



Jagan Institute of Management Studies

(Affiliated to Guru Gobind Singh Indraprastha University, Sector-16C, Dwarka, Delhi)

PLACEMENT OFFICE

A PROJECT REPORT

Submitted by

ADITYA RAJ MOURYA (09114004423)

in partial fulfillment for the award of the degree

of

MASTER OF COMPUTER APPLICATIONS



2024

JAGAN INSTITUTE OF MANAGEMENT STUDIES

BONAFIDE CERTIFICATE

It is certified that this project report **“PLACEMENT OFFICE”** is the bonafide work of **“ADITYA RAJ MOURYA (09114004423)”** of MCA third Semester, Section A, Shift-I who carried out the project work under my supervision.

SIGNATURE

Dr. Sanjeev Saxena

Professor (IT)

Jagan Institute of Management Studies,

Sector-5, Rohini, New Delhi

ACKNOWLEDGEMENT

First and foremost, I would like to thank Jagan Institute of Management Studies for giving us the opportunity to work on a project, and to present a comprehensive report on the same.

I would like to show my gratitude to *Dr.Sanjeev Saxena*, who allowed me to work under her expert guidance and invaluable supervision. She guided us from the first step in this project, to the last step of this report, helping us all the way to the finish line. We are also thankful to our peers for their extremely important insights into how the project could be improved and enhanced.

Most importantly, we are lucky to have such dedicated and understanding teachers at JIMS, and would like to thank them for the knowledge they have imparted in us, during the course of our curriculum.

Aditya Raj Mourya (09114004423)

Signature: _____

ABSTRACT

"In the dynamic realm of modern education, the Placement Office web application emerges as a pivotal conduit between educational institutions and ambitious students. Acknowledging the paramount significance of campus placements, Placement Office optimizes the process, fortifying the symbiotic relationship between students and institutes.

For students, campus placements signify more than mere job opportunities – they epitomize a gateway into the professional sphere, shielded from the rigors of traditional job hunting. Placement Office empowers students to construct sturdy career trajectories, alleviating the pressures of peer competition and familial expectations. The platform ensures that students not only secure a foothold in the professional arena but also cultivate connections with industry professionals during placement drives, fostering invaluable networks for their future endeavours.

On the institutional front, a robust placement record becomes a cornerstone of competitive advantage in the educational landscape. Institutes and universities leveraging Placement Office gain a distinctive edge by seamlessly connecting with eligible students and streamlining the selection process. The application furnishes institutes with the autonomy to meticulously select candidates, thereby facilitating a tailored approach to talent acquisition.

The Placement Office system transcends conventional placement services, serving as a comprehensive application software that equips institutes with the tools to independently identify eligible candidates. As the educational landscape continues to evolve, Placement Office stands as a testament to our unwavering commitment to nurturing meaningful connections between educational institutions and students, sculpting successful futures, and propelling careers forward."

LIST OF FIGURES AND TABLES

LIST OF FIGURES

1.1	Iterative waterfall model	5
2.1	Application Architecture	8
3.1.1	Use Case Diagram	10
3.1.2	Data Flow Diagram	11
3.1.3	ER Diagram	12
3.1.4	Object Diagram	13
3.1.5	Class Diagram	14
3.1.6	Activity Diagram	15
3.1.7	Sequence Diagram	16
3.1.8	State Transition Diagram	18
4.7	Unit Testing of Placement Office	50

LIST OF TABLES

3.1	Users Table	19
3.2	Applications Table	19
3.3	Job Table	19
3.4	Question Table	19
3.2	Applications Table	19
4.2.1	Test Case 1: Register Module	51
4.2.2	Test Case 2: Login Module	51
4.2.3	Test Case 3: Post Job Module	52

TABLE OF CONTENT

CHAPTER 1: INTRODUCTION.....	1
1.1 Brief description of the system	1
1.2 About the proposed system	2
1.3 Methodology used for Analysis, design and development.....	3
1.4 Methodology used for Data Collection	5
1.5 System Requirement tools	6
CHAPTER-2: SYSTEM SPECIFICATIONS.....	7
2.1 Project Design and Architecture	8
2.2 Frontend Approach	8
2.3 Backend Approach.....	9
CHAPTER 3: SYSTEM DESIGN... ..	10
3.1 Physical Design.....	10
3.1.1 Use Case Diagram	10
3.1.2 Data Flow Diagram	11
3.1.3 ER Diagram.....	12
3.1.4 Object Diagram	13
3.1.5 Class Diagram	14
3.1.6 Activity Diagram	15
3.1.7 Sequence Diagram.....	16
3.1.8 State Transition Diagram.....	18
3.2 Database Design.....	19
CHAPTER-4: SYSTEM DEVELOPMENT AND TESTING.....	20
4.1 System Development.	20
4.1.1 Files in the project	20

4.1.2 Dependencies in the project.....	22
4.2 System Testing.....	48
4.2.1 Introduction.....	48
4.2.2 Unit testing and System testing of Placement Office	48
CHAPTER-5: INTERFACE DESIGN	56
CHAPTER 6: FUTURE SCOPE, SUMMARY AND CONCLUSION	65

CHAPTER-1

INTRODUCTION

1.1 BRIEF DESCRIPTION OF THE SYSTEM

Placement Office is a web-based application that establishes a comprehensive network between educational institutes and their students. In the contemporary academic landscape, campus placements hold paramount importance for both students and institutions. While placements help students kickstart their careers without the typical job market challenges, a strong placement record gives institutes a competitive edge in the education market.

The Placement Office system aims to provide a streamlined and efficient solution for managing the campus placement process. Recognizing the current industry scenario, wherein educational institutions often rely on manual or disparate digital methods for placement tracking, the Placement Office project seeks to address these challenges and deliver a unified, automated, and user-friendly platform.

Need of the Project

In the contemporary educational landscape, the process of campus placements plays a pivotal role for both students and institutions. The demand for streamlined and efficient placement tracking systems has intensified, recognizing the need for a comprehensive solution. Presently, educational institutions grapple with manual and disparate methods for recording and managing placement data, leading to inefficiencies, inaccuracies, and time-consuming processes. Recognizing these challenges, the Placement Office project emerges as a vital solution to address the current gaps in the industry.

Current Scenario

The prevailing scenario in the education sector involves a fragmented approach to placement tracking. Many institutions rely on traditional, paper-based systems or rudimentary digital tools that lack cohesion and advanced features. This disjointed approach results in difficulties for both students and institutions in accessing, updating, and analyzing placement data efficiently. Additionally, manual processes contribute to a lack of real-time insights, hindering strategic decision-making for educational institutions. The industry yearns for a unified, automated, and user-friendly Placement Office system that aligns with modern technological standards.

Project Ideation

The ideation of the Placement Office project stems from a visionary approach to revolutionize the way educational institutions manage and track placements. The project envisions a user-friendly interface that caters to the specific needs of both students and institutions. By harnessing cutting-edge technologies, the Placement Office aims to simplify data storage and retrieval, eliminate wastage of time associated with manual processes, and provide instantaneous results. The project aligns with the objectives of enhancing efficiency, reducing paperwork, and delivering an

immediate, accurate, and user-centric placement tracking experience. The Placement Office aspires to be a transformative solution, bringing innovation to the realm of campus placements and establishing a seamless connection between institutes and their students.

1.2 ABOUT THE PROPOSED SYSTEM

Objective

1. Develop an intuitive user interface.
2. Facilitate seamless data storage and retrieval processes.
3. Minimize time consumption for all stakeholders.
4. Eliminate paperwork, ensuring swift and immediate outcomes.

Purpose

The Placement Office addresses the imperative for educational institutes to efficiently manage placement data without the need for custom software development. Recognized as a premier provider of research, data, and analytics, the Placement Office offers comprehensive insights into campus recruitment and placement markets. It serves as a valuable resource, furnishing educational institutions with distinctive daily, weekly, and monthly data.

Aim

The Placement Office is introduced to meet the needs of both educational institutes and their students. The platform enables institutes to review past placement records without the necessity of creating separate sites. Students can access their placement history and streamline job applications through the Placement Application Form. This initiative aims to rectify existing drawbacks in the manual system of reviewing placement records and job applications.

Scope

The system offers educational institutes the capability to assess the number of students placed by presenting comprehensive placement records. This allows for quick access by students and faculty to review placement data with a single click. Additionally, students can effortlessly apply for jobs by submitting resumes and relevant details.

Benefits

The Placement Office system significantly reduces manual workload, enhances accuracy, improves efficiency, and saves time. It contributes to advancing student placement and retention strategies while effectively tracking the campus placement process. The implementation results in time savings, reduced human effort, heightened security, reliability, and streamlined scheduling of interview details and responsibilities.

1.3 METHODOLOGY USED FOR ANALYSIS, DESIGN & DEVELOPMENT

The application of robust engineering principles to achieve the economic development of software that is both reliable and efficiently operational on tangible hardware is recognized as software engineering.

Software engineering serves as a discipline with the following objectives:

1. Ensuring the production of high-quality software.
2. Timely delivery of software products
3. Adherence to budget constraints
4. Fulfillment of all specified requirements.

A software life cycle encompasses the discernible stages that a software product, such as the Placement Office, traverses throughout its existence. This life cycle is articulated through a descriptive and diagrammatic model, representing the sequence of activities essential for guiding a software product through its various phases. Moreover, a life cycle model not only encapsulates all requisite activities but also delineates the prescribed order in which these activities should be executed.

LIFE CYCLE MODEL:

There are various life cycle models to improve the software processes.

- ✓ Waterfall model
- ✓ Prototype model
- ✓ Iterative Enhancement model
- ✓ Evolutionary model
- ✓ Spiral model

ITERATIVE WATERFALL MODEL

There are 6 phases in this Development model.

1. **Feasibility Study:**
 - This pivotal phase entails a meticulous analysis of the problem, coupled with the collection of relevant information pertinent to the product. Its paramount objective is to ascertain the financial and technical feasibility of product development.
2. **Requirement Analysis and Specification:**
 - This phase is dedicated to a comprehensive understanding of the customer's precise requirements, meticulously documented in the form of a Software Requirements Specification (SRS).
3. **Design:**
 - The design phase seeks to adeptly translate the requirement specifications into a structured format amenable for implementation in a chosen programming language.

4. Implementation and Unit Testing:

- This critical phase involves the actual implementation of the design. Initial testing focuses on isolated modules before integration into the larger software product.

5. Integration and System Testing:

- This phase encompasses the comprehensive integration of all modules, subjecting the entire system to rigorous testing to ensure cohesion and functionality.

6. Operation and Maintenance:

- With the software release, this phase initiates the operational life cycle, encompassing ongoing activities and maintenance tasks to ensure sustained performance and adaptability.

The outlined process adheres to the iterative waterfall model, emphasizing a sequential progression through well-defined phases, with provisions for iterative refinement based on feedback.

WHY ITERATIVE WATERFALL MODEL?

Clear Project Objectives: The project's objectives are well-defined and unlikely to undergo significant changes. The Waterfall Model is effective when project requirements are stable from the outset.

Predictable and Sequential Process: The Waterfall Model offers a linear and sequential approach, making it highly predictable. This is advantageous when a step-by-step progression is preferred.

Document-Driven Approach: The model's emphasis on documentation aligns with the project's need for thorough documentation of requirements, designs, and testing procedures.

Well-Defined Phases: Each phase in the Waterfall Model has distinct goals and outcomes. This structured approach is suitable for a project with clear milestones and deliverables.

Client Involvement During Initial Phases: Client involvement is crucial during the feasibility study, requirement analysis, and specification phases. The Waterfall Model allows clients to provide input at the beginning of the project.

