RAMAIAH
Institute of Technology

MANAGE CORP

Submitted to the

Department of Master of Computer Applications in partial fulfilment of the requirements for the Mini Project in

Full Stack Development – MCA26

by

Esha Patel

1MS24MC028

Under the Guidance of

Ms Komal S

Department of Master of Computer Applications

RAMAIAH INSTITUTE OF TECHNOLOGY

(Autonomous Institute, Affiliated to VTU)

Accredited by National Board of Accreditation & NAAC with 'A+' Grade

MSR Nagar, MSRIT Post, Bangalore-560054

www.msrit.edu

2025

RAMAIAH
Institute of Technology

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that the project entitled **“Manage Corp”** is carried out by **Esha Patel - 1MS24MC028** student of 2nd semester, in partial fulfillment for the Mini Project in Full Stack development – MCA26, during the academic year 2024-2025.

Ms Komal S	**Dr. S Ajitha**
Guide	**Head of the Department**

Name of Examiners	**Signature with Date**
1.	
2.	

ACKNOWLEDGEMENT

With deep sense of gratitude, I am thankful to our Respected HoD, **Dr. S Ajitha** for her pillared support and encouragement. I would like to convey my heartfelt thanks to my Project Guide **Ms Komal S**, Department of Master of Computer Applications, for giving me an opportunity to embark upon this topic and for her/his constant encouragement.

It gives me great pleasure to acknowledge the contribution of all people who helped me directly or indirectly realize it. First I would like to acknowledge the love and tremendous sacrifices that my Parents made to ensure that I always get an excellent education. For this and much more, I am forever in their debt.

I am thankful to all my friends for their moral and material support throughout the course of my project. Finally, I would like to thank all the Teaching and Non-Teaching Staff of Department of Master of Computer Applications for their assistance during various stages of my project work.

Esha Patel

ABSTRACT

Efficient management of employees, job postings, and administrative workflows is a critical need in today's organizational environments. To address this challenge, a full-stack web application has been created using Node.js, Express.js, MongoDB, and EJS. It enables role-specific access where employees can register, log in, and manage resumes; companies can post job openings and browse candidates; and administrators can oversee system-wide activity. The system offers a responsive, browser-based interface with secure session management using express-session and connect-mongo.

Within this platform, Manage Corp serves as the centralized solution that bridges the gap between workforce management and recruitment processes. Its modular codebase separates concerns through distinct route files and Mongoose data models for employees, companies, and admins. The application employs EJS templates to dynamically render views and incorporates a custom color palette to deliver a visually consistent and user-friendly experience. Admin access is protected with default credentials, allowing system-wide control over user activity and platform integrity.

Beyond basic CRUD operations, Manage Corp supports a scalable architecture that can be easily extended to meet future business requirements. Its design emphasizes usability, security, and maintainability, making it suitable for small to mid-sized enterprises seeking an all-in-one digital infrastructure for HR and recruitment tasks. The project demonstrates how modern web technologies can be combined to deliver real-world business solutions efficiently.

In addition to its technical strengths, the application also emphasizes user experience and accessibility. The responsive design ensures seamless usability across devices, from desktops to smartphones, making Manage Corp a practical solution for remote teams and on-the-go users. The intuitive navigation, clean interface, and clear role-based access help minimize the learning curve for first-time users. Whether accessed by HR personnel, job seekers, or administrators, the platform delivers a consistent and efficient workflow that supports real-world business operations in a digital-first environment.

Table of Contents

1. Introduction

1.1. Problem Definition

In today’s fast-paced digital economy, organizations face growing demands to manage employees, recruitment, and internal operations through streamlined, technology-driven platforms. Traditional human resource management systems often suffer from fragmentation, manual processes, or lack of integration across different user roles. This can lead to inefficiencies, data loss, and poor user experience. Businesses increasingly require centralized systems that support real-time data access, secure role-based interactions, and intuitive user interfaces.

To address these needs, Manage Corp has been developed as a modern, full-stack web application that brings together employees, companies, and administrators under a unified system. Built using Node.js, Express.js, MongoDB, and EJS, the application enables seamless interaction between different user types while maintaining a secure and scalable backend. Employees can manage personal profiles and resumes; companies can post jobs and filter applicants; and administrators can monitor and manage overall platform activity. The system emphasizes both functionality and user experience, offering a responsive UI styled with a custom color palette, modular codebase, and session-based access control.

In addition to its core features, Manage Corp is designed with scalability and maintainability in mind. The application's modular architecture—separating routes, models, and views—allows for easy updates and future enhancements without disrupting existing functionality. Technologies like express-session and connect-mongo ensure secure and persistent session handling, while Mongoose schemas enforce data consistency. Whether deployed in small businesses or scaled for larger organizations, the platform provides a solid foundation for managing human resources and recruitment digitally, with the flexibility to evolve alongside changing business requirements.

The system is easy to use and flexible, making it suitable for a wide range of organizations. With a strong focus on performance, security, and user-friendly design, *Manage Corp* helps solve common problems in HR and recruitment. It cuts down on manual work, improves communication between users, and keeps data safe and organized. Overall, the platform is a reliable and modern solution for managing employees and hiring processes effectively.

2. Implementation

2.1. Code snippets

➢ **Admin.js**

```
const express = require('express');
const router = express.Router();
const Admin = require('../models/Admin');
const Employee = require('../models/Employee');
const Company = require('../models/Company');
// Admin login
router.get('/login', (req, res) => {
  res.render('admin/login');
});
router.post('/login', async (req, res) => {
  const { username, password } = req.body;
  // Allow hardcoded default admin login
  if (username === 'admin' && password === 'admin') {
    req.session.admin = { username: 'admin' };
    return res.redirect('/admin/dashboard');
  }
  // Fallback to DB check for any other admin
  const admin = await Admin.findOne({ username, password });
  if (admin) {
    req.session.admin = admin;
    res.redirect('/admin/dashboard');
  } else {
    res.render('admin/login', { error: 'Invalid credentials' });
  }
});
// Admin dashboard
router.get('/dashboard', async (req, res) => {
  if (!req.session.admin) return res.redirect('/admin/login');
  const employees = await Employee.find();
  const companies = await Company.find();
  res.render('admin/dashboard', { employees, companies });
});
module.exports = router;
```

