 Port Management Systems

**MINI PROJECT REPORT**

***Submitted By***

**AARTHI.M**

**(2303737724422037)**

# 60 CB 3P1- Database Management Systems Lab

***in partial fulfillment of the requirement for the award of the degree***

***of***

# BACHELOR OF TECHNOLOGY

*in*

# COMPUTER SCIENCE AND BUSINESS SYSTEMS

**K.S. RANGASAMY COLLEGE OF TECHNOLOGY**

(Autonomous)

**TIRUCHENGODE – 637 215**

**NOVEMBER 2024**

**K.S. RANGASAMY COLLEGE OF TECHNOLOGY**

**TIRUCHENGODE - 637 215**

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“PORT MANAGEMENT SYSTEMS”** is the bonafide work of **AARTHI M (2303737724422037)** who carried out the project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form part of any other project report or dissertation based on which a degree or award was conferred on an earlier occasion on this or any other candidate.

|  |  |
| --- | --- |
| **SIGNATURE**  Dr. M. TAMILARASI M.E., Ph.D.,  **FACULTY IN CHARGE**  Associate Professor  Department of Computer Science and Business Systems  K.S. Rangasamy College of Technology  Tiruchengode - 637 215 | **SIGNATURE**  Dr. K. SAKTHIVEL M.E., Ph.D.,  **HEAD OF THE DEPARTMENT**  Professor  Department of Computer Science and Business Systems  K.S. Rangasamy College of Technology  Tiruchengode - 637 215 |

Submitted for the viva-voce examination held on ……………………..…………

**Internal Examiner - 1 Internal Examiner - 2**

**DECLARATION**

I declare that the project report on **“PORT MANAGEMENT SYSTEMS”** is the result of original work done by me and best of my knowledge, similar work has not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of Bachelor of Technology. This project report is submitted on the partial fulfillment of the requirement of the award of Degree of Bachelor of Technology.

# SIGNATURE

AARTHI M

Place : Tiruchengode Date :

# ACKNOWLEDGEMENT

I wish to express my sincere gratitude to our honorable Chairman **Thiru. R. SRINIVASAN, B.B.M., MISTE.,** for providing immense facilities at our institution

I am proudly rendering my thanks to our Principal **, R. GOPALAKRISHNAN, M.E., Ph.D.,** for the facilities and the encouragement given by him to the progress and completion of my project.

I proudly render my immense gratitude to the Head of the Department **Dr. K. SAKTHIVEL M.E., Ph.D.,** for his effective leadership, encouragement, and guidance in the project. Your insightful feedback and expertise have been instrumental in refining my approach and methodology.

I am highly indebted to provide my heartful thanks to our course handler **Dr. M. TAMILARASI M.E., Ph.D.,** Associate Professor for her valuable ideas, encouragement, and supportive guidance throughout the project.

I would also like to extend my appreciation to all those who have directly or indirectly contributed to this project. Your contributions have been invaluable in making this endeavor a successful one.

# ABSTRACT

The **Port Management System** is a web-based application designed to streamline port operations by managing vessel records, cargo details, dock assignments, employee information, and billing processes. Developed using Flask and MySQL, it offers a responsive and user-friendly interface built with HTML, CSS, and Bootstrap. The system supports essential CRUD operations, enabling efficient management of records and providing real-time updates on dock and vessel availability. Key features include a search functionality by vessel ID or cargo type and confirmation prompts for critical actions to ensure data integrity. Optimized for secure and reliable data handling, this application centralizes port operations, enhancing efficiency and scalability for ports of all sizes.

4o

# TABLE OF CONTENTS

|  |  |  |  |
| --- | --- | --- | --- |
| **CHAPTER** | | **TITLE** | **PAGE NO.** |
| **ABSTRACT** | |  |  |
| **1 INTRODUCTION** | |  | 1 |
| **2 TECHNOLOGIES** | |  |  |
| 2.1 | INTRODUCTION |  | 2 |
| 2.2 | HTML |  | 2 |
| 2.3 | CSS |  | 2 |
| 2.4 | SQL |  | 3 |
| 2.5 | PYTHON FLASK |  | 3 |
| **3 DESIGN AND IMPLEMENTATION** | |  |  |
| 3.1 | SYSTEM DESIGN |  | 4 |
| 3.2 | SYSTEM IMPLEMENTATION |  | 5 |

|  |  |  |  |
| --- | --- | --- | --- |
| **4** | **RESULT** |  | 7 |
| **5** | **CONCLUSION** |  | 10 |
| **6** | **REFERENCE** |  | 11 |
| **6** | **APPENDIX** |  | 13 |

**CHAPTER 1**

**INTRODUCTION**

The **Port Management Systems** is a comprehensive web-based application designed to simplify and streamline the management of port operations. Built using the Flask framework and MySQL as the database backend, the system provides a robust, scalable, and user-friendly platform for efficiently managing vessel records, cargo details, dock assignments, employee information, and billing processes. This application caters to port administrators and staff, enabling seamless handling of daily operations through its intuitive interface and powerful functionality.

