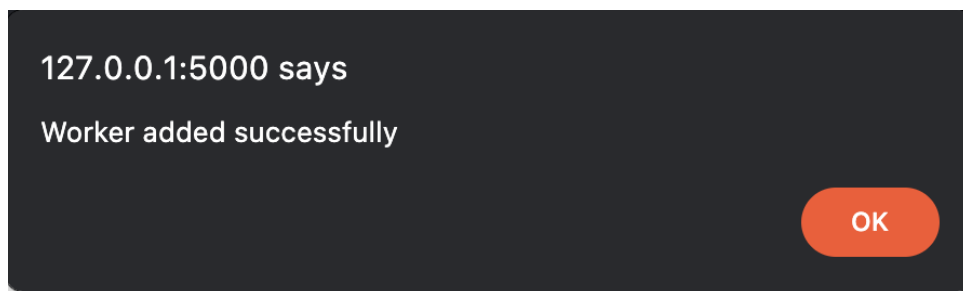


Figure 33: Added Successfully Dialog Box

400	Liberty	Alia	Howe	10	Available on weekends and evenings	5678901283	<a href="#">Update</a>	<a href="#">Delete</a>
405	Shubh	a	Agarwal	2	Anytime	7043236104	<a href="#">Update</a>	<a href="#">Delete</a>
407	Pulkit	Raja	Ram	2	Available on weekends	9867504823	<a href="#">Update</a>	<a href="#">Delete</a>

Figure 34: updated webpage

400	Liberty	Alia	Howe	10	Available on weekends and evenings	5678901283	
405	Shubh	a	Agarwal	2	Anytime	7043236104	
407	Pulkit	Raja	Ram	2	Available on weekends	9867504823	
NULL	NULL	NULL	NULL	NULL	NULL	NULL	

Figure 35: updated database

## DELETE operations

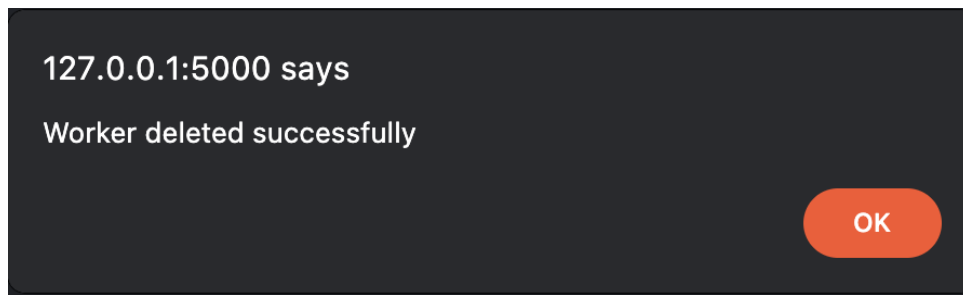


Figure 36: Deleted Successfully Dialog Box

## UPDATE operations

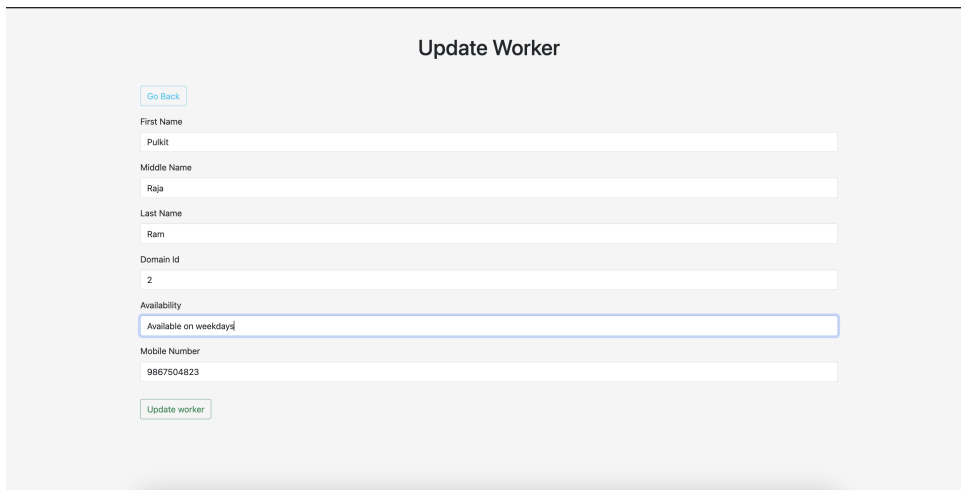
A light gray form titled "Update Worker" in bold. At the top left is a "Go Back" button. Below the title are several input fields: "First Name" (Pulkit), "Middle Name" (Raja), "Last Name" (Ram), "Domain Id" (2), "Availability" (Available on weekday), and "Mobile Number" (9867504823). At the bottom is an "Update worker" button.