- **Company.js**

```
const express = require('express');
const router = express.Router();
const Company = require('../models/Company');
const Employee = require('../models/Employee');
// Registration form
router.get('/register', (req, res) => {
  res.render('company/register');
});
// Register company
router.post('/register', async (req, res) => {
  const { name, email, password, projects, achievements } = req.body;
  const company = {
    name,
    email,
    password,
    projects: projects ? projects.split('\n').map(p => p.trim()).filter(p => p) : [],
    achievements: achievements ? achievements.split('\n').map(a => a.trim()).filter(a => a) : []
  };
  await Company.create(company);
  res.redirect('/company/login');
});
// Edit company profile form
router.get('/edit', (req, res) => {
  if (!req.session.company) return res.redirect('/company/login');
  res.render('company/edit', { company: req.session.company });
});
// Handle company profile edit
router.post('/edit', async (req, res) => {
  if (!req.session.company) return res.redirect('/company/login');
  const { name, email, password, projects, achievements } = req.body;
  const update = {
    name,
    email
  };
  if (password && password.trim() !== '') {
    update.password = password;
  }
  update.projects = projects ? projects.split('\n').map(p => p.trim()).filter(p => p) : [];
  update.achievements = achievements ? achievements.split('\n').map(a => a.trim()).filter(a
=> a) : [];
  const company = await Company.findByIdAndUpdate(req.session.company._id, update,
{ new: true });
  req.session.company = company;
  res.redirect('/company/dashboard');
});
// Login form
router.get('/login', (req, res) => {
  res.render('company/login');
});
// Login logic
router.post('/login', async (req, res) => {
  const { email, password } = req.body;
  const company = await Company.findOne({ email, password });
  if (company) {
    req.session.company = company;
    res.redirect('/company/dashboard');
  } else {
    res.render('company/login', { error: 'Invalid credentials' });
  }
```

```
});
// Dashboard
router.get('/dashboard', (req, res) => {
  if (!req.session.company) return res.redirect('/company/login');
  res.render('company/dashboard', { company: req.session.company });
});
// Post job requirements
router.get('/post', (req, res) => {
  if (!req.session.company) return res.redirect('/company/login');
  res.render('company/post');
});
router.post('/post', async (req, res) => {
  if (!req.session.company) return res.redirect('/company/login');
  const { title, location, experience, skills, description } = req.body;
  await Company.findByIdAndUpdate(req.session.company._id, {
    jobPost: { title, location, experience, skills, description }
  });
  res.redirect('/company/search');
});
// Search for candidates
router.get('/search', async (req, res) => {
  if (!req.session.company) return res.redirect('/company/login');
  const company = await Company.findById(req.session.company._id);
  let query = {};
  if (company.jobPost && company.jobPost.skills) {
    query.skills = { $regex: company.jobPost.skills + '', $options: 'i' };
  }
  if (company.jobPost && company.jobPost.experience) {
    query.experience = { $regex: company.jobPost.experience + '', $options: 'i' };
  }
  const candidates = await Employee.find(query);
  res.render('company/search', { candidates });
});
module.exports = router;
```

- **Employee.js**

```
const express = require('express');
const router = express.Router();
const Employee = require('../models/Employee');
// Registration form
router.get('/register', (req, res) => {
  res.render('employee/register');
});
// Register employee
router.post('/register', async (req, res) => {
  const { name, email, password, phone, education, experience, skills, summary, projects,
achievements } = req.body;
  await Employee.create({
    name,
    email,
    password,
    phone,
    education,
    experience,
    skills,
    summary,
    projects: projects ? projects.split('\n').map(p => p.trim()).filter(p => p) : [],
    achievements: achievements ? achievements.split('\n').map(a => a.trim()).filter(a => a) : []
  });
  res.redirect('/employee/login');
});
// Login form
router.get('/login', (req, res) => {
  res.render('employee/login');
});
// Login logic
router.post('/login', async (req, res) => {
  const { email, password } = req.body;
  const user = await Employee.findOne({ email, password });
  if (user) {
    req.session.user = user;
    res.redirect('/employee/dashboard');
  } else {
    res.render('employee/login', { error: 'Invalid credentials' });
  }
});
router.get('/dashboard', (req, res) => {
  if (!req.session.user) return res.redirect('/employee/login');
  res.render('employee/dashboard', { user: req.session.user });
});
router.get('/resume', (req, res) => {
  if (!req.session.user) return res.redirect('/employee/login');
  res.render('employee/resume', { user: req.session.user });
});
router.get('/edit', (req, res) => {
  if (!req.session.user) return res.redirect('/employee/login');
  res.render('employee/edit', { user: req.session.user });
});
router.post('/edit', async (req, res) => {
  if (!req.session.user) return res.redirect('/employee/login');
  const { name, email, phone, education, experience, skills, summary, password, projects,
achievements } = req.body;
  const update = {
    name,
    email,
    phone,
    education,
    experience,
    skills,
    summary
  };
  if (password && password.trim() !== '') {
    update.password = password;
  }
  update.projects = projects ? projects.split('\n').map(p => p.trim()).filter(p => p) : [];
  update.achievements = achievements ? achievements.split('\n').map(a => a.trim()).filter(a
=> a) : [];
  const user = await Employee.findByIdAndUpdate(req.session.user._id, update, { new:
true });
  req.session.user = user;
  res.redirect('/employee/resume');
});
module.exports = router;
```

- **Main.js**

```
const express = require('express');
const router = express.Router();
router.get('/', (req, res) => {
  res.render('index');
});
module.exports = router;
```

- **Admin Dashboard**

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Admin Dashboard | Manage Corp</title>
  <link rel="stylesheet" href="/style.css">
</head>
<body>
  <header class="main-header">
    <div class="nav-bar">
      <span class="logo">Manage Corp</span>
      <nav>
        <a href="/">Home</a>
      </nav>
    </div>
  </header>
  <main class="container">
    <div class="portal-card" style="margin:2rem auto; max-width:600px;">
      <h2 style="text-align:center; margin-bottom:1.5rem;">Admin Dashboard</h2>
      <div style="display:flex; gap:2rem; flex-wrap:wrap; justify-content:center;">
        <section style="flex:1; min-width:220px;">
          <h3 style="color:#3949ab;">Employees</h3>
          <ul style="padding-left:1.2rem;">
            <% employees.forEach(function(emp) { %>
              <li><strong><%= emp.name %></strong> - <%= emp.email %></li>
            <% }) %>
          </ul>
        </section>
        <section style="flex:1; min-width:220px;">
          <h3 style="color:#3949ab;">Companies</h3>
          <ul style="padding-left:1.2rem;">
            <% companies.forEach(function(comp) { %>
              <li><strong><%= comp.name %></strong> - <%= comp.email %></li>
            <% }) %>
          </ul>
        </section>
      </div>
      <a href="/" class="portal-btn secondary" style="margin-top:2rem;">Back to Home</a>
    </div>
  </main>
    </div>
  </main>
</body>
</html>
```

➢ **Company Registration Page**

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Company Registration | Manage Corp</title>
  <link rel="stylesheet" href="/style.css">
</head>
<body>
  <header class="main-header">
    <div class="nav-bar">
      <span class="logo">Manage Corp</span>
      <nav>
        <a href="/">Home</a>
      </nav>
    </div>
  </header>
  <main class="container">
    <div class="portal-card" style="margin:2rem auto; max-width:400px;">
      <h2 style="text-align:center; margin-bottom:1rem;">Company Registration</h2>
      <% if (typeof error !== 'undefined') { %><div class="error"><%=
error %></div><% } %>
      <form method="POST" action="/company/register" style="display:flex; flex-
direction:column; gap:1rem;">
        <input name="name" placeholder="Company Name" required />
        <input name="email" placeholder="Email Address" type="email" required />
        <input name="password" type="password" placeholder="Password" required />
        <label>Projects (one per line):</label>
        <textarea name="projects" placeholder="Project 1\nProject 2" rows="3"></textarea>
        <label>Achievements (one per line):</label>
        <textarea name="achievements" placeholder="Achievement 1\nAchievement 2"
rows="3"></textarea>
        <button type="submit" class="portal-btn">Register</button>
      </form>
      <a href="/company/login" class="portal-btn secondary" style="margin-
top:1rem;">Already have an account? Login</a>
    </div>
  </main>
</body>
</html>
```