Minimal Changes Expected: The project scope is well-understood, and there is minimal anticipation of significant changes. The Waterfall Model is less adaptable to changes compared to more iterative models.

Resource Availability: Resources, including personnel and technology, are available and committed for the entire duration of the project, aligning with the sequential nature of the Waterfall Model.

Risk Assessment: The project's risks can be assessed and mitigated at each phase before progressing to the next, contributing to a more controlled development process.

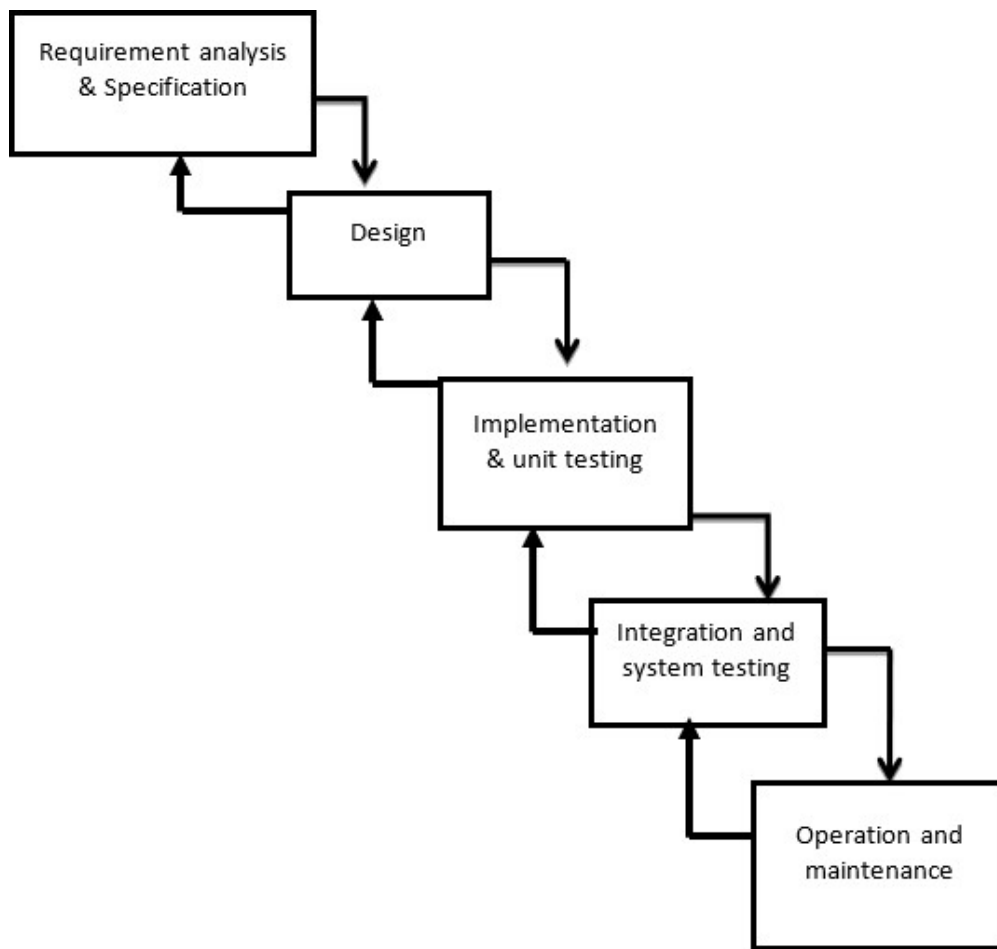


Figure 1.1: Iterative Waterfall Model

1.4 METHODOLOGY USED FOR DATA COLLECTION

Data collection is the systematic process of gathering information, encompassing two distinct types:

- **Primary Data**
- **Secondary Data**

Primary Data Collection: Primary data refers to original information gathered specifically for the intended purpose. It entails data obtained firsthand from direct experiences.

Secondary Data Collection: Secondary data encompasses information that has been previously collected and is readily available from alternative sources in various formats. Secondary data proves to be cost-effective and more rapidly accessible compared to primary data.

Sources of Secondary Data for Project Implementation:

Web Search:

Platforms such as Wikipedia.org serve as valuable sources for comprehensive information.

References:

Reputable texts like "Software Engineering" by K. K. Aggarwal and "Database Management System" by Navathe contribute to the secondary data pool.

1.5 SYSTEM REQUIREMENT TOOLS:

- ✓ Software requirements-
 1. Operating system
 2. Database- Mongo DB
 3. Platform – Web Browser
 4. Technology – MERN Stack (MongoDB, Express, React, Node)
- ✓ Hardware requirement-
 1. 2 GB RAM
 2. VGA monitor
 3. Intel 3.0 GHz (or equivalent) or higher processor
 4. Keyboard and mouse

No other special hardware interfaces are used in the project.

CHAPTER-2

SYSTEM SPECIFICATION

FOR APPLICATION SERVER

HARDWARE REQUIREMENTS (Minimum Requirement)

Minimum RAM: 2 GB

Free Hard Disk Space: 250 MB

Processor: 2 GHz

Connectivity: 3G Internet

SOFTWARE REQUIREMENTS (Minimum Requirement)

Operating System: Windows

Backend: NodeJS, ExpressJS

Database: MongoDB

Frontend: HTML, CSS, JavaScript, React, Tailwind

FOR WEB APPLICATION USERS

HARDWARE REQUIREMENTS (Minimum Requirement)

Minimum RAM: 256 MB

Free Hard Disk Space: 250 MB

Processor: 512MHz

Connectivity: 3G Internet

SOFTWARE REQUIREMENTS (Minimum Requirement)

Internet Browser: Google Chrome, Mozilla Firefox etc.

FOR SMARTPHONE USERS

HARDWARE REQUIREMENTS (Minimum Requirement)

Connectivity: Internet

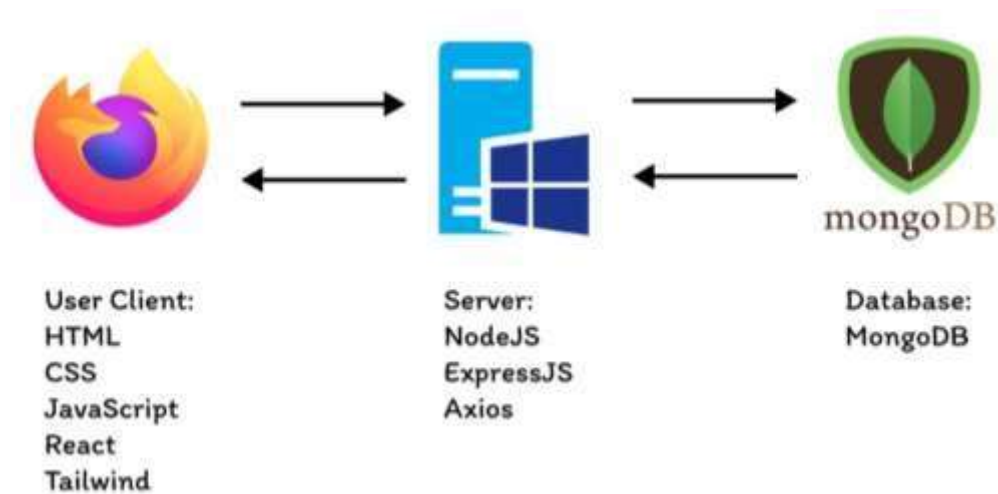
SOFTWARE REQUIREMENTS (Minimum Requirement)

Operating System: Android 6.0 (Marshmallow) or up

Internet browser: Any

2.1 PROJECT DESIGN AND ARCHITECTURE

Application architecture



2.2 FRONTEND APPROACH

The basic frontend development of the Placement Tracker project was to ensure a user-friendly and visually appealing interface for both students and educational institutions. The primary goal was to create an intuitive platform that facilitates easy navigation, data interaction, and a seamless experience.

Responsive Design:

Implemented a responsive design to ensure the platform is accessible across various devices, including desktops, tablets, and mobile phones. Utilized media queries and a responsive framework (Tailwind CSS) to adapt the layout to different screen sizes.

User Authentication:

Implemented a secure user authentication system to differentiate between students and educational institutions. Provide a smooth and secure login and registration process for users.

Dashboard for Students:

Developed a personalized dashboard for students after login, to fill an application form, prepare for upcoming interviews with Subject-specific Interview Questions.

Placement Records:

Added the major recruiters of the institution with a link to navigate directly to the company's portal to know more about them.

Application Forms:

Enable students to fill out online application forms for placement opportunities. Implement form validation to ensure accurate and complete submission.

Feedback and Helpdesk:

Integrated a feedback and contact system for students to provide insights on their placement experiences or contact the administrator in case of any issue or problem faced by them.

Visual Design:

Implemented a clean and professional visual design with a consistent colour scheme and typography. Used high-quality images and icons to enhance the overall aesthetics.

2.3 BACKEND APPROACH

Technology Stack:

Use Node.js with Express.js as the backend framework.

Employed MongoDB for data storage.

RESTful API Design:

Developed a RESTful API to enable communication between the frontend and backend.

Defined clear and intuitive endpoints for various operations, such as user authentication, placement record management, and application submissions.

User Authentication and Authorization:

Implemented user authentication using contexts for secure authentication. Enforce role-based access control to ensure that users have appropriate permissions.

Database Schema Design:

Create a well-structured database schema with tables for users, placement records, applications, and related entities. Establish relationships between tables for efficient data retrieval.

Data Validation and Sanitization:

Implement input validation and sanitization at the API level to prevent common security vulnerabilities.

Application Submission Workflow:

Develop a workflow for handling student applications, including form validation and storage of application details.

Feedback and Query Handling:

Used EmailJS to manage feedback and queries submitted by students. Store feedback securely and provide administrator to view, respond to, and analyse feedback.

Security Measures:

Enforce HTTPS to secure data transmission between the frontend and backend. Implement measures like rate limiting and input validation to enhance security.

Scalability Considerations:

Design the backend with scalability in mind, allowing the system to handle an increasing number of users and data.