Figure 37: Updating the value

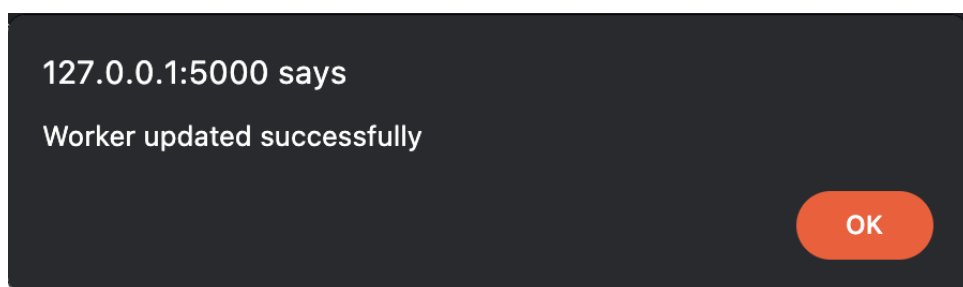


Figure 38: Updated successfully dialog box

400	Liberty	Alia	Howe	10	Available on weekends and evenings	5678901283	<a href="#">Update</a>	<a href="#">Delete</a>
405	Shubh	a	Agarwal	2	Anytime	7043236104	<a href="#">Update</a>	<a href="#">Delete</a>
407	Pulkit	Raja	Ram	2	Available on weekdays	9867504823	<a href="#">Update</a>	<a href="#">Delete</a>

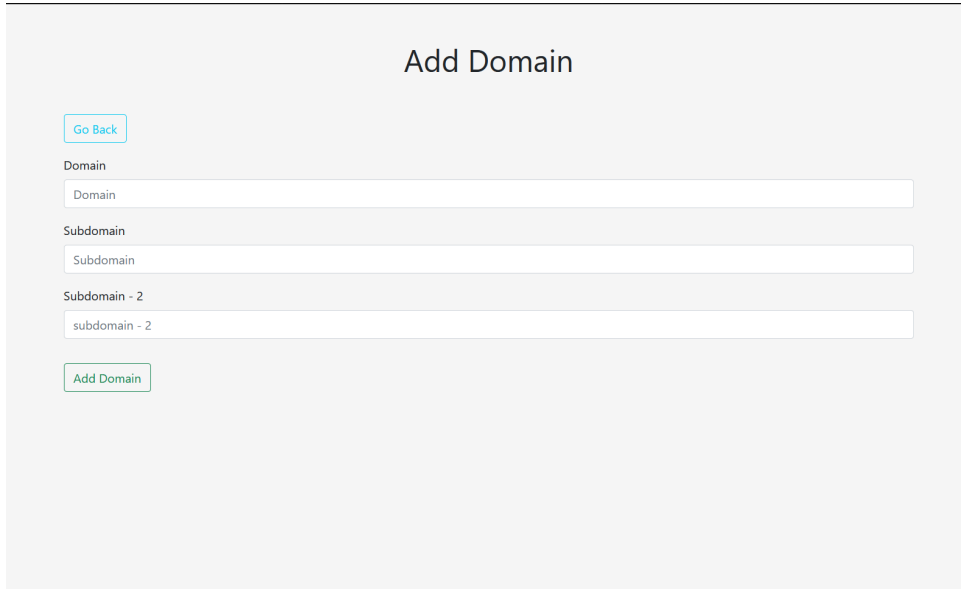
Figure 39: Updated webpage

400	Liberty	Alia	Howe	10	Available on weekends and evenings	5678901283	
405	Shubh	a	Agarwal	2	Anytime	7043236104	
407	Pulkit	Raja	Ram	2	Available on weekdays	9867504823	
NULL	NULL	NULL	NULL	NULL	NULL	NULL	