➢ **Employee Registration Page**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Employee Registration | Manage Corp</title>
  <link rel="stylesheet" href="/style.css">
</head>
<body>
```

```
  <header class="main-header">
    <div class="nav-bar">
      <span class="logo">Manage Corp</span>
      <nav>
        <a href="/">Home</a>
      </nav>
    </div>
  </header>
  <main class="container">
    <div class="portal-card" style="margin:2rem auto; max-width:400px;">
      <h2 style="text-align:center; margin-bottom:1rem;">Employee Registration</h2>
      <% if (typeof error !== 'undefined') { %><div class="error"><%=
error %></div><% } %>
      <form method="POST" action="/employee/register" style="display:flex; flex-
direction:column; gap:1rem;">
        <input name="name" placeholder="Full Name" required />
        <input name="email" placeholder="Email Address" type="email" required />
        <input name="password" type="password" placeholder="Password" required />
        <input name="phone" placeholder="Phone Number" required />
        <input name="education" placeholder="Education (e.g. BSc Computer Science)"
required />
        <input name="experience" placeholder="Experience (e.g. 2 years)" required />
        <input name="skills" placeholder="Skills (comma separated)" required />
        <textarea name="summary" placeholder="Short Professional Summary"
required></textarea>
```

```
        <label>Projects (one per line):</label>

        <textarea name="projects" placeholder="Project 1\nProject 2" rows="3"></textarea>

        <label>Achievements (one per line):</label>

        <textarea name="achievements" placeholder="Achievement 1\nAchievement 2"
rows="3"></textarea>

        <button type="submit" class="portal-btn">Register</button>

      </form>

      <a href="/employee/login" class="portal-btn secondary" style="margin-
top:1rem;">Already have an account? Login</a>

    </div>

  </main>

</body>

</html>
```

➢ **Index.ejs**

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

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

  <title>Welcome | Manage Corp</title>

  <link rel="stylesheet" href="/style.css">

</head>

<body>

  <header class="main-header">
```

```
    <div class="nav-bar">
      <span class="logo">Manage Corp</span>
      <nav>
        <a href="/">Home</a>
        <a href="/employee/login">Employee</a>
        <a href="/company/login">Company</a>
        <a href="/admin/login">Admin</a>
      </nav>
    </div>
  </header>
  <main class="container">
    <h1 style="text-align:center; margin-bottom:0.5rem;">Welcome to Manage Corp</h1>
      <div style="max-width:400px;margin:0 auto 1.5rem auto;">
      <div style="background:#393E46;border:1px solid #00ADB5;border-
radius:8px;padding:1rem 1.5rem;text-align:center;font-size:1rem;color:#EEEEEE;">
        <strong>Default Admin Login:</strong><br>
        Username: <span style="font-family:monospace;">admin</span><br>
        Password: <span style="font-family:monospace;">admin</span>
      </div>
    </div>
    <div class="portal-cards">
      <div class="portal-card">
        <h3>Employee Portal</h3>
        <p>Build your resume, manage your profile, and get matched to jobs that fit your skills
and experience.</p>
```

```
        <a href="/employee/login" class="portal-btn">Login</a>
        <a href="/employee/register" class="portal-btn secondary">Register</a>
      </div>
      <div class="portal-card">
        <h3>Company Portal</h3>
        <p>Post job requirements, search for the best candidates, and manage your company profile.</p>
        <a href="/company/login" class="portal-btn">Login</a>
        <a href="/company/register" class="portal-btn secondary">Register</a>
      </div>
      <div class="portal-card">
        <h3>Admin Portal</h3>
        <p>Manage employees, companies, and keep the platform running smoothly.</p>
        <a href="/admin/login" class="portal-btn">Admin Login</a>
      </div>
    </div>
  </main>
</body>
</html>
```