The system supports complete CRUD operations, allowing users to add, view, update, search, and delete records effortlessly. With an aesthetically designed frontend using HTML, CSS, and Bootstrap, the application ensures a visually appealing and responsive interface, offering smooth navigation and accessibility across devices. The dashboard provides an organized view of critical port data, including vessel schedules, dock availability, and cargo details, with options for quick edits and real-time updates.

A key feature of the system is its advanced search functionality, allowing port staff to quickly locate specific records based on vessel ID, cargo type, or dock number. Real-time updates ensure that changes to vessel schedules, dock assignments, and other operational data are immediately reflected, maintaining accuracy and up-to-date information across the system.

Security and scalability are central to the design of the Port Management System. Sensitive data, such as cargo details and billing information, is securely managed using MySQL for reliable storage and retrieval. Flask's lightweight framework ensures the application can efficiently handle an increasing number of users and records, making it adaptable to ports of varying sizes and capacities.

This system not only optimizes operational efficiency but also lays the groundwork for future enhancements, such as integrating analytics for business insights or automating resource allocation. By prioritizing functionality, user experience, and security, the Port Management System serves as a vital tool for creating an organized, efficient, and future-ready port environment.

# CHAPTER 2

# TECHNOLOGIES

* 1. **INTRODUCTION**

The **Port Management Systems** is a comprehensive web-based application designed to simplify and streamline the management of port operations. Built using the Flask framework and MySQL as the database backend, the system provides a robust, scalable, and user-friendly platform for efficiently managing vessel records, cargo details, dock assignments, employee information, and billing processes. This application caters to port administrators and staff, enabling seamless handling of daily operations through its intuitive interface and powerful functionality.

# 2.1 HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages. It structures the web content and provides the foundation for the layout of a webpage. HTML elements are the building blocks of web pages, and they can be enhanced by technologies such as CSS and JavaScript to create a richer user experience. Web browsers fetch HTML documents from a web server or local storage and render them into interactive multimedia web pages.

* 1. **CSS**

Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in HTML or XML. CSS is essential for controlling the layout, colors, fonts, and overall visual aesthetics of a web page. In the Freelancer Connect web application, CSS is employed to ensure a consistent and appealing look across all devices. By separating the content structure (HTML) from the visual presentation (CSS), developers can maintain a clean and efficient codebase, making the website more accessible and easier to manage.

## MYSQL

MySQL plays a crucial role in the Port **Record Management System** by storing and managing student data. It handles operations such as retrieving, inserting, updating, and deleting student records. MySQL ensures data integrity, efficient querying, and secure storage, enabling seamless interaction between the back-end application and the database.