Figure 40: Updated database

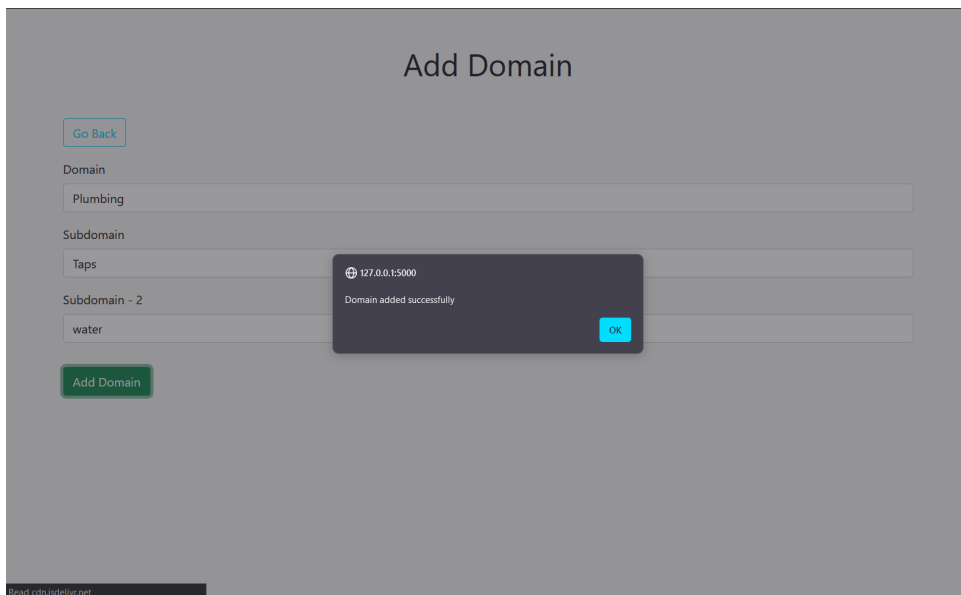
## Operations in Domain List

### INSERT operation



The screenshot shows a web form titled "Add Domain". At the top left is a "Go Back" button. Below it are three input fields: "Domain" (empty), "Subdomain" (empty), and "Subdomain - 2" (containing the text "subdomain - 2"). At the bottom left is an "Add Domain" button.

Figure 41: Insert Operation on Domain Table



The screenshot shows the same "Add Domain" form, but now the input fields are filled: "Domain" contains "Plumbing", "Subdomain" contains "Taps", and "Subdomain - 2" contains "water". The "Add Domain" button is now green. A dark gray dialog box is overlaid in the center, displaying a green plus icon, the IP address "127.0.0.1:5000", the message "Domain added successfully", and an "OK" button. In the bottom left corner, there is a small link that reads "Read cctvjsdelivr.net".

Figure 42: Added Successfully Dialog Box

10	securitysystems	security	systems	<a href="#">Update</a>	<a href="#">Delete</a>
12	Electrical	Room	Tubelights Fans	<a href="#">Update</a>	<a href="#">Delete</a>
13	Plumbing	Taps	water	<a href="#">Update</a>	<a href="#">Delete</a>

Figure 43: Changes reflected on WebApp

9	pestcontrol	pest	control
10	securitysystems	security	systems
13	Plumbing	Taps	liquid

