COLLEGE CODE : 9111

COLLEGE NAME : SRM Madurai College for Engineering and Technology

DEPARTMENT : B.Tech Information Technology

STUDENT NM-ID : 8415E44C1D9006181424F999A9D508CA

ROLL NO : 911123205045

DATE : 15-09-2025

Completed the project named as

Phase 3 – MVP Implementation

TECHNOLOGY PROJECT NAME :

IBM-FE-Employee Directory with Search

SUBMITTED BY,

NAME : RUJMAL M

MOBILE NO : 7904425103

Introduction

Phase 3 Overview

In Phase 2, we designed the solution by selecting the tech stack, creating the UI structure, API schema, data handling approach, and component/module diagram.
Now in **Phase 3**, we focus on implementing the **Minimum Viable Product (MVP)** of the project. The MVP is the first working version that connects the frontend, backend, and database to deliver core features such as employee search and filter.

The key tasks of Phase 3 are:

- Project Setup
- Core Features Implementation
- Data Storage (Local State / Database)
- Testing Core Features
- Version Control (GitHub)

Project Setup

Frontend Setup

- React.js initialized using `create-react-app`.
- Installed required packages: React Router, Axios, Bootstrap.
- Created components: `Navbar.js`, `Filter.js`, `EmployeeList.js`, `EmployeeCard.js`.

Backend Setup

- Node.js project initialized with `npm init`.
- Installed dependencies: `express`, `mongoose`, `cors`.
- Created REST API endpoints in `server.js`.

Database Setup

- Used **MongoDB Atlas** for cloud storage of employee details.
- Employee schema includes: ID, Name, Role, Department, Email, Phone.

Output

```
Success! Created employee-directory at C:\Users\student\employee-directory
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd employee-directory
  npm start

Happy hacking!
PS C:\Users\student>
```

```
PS C:\Users\student\employee-directory\backend> npm install express mongoose cors
>>

added 87 packages, and audited 88 packages in 7s

17 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\student\employee-directory\backend>
```

Core Features Implementation

Implemented Features

1. **Employee Search** – Search by employee name.
2. **Filter Employees** – Filter by department and role.
3. **Employee Listing** – Display employee details in cards.

React Example – EmployeeCard Component

```
// EmployeeCard.js
import React from "react";

function EmployeeCard({ employee }) {
  return (
    <div className="card">
      <h3>{employee.name}</h3>
      <p>Role: {employee.role}</p>
      <p>Department: {employee.department}</p>
      <p>Email: {employee.email}</p>
      <p>Phone: {employee.phone}</p>
    </div>
  );
}

export default EmployeeCard;
```

Output

John Doe

Role: Software Engineer

Department: IT

Email: john.doe@example.com

Phone: +91 9876543210

Data Storage (Local State / Database)

Local State (Frontend)

- React `useState` hook used to store employee data temporarily.
- API responses stored in state and rendered dynamically.

Database (Backend – MongoDB)

- Employee details stored in MongoDB collection.
- Data fetched using Node.js Express API.

