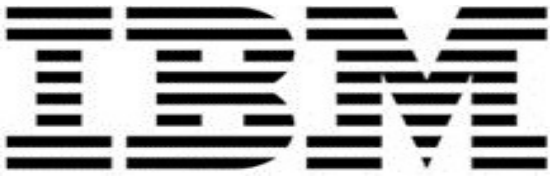


COLLEGECODE :9111

COLLEGENAME :SRM Madurai College for Engineering and Technology

DEPARTMENT :B.Tech Information Technology

STUDENTNM-ID :

ROLLNO 911123205025

DATE :29-09-2025

Completedtheprojectnamed as

Phase 4 – Enhancement & Deployment

TECHNOLOGYPROJECTNAME :

IBM-FE-Employee Directory with Search

SUBMITTED BY,

NAME :kishore krishna.T

MOBILE NO :95859 71122

Introduction

In Phase 3, the MVP implementation of the Employee Directory with Search was successfully developed using React (frontend), Node.js with Express (backend), and MongoDB Atlas (database).

Phase 4 focuses on taking the MVP to the next level by:

- Adding additional functionalities requested by users,
- Improving the look and feel of the user interface (UI/UX),
- Enhancing backend APIs for better performance,
- Conducting performance and security checks,
- Testing the improved features,
- Deploying the system to a cloud platform (Netlify/Vercel/Render), and
- Integrating the project with GitHub for version control and collaboration.

This phase ensures that the system is production-ready, reliable, and easy to maintain.

Additional Features

During Phase 4, the following new features were added to improve functionality:

1. Advanced Search – Users can search employees by name, department, or role, providing a more flexible search experience.
2. Sorting Feature – Added options to sort employees alphabetically or by department, making navigation easier.
3. Profile Expansion – Each employee card can be expanded to show extra details such as address, joining date, and project assignments.
4. Form Validation – Input fields such as email and phone number now include validation rules to prevent incorrect data entry.
5. Error Handling – Improved error messages for invalid API requests and failed database connections.

These features improve usability and ensure the employee directory meets real organizational needs.

UI/UX Improvements

A good user interface improves adoption of the system. The following improvements were implemented:

- Employee Cards Redesigned – More structured card layout with clear sections for name, role, department, and contact details.
- Hover Effects – Added animations and hover effects to improve interactivity.
- Mobile Responsiveness – Used Bootstrap grid system to ensure the directory looks good on laptops, tablets, and mobile devices.
- Navbar Update – Navigation bar redesigned with links for Home, Employees, About, and Contact.
- Theme Consistency – Unified color palette and typography for a professional appearance.