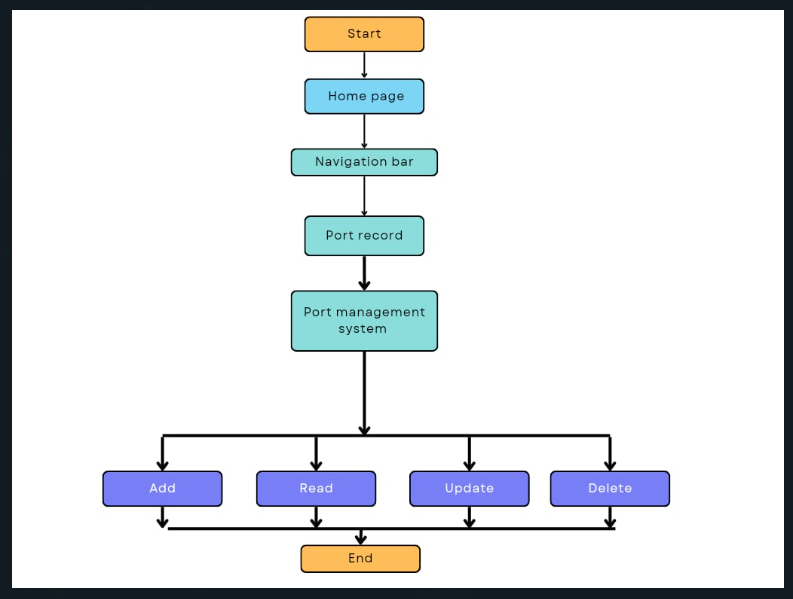
## PYTHON FLASK

The Port management systems are built using Python and the Flask framework, which offers a lightweight yet powerful platform for developing web applications. Flask's modular design and ease of use make it ideal for creating dynamic features such as user authentication, session management, and role-based access control. The application utilizes Flask's routing capabilities to define clear pathways for user and administrator interactions, while its templating engine enables the creation of responsive and user-friendly interfaces. Flask extensions like FlaskMySQLdb integrate seamlessly to facilitate database operations, ensuring efficient interaction with the MySQL backend. Additionally, Flask's built-in support for security features, such as session handling and password hashing, enhances the application's reliability and protects sensitive data. This combination of Python's versatility and Flask's simplicity provides a robust foundation for scalable and secure web application development.

# CHAPTER 3

**DESIGN AND IMPLEMENTATION**

### SYSTEM DESIGN

****

**Fig. 3.1 SYSTEM FLOW DIAGRAM**

### SYSTEM IMPLEMENTATION

The implementation of the Port Management System involves both the front-end and back-end components working together to provide a seamless experience. The back-end is developed using Flask, a lightweight Python web framework, while the front-end is built with HTML, CSS, and Bootstrap to ensure a responsive and user-friendly design. MySQL is used for storing and managing data related to room information, such as room IDs, types, prices, and availability. The system supports efficient data handling and retrieval, allowing users to perform CRUD operations—adding, viewing, updating, and deleting room records—through an intuitive web interface. This integration of front-end and back-end technologies ensures smooth operation and effective management of hotel room data.

**1. Backend Implementation:**

The backend of the Port Management System is built using Flask, which facilitates routing and interaction with the MySQL database. The key routes for the backend are:

* **Home Route (/):** This route renders the main index page, which could feature a navigation bar with links to other pages, such as vessel listings and port management options.
* **Vessels Route:** Fetches all vessel records (including vessel IDs, cargo types, dock assignments, and schedules) from the MySQL database and displays them in a table or card format on the front-end for users to view and manage port operations.
* **Search Route:** Allows users to search for vessels based on parameters such as vessel ID, cargo type, or dock number. It accepts input from the user and queries the database to find matching records.
* **Insert Route:** Facilitates the addition of new vessel or cargo records into the database. The user fills out a form with details (e.g., vessel ID, cargo type, dock number, schedule), and upon submission, the data is inserted into the MySQL database.
* **Edit Route:** Displays a form pre-populated with existing details, allowing users to update vessel or dock information. This route fetches the current record based on the vessel ID, and after submission, updates the data in the database.
* **Delete Route:** Deletes a specific vessel or cargo record from the MySQL database based on the vessel ID when selected by the user.

These routes allow users to interact with the port's operational data through a simple web interface, providing full CRUD (Create, Read, Update, Delete) functionality for port management. The backend leverages Flask to handle HTTP requests and MySQL to store and manage the port data efficiently.

The **Port Management System** uses **MySQL** to store and manage essential port-related data. Key tables include:

* **Ships:** Stores ship details such as ship ID, name, type, capacity, and current status.
* **DockingSlots:** Contains information about docking slots, including slot ID, location, availability status, and size compatibility.
* **Bookings:** Records docking reservations, linking ships to their assigned slots with booking dates and durations.
* **Staff:** Manages staff details, including employee ID, name, position, and assigned responsibilities.

The database structure supports efficient CRUD operations, ensures data integrity through foreign keys (e.g., linking ships to bookings and docking slots), and facilitates fast retrieval of records. This setup enables real-time management of docking availability, ship bookings, and staff assignments, ensuring smooth port operations and optimized resource utilization.

**3. Front-end Implementation:**

The front-end of the **Port Management System** is responsive and user-friendly, built using **Bootstrap** for consistency.

#### ****Key Components:****

**Navbar:**

* Includes links to **Home**, **Sign-Up**, **Login**, and **Ports** for easy navigation.