CHAPTER 3

SYSTEM DESIGN

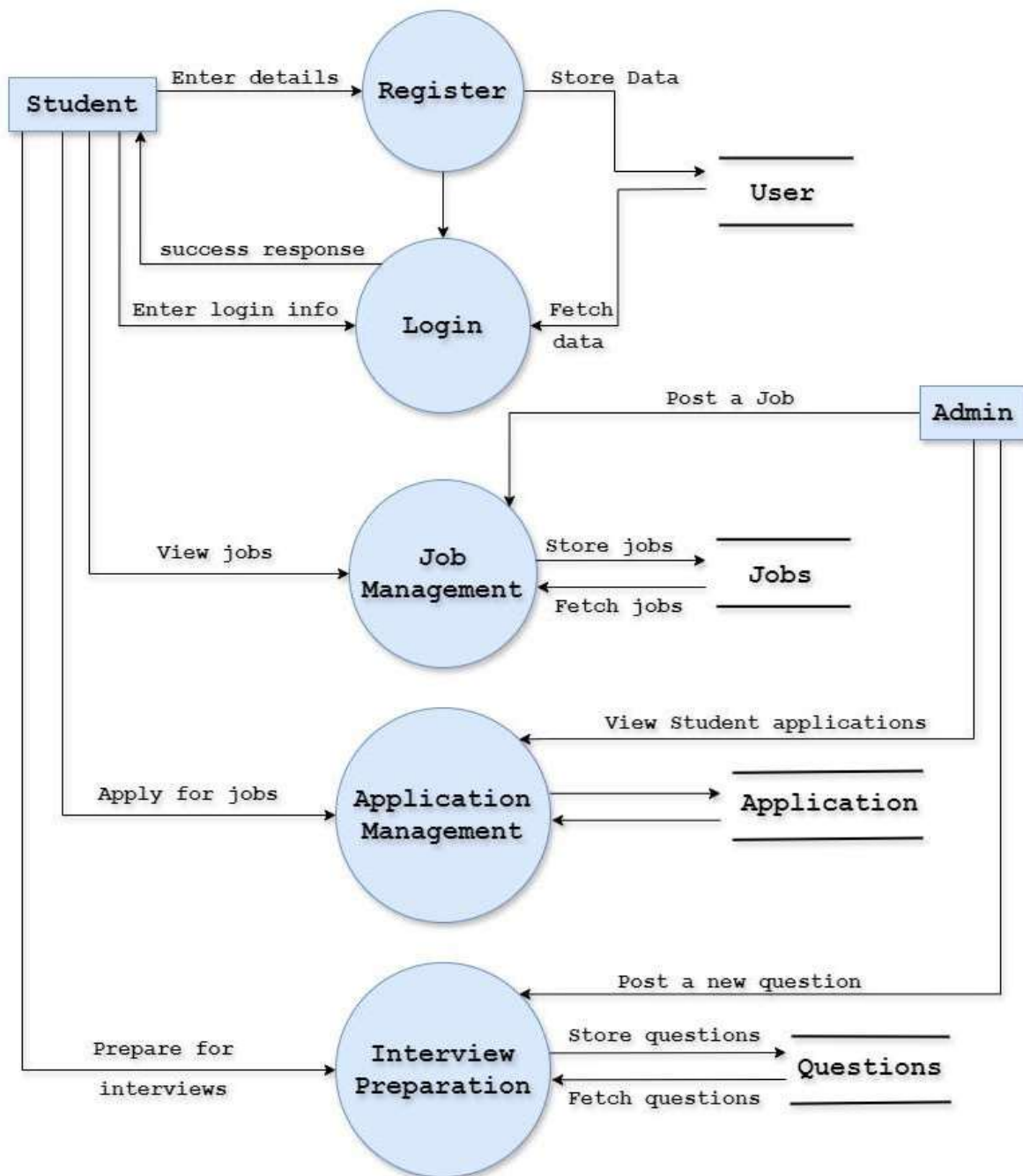
Systems design is a critical phase in the development process where the structure, components, modules, interfaces, and data of a system are meticulously defined to fulfill specific requirements.

3.1 Physical Design

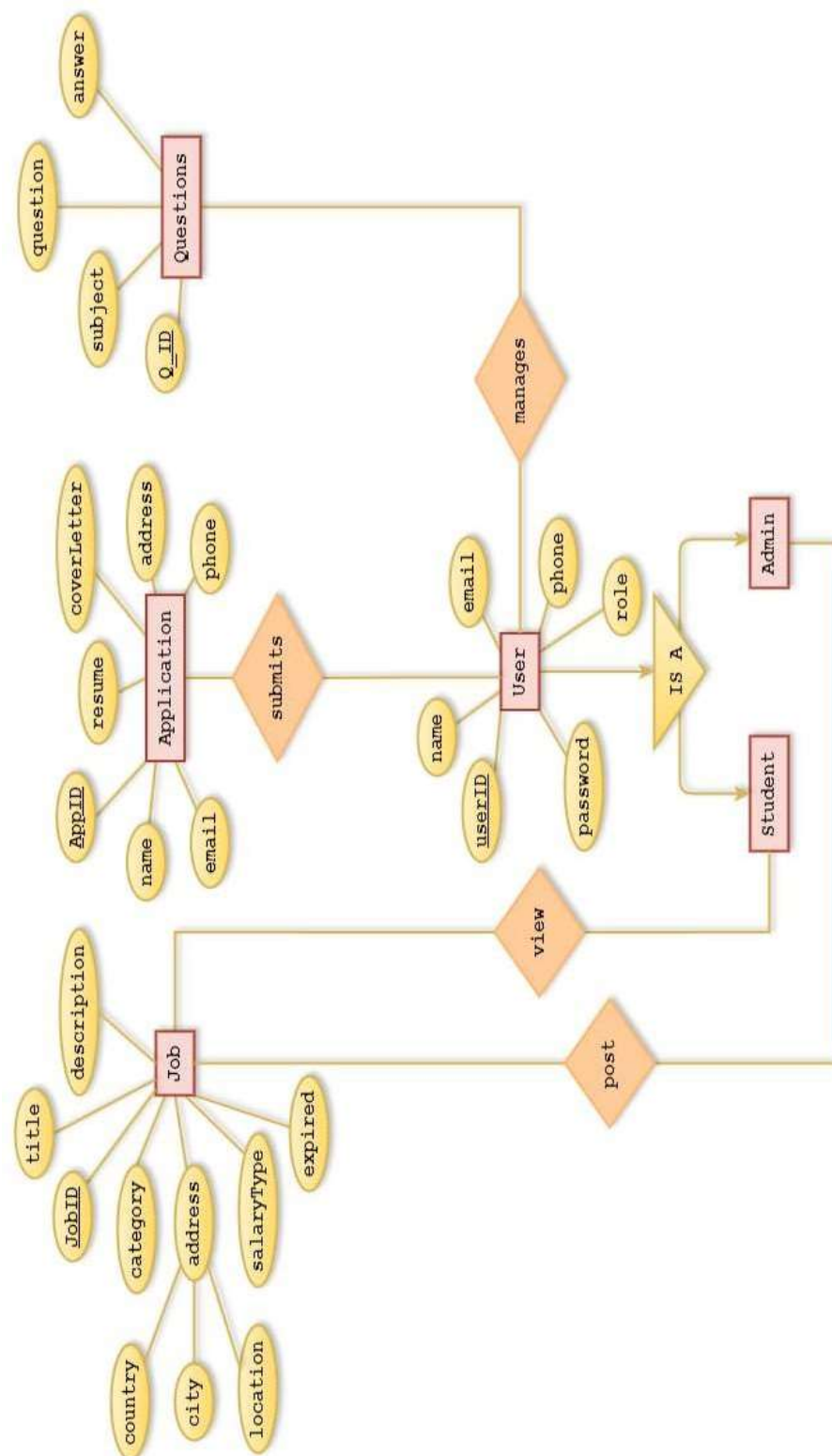
3.1.1 Use case Diagram



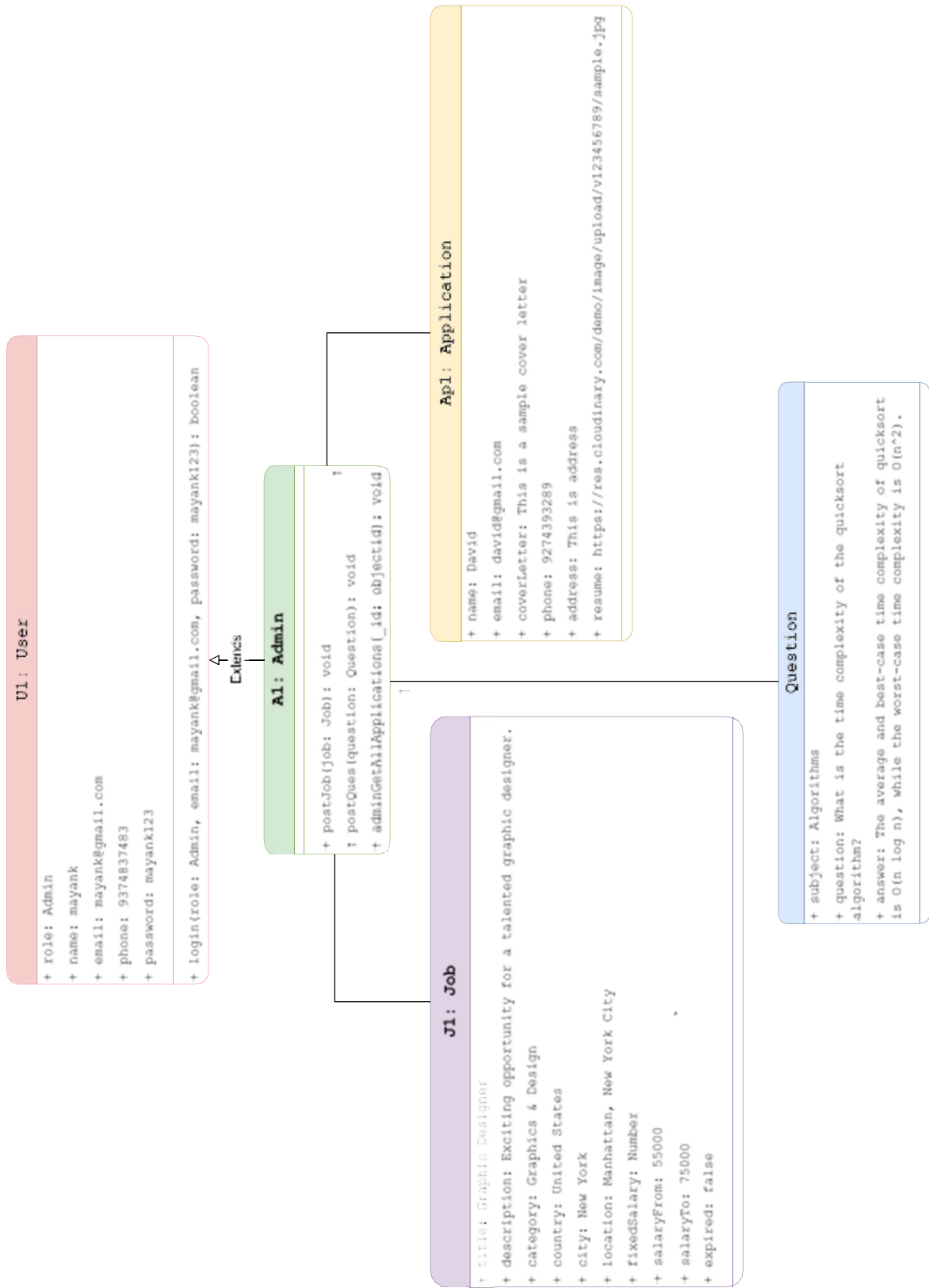
3.1.2 DFD (Data Flow Diagram)