- **Layout.ejs**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title><%= typeof title !== 'undefined' ? title : 'Manage Corp' %></title>
  <link rel="stylesheet" href="/style.css">
</head>
<body>
  <header class="main-header">
    <div class="nav-bar">
      <span class="logo">Manage Corp</span>
      <nav>
        <a href="/">Home</a>
        <% if (typeof userType !== 'undefined' && userType === 'employee') { %>
          <a href="/employee/dashboard">Dashboard</a>
          <a href="/employee/logout">Logout</a>
        <% } else if (typeof userType !== 'undefined' && userType === 'company') { %>
          <a href="/company/dashboard">Dashboard</a>
          <a href="/company/logout">Logout</a>
        <% } else if (typeof userType !== 'undefined' && userType === 'admin') { %>
          <a href="/admin/dashboard">Dashboard</a>
          <a href="/admin/logout">Logout</a>
        <% } else { %>
          <a href="/employee/login">Employee</a>
          <a href="/company/login">Company</a>
          <a href="/admin/login">Admin</a>
        <% } %>
      </nav>
    </div>
  </header>
  <main class="container">
    <% if (typeof message !== 'undefined') { %>
      <div class="flash"><%= message %></div>
    <% } %>
    <%- body %>
  </main>
  <footer style="text-align:center; margin:2rem 0; color:#888;">&copy; <%= new
Date().getFullYear() %> Manage Corp</footer>
</body>
</html>
```

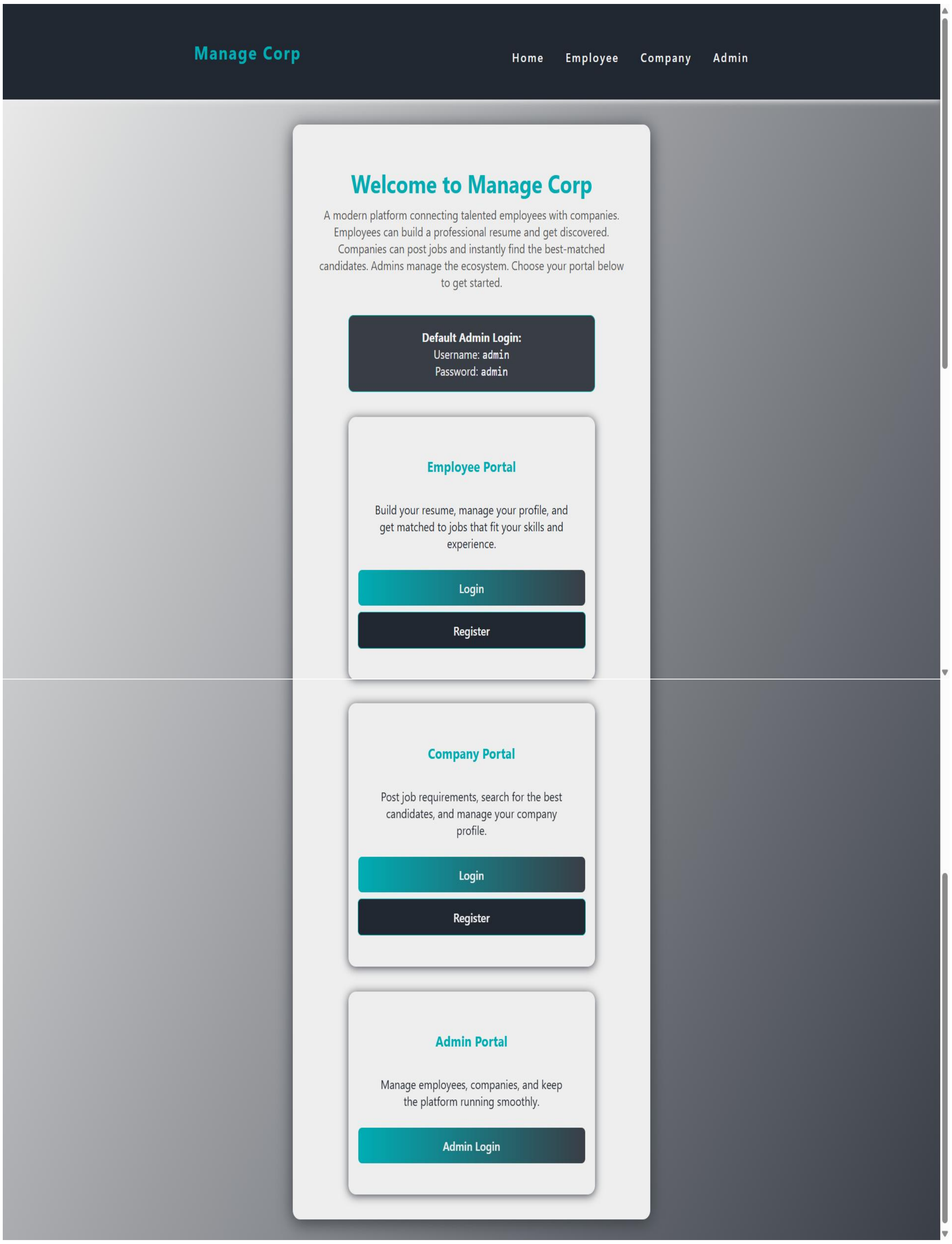
- **App.js**

```
require('dotenv').config();
const express = require('express');
const mongoose = require('mongoose');
const session = require('express-session');
const MongoStore = require('connect-mongo');
const bodyParser = require('body-parser');
const path = require('path');
const app = express();
app.use(bodyParser.urlencoded({ extended: true }));
app.use(express.static(path.join(__dirname, 'public')));
app.set('view engine', 'ejs');


// Session
app.use(session({
  secret: 'managecorpsecret',
  resave: false,
  saveUninitialized: false,
  store: MongoStore.create({ mongoUrl: process.env.MONGO_URI ||
'mongodb://localhost:27017/managecorp' })
}));
// MongoDB connection
mongoose.connect(process.env.MONGO_URI || 'mongodb://localhost:27017/managecorp', {
  useNewUrlParser: true,
  useUnifiedTopology: true
}).then(() => console.log('MongoDB connected'))
  .catch(err => console.log(err));
// Routes
app.use('/', require('./routes/main'));
app.use('/employee', require('./routes/employee'));
app.use('/company', require('./routes/company'));
app.use('/admin', require('./routes/admin'));