Example Node.js API Code

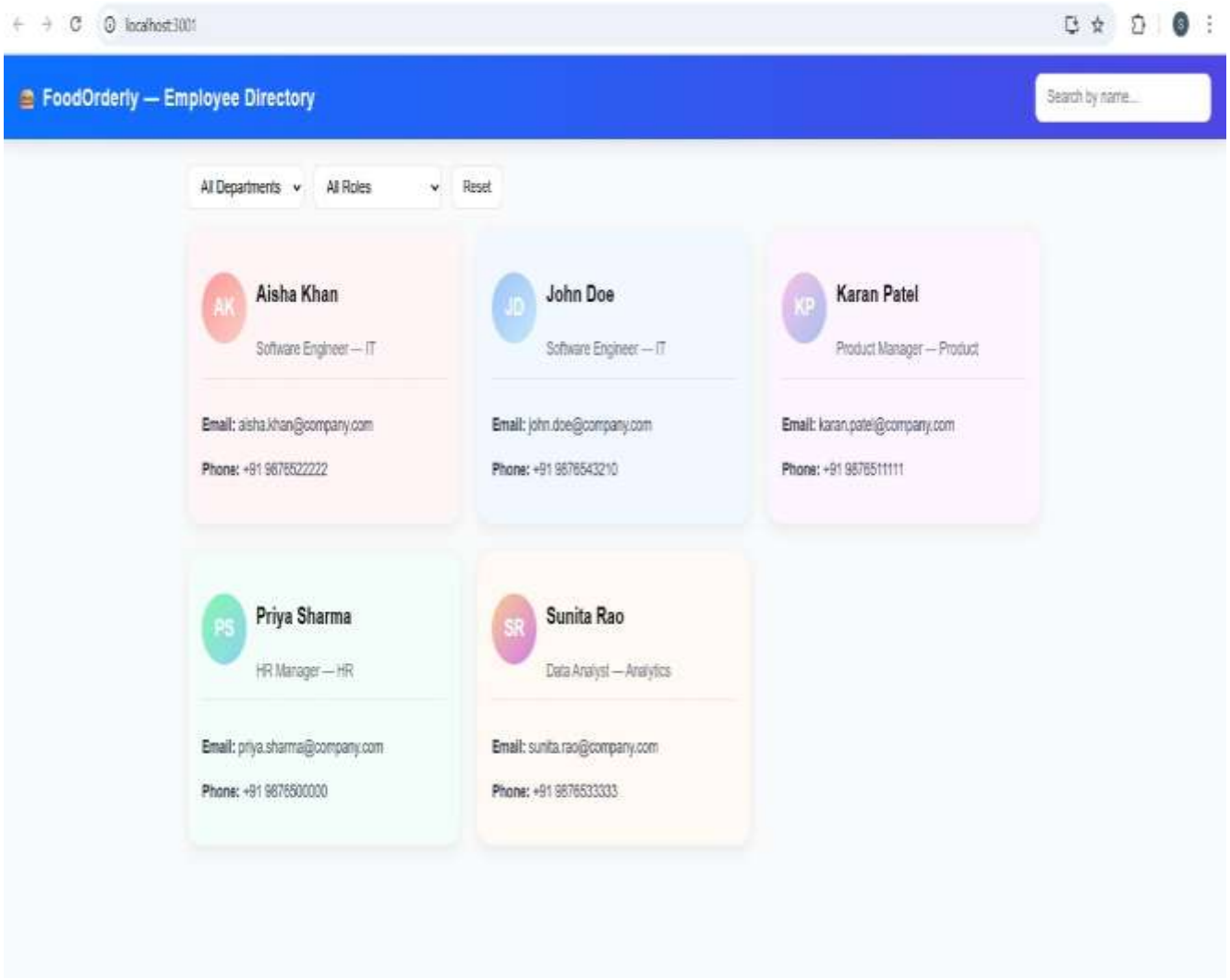
```
// server.js
const express = require("express");
const app = express();

const employees = [
  { id: "E101", name: "John Doe", role: "Software
Engineer", department: "IT", email:
"john.doe@company.com", phone: "+91 9876543210" },
  { id: "E102", name: "Priya Sharma", role: "HR
Manager", department: "HR", email:
"priya@company.com", phone: "+91 9123456780" }
];

app.get("/employees", (req, res) => {
  res.json(employees);
});

app.listen(5000, () => console.log("Server running on
port 5000"));
```