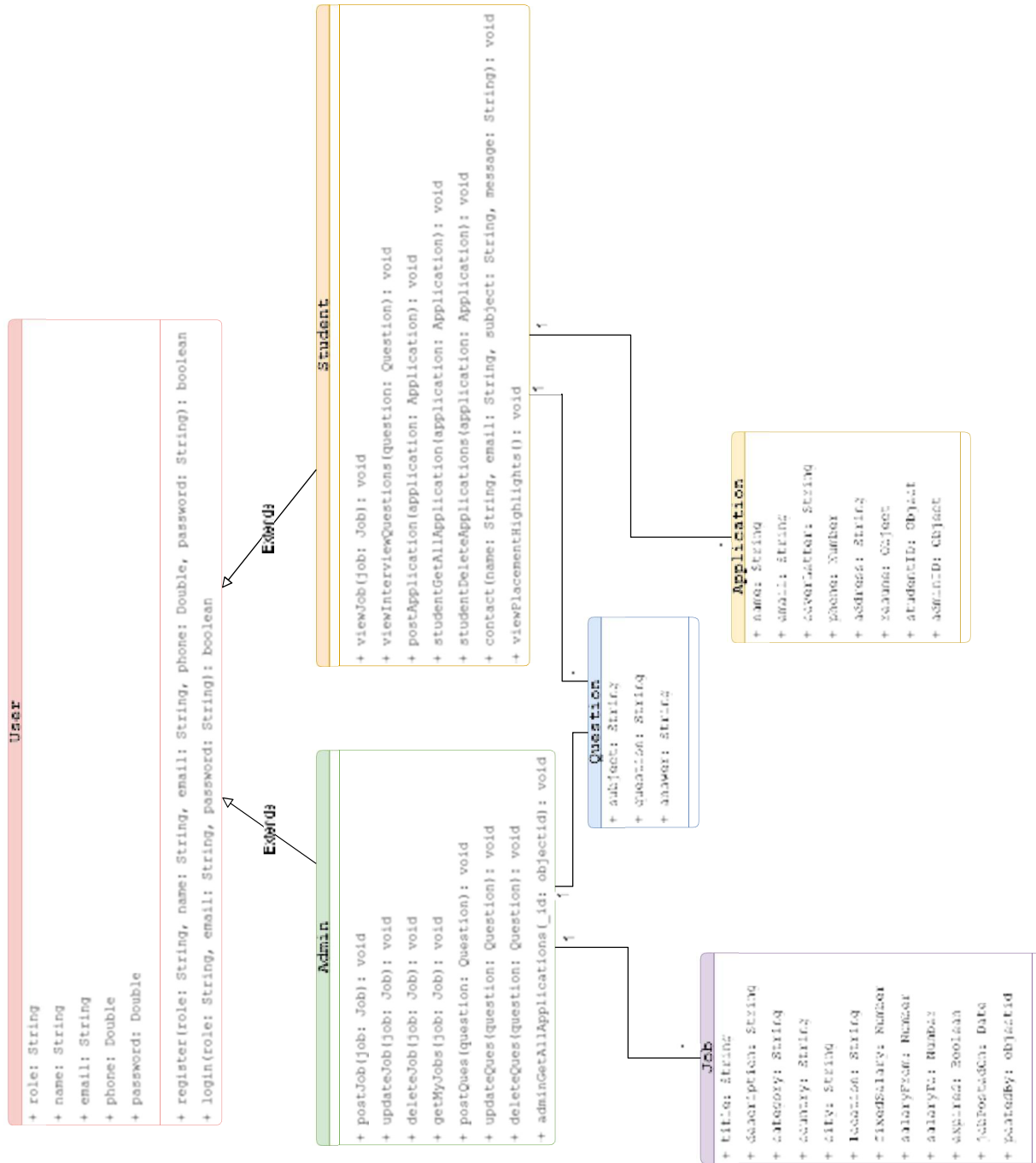
3.1.3 ER Diagram



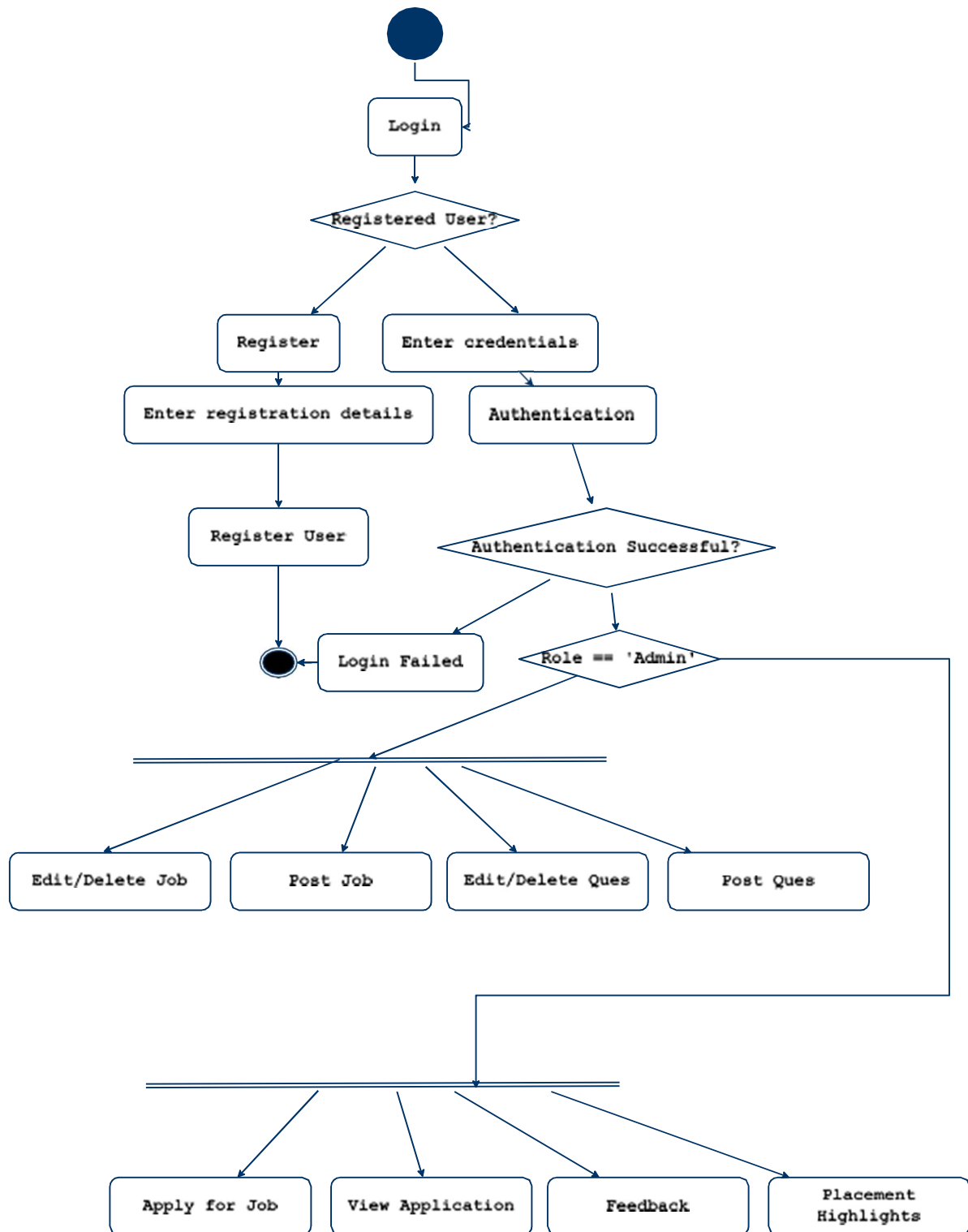
3.1.4 Object Diagram



3.1.5 Class Diagram

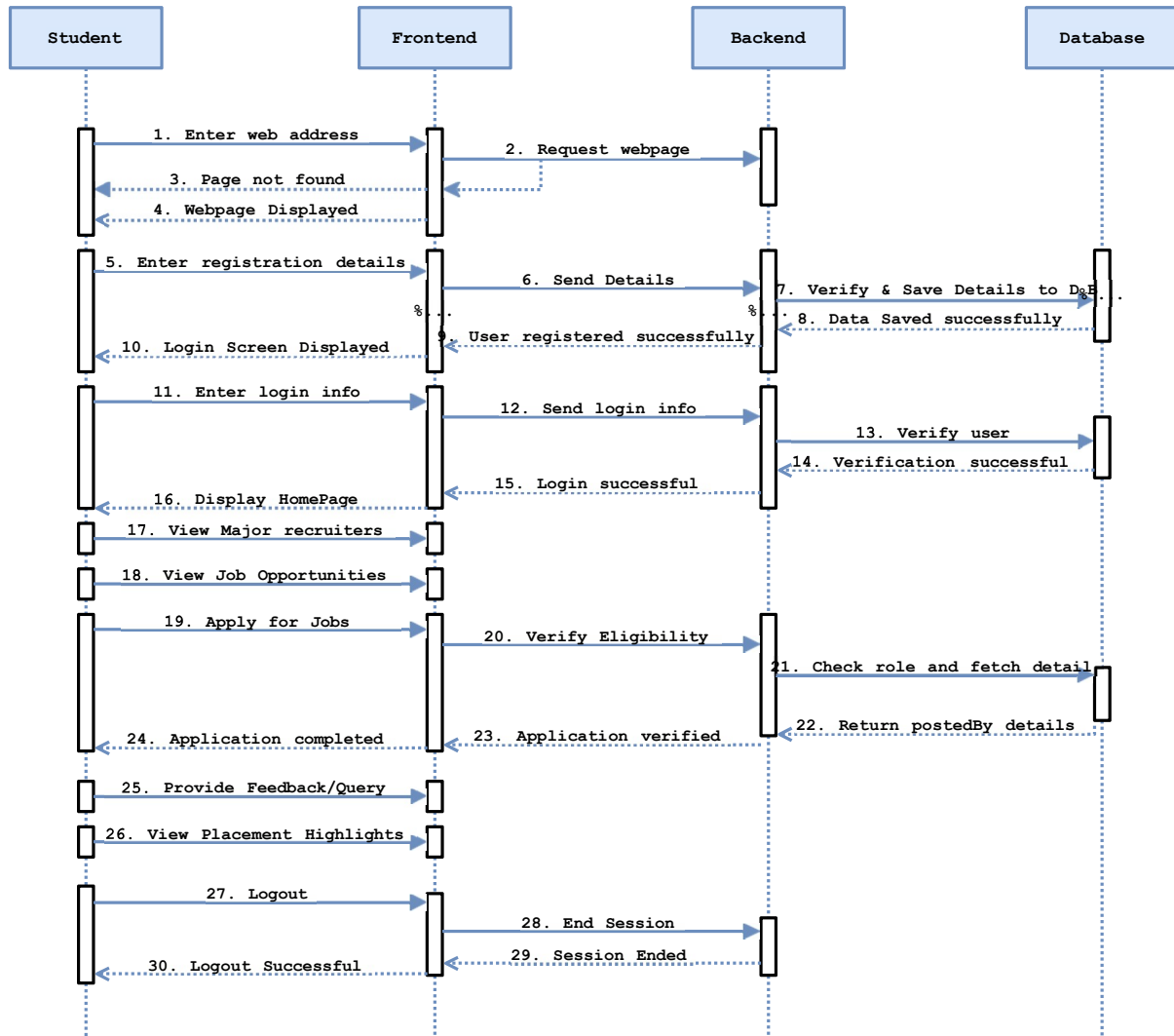


3.1.6 Activity Diagram

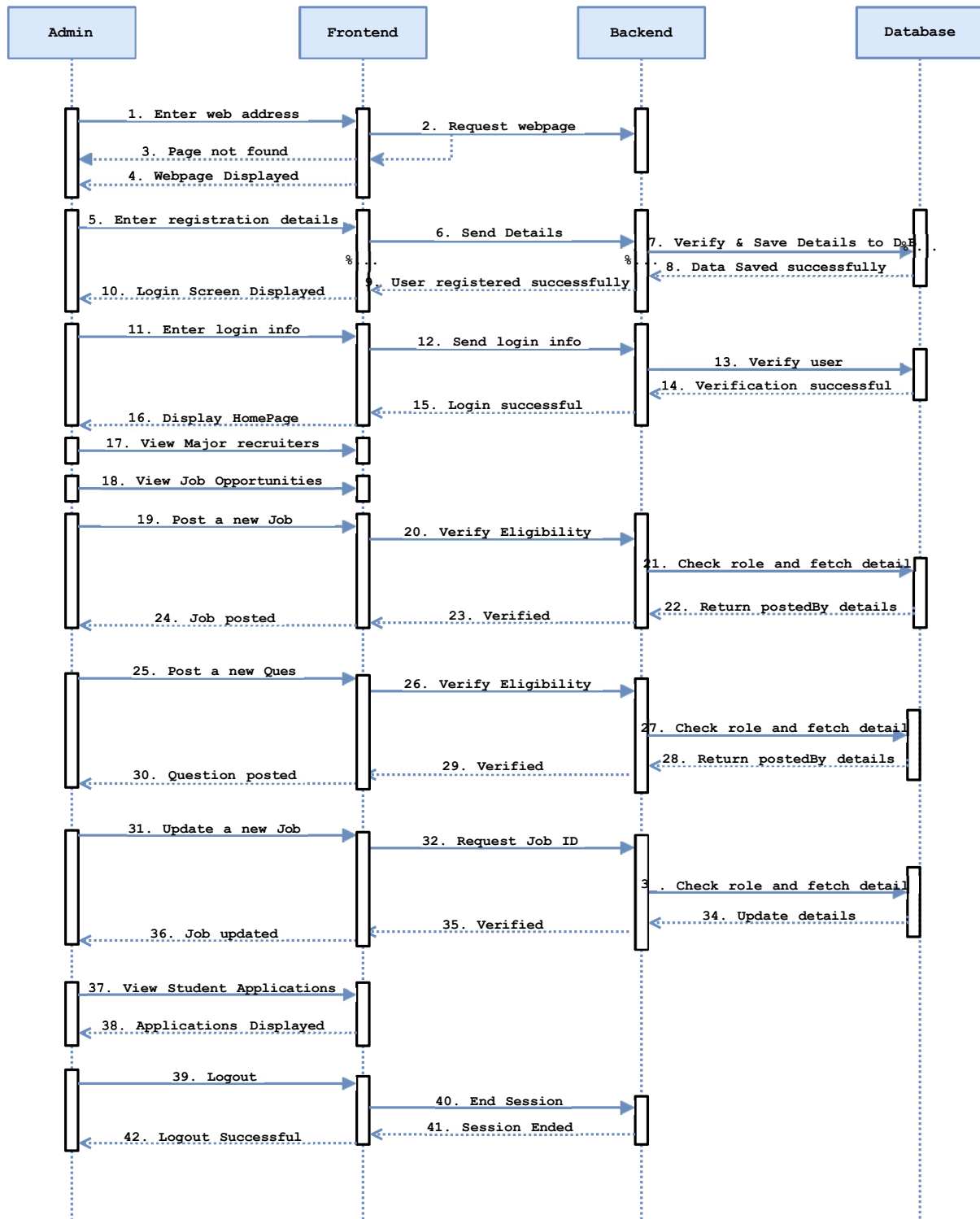


3.1.7 Sequence Diagram

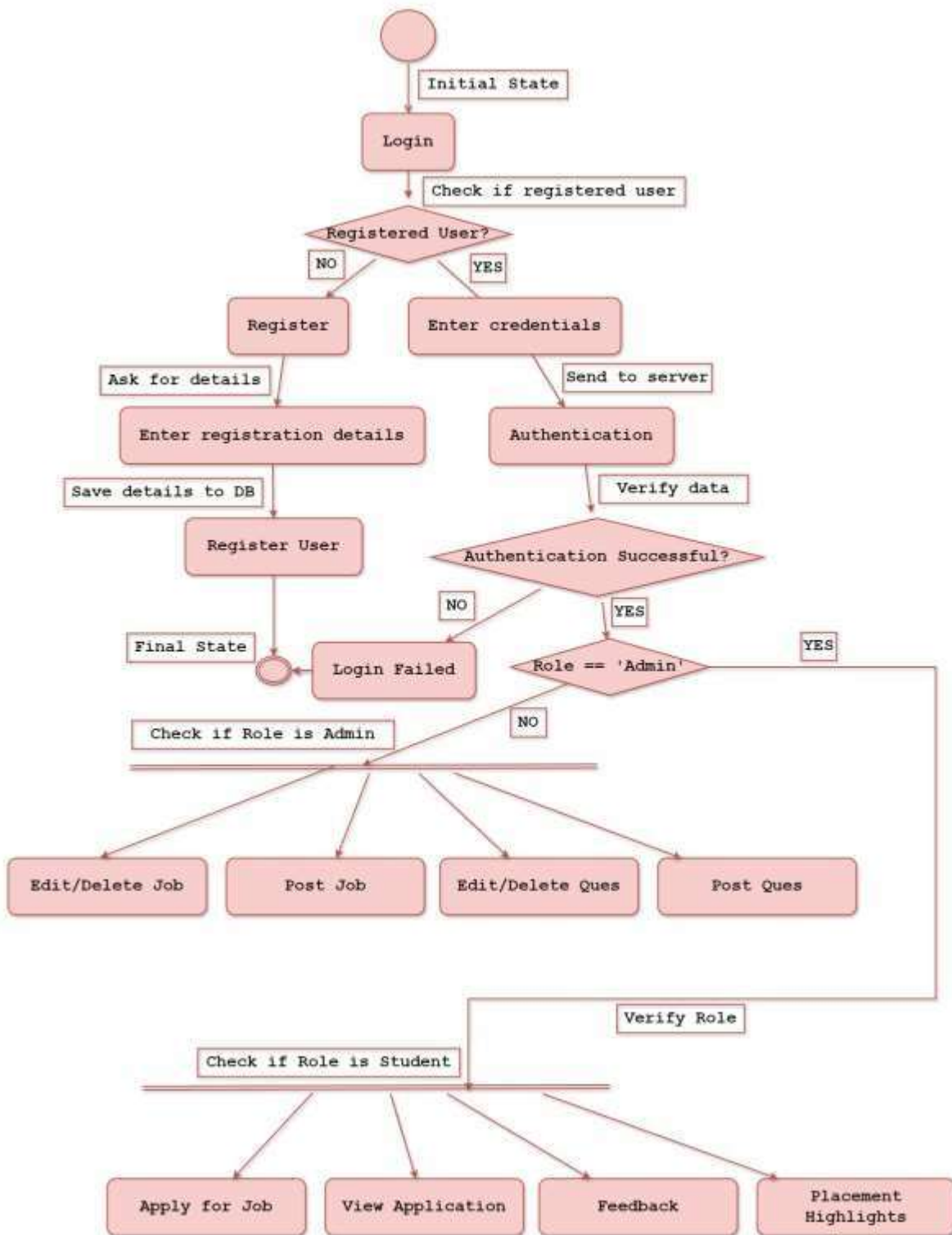
a) Student



b) Admin



3.1.8 State Transition Diagram



3.2 Database Design

Users Table:

S.NO.	Field Name	Field Type	Description
1	Name	Text	Contains user name
2	Email	Text	Contains users Email IDs
3	Password	Text	Contains Password
4	Phone	Number	Contains user contact
5	Role	Enum	[Admin,Student]

Table 3.1 Users Table

Applications Table:

S.NO.	Field Name	Field Type	Description
1	name	Text	Contains user's first name
2	email	Email	Contains user Email IDs
4	phone	Text	Contains user Phone Number
5	address	Text	Contains user Address
6	coverLetter	Text	Contains job cover letter
7	resume	Document	Contains user's resume

Table 3.2 Applications Table

Job Table:

S.NO.	Field Name	Field Type	Description
1	title	Text	Contains job title