Figure 44: Changes reflected on Terminal



## DELETE operations

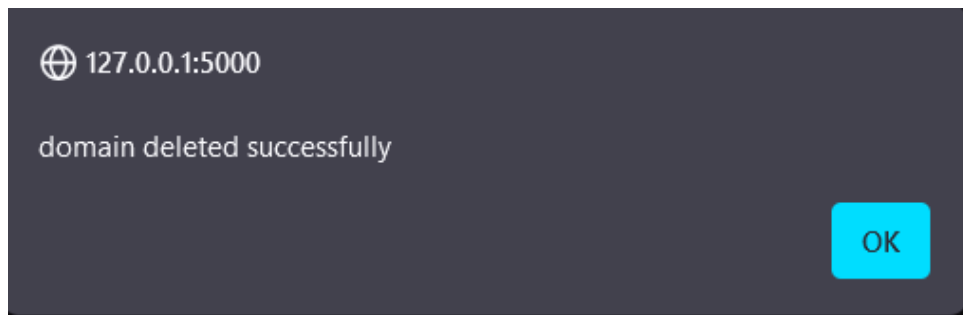


Figure 45: Deleting the value

8		janitorial		janitorial		services	
9		pestcontrol		pest		control	
10		securitysystems		security		systems	
<hr/>							