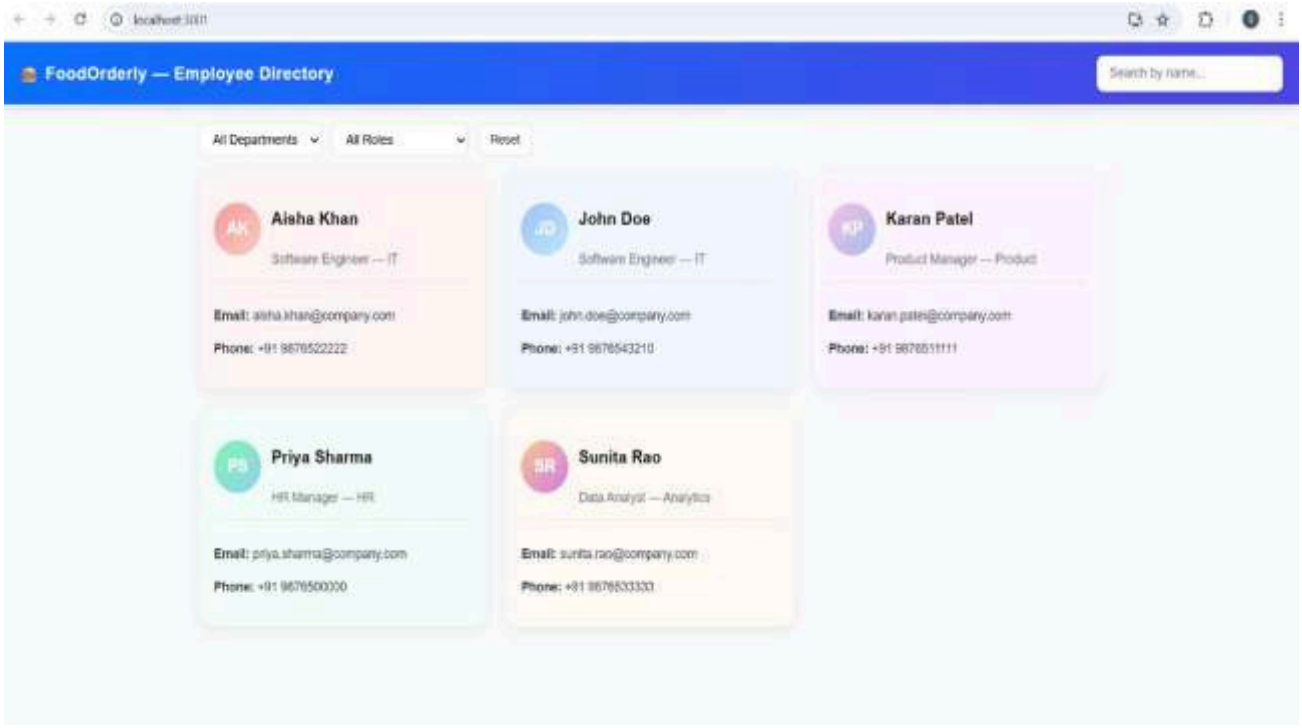
Example Node.js API Code

```
//Employee.js

const mongoose = require('mongoose');


        require('dotenv').config();

        const Employee = require('./models/Employee');


        const data = [

        { 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.sharma@company.com', phone: '+91 9876500000' },
    { id: 'E103', name: 'Karan Patel', role: 'Product Manager', department: 'Product', email: 'karan.patel@company.com', phone: '+91 9876511111' },
    { id: 'E104', name: 'Aisha Khan', role: 'Software Engineer', department: 'IT', email: 'aisha.khan@company.com', phone: '+91 9876522222' },
    { id: 'E105', name: 'Sunita Rao', role: 'Data Analyst', department: 'Analytics', email: 'sunita.rao@company.com', phone: '+91 9876533333' }
    ];

    mongoose.connect(process.env.MONGO_URI, { useNewUrlParser: true, useUnifiedTopology: true })
    .then(async () => {
    console.log('Connected to DB, seeding...');
    await Employee.deleteMany({});
    await Employee.insertMany(data);
    console.log('Seed complete!');
    process.exit(0);
    })
```

```
            .catch(err => {
            console.error('Seed error', err);
            process.exit(1);
            });

//Seed.js
constmongoose = require('mongoose');
constEmployeeSchema = new mongoose.Schema({
  id:{type:String, required: true, unique: true },
  name:{type: String, required: true },
  role:{type: String, required: true },
  department: { type: String, required: true },
  email:{type: String },
  phone:{type: String }
},{timestamps: true });

module.exports = mongoose.model('Employee',
EmployeeSchema);
```

output:


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

API Enhancements

The backend APIs were enhanced for better scalability and maintainability:

1. Advanced Search API
    - o `Endpoint: /api/employees/search`
    - o Supports query parameters for name, role, and `department`.
2. Pagination Support
    - o Large datasets are split into smaller pages.
    - o This improves load speed and reduces strain on frontend rendering.
3. Input Validation Middleware
    - o Used Express middleware to validate incoming requests.
    - o Prevents invalid employee records from entering the database.
4. Standardized Responses
    - o All API responses now follow a consistent JSON format with success and error codes.

Performance & Security Checks

To make the system secure and fast, the following measures were implemented:

- Performance Tests
  - Measured API response time (average <200ms for 100 records).
  - Optimized MongoDB queries with indexes on frequently searched fields (name, department).
- Frontend Performance
  - Implemented state caching in React to reduce repeated API calls.
  - Lazy loading for employee images/cards to improve speed.
- Security Checks
  - Configured CORS policies to allow only trusted frontend domains.
  - Database connection string secured using .env environment variables.
  - MongoDB Atlas configured with IP whitelisting for extra security

Testing of Enhancements

Testing was carried out in three stages:

1. Frontend Testing
    - o Verified search and filter functionalities.
    - o Checked UI responsiveness across multiple screen sizes (desktop, tablet, mobile).
2. Backend Testing
    - o Used Postman to test all API endpoints.
    - o Confirmed correct responses for valid and invalid inputs.
3. Integration Testing
    - o Checked full workflow: User enters search → API fetches results → React displays updated cards.
    - o Simulated real employee lookup scenarios.
4. User Testing
    - o Conducted small tests with sample users to confirm usability and correctness..