**Port Management Table:**

* Displays docking slot data (e.g., slot ID, availability) with options to **edit** or **delete** records.

**Forms:**

* **Add New Docking Slot:** For creating new slots by submitting data via **POST** requests.
* **Edit Docking Slot:** Pre-filled forms to update slot details.

This setup ensures smooth interaction with the backend for real-time port and docking slot management.

**4. Integration:**

The front-end and back-end are integrated such that the Flask application communicates with the MySQL database, allowing real-time updates to the student records displayed on the front-end. Each operation (view, add, edit, delete) is handled through HTTP requests, which are processed by Flask routes that query and manipulate the database accordingly.

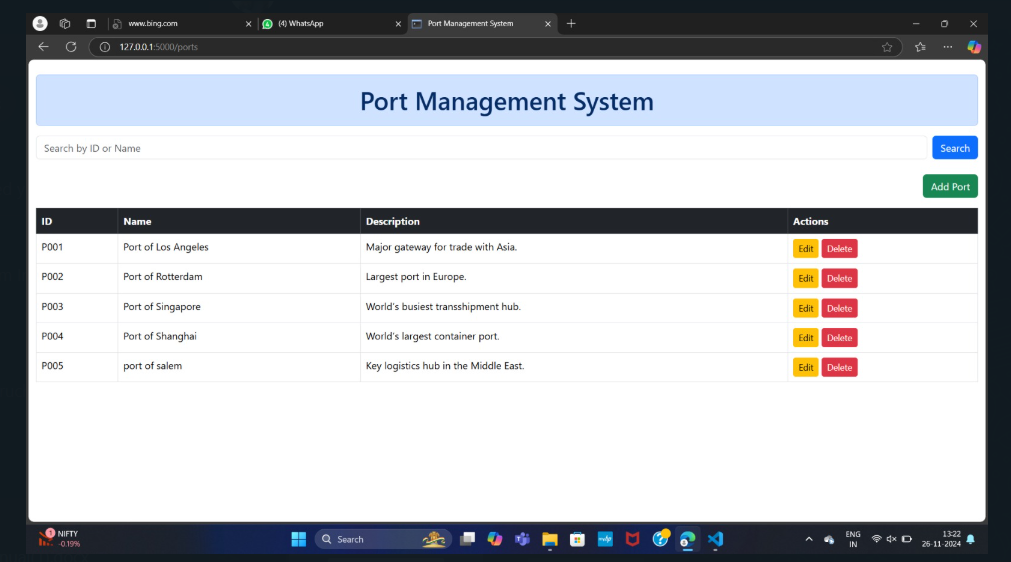
**5. Security Considerations:**

To prevent SQL injection attacks, all user inputs are properly sanitized and parameterized queries are used for interacting with the MySQL database. Additionally, the application uses Flask’s built-in session management and secure cookies to maintain user sessions. In summary, the implementation of this system demonstrates a solid understanding of web development principles, database management, and the integration of front-end and back-end technologies .