2	description	Text	Contains job description
4	category	Text	Contains job category
5	country	Text	Contains job country
6	city	Text	Contains job city
7	location	Text	Contains job location
8	expired	Boolean	[True, False]
9	salary	Number	Contains job salary (Fixed/Ranged)

Table 3.3 Job Table

Question Table:

S.NO.	Field Name	Field Type	Description
1	subject	Text	Contains interview subject
2	question	Text	Contains interview question
4	answer	Text	Contains interview answer

Table 3.4 Question Table

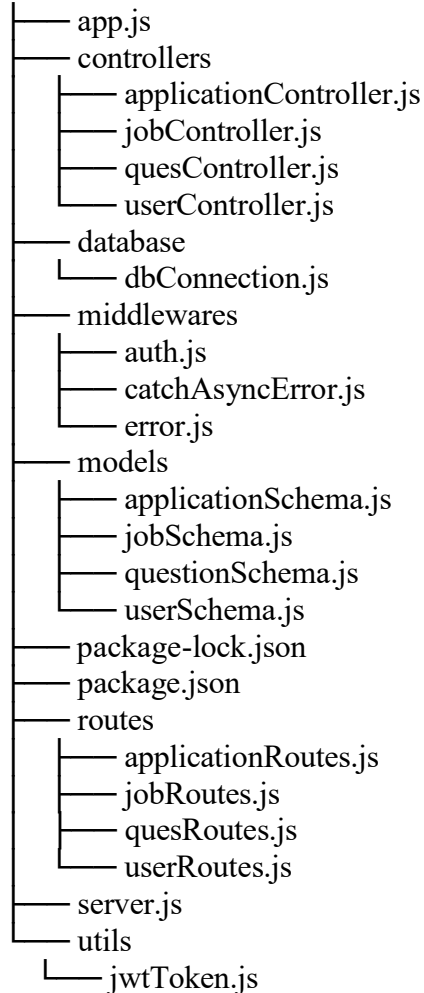
CHAPTER - 4

SYSTEM DEVELOPMENT AND TESTING

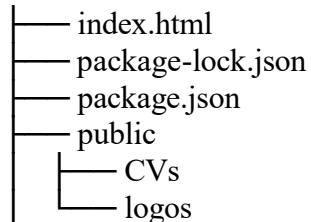
4.1 SYSTEM DEVELOPMENT

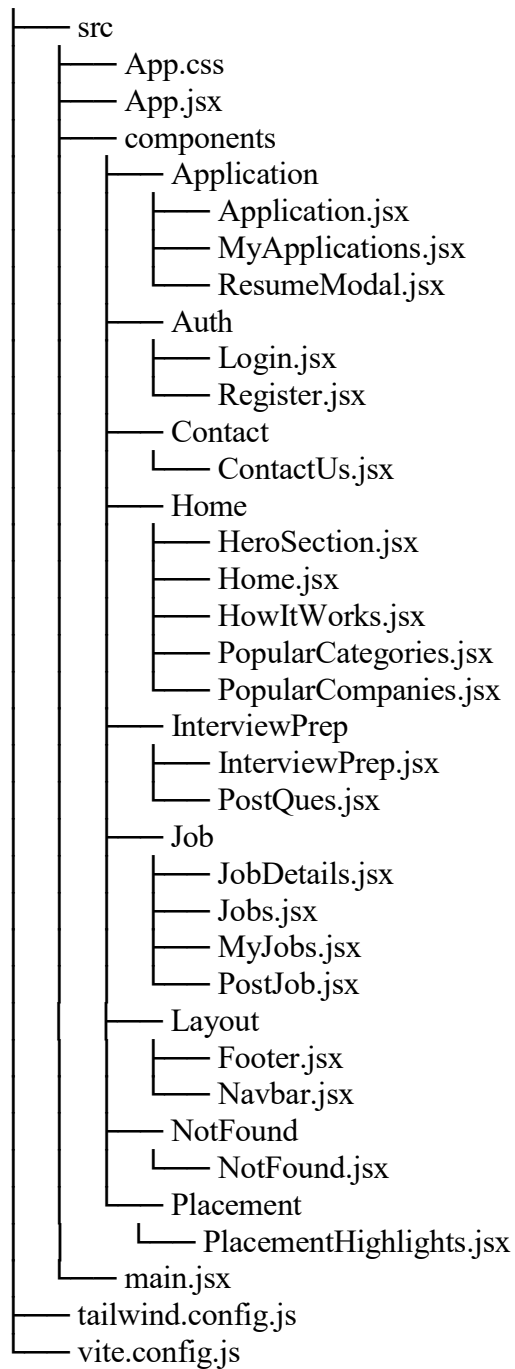
4.1.1 Files in the Project: An overview

backend



frontend





4.1.2 Dependencies in the project:

Server-side dependencies

```
"bcrypt": "^5.1.1",  
"cloudinary": "^1.41.2",  
"cookie-parser": "^1.4.6",  
"cors": "^2.8.5",  
"dotenv": "^16.3.1",  
"express": "^4.18.2",  
"express-fileupload": "^1.4.3",  
"jsonwebtoken": "^9.0.2",  
"mongoose": "^8.0.3",  
"validator": "^13.11.0"
```