Output

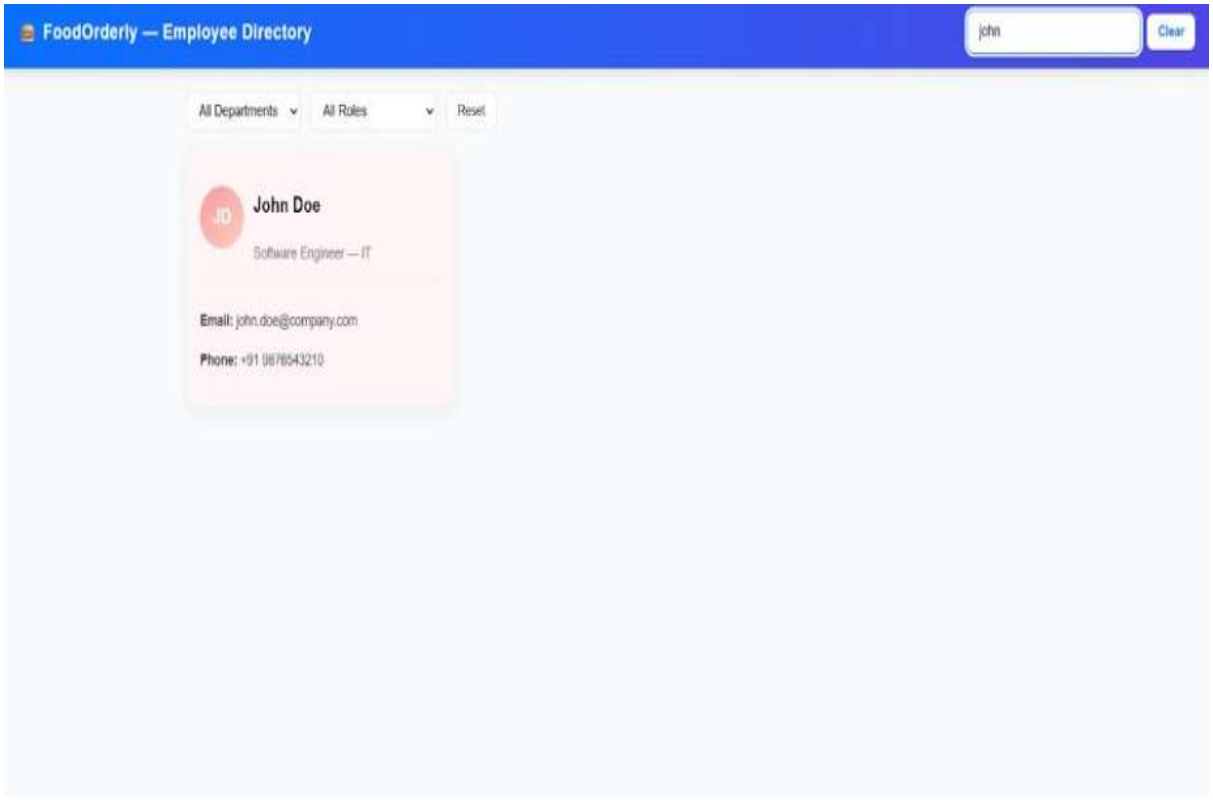

localhost:3001
FoodOrderly — Employee Directory
Search by name...
All Departments
All Roles
Reset
AK
Aisha Khan
Software Engineer — IT
Email: aisha.khan@company.com
Phone: +91 9876522222
JD
John Doe
Software Engineer — IT
Email: john.doe@company.com
Phone: +91 9876543210
KP
Karan Patel
Product Manager — Product
Email: karan.patel@company.com
Phone: +91 9876511111
PS
Priya Sharma
HR Manager — HR
Email: priya.sharma@company.com
Phone: +91 9876500000
SR
Sunita Rao
Data Analyst — Analytics
Email: sunita.rao@company.com
Phone: +91 9876533333

Testing Core Features

Test Cases

1. **Search Feature**
    - Input: “John”
    - Output: Employee card of John Doe displayed.
2. **Filter Feature**
    - Input: Department = IT
    - Output: Only IT employees displayed.
3. **API Test**
    - `/employees` endpoint returns JSON array of employees.

Output


FoodOrderly — Employee Directory
john
Clear
All Departments
All Roles
Reset
JD
John Doe
Software Engineer — IT
Email: john.doe@company.com
Phone: +91 9876543210

Version Control (GitHub)

- **GitHub Repository:** A centralized repository was created on GitHub to manage the entire project's codebase.

- **Version History:** All changes to the project are tracked through commits, ensuring that the complete development history is preserved.

- **Regular Commits:** Developers made frequent commits for frontend, backend, and database changes. Each commit was accompanied by meaningful commit messages for better understanding.

- **Branching Strategy:** Feature branches were created for individual modules or new functionalities. After testing, these branches were merged into the main branch to maintain stability

- **Issue Tracking:** GitHub Issues and Projects were utilized to assign tasks, track bugs, and manage progress.

Conclusion

- In **Phase 3**, the **MVP (Minimum Viable Product)** of the *Employee Directory with Search* application was successfully implemented.

- The **frontend (React.js)**, **backend (Node.js with Express.js)**, and **database (MongoDB Atlas)** were integrated and made to work together.

- Core features such as **search by name**, **filter by department/role**, and **employee list display** were designed, implemented, and tested with sample data.

- **Version Control using GitHub** ensured that the project development was systematic, with proper commit history and collaboration.

- The MVP provides a **basic but functional system**, which can now be enhanced in **Phase 4** with advanced features like authentication, role-based access, and UI improvements.