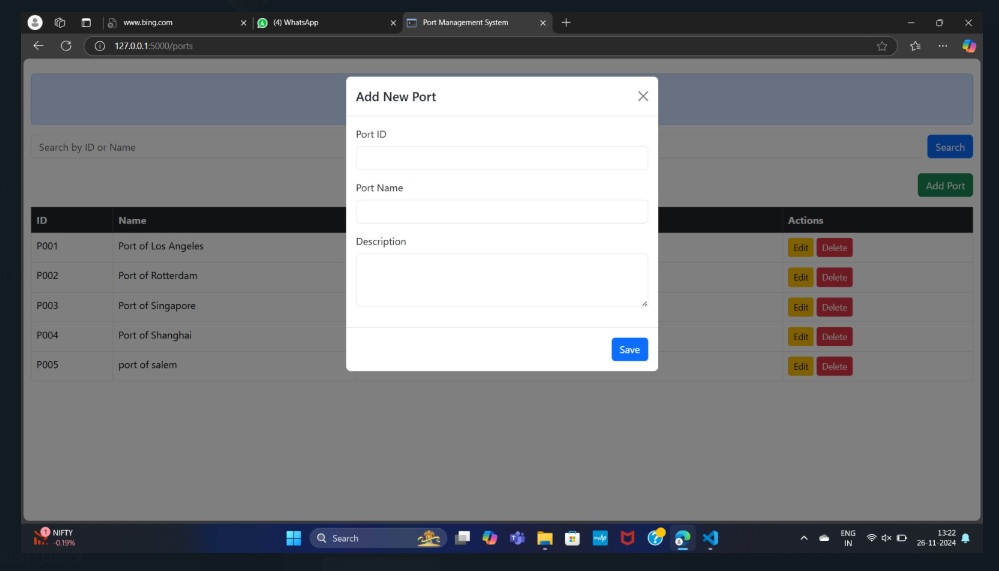
# HOME PAGE

# {9BACEDCD-832E-4476-941A-601ACA1FEA81}

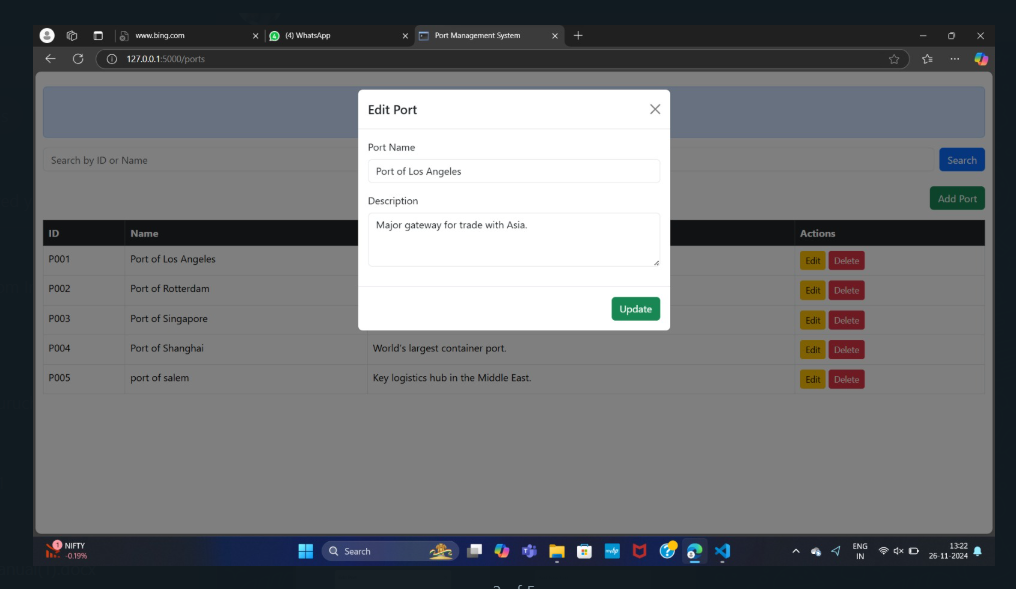
**DASHBOARD**



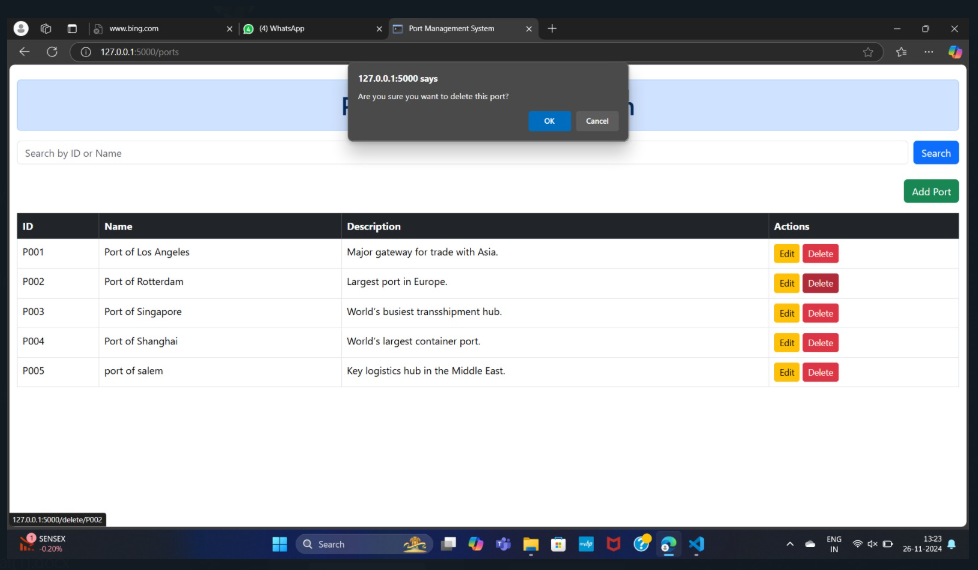
**ADD PRODUCT**

****

**EDIT PRODUCT**

****

**DELETE**



# CHAPTER 5

# CONCLUSION

The **Port Management System** project exemplifies the practical implementation of a web-based application for efficiently managing port operations and docking slot data. This system enables users to perform essential operations such as adding, updating, deleting, and searching docking slot records through an intuitive interface powered by **Flask** and **MySQL**. The **Flask** framework handles backend logic, ensuring seamless interaction with the **MySQL** database for secure storage and management of slot details and bookings.The front-end, designed with **HTML**, **Bootstrap**, and **CSS**, ensures a responsive design and user-friendly navigation. With features like search functionality, CRUD operations (Create, Read, Update, Delete), and a well-structured docking slot records table, this system provides a robust solution for managing port activities efficiently.Overall, the project demonstrates the effective integration of web development and database management skills to create a dynamic and functional application tailored for real-world port management needs.

# CHAPTER 6

# REFERENCES

1. <https://github.com/mtamilarasi589/EmpIoyeeManagement-FlaskMysqlProject>
2. <https://www.w3schools.com/html/html_css.asp>
3. <https://flask.palletsprojects.com/en/stable/>
4. https://www.mysql.com/
5. https://getbootstrap.com/docs/5.3/getting-started/introduction/

# APPENDIX 1

# SOURCE CODE

### FRONTEND

#### Template:

**Index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">

<title>Port Management System</title>

</head>

<body>

<header>

<nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top">

<div class="container-fluid">

<a class="navbar-brand" href="">Port Management System</a>

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNav">

<ul class="navbar-nav ms-auto">

<li class="nav-item">

<a class="nav-link active" aria-current="page" href="">Home</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#">Sign Up</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#">Login</a>

</li>

<li class="nav-item">

<a class="nav-link" href="{{ url\_for('ports') }}">Ports</a>

</li>

</ul>

</div>

</div>

</nav>

</header>

<div>