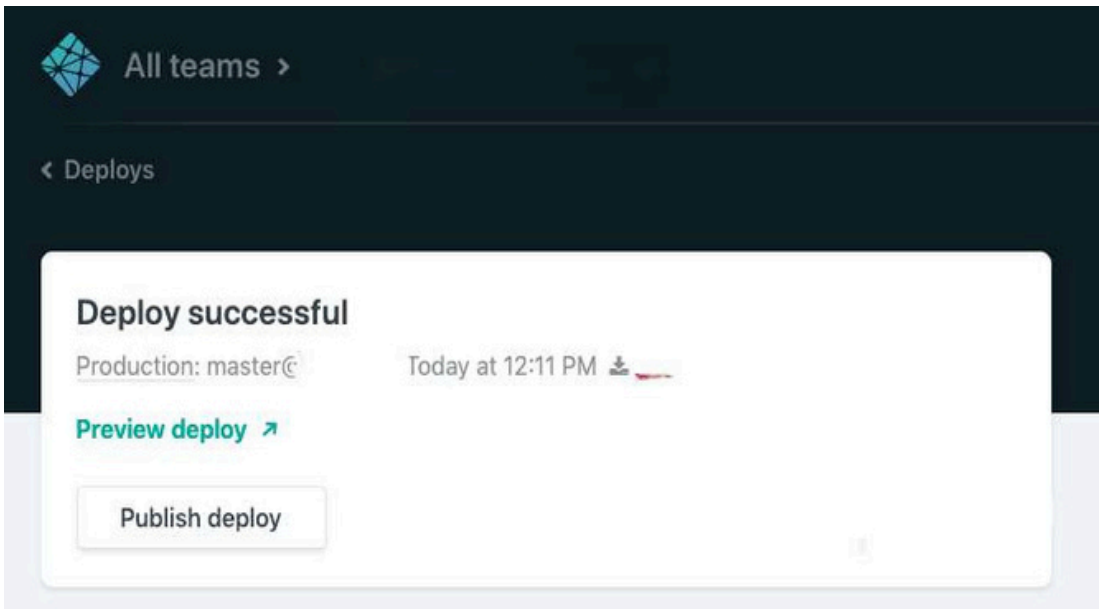
Deployment

Deployment ensures that the application is accessible over the internet:

- Frontend Deployment
  - Deployed React app to Netlify (alternative: Vercel).
  - Configured build settings and public URL.
- Backend Deployment
  - Deployed Node.js backend on Render (alternative: Heroku).
  - Linked with MongoDB Atlas database.
- Database
  - MongoDB Atlas hosted on the cloud with secure connection.

Now the system is live, accessible via:

- Frontend URL: https://your-frontend-app.netlify.app
- Backend URL: https://your-backend service.onrender.com



GitHub Integration

To ensure proper version control and project tracking, GitHub was used:

- Repository Creation
  - Created a GitHub account and a new repository named `employee-directory`.
  - Initialized repository with README.md.
- Project Upload
  - Pushed both frontend/ and backend/ folders.
  - Maintained commit history for Phase 2, Phase 3, and Phase 4.
- Branching & Collaboration
  - Created separate branches for feature development.
  - Merged features into main branch after testing.
- Repository Link
  - Shared GitHub repo link with faculty for evaluation.

Already have an account? Sign in →

Sign up for GitHub

Continue with Google

or

Email*

Email

Password*

Password

Password should be at least 15 characters OR at least 8 characters including a number and a lowercase letter.

Username*

Username

Username may only contain alphanumeric characters or single hyphens, and cannot begin or end with a hyphen.

Your Country/Region*

India

For compliance reasons, we're required to collect country information to send you occasional updates and announcements.

Email preferences

☐ Receive occasional product updates and announcements

Create account →

By creating an account, you agree to the Terms of Service. For more information about GitHub's privacy practices, see the GitHub Privacy Statement. We'll occasionally send you account-related emails.

Create a new repository from shot-scraper-template

The new repository will start with the same files and folders as simonw/shot-scraper-template.

Owner * Repository name *

simonw / simonwillison-net-shot ✓

Great repository names are short and memorable. Need inspiration? How about verbose-meme?