const PORT = process.env.PORT || 3000;
app.listen(PORT, () => console.log(`Server running on port: http://localhost:${PORT}`));
```

3. User Interface Screenshots



Welcome to Manage Corp

A modern platform connecting talented employees with companies. Employees can build a professional resume and get discovered. Companies can post jobs and instantly find the best-matched candidates. Admins manage the ecosystem. Choose your portal below to get started.

Default Admin Login:
Username: admin
Password: admin

Employee Portal

Build your resume, manage your profile, and get matched to jobs that fit your skills and experience.

Company Portal

Post job requirements, search for the best candidates, and manage your company profile.

Admin Portal

Manage employees, companies, and keep the platform running smoothly.

Fig 3.1 - Home Page

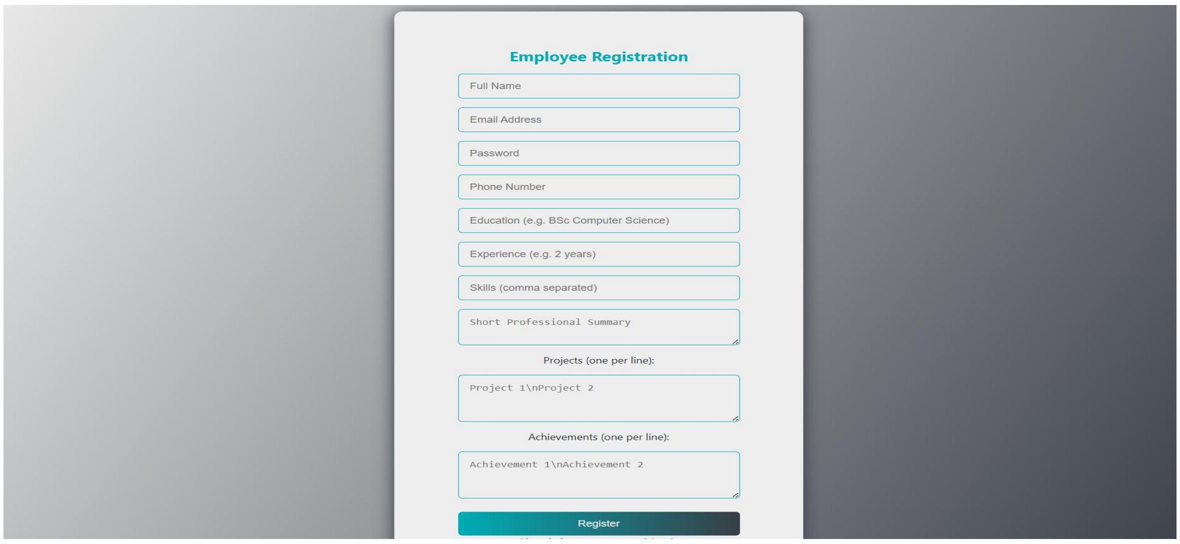

Employee Registration
Full Name
Email Address
Password
Phone Number