Client-side dependencies

```
"axios": "^1.6.5",  
"emailjs-com": "^3.2.0",  
"react": "^18.2.0",  
"react-dom": "^18.2.0",  
"react-hot-toast": "^2.4.1",  
"react-icons": "^4.12.0",  
"react-router-dom": "^6.21.1",  
"recharts": "^2.12.3"
```

server.js

```
import app from "./app.js";
import cloundinary from "cloudinary";

cloudinary.v2.config({
  cloud_name: process.env.CLOUDINARY_CLIENT_NAME,
  api_key: process.env.CLOUDINARY_CLIENT_API,
  api_secret: process.env.CLOUDINARY_CLIENT_SECRET,
});

app.listen(process.env.PORT, () => {
  console.log(`Server running at port ${process.env.PORT}`);
});
```

dbConnection.js

```
import mongoose from "mongoose";

export const dbConnection = () => {
  mongoose
    .connect(process.env.MONGO_URI, {
      dbName: "Placement_Office",
    })
    .then(() => {
      console.log("Connected to MongoDB.");
    })
    .catch((err) => {
      console.log(`Some Error occured. ${err}`);
    });
};
```

app.js

```
import express from "express";
import { dbConnection } from "../database/dbConnection.js";
import jobRouter from "../routes/jobRoutes.js";
import userRouter from "../routes/userRoutes.js";
import applicationRouter from "../routes/applicationRoutes.js";
import quesRouter from "../routes/quesRoutes.js";
```

```

import { config } from "dotenv";
import cors from "cors";
import { errorMiddleware } from "../middlewares/error.js";
import cookieParser from "cookie-parser";
import fileUpload from "express-fileupload";

const app = express();
config({ path: "../config/config.env" });

app.use(
  cors({
    origin: [process.env.FRONTEND_URL],
    method: ["GET", "POST", "DELETE", "PUT"],
    credentials: true,
  })
);

app.use(cookieParser());
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

app.use(
  fileUpload({
    useTempFiles: true,
    tempFileDir: "/tmp/",
  })
);

app.use("/user", userRouter);
app.use("/job", jobRouter);
app.use("/application", applicationRouter);
app.use("/interview", quesRouter);

app.get('/download-image', async (req, res) => {
  const imageUrl = req.query.url; // Get the image URL from query
  params
  try {
    const response = await axios({
      url: imageUrl,

```

```

        method: 'GET',
        responseType: 'stream',
    });
    res.setHeader('Content-Disposition', `attachment;
filename="${decodeURIComponent(imageUrl.split('/').pop())}"`);
    response.data.pipe(res);
} catch (error) {
    res.status(500).send("Error downloading the image");
}
});
dbConnection();

app.use(errorMiddleware);
export default app;

```

auth.js

```

import { User } from "../models/userSchema.js";
import { catchAsyncErrors } from "../catchAsyncError.js";
import ErrorHandler from "../error.js";
import jwt from "jsonwebtoken";

export const isAuthenticated = catchAsyncErrors(async (req, res, next)
=> {
    const { token } = req.cookies;
    if (!token) {
        return next(new ErrorHandler("User Not Authorized", 401));
    }
    const decoded = jwt.verify(token, process.env.JWT_SECRET_KEY);

    req.user = await User.findById(decoded.id);

    next();
});

```

userSchema.js

```

import mongoose from "mongoose";
import validator from "validator";
import bcrypt from "bcrypt";

```



```

import jwt from "jsonwebtoken";
const userSchema = new mongoose.Schema({
  name: {
    type: String,
    required: [true, "Please enter your Name!"],
    minLength: [3, "Name must contain at least 3 Characters!"],
    maxLength: [30, "Name cannot exceed 30 Characters!"],
  },
  email: {
    type: String,
    required: [true, "Please enter your Email!"],
    validate: [validator.isEmail, "Please provide a valid Email!"],
  },
  phone: {
    type: Number,
    required: [true, "Please enter your Phone Number!"],
  },
  password: {
    type: String,
    required: [true, "Please provide a Password!"],
    minLength: [8, "Password must contain at least 8 characters!"],
    maxLength: [32, "Password cannot exceed 32 characters!"],
    select: false,
  },
  role: {
    type: String,
    required: [true, "Please select a role"],
    enum: ["Student", "Admin"],
  },
  createdAt: {
    type: Date,
    default: Date.now,
  },
});

```

```

userSchema.pre("save", async function (next) {
  if (!this.isModified("password")) {
    next();
  }
});

```

```

    }
    this.password = await bcrypt.hash(this.password, 10);
  });

userSchema.methods.comparePassword = async function (enteredPassword)
{
  return await bcrypt.compare(enteredPassword, this.password);
};

userSchema.methods.getJWTToken = function () {
  return jwt.sign({ id: this._id }, process.env.JWT_SECRET_KEY, {
    expiresIn: process.env.JWT_EXPIRES,
  });
};

export const User = mongoose.model("User", userSchema);

```

UserController.js

```

import { catchAsyncErrors } from "../middlewares/catchAsyncError.js";
import { User } from "../models/userSchema.js";
import ErrorHandler from "../middlewares/error.js";
import { sendToken } from "../utils/jwtToken.js";

export const register = catchAsyncErrors(async (req, res, next) => {
  const { name, email, phone, password, role } = req.body;
  if (!name || !email || !phone || !password || !role) {
    return next(new ErrorHandler("Please fill full form!"));
  }
  const isEmail = await User.findOne({ email });
  if (isEmail) {
    return next(new ErrorHandler("Email already registered!"));
  }
  const user = await User.create({
    name,
    email,
    phone,
    password,
    role,
  });
  sendToken(user, 201, res);
});

```

```

    });
    sendToken(user, 201, res, "User Registered!");
  });

export const login = catchAsyncErrors(async (req, res, next) => {
  const { email, password, role } = req.body;
  if (!email || !password || !role) {
    return next(new ErrorHandler("Please provide email, password and role."));
  }
  const user = await User.findOne({ email }).select("+password");
  if (!user) {
    return next(new ErrorHandler("Invalid Email Or Password.", 400));
  }
  const isPasswordMatched = await user.comparePassword(password);
  if (!isPasswordMatched) {
    return next(new ErrorHandler("Invalid Email Or Password.", 400));
  }
  if (user.role !== role) {
    return next(
      new ErrorHandler(`User with provided email and ${role} not found!`, 404)
    );
  }
  sendToken(user, 201, res, "User Logged In!");
});

export const logout = catchAsyncErrors(async (req, res, next) => {
  res
    .status(201)
    .cookie("token", "", {
      httpOnly: true,
      expires: new Date(Date.now()),
    })
    .json({
      success: true,
      message: "Logged Out Successfully.",
    });
});

```

```
});
```

```
export const getUser = catchAsyncErrors((req, res, next) => {  
  const user = req.user;  
  res.status(200).json({  
    success: true,  
    user,  
  });  
});
```

userRoutes.js

```
import express from "express";  
import { login, register, logout, getUser } from  
  "../controllers/userController.js";  
import { isAuthenticated } from "../middlewares/auth.js";  
  
const router = express.Router();  
  
router.post("/register", register);  
router.post("/login", login);  
router.get("/logout", isAuthenticated, logout);  
router.get("/getuser", isAuthenticated, getUser);  
  
export default router;
```

main.jsx

```
import React, { createContext, useState } from "react";  
import ReactDOM from "react-dom/client";  
import App from "./App.jsx";  
  
export const Context = createContext({  
  isAuthorized: false,  
});  
  
const AppWrapper = () => {  
  const [isAuthorized, setIsAuthorized] = useState(false);  
  const [user, setUser] = useState({});
```

```

    return (
      <Context.Provider
        value={{
          isAuthenticated,
          setIsAuthorized,
          user,
          setUser,
        }}
      >
        <App />
      </Context.Provider>
    );
  };

ReactDOM.createRoot(document.getElementById("root")).render(
  <React.StrictMode>
    <AppWrapper />
  </React.StrictMode>
);

```

App.jsx

```

import React, { useContext, useEffect } from "react";
import "./App.css";
import { Context } from "./main";
import { BrowserRouter, Route, Routes } from "react-router-dom";
import Login from "./components/Auth/Login";
import Register from "./components/Auth/Register";
import { Toaster } from "react-hot-toast";
import axios from "axios";
import Navbar from "./components/Layout/Navbar";
import Footer from "./components/Layout/Footer";
import Home from "./components/Home/Home";
import Jobs from "./components/Job/Jobs";
import JobDetails from "./components/Job/JobDetails";
import Application from "./components/Application/Application";
import MyApplications from "./components/Application/MyApplications";
import PostJob from "./components/Job/PostJob";

```

```

import NotFound from "../components/NotFound/NotFound";
import MyJobs from "../components/Job/MyJobs";
import InterviewPrep from "../components/InterviewPrep/InterviewPrep";
import PostQues from "../components/InterviewPrep/PostQues";
import ContactUs from "../components/Contact/ContactUs";
import PlacementHighlights from
"../components/Placement/PlacementHighlights";

const App = () => {

  const { isAuthorized, setIsAuthorized, setUser } =
  useContext(Context);
  useEffect(() => {
    const fetchUser = async () => {
      try {
        const response = await axios.get(
          "http://localhost:3001/user/getuser",
          {
            withCredentials: true,
          }
        );
        setUser(response.data.user);
        setIsAuthorized(true);
      } catch (error) {
        setIsAuthorized(false);
      }
    };
    fetchUser();
  }, [isAuthorized]);

  return (
    <>
      <BrowserRouter>
        <Navbar />
        <Routes>
          <Route path="/login" element={<Login />} />
          <Route path="/register" element={<Register />} />
          <Route path="/" element={<Home />} />
        </Routes>
      </BrowserRouter>
    </>
  );
};

```

```

    <Route path="/job/getall" element={<Jobs />} />
    <Route path="/job/:id" element={<JobDetails />} />
    <Route path="/application/:id" element={<Application />} />
    <Route path="/applications/me" element={<MyApplications />}
  />

  <Route path="/job/post" element={<PostJob />} />
  <Route path="/job/me" element={<MyJobs />} />
  <Route path="/interview/getallques" element={<InterviewPrep
/>} />

  <Route path="/interview/postques" element={<PostQues />} />
  <Route path="/contact" element={<ContactUs />} />
  <Route path="/placement" element={<PlacementHighlights />}
/>

  <Route path="*" element={<NotFound />} />
</Routes>
<Footer />
<Toaster />
</BrowserRouter>
</>
);
};

```

```
export default App;
```

Login.jsx

```

import React, { useContext, useState } from "react";
import { MdOutlineMailOutline } from "react-icons/md";
import { RiLock2Fill } from "react-icons/ri";
import { Link, Navigate } from "react-router-dom";
import { FaRegUser } from "react-icons/fa";
import axios from "axios";
import toast from "react-hot-toast";
import { Context } from "../../main";

const Login = () => {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [role, setRole] = useState("");