Figure 46: Deleting the value

## UPDATE operations

### Update Domain

Go Back

Domain

Plumbing

Subdomain

Taps

Subdomain - 2

water

Update Domain

Figure 47: Updating the value

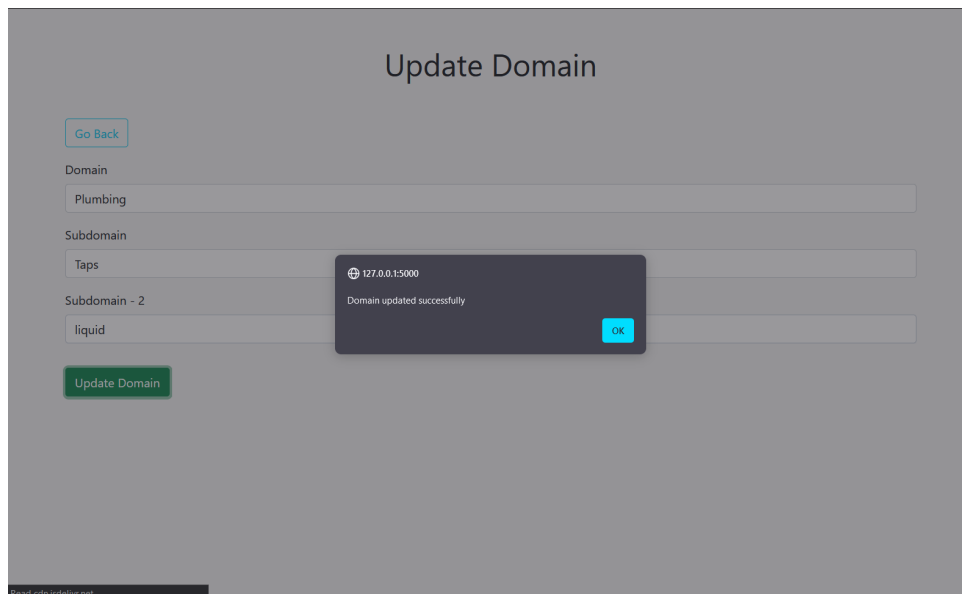


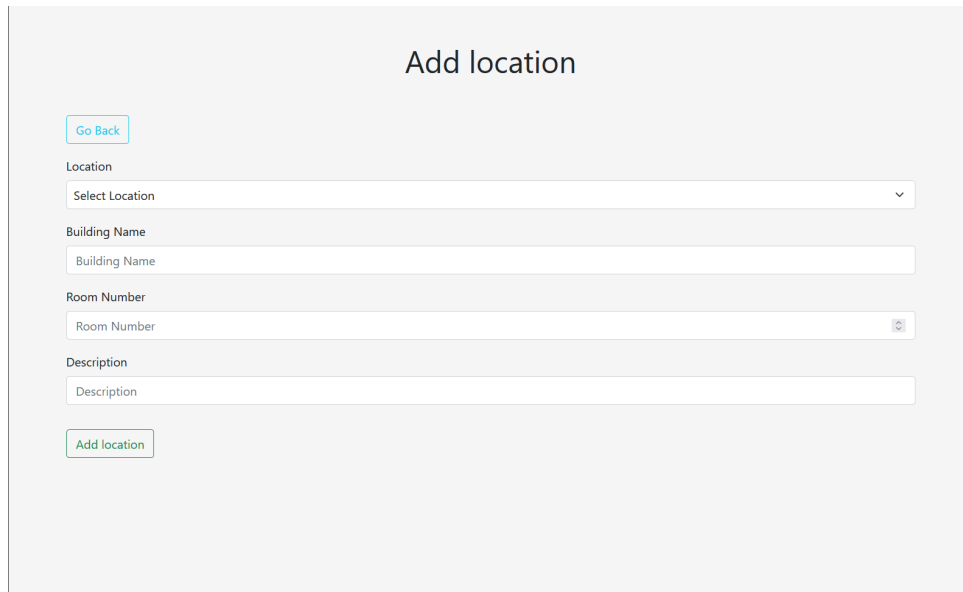
Figure 48: Updated successfully dialog box

9	pestcontrol	pest	control	<a href="#">Update</a>	<a href="#">Delete</a>
10	securitysystems	security	systems	<a href="#">Update</a>	<a href="#">Delete</a>
13	Plumbing	Taps	water	<a href="#">Update</a>	<a href="#">Delete</a>

Figure 49: Update reflected on the webApp

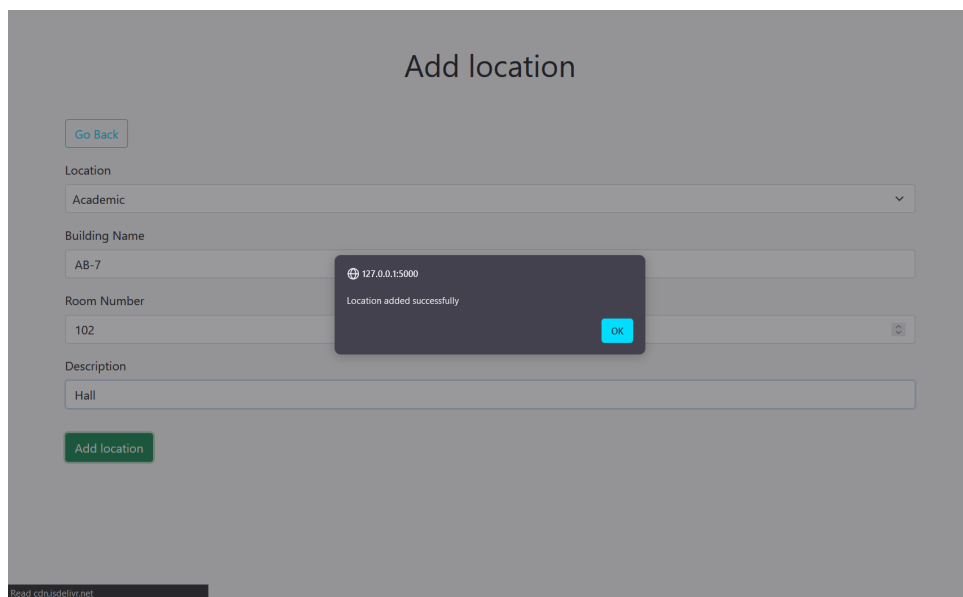
## Operations in Location Table

### INSERT operations



The screenshot shows a web form titled "Add location". At the top left is a "Go Back" button. Below it are four input fields: "Location" (a dropdown menu with "Select Location" as the placeholder), "Building Name" (a text input field), "Room Number" (a text input field with a small icon on the right), and "Description" (a text input field). At the bottom left of the form is an "Add location" button.

Figure 50: Inserting Values in Location Table



This screenshot shows the same "Add location" form as Figure 50, but with a success dialog box overlay. The dialog box is dark gray with a white border and contains the text "127.0.0.1:5000" and "Location added successfully". There is an "OK" button in the bottom right corner of the dialog box. The form fields behind the dialog box are filled with the following values: "Location" is set to "Academic", "Building Name" is "AB-7", "Room Number" is "102", and "Description" is "Hall". The "Add location" button is now green. In the bottom left corner of the page, there is a small text link: "Read cdb.jsdelivr.net".