Education (e.g. BSc Computer Science)
Experience (e.g. 2 years)
Skills (comma separated)
Short Professional Summary
Projects (one per line):
Project 1\nProject 2
Achievements (one per line):
Achievement 1\nAchievement 2
Register

Fig 3.2 - Employee Registration Page

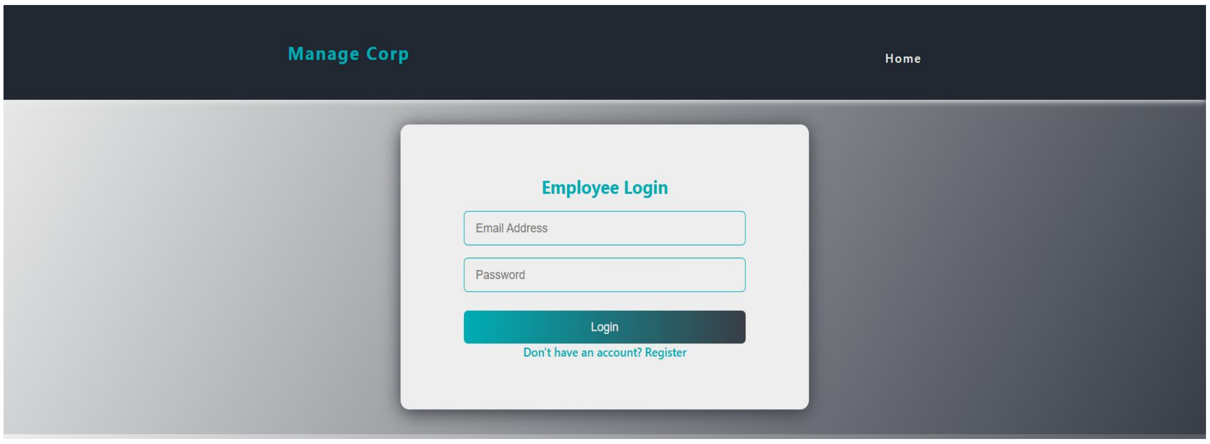


Fig 3.3 - Employee Login Page

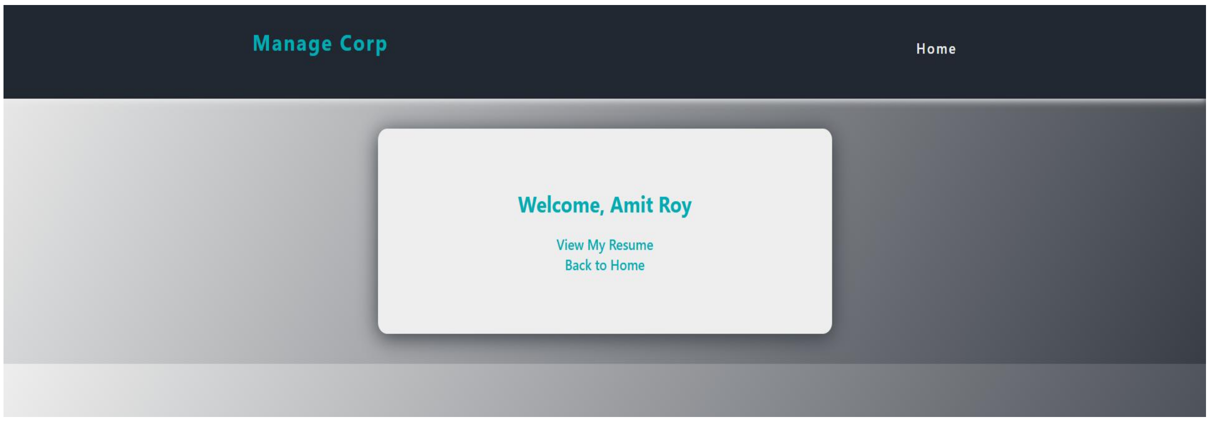
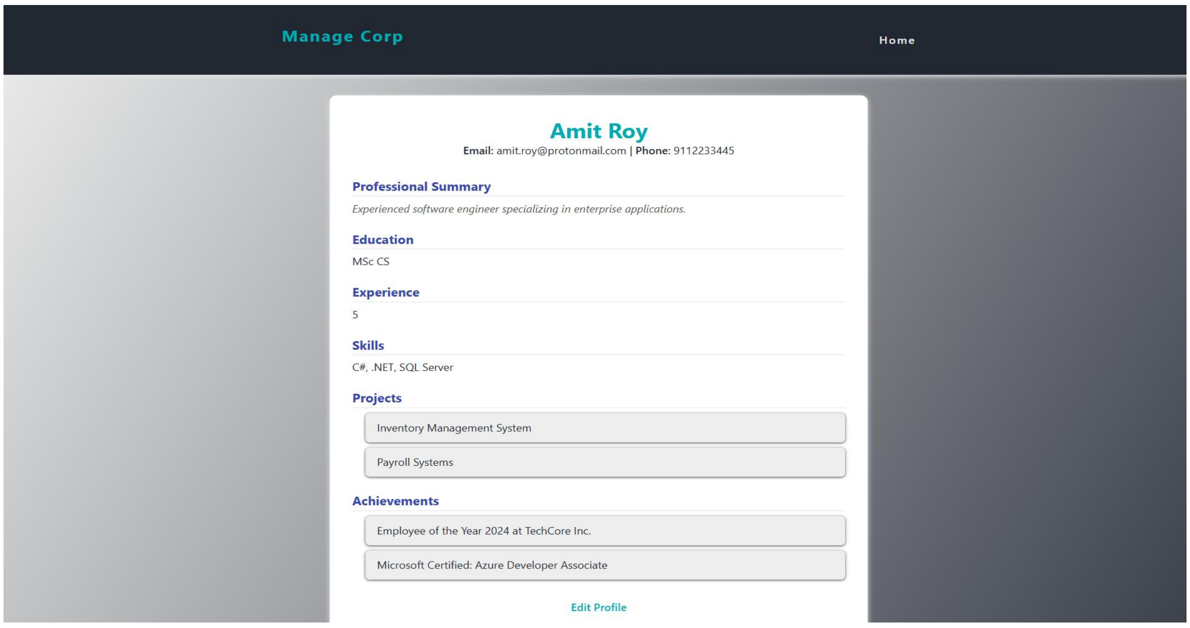


Fig 3.4 - Employee Dashboard



Amit Roy

Email: amit.roy@protonmail.com | **Phone:** 9112233445

Professional Summary

Experienced software engineer specializing in enterprise applications.

Education

MSc CS

Experience

5

Skills

C#, .NET, SQL Server

Projects

Inventory Management System

Payroll Systems

Achievements

Employee of the Year 2024 at TechCore Inc.

Microsoft Certified: Azure Developer Associate

Fig 3.5 - Employee Resume

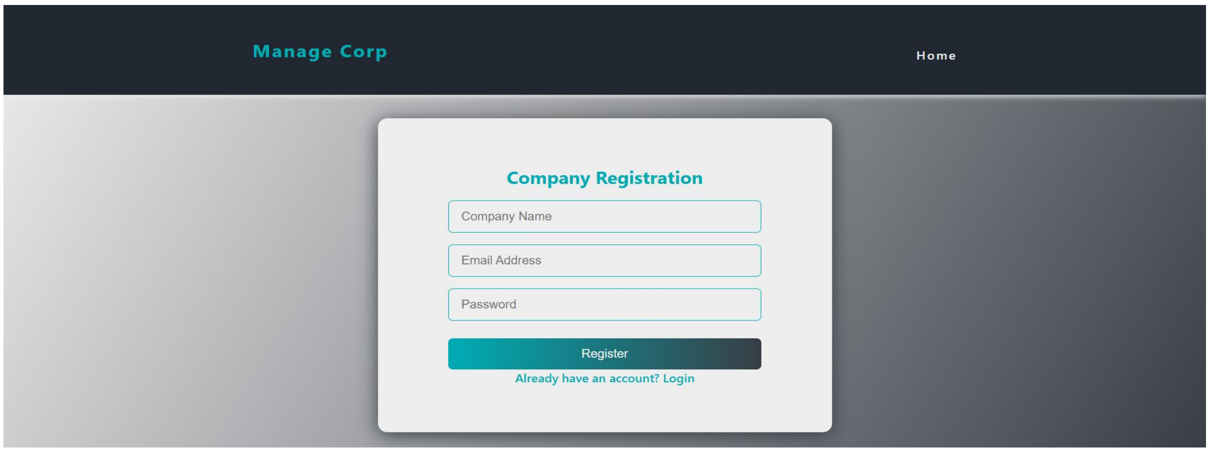


Fig 3.6 - Company Registration Page

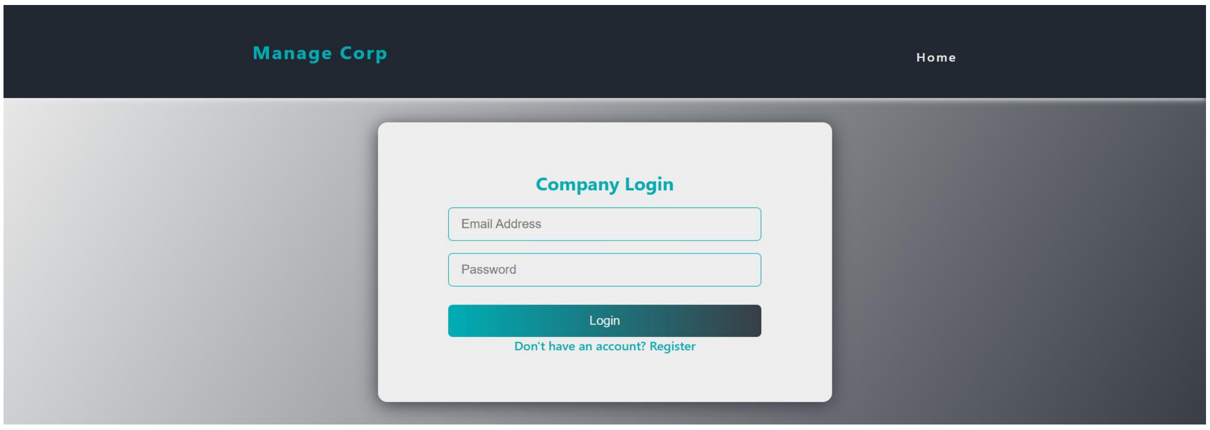