```

```

const { isAuthorized, setIsAuthorized } = useContext(Context);

const handleLogin = async (e) => {
  e.preventDefault();
  try {
    const { data } = await axios.post(
      "http://localhost:3001/user/login",
      { email, password, role },
      {
        headers: {
          "Content-Type": "application/json",
        },
        withCredentials: true,
      }
    );
    toast.success(data.message);
    setIsAuthorized(true);
  } catch (error) {
    toast.error(error.response.data.message);
  }
};

if (isAuthorized) {
  return <Navigate to="/" />;
}

return (
  <div className="flex flex-col md:flex-row justify-center items-center min-h-screen gap-10">
    <div className="flex-1 max-w-md px-4 py-8 bg-white shadow-lg rounded-lg mx-4 md:mx-8 lg:mx-16 mt-8">
      <h2 className="mb-8 text-center text-4xl font-bold text-gray-900">Login to your account</h2>
      <form className="space-y-6" onSubmit={handleLogin}>
        <div className="space-y-4">
          <div className="relative">
            <select

```



```

        id="role"
        name="role"
        className="appearance-none block w-full px-3 py-2
border border-gray-300 rounded-md shadow-sm placeholder-gray-400
focus:outline-none focus:ring-indigo-500 focus:border-indigo-500
sm:text-sm"
        value={role}
        onChange={(e) => setRole(e.target.value)}
      >
        <option value="">Select Role</option>
        <option value="Admin">Admin</option>
        <option value="Student">Student</option>
      </select>
      <div className="pointer-events-none absolute inset-y-0
right-0 flex items-center px-2 text-gray-400">
        <FaRegUser className="h-5 w-5"/>
      </div>
    </div>

    <div className="relative">
      <input
        type="email"
        name="email"
        id="email-address"
        autoComplete="email"
        required
        className="appearance-none block w-full px-3 py-2
border border-gray-300 rounded-md shadow-sm placeholder-gray-400
focus:outline-none focus:ring-indigo-500 focus:border-indigo-500
sm:text-sm"
        placeholder="Email address"
        value={email}
        onChange={(e) => setEmail(e.target.value)}
      />
      <div className="pointer-events-none absolute inset-y-0
right-0 flex items-center px-2 text-gray-400" >
        <MdOutlineMailOutline className="h-6 w-6"/>
      </div>
    </div>

```

```

    </div>

    <div className="relative">
      <input
        type="password"
        name="password"
        id="password"
        autoComplete="current-password"
        required
        className="appearance-none block w-full px-3 py-2
border border-gray-300 rounded-md shadow-sm placeholder-gray-400
focus:outline-none focus:ring-indigo-500 focus:border-indigo-500
sm:text-sm"
        placeholder="Password"
        value={password}
        onChange={(e) => setPassword(e.target.value)}
      />
      <div className="pointer-events-none absolute inset-y-0
right-0 flex items-center px-2 text-gray-400" >
        <RiLock2Fill className="h-6 w-6" />
      </div>
    </div>
  </div>

  <div>
    <button
      type="submit"
      className="group relative w-full flex justify-center py-
2 px-4 border border-transparent text-sm font-medium rounded-md text-
white bg-indigo-600 hover:bg-indigo-700 focus:outline-none focus:ring-
2 focus:ring-offset-2 focus:ring-indigo-500"
    >
      Login
    </button>
  </div>

  <div className="text-sm text-center">

```

```

        <Link to="/register" className="font-medium text-indigo-600 hover:text-indigo-500">
            Don't have an account? Register
        </Link>
    </div>
</form>
</div>

<div className="flex-1 flex justify-center items-center w-full max-w-lg px-4 md:px-0 ">
    
</div>
</div>
);
};

```

```
export default Login;
```

Home.jsx

```

import React from "react";
import { useContext } from "react";
import { Context } from "../main";
import { Navigate } from "react-router-dom";
import HeroSection from "./HeroSection";
import HowItWorks from "./HowItWorks";
import PopularCategories from "./PopularCategories";
import PopularCompanies from "./PopularCompanies";

const Home = () => {
    const { isAuthorized } = useContext(Context);
    if (!isAuthorized) {
        return <Navigate to="/login" />;
    }
    return (

```

```

    <>
      <section>
        <HeroSection />
        <HowItWorks />
        <PopularCategories />
        <PopularCompanies />
      </section>
    </>
  );
};

export default Home;

```

PostJob.jsx

```

import React, { useContext, useEffect, useState } from 'react';
import axios from 'axios';
import toast from 'react-hot-toast';
import { useNavigate } from 'react-router-dom';
import { Context } from '../main';

const PostJob = () => {
  const [title, setTitle] = useState('');
  const [description, setDescription] = useState('');
  const [category, setCategory] = useState('');
  const [country, setCountry] = useState('');
  const [city, setCity] = useState('');
  const [location, setLocation] = useState('');
  const [salaryFrom, setSalaryFrom] = useState('');
  const [salaryTo, setSalaryTo] = useState('');
  const [fixedSalary, setFixedSalary] = useState('');
  const [salaryType, setSalaryType] = useState('default');

  const { isAuthorized, user } = useContext(Context);
  const navigateTo = useNavigate();

  useEffect(() => {
    if (!isAuthorized || (user && user.role !== 'Admin')) {
      navigateTo('/');
    }
  });

```

```

    }
  }, [isAuthorized, user, navigateTo]));

const handleJobPost = async (e) => {
  e.preventDefault();
  const postData = salaryType === 'Fixed Salary'
    ? { title, description, category, country, city, location,
fixedSalary }
    : { title, description, category, country, city, location,
salaryFrom, salaryTo };

  try {
    const response = await axios.post(
      'http://localhost:3001/job/post',
      postData,
      {
        withCredentials: true,
        headers: { 'Content-Type': 'application/json' },
      }
    );
    toast.success(response.data.message);
  } catch (err) {
    toast.error(err.response?.data.message || 'An error occurred');
  }
};

return (
  <div className="min-h-screen pt-8">
    <div className="container mx-auto max-w-4xl p-8 bg-white bg-
opacity-70 rounded-xl shadow-2xl">
      <h2 className="text-2xl font-bold text-center text-gray-800
mb-8">Post New Job</h2>
      <form onSubmit={handleJobPost} className="space-y-6">
        <div className="space-y-4">
          <input
            type="text"
            value={title}
            onChange={(e) => setTitle(e.target.value)}

```

```

        placeholder="Job Title"
        className="w-full px-4 py-2 border border-gray-300
rounded-md focus:ring-blue-500 focus:border-blue-500"
    />
    <select
        value={category}
        onChange={(e) => setCategory(e.target.value)}
        className="w-full px-4 py-2 border border-gray-300
rounded-md focus:ring-blue-500 focus:border-blue-500"
    >
        <option value="">Select Category</option>
        <option value="Graphics & Design">Graphics &
Design</option>
        <option value="Mobile App Development">Mobile App
Development</option>
        <option value="Frontend Web Development">Frontend Web
Development</option>
        <option value="MERN Stack Development">MERN STACK
Development</option>
        <option value="Account & Finance">Account &
Finance</option>
        <option value="Artificial Intelligence">Artificial
Intelligence</option>
        <option value="Video Animation">Video Animation</option>
        <option value="MEAN Stack Development">MEAN STACK
Development</option>
        <option value="MEVN Stack Development">MEVN STACK
Development</option>
        <option value="Data Entry Operator">Data Entry
Operator</option>
    </select>
    <div className="flex gap-4">
        <input
            type="text"
            value={country}
            onChange={(e) => setCountry(e.target.value)}
            placeholder="Country"

```

```

        className="w-1/2 px-4 py-2 border border-gray-300
rounded-md focus:ring-blue-500 focus:border-blue-500"
      />
      <input
        type="text"
        value={city}
        onChange={(e) => setCity(e.target.value)}
        placeholder="City"
        className="w-1/2 px-4 py-2 border border-gray-300
rounded-md focus:ring-blue-500 focus:border-blue-500"
      />
    </div>
    <input
      type="text"
      value={location}
      onChange={(e) => setLocation(e.target.value)}
      placeholder="Location"
      className="w-full px-4 py-2 border border-gray-300
rounded-md focus:ring-blue-500 focus:border-blue-500"
    />
  </div>
  <div className="space-y-4">
    <select
      value={salaryType}
      onChange={(e) => setSalaryType(e.target.value)}
      className="w-full px-4 py-2 border border-gray-300
rounded-md focus:ring-blue-500 focus:border-blue-500"
    >
      <option value="default">Select Salary Type</option>
      <option value="Fixed Salary">Fixed Salary</option>
      <option value="Ranged Salary">Ranged Salary</option>
    </select>
    {salaryType === 'Fixed Salary' && (
      <input
        type="number"
        placeholder="Enter Fixed Salary"
        value={fixedSalary}
        onChange={(e) => setFixedSalary(e.target.value)}

```

```

        className="w-full px-4 py-2 border border-gray-300
rounded-md focus:ring-blue-500 focus:border-blue-500"
      />
    )}
    {salaryType === 'Ranged Salary' && (
      <div className="flex gap-4">
        <input
          type="number"
          placeholder="Salary From"
          value={salaryFrom}
          onChange={(e) => setSalaryFrom(e.target.value)}
          className="w-1/2 px-4 py-2 border border-gray-300
rounded-md focus:ring-blue-500 focus:border-blue-500"
        />
        <input
          type="number"
          placeholder="Salary To"
          value={salaryTo}
          onChange={(e) => setSalaryTo(e.target.value)}
          className="w-1/2 px-4 py-2 border border-gray-300
rounded-md focus:ring-blue-500 focus:border-blue-500"
        />
      </div>
    )}
  </div>
  <textarea
    rows="5"
    value={description}
    onChange={(e) => setDescription(e.target.value)}
    placeholder="Job Description"
    className="w-full p-4 border border-gray-300 rounded-md
focus:ring-blue-500 focus:border-blue-500"
  />
  <button
    type="submit"
    className="w-full px-4 py-2 bg-blue-600 text-white
rounded-md hover:bg-blue-700 focus:outline-none focus:ring-4
focus:ring-blue-500 focus:ring-opacity-50"

```



```

        >
        Create Job
      </button>
    </form>
  </div>
</div>
);
};
export default PostJob;

```

HeroSection.jsx

```

import React from "react";
const HeroSection = () => {
  return (
    <div className=" text-gray-600 px-10">
      <div className="container mx-auto px-4 py-12">
        <div className="md:flex md:items-center md:justify-between">
          <div className="mb-4 md:mb-0 md:w-1/2">
            <h1 className="text-3xl md:text-4xl font-bold text-gray-800">Find a job that suits your interests and skills</h1>
            <p className="mt-2 text-base md:text-lg">
              Embark on a journey where your skills meet purpose,
              transforming passion into profession. Make everyday an opportunity to
              excel and inspire.
            </p>
          </div>
          <div className="md:w-1/2">
            
          </div>
        </div>
      </div>
    </div>
  );
};
export default HeroSection;