<img class="bkimg" src="{{ url\_for('static', filename='images/port.jpg') }}" style="width: 100%; height: auto;">

</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN6jIeHz" crossorigin="anonymous"></script>

</body>

</html>

**PORT.HTML**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Port Management System</title>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" crossorigin="anonymous">

</head>

<body>

<div class="container-fluid">

<h1 class="text-center alert alert-primary mt-4">Port Management System</h1>

<!-- Search Form -->

<div class="row mb-4">

<div class="col-12">

<form action="{{ url\_for('ports') }}" method="POST" class="d-flex">

<input type="text" class="form-control me-2" name="portid" placeholder="Search by ID or Name">

<button type="submit" class="btn btn-primary">Search</button>

</form>

</div>

</div>

<div class="row">

<div class="col-12 text-end">

<button class="btn btn-success mb-3" data-bs-toggle="modal" data-bs-target="#addPortModal">Add Port</button>

</div>

</div>

<!-- Add New Port Modal -->

<div class="modal fade" id="addPortModal" tabindex="-1" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title">Add New Port</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<form action="{{ url\_for('insert') }}" method="POST">

<div class="modal-body">

<div class="mb-3">

<label for="portid" class="form-label">Port ID</label>

<input type="text" class="form-control" name="portid" required>

</div>

<div class="mb-3">

<label for="name" class="form-label">Port Name</label>

<input type="text" class="form-control" name="name" required>

</div>

<div class="mb-3">

<label for="description" class="form-label">Description</label>

<textarea class="form-control" name="description" rows="3" required></textarea>

</div>

</div>

<div class="modal-footer">

<button type="submit" class="btn btn-primary">Save</button>

</div>

</form>

</div>

</div>

</div>

<!-- Port List Table -->

<table class="table table-bordered">

<thead class="table-dark">

<tr>

<th>ID</th>

<th>Name</th>

<th>Description</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

{% for row in ports %}

<tr>

<td>{{ row[0] }}</td>

<td>{{ row[1] }}</td>

<td>{{ row[2] }}</td>

<td>

<button class="btn btn-warning btn-sm" data-bs-toggle="modal" data-bs-target="#editPortModal{{ row[0] }}">Edit</button>

<a href="{{ url\_for('delete', portid=row[0]) }}" onclick="return confirm('Are you sure you want to delete this port?');" class="btn btn-danger btn-sm">Delete</a>