Fig 3.7 - Company Login Page

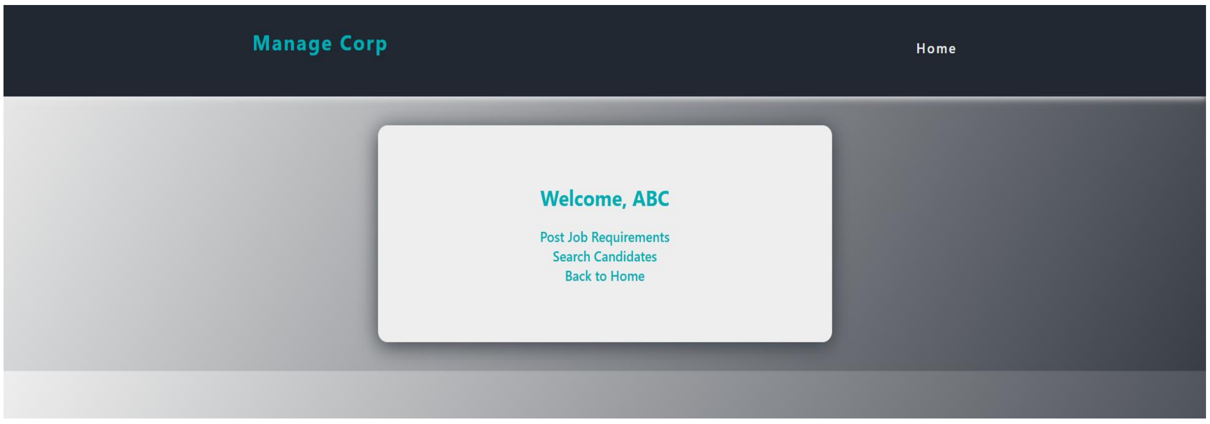


Fig 3.8 - Company Dashboard

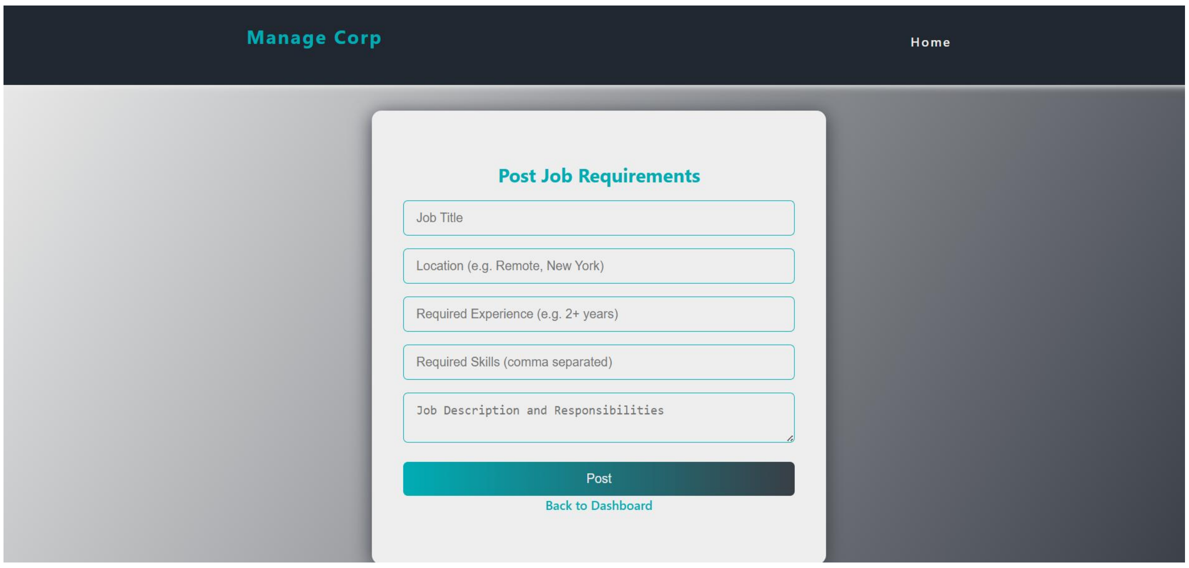
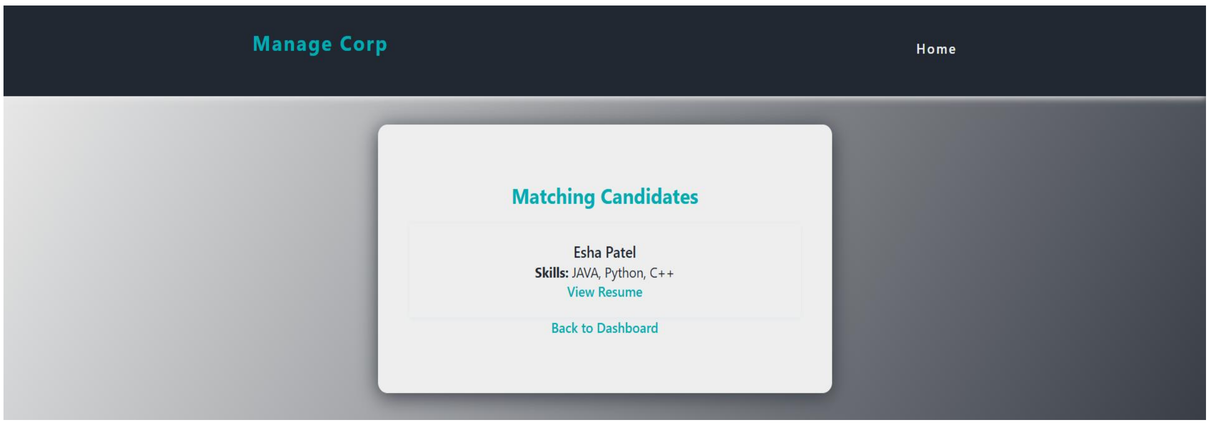

Manage Corp
Home
Post Job Requirements
Job Title
Location (e.g. Remote, New York)
Required Experience (e.g. 2+ years)
Required Skills (comma separated)
Job Description and Responsibilities
Post
Back to Dashboard

Fig 3.9 - Page for Posting Jobs



Esha Patel
Skills: JAVA, Python, C++

Fig 3.10 - Displaying Matching Candidate

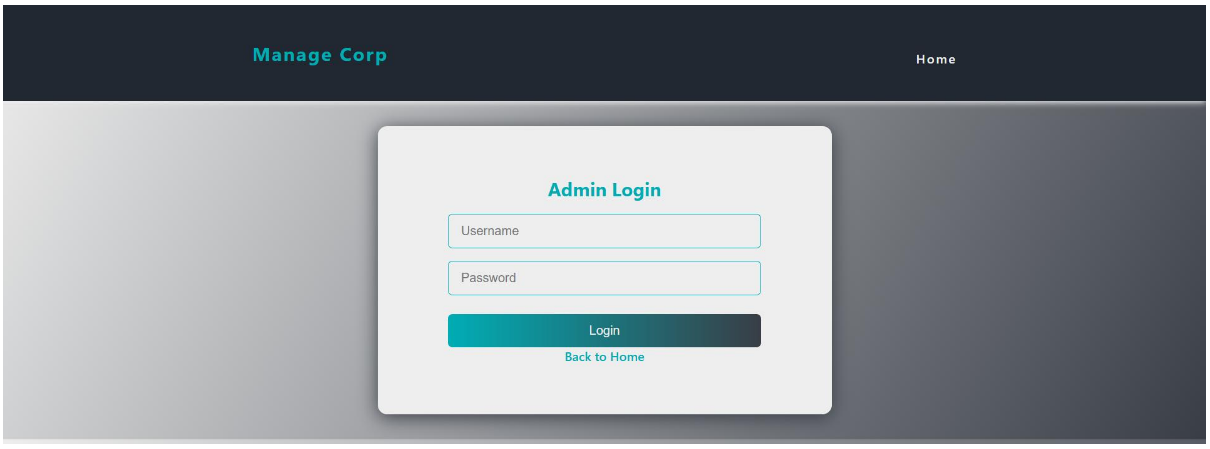
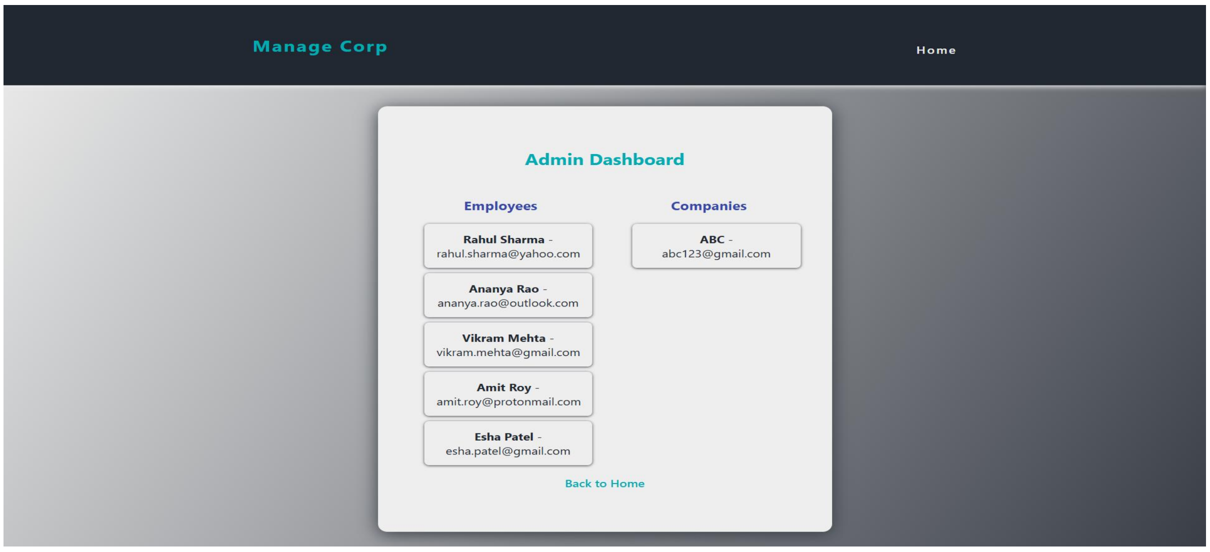


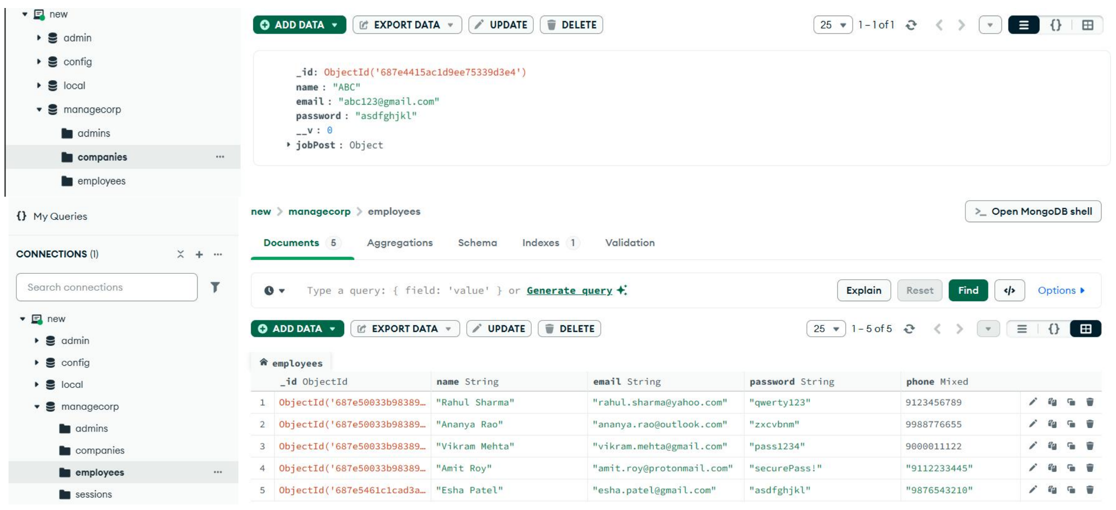
Fig 3.11 - Admin Login Page



Admin Dashboard

Employees	Companies
Rahul Sharma - rahul.sharma@yahoo.com	**ABC** - abc123@gmail.com
Ananya Rao - ananya.rao@outlook.com	
Vikram Mehta - vikram.mehta@gmail.com	
Amit Roy - amit.roy@protonmail.com	
Esha Patel - esha.patel@gmail.com	

Fig 3.12 - Admin Dashboard