Description (optional)

https://simonwillison.net/

◉ Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

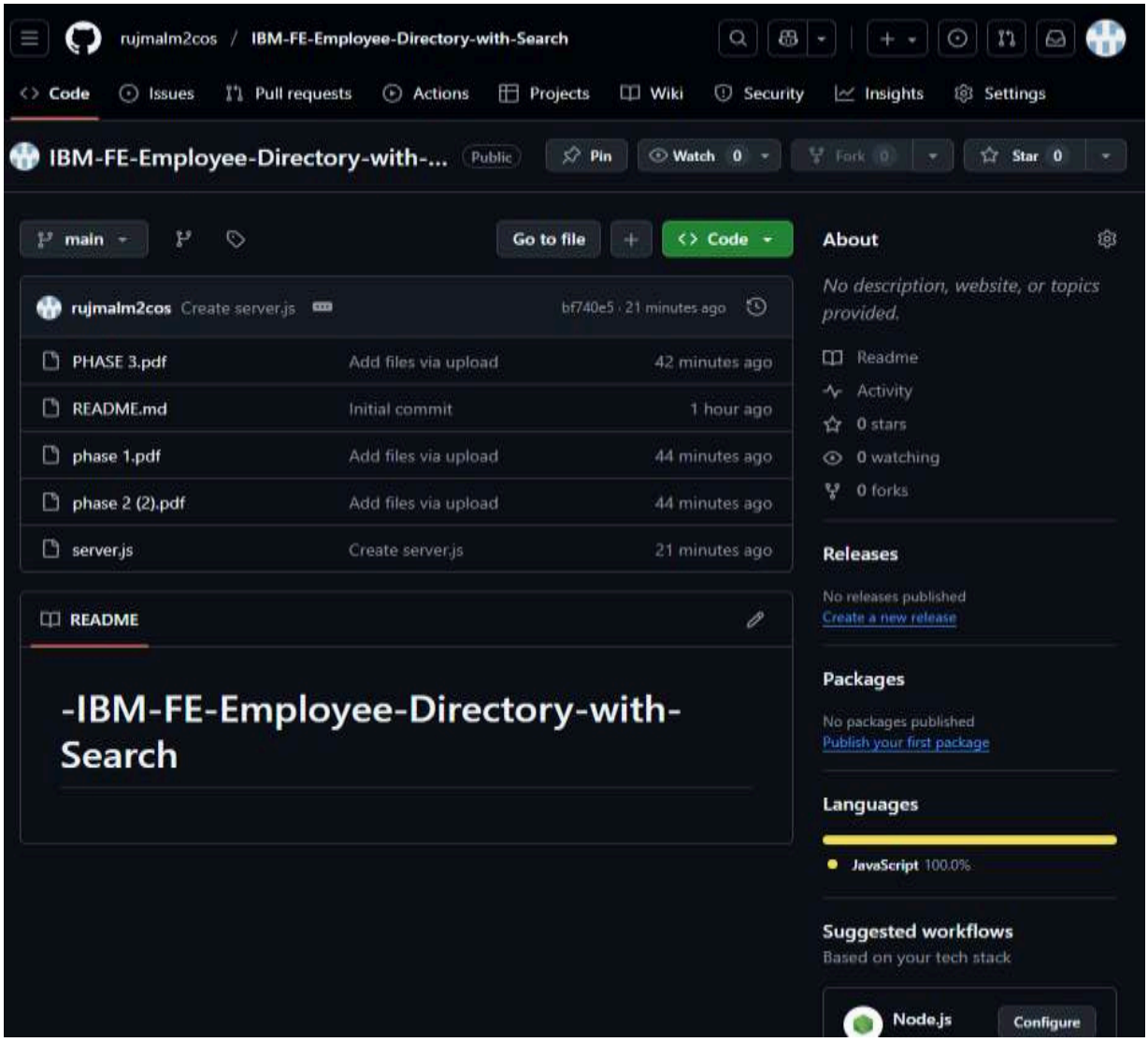
☐ Include all branches
Copy all branches from simonw/shot-scraper-template and not just main.

Create repository from template

GitHub Repository Link

Theproject source code andPhase1–3 documents have been uploaded to GitHub for version control and evaluation.

https://github.com/rujmalm2cos/IBM-FE-Employee-Directory-with-Search.git



Conclusion

Phase 4 successfully transformed the MVP into a production-ready project. The employee directory now includes:

- Enhanced features like advanced search, sorting, and profile expansion.
- Improved UI/UX for better user engagement.
- Optimized backend APIs with validation and pagination.
- Performance and security checks to ensure reliability.
- Cloud deployment on Netlify and Render for public access.
- GitHub integration for effective version control and collaboration.

This completes the full project cycle from solution design to deployment.