</td>

</tr>

<!-- Edit Port Modal -->

<div class="modal fade" id="editPortModal{{ row[0] }}" tabindex="-1" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title">Edit Port</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<form action="{{ url\_for('update') }}" method="POST">

<div class="modal-body">

<input type="hidden" name="portid" value="{{ row[0] }}">

<div class="mb-3">

<label for="name" class="form-label">Port Name</label>

<input type="text" class="form-control" name="name" value="{{ row[1] }}" required>

</div>

<div class="mb-3">

<label for="description" class="form-label">Description</label>

<textarea class="form-control" name="description" rows="3" required>{{ row[2] }}</textarea>

</div>

</div>

<div class="modal-footer">

<button type="submit" class="btn btn-success">Update</button>

</div>

</form>

</div>

</div>

</div>

{% endfor %}

</tbody>

</table>

</div>

<scriptsrc="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js" crossorigin="anonymous"></script>

</body>

</html>

**BACKEND**

**APP.PY**

from flask import Flask, render\_template, redirect, request, url\_for

from flask\_mysqldb import MySQL

app = Flask(\_name\_)

app.secret\_key = 'secure\_key'

# Database Configuration

app.config['MYSQL\_HOST'] = 'localhost'

app.config['MYSQL\_USER'] = 'root'

app.config['MYSQL\_PASSWORD'] = 'root'

app.config['MYSQL\_DB'] = 'portdb'

mysql = MySQL(app)

@app.route('/')

def index():

return render\_template('index.html')

@app.route('/ports', methods=['GET', 'POST'])

def ports():

search\_results = []

if request.method == 'POST':

search\_term = request.form['portid']

cur = mysql.connection.cursor()

query = "SELECT \* FROM portdetails WHERE id LIKE %s OR name LIKE %s"

cur.execute(query, ('%' + search\_term + '%', '%' + search\_term + '%'))

search\_results = cur.fetchall()

cur.close()

else:

cur = mysql.connection.cursor()

cur.execute("SELECT \* FROM portdetails")

search\_results = cur.fetchall()

cur.close()

return render\_template('ports.html', ports=search\_results)

@app.route('/insert', methods=['POST'])

def insert():

if request.method == "POST":

portid = request.form['portid']

name = request.form['name']

description = request.form['description']

cur = mysql.connection.cursor()

cur.execute("INSERT INTO portdetails (id, name, description) VALUES (%s, %s, %s)", (portid, name, description))

mysql.connection.commit()

return redirect(url\_for('ports'))

@app.route('/delete/<string:portid>', methods=['GET'])

def delete(portid):

cur = mysql.connection.cursor()

cur.execute("DELETE FROM portdetails WHERE id = %s", (portid,))

mysql.connection.commit()

return redirect(url\_for('ports'))

@app.route('/update', methods=['POST'])

def update():

if request.method == 'POST':

portid = request.form['portid']

name = request.form['name']

description = request.form['description']

cur = mysql.connection.cursor()

cur.execute("UPDATE portdetails SET name = %s, description = %s WHERE id = %s", (name, description, portid))

mysql.connection.commit()

return redirect(url\_for('ports'))

if \_name\_ == "\_main\_":

app.run(debug=True)

**db.sql**

CREATE DATABASE portdb;

USE portdb;

CREATE TABLE portdetails (

id VARCHAR(10) PRIMARY KEY,

name VARCHAR(100) NOT NULL,

description VARCHAR(255)

);

INSERT INTO portdetails (id, name, description)

VALUES

('P001', 'Port of Los Angeles', 'Major gateway for trade with Asia.'),

('P002', 'Port of Rotterdam', 'Largest port in Europe.'),

('P003', 'Port of Singapore', 'World’s busiest transshipment hub.'),

('P004', 'Port of Shanghai', 'World’s largest container port.'),

('P005', 'Port of Dubai', 'Key logistics hub in the Middle East.');

select \* from portdetails;

**Static\styles.css**

.bkimg {

background-size: cover;

background-repeat: no-repeat;

height: 900px;

}

# APPENDIX 2

# PROJECT LINKS

### GOOGLE DRIVE LINK:

https://drive.google.com/file/d/1Pg--PWcxwh\_RgQpX9Bev13pScQfiMvx2/view?usp=drivesdk

### GITHUB LINK:

https://github.com/Devadharshinimahendran/DBMS-MINI-PROJECT.git