Figure 51: Successfully inserted dialog box


### DELETE operation

85	Research Park	Research Park 3	303	Laboratory complex	<a href="#">Update</a>	<a href="#">Delete</a>
86	Research Park	Research Park 4	404	Incubation centers	<a href="#">Update</a>	<a href="#">Delete</a>
90	Academic	AB-7	102	Hall	<a href="#">Update</a>	<a href="#">Delete</a>

Figure 52: Changes reflected in webpage

85	Research Park	Research Park 3	303	Laboratory complex
86	Research Park	Research Park 4	404	Incubation centers
90	Academic	AB-7	102	Hall

Figure 53: Changes reflected in Terminal

 127.0.0.1:5000

Location deleted successfully

OK

Figure 54: Deleting Values

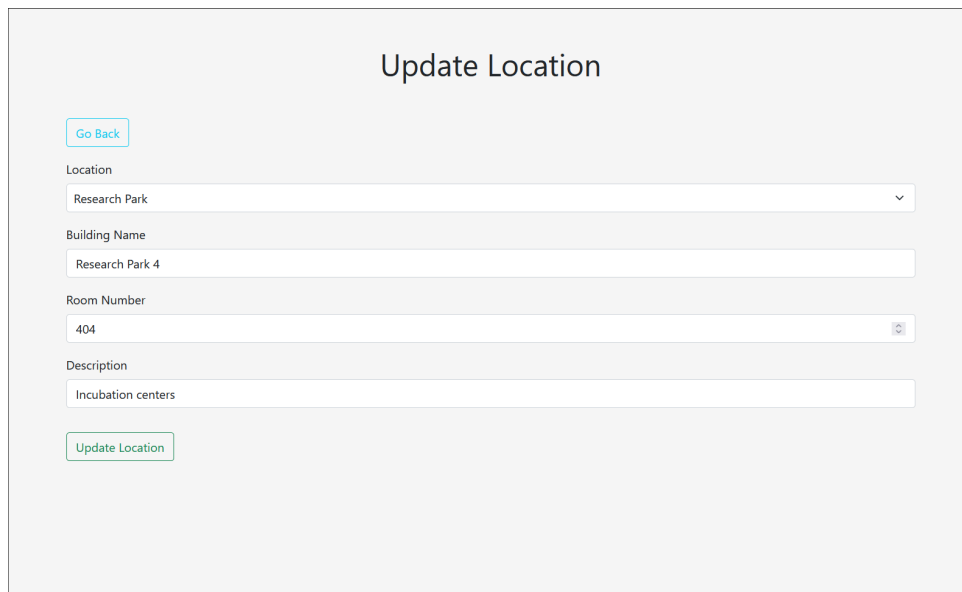
84	Research Park	Research Park 2	202	Collaborative space	<a href="#">Update</a>	<a href="#">Delete</a>
85	Research Park	Research Park 3	303	Laboratory complex	<a href="#">Update</a>	<a href="#">Delete</a>
86	Research Park	Research Park 4	404	Incubation centers	<a href="#">Update</a>	<a href="#">Delete</a>

Figure 55: Changes Reflected on WebApp

84	Research Park	Research Park 2	202	Collaborative space
85	Research Park	Research Park 3	303	Laboratory complex
86	Research Park	Research Park 4	404	Incubation centers