```
_id: ObjectId('687e4415ac1d9ee75339d3e4')
name : "ABC"
email : "abc123@gmail.com"
password : "asdfghjkl"
__v : 0
jobPost : Object
```

employees

	_id ObjectId	name String	email String	password String	phone Mixed
1	ObjectId('687e50033b98389…	"Rahul Sharma"	"rahul.sharma@yahoo.com"	"qwerty123"	9123456789
2	ObjectId('687e50033b98389…	"Ananya Rao"	"ananya.rao@outlook.com"	"zxcvbnm"	9988776655
3	ObjectId('687e50033b98389…	"Vikram Mehta"	"vikram.mehta@gmail.com"	"pass1234"	9000011122
4	ObjectId('687e50033b98389…	"Amit Roy"	"amit.roy@protonmail.com"	"securePass!"	"9112233445"
5	ObjectId('687e5461c1cad3a…	"Esha Patel"	"esha.patel@gmail.com"	"asdfghjkl"	"9876543210"

Fig 3.13 - Database

4. Conclusion and Future Enhancement

The Manage Corp web application successfully addresses the challenges faced by organizations in managing employees, job postings, and administrative workflows. By integrating employee, company, and admin roles into a single, unified platform, the system ensures smooth interaction, secure data handling, and a responsive user experience. Built with modern web technologies such as Node.js, Express.js, MongoDB, and EJS, the application offers a scalable, maintainable, and modular architecture that supports core HR and recruitment functionalities. Secure session management and a clean codebase further enhance its reliability and usability. The modular folder structure and use of standardized practices also make the application easier to maintain and extend in the long term. Overall, Manage Corp proves to be an effective, real-world solution for digital workforce management. It not only simplifies day-to-day HR operations but also promotes better coordination between companies and candidates. The platform demonstrates how open-source technologies can be combined to build functional, user-friendly tools tailored to real organizational needs.

Future Enhancement

While the current version of Manage Corp meets essential requirements, there are several opportunities to enhance its functionality in future iterations. Introducing an email and notification system would enable alerts for job application updates, password resets, and admin messages. Adding secure file upload support would allow employees to attach PDF resumes and profile pictures to their profiles. Implementing role-based authorization middleware can further strengthen access control across different user types. An analytics dashboard with charts and metrics would help companies and admins track hiring trends and user activity more effectively. Additionally, exposing key features as Restful APIs would enable integration with mobile apps or third-party platforms. Improving the search functionality with advanced filters and full-text search can enhance the job and candidate lookup experience. Finally, upgrading the user interface using component-based libraries or frameworks like React would modernize the platform and provide a more dynamic, responsive user experience. These enhancements would significantly increase the platform's usability, performance, and adaptability to future organizational needs.