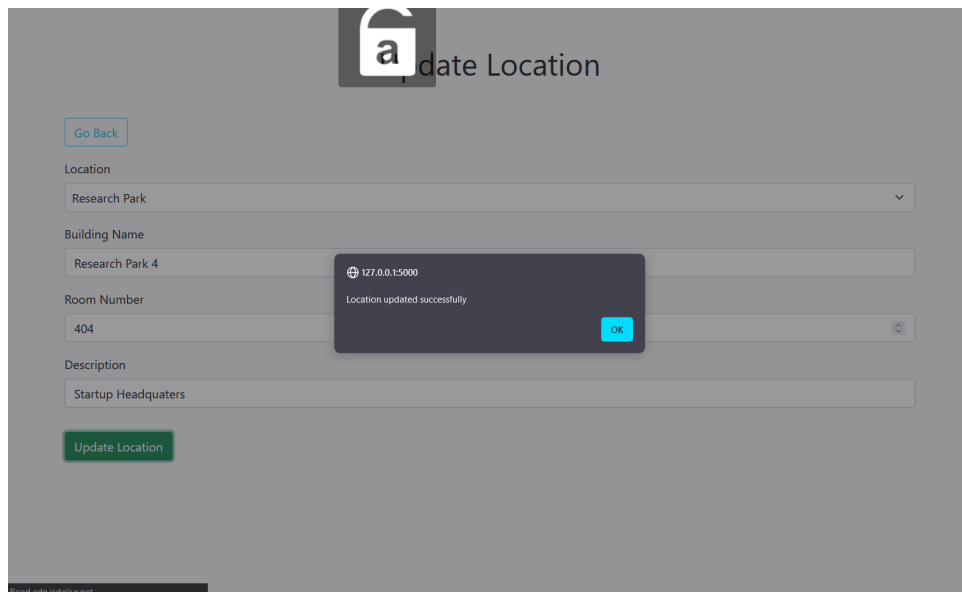
Figure 56: Changes Reflected on Terminal

## UPDATE operation



The screenshot shows a web form titled "Update Location". At the top left is a "Go Back" button. Below it are four input fields: "Location" (a dropdown menu showing "Research Park"), "Building Name" (a text field showing "Research Park 4"), "Room Number" (a text field showing "404"), and "Description" (a text field showing "Incubation centers"). At the bottom left is an "Update Location" button.

Figure 57: Updating Entries in Location Table



This screenshot shows the same "Update Location" form as Figure 57, but with a modal dialog box overlaid in the center. The dialog box has a dark background and contains the text "127.0.0.1:5000" and "Location updated successfully". There is an "OK" button in the bottom right corner of the dialog box. The form fields behind the dialog are dimmed. The "Location" dropdown now shows "Research Park", "Building Name" shows "Research Park 4", "Room Number" shows "404", and "Description" shows "Startup Headquarters". The "Update Location" button is still visible at the bottom left.

Figure 58: Successfully changed dialog box

84	Research Park	Research Park 2	202	Collaborative space	Update	Delete
85	Research Park	Research Park 3	303	Laboratory complex	Update	Delete
86	Research Park	Research Park 4	404	Startup Headquarters	Update	Delete

Figure 59: Changes Reflected on WebApp

84	Research Park	Research Park 2	202	Collaborative space
85	Research Park	Research Park 3	303	Laboratory complex
86	Research Park	Research Park 4	404	Startup Headquarters

Figure 60: Changes Reflected on Terminal

## Contributions

### Group 1

- **Pulkit Gautam:** Led Group 1, designed the overall structure of the frontend. Helped with integration of key features. Also, contributed to the documentation.
- **Manav Parmar:** Responsible for designing the frontend for the Request Page and Worker Page. Also, contributed in the documentation.
- **Gaurav Rawat:** Responsible for designing the forms and helped with the documentations.
- **Hrishikesh Birje:** Responsible for designing User and Domain Page.

### Group 2

- **Shubh Agarwal:** Led Group 2, designed the overall structure of the backend, worked on integrating the frontend with the backend, as well as integrating Flask with MySQL. Also responsible for writing APIs for the Login and User Table.
- **Atal Gupta:** Responsible for writing APIs for the Request and Worker Tables. Worked on sending and displaying images from the backend to the frontend. Also contributed significantly to the documentation.
- **Ishva Patel:** Wrote APIs for the Location Table and assisted in integrating MySQL with Flask. Also provided support for documentation tasks.
- **Aman Singh:** Developed APIs for the Domain Table and assisted in integrating MySQL with Flask. Also played a role in documentation efforts.