```

Navbar.jsx

```
import React, { useContext, useState } from "react";
import { Context } from "../../main";
import { Link, useNavigate } from "react-router-dom";
import axios from "axios";
import toast from "react-hot-toast";
import { GiHamburgerMenu } from "react-icons/gi";
import { AiOutlineClose } from "react-icons/ai";

const Navbar = () => {
  const [show, setShow] = useState(false);
  const { isAuthorized, setIsAuthorized, user } = useContext(Context);
  const navigateTo = useNavigate();

  const handleLogout = async () => {
    try {
      const response = await
    axios.get("http://localhost:3001/user/logout", {
      withCredentials: true,
    });
    toast.success(response.data.message);
    setIsAuthorized(false);
    navigateTo("/login");
    } catch (error) {
      toast.error(error.response.data.message);
      setIsAuthorized(true);
    }
  };

  return (
    <nav className={`bg-gray-800 p-5 ${isAuthorized ? "block" :
    "hidden"} `>
      <div className="container mx-auto flex justify-between items-
    center">
        <Link to="/" onClick={() => setShow(false)} className="text-
    white hover:text-gray-200 transition duration-150 flex items-center
    mr-3 ">

          
```

```

        <span className="font-semibold text-xl text-blue-50 font-
serif">Placement Office</span>
    </Link>
    <div className="md:hidden">
        {show ? (
            <AiOutlineClose onClick={() => setShow(false)}
className="text-white text-2xl" />
        ) : (
            <GiHamburgerMenu onClick={() => setShow(true)}
className="text-white text-2xl" />
        )}
    </div>
    <div
        className={`absolute top-16 right-0 bg-gray-800 p-2 z-20
transform ${show ? "translate-x-0" : "translate-x-full"} transition-
transform duration-300 ease-in-out md:relative md:top-0 md:translate-
x-0 md:flex md:flex-row md:items-center md:space-x-6`}
    >
        <ul className="flex flex-col md:flex-row items-center space-
y-5 md:space-y-0 md:space-x-6 w-full text-center">
            <li>
                <Link to="/job/getall" onClick={() => setShow(false)}
className="text-white hover:text-gray-200 transition duration-150">
                    All Jobs
                </Link>
            </li>
            <li>
                <Link to="/interview/getallques" onClick={() =>
setShow(false)} className="text-white hover:text-gray-200 transition
duration-150">
                    Interview Ques
                </Link>
            </li>
            {user && user.role === "Admin" && (
                <>
                    <li>

```

```

        <Link to="/interview/postques" onClick={() =>
setShow(false)} className="text-white hover:text-gray-200 transition
duration-150">
            Post Ques
        </Link>
    </li>
    <li>
        <Link to="/job/post" onClick={() => setShow(false)}
className="text-white hover:text-gray-200 transition duration-150">
            Post Job
        </Link>
    </li>
    <li>
        <Link to="/job/me" onClick={() => setShow(false)}
className="text-white hover:text-gray-200 transition duration-150">
            My Jobs
        </Link>
    </li>
</>
)}
<li>
    <Link to={"/applications/me"} onClick={() =>
setShow(false)} className="text-white hover:text-gray-200 transition
duration-150">
        {user && user.role === "Admin" ? "Student's
Applications" : "My Applications"}
    </Link>
</li>
{user && user.role === "Student" && (
    <>
        <li>
            <Link to="/contact" onClick={() => setShow(false)}
className="text-white hover:text-gray-200 transition duration-150">
                Contact
            </Link>
        </li>
        <li>

```

```

        <Link to="/placement" onClick={() => setShow(false)}
className="text-white hover:text-gray-200 transition duration-150">
            Placement Highlights
        </Link>
    </li>
</>
)}
<li>
    <button onClick={handleLogout} className="bg-red-500
text-white py-2 px-4 rounded hover:bg-red-700 transition duration-
150">
        LOGOUT
    </button>
</li>
</ul>
</div>
</div>
</nav>
);
};
export default Navbar;

```

Footer.jsx

```

import React, { useContext } from "react";
import { Context } from "../../main";
import { FaTwitter, FaLinkedinIn, FaFacebook } from "react-icons/fa";
const Footer = () => {
    const { isAuthorized } = useContext(Context);
    return (
        <footer className={`bg-gray-800 text-white p-4 mt-4 ${isAuthorized
? "block" : "hidden"}`} >
            <div className="container mx-auto">
                <div className="flex flex-col md:flex-row items-center
justify-between">
                    <div className="text-center md:text-left mb-4 md:mb-0">
                        <p className="text-2xl font-bold">Placement Office</p>
                        <p className="mt-2">Your go-to platform for all placement
related activities.</p>

```

```

    </div>
    <div className="text-center md:text-left mb-4 md:mb-0">
      <p className="text-lg font-semibold">Follow Us</p>
      <div className="flex justify-center md:justify-start
space-x-4">
        <a href="https://www.facebook.com" className="text-
white-500 hover:text-gray-700" target="_blank" rel="noopener
noreferrer">
          <FaFacebook />
        </a>
        <a href="https://www.twitter.com" className="text-white-
400 hover:text-gray-600" target="_blank" rel="noopener noreferrer">
          <FaTwitter />
        </a>
        <a href="https://www.linkedin.com" className="text-
white-800 hover:text-gray-900" target="_blank" rel="noopener
noreferrer">
          <FaLinkedinIn />
        </a>
      </div>
    </div>
    <div className="text-center md:text-left">
      <p className="text-lg font-semibold">Contact Us</p>
      <p>Email: info@placementtrackerapp.com</p>
      <p>Phone: +1 (123) 456-7890</p>
    </div>
  </div>
  <div className="mt-6 text-center">
    <p className="text-sm">&copy; {new Date().getFullYear()}
Placement Office App. All rights reserved.</p>
  </div>
</div>
</footer>
);
};
export default Footer;

```

4.2 SYSTEM TESTING

4.2.1 Introduction:

Software testing is a thorough investigation carried out to furnish stakeholders with insights into the quality of the tested product or service. Its objective nature provides an independent viewpoint, enabling businesses to comprehend and evaluate the risks associated with software implementation. The methodologies for testing encompass the execution of a program or application, aiming to uncover bugs, errors, or other defects.

Describing software testing involves the validation and verification of a computer program, application, or product to ensure it:

- ❖ Aligns with the requirements that guided its design and development.
- ❖ Functions as anticipated.
- ❖ Can be implemented with consistent characteristics, meeting the stakeholders' needs.

It is crucial to acknowledge that testing cannot completely identify all defects within software. Instead, it offers a critical evaluation by comparing the state and behaviour of the product against established principles or mechanisms, known as oracles. These oracles may include specifications, contracts, comparable products, past versions, expected purposes, user expectations, standards, laws, or other relevant criteria.

The primary goal of testing is to identify software failures to facilitate the discovery and correction of defects. However, it cannot guarantee that a product functions flawlessly under all conditions; it can only determine that it malfunctions under specific circumstances. The scope of software testing encompasses code examination, code execution in various environments, and evaluating whether the code performs its intended and necessary functions.

In the contemporary landscape of software development, testing organizations are often distinct from development teams, with various roles assigned to testing team members. Insights derived from software testing play a pivotal role in refining the software development process.

A fundamental challenge in software testing is the impracticality of testing under all combinations of inputs and preconditions, even for a simple product. This implies that the number of defects in a software product can be extensive, and infrequently occurring defects are challenging to detect through testing. Moreover, non-functional quality dimensions, such as usability, scalability, performance, compatibility, and reliability, can be subjective, with value perceptions varying among individuals.

Types of Testing:

- a) Unit Testing
- b) System Testing
- c) Integration testing

4.2.2 Unit Testing and System Testing of Placement Office App

A comprehensive testing phase for the Placement Tracker App was done, employing a meticulous approach to ensure its robustness and functionality. This testing process involved two key aspects: Unit Testing and System Testing.

Unit Testing:

In the Unit Testing phase, each module of the Placement Tracker App underwent rigorous examination in isolation. This granular testing approach allowed me to scrutinize the individual components and functionalities of the app to ensure they operate as intended. By breaking down the application into smaller, manageable units, I could pinpoint and address any issues or discrepancies within specific modules.

During Unit Testing, specific test cases were designed to assess the correctness of functions, methods, and procedures within each module. This process facilitated the identification and resolution of errors, ensuring that every unit of the app functioned independently and seamlessly.

System Testing:

Following the successful completion of Unit Testing, the Placement Tracker App underwent System Testing. This phase involved testing the entire application as an integrated system to evaluate its overall performance and compliance with specified requirements.

System Testing provided a holistic view of the app's behaviour, examining how different modules interacted with each other and identifying any potential integration issues. The goal was to verify that the app's components collaboratively delivered the intended outcomes and that the system, as a whole, met the defined objectives.

Throughout the System Testing process, various scenarios were simulated to mimic real-world usage, ensuring the app's resilience under different conditions. This included testing functionalities such as user registration, data storage and retrieval, and the overall user interface.

Key Outcomes:

The Unit Testing and System Testing phases were instrumental in enhancing the reliability and performance of the Placement Tracker App. Identified issues were promptly addressed, and the testing process provided valuable insights into the app's strengths and areas for improvement.

This rigorous testing approach aligns with industry best practices, emphasizing the importance of both unit-level and system-level evaluations to deliver a robust and reliable software application. The Placement Tracker App is now better positioned to provide a seamless experience for users, meeting the high standards expected in the dynamic field of placement tracking.

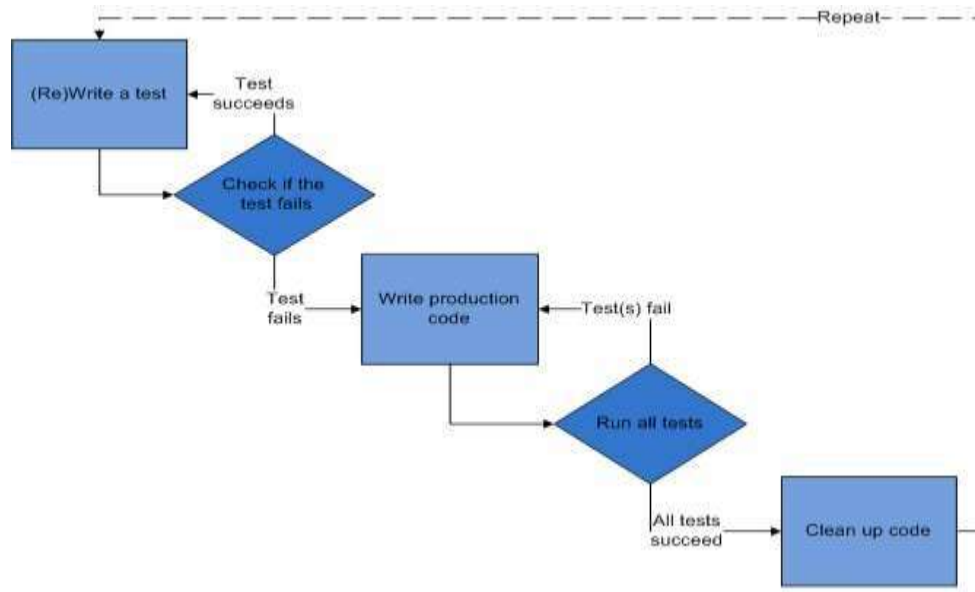


Fig 4.7: Unit Testing of Placement Office

Integration Testing

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing. The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items", i.e. assemblages (or groups of units), are exercised through their interfaces using black box testing, success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e. unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.

Test Case 1: Register Module

Test Description: Testing to see if the Registration is Successful or Unsuccessful

TEST INPUT	Expected Result	Actual Result	Pass/Fail
Sign up is not successful because of empty field	Warning while attempting to register	Warning “Please fill out this field”	Pass
Email field is empty	Warning while attempting to register	Warning “please include @ in the email address column or fill out this field”	Pass
Signup Successful	User can browse on the website and order product	“User Registered!”	Pass

Table 4.2.1

Test Case 2: Login Module

Test Description: Testing to see if the login is Successful or Unsuccessful

TEST INPUT	Expected Result	Actual Result	Pass/Fail
Login is not successful Because of wrong details	Error while login due to wrong credentials	Error Message “Login unsuccessful”	Pass
Email field is empty	Warning while attempting to login	Warning “please include @ in the email address column or fill out this field”	Pass
Login Successful	User can browse on the website and view homepage	"User Logged In!"	Pass

Table 4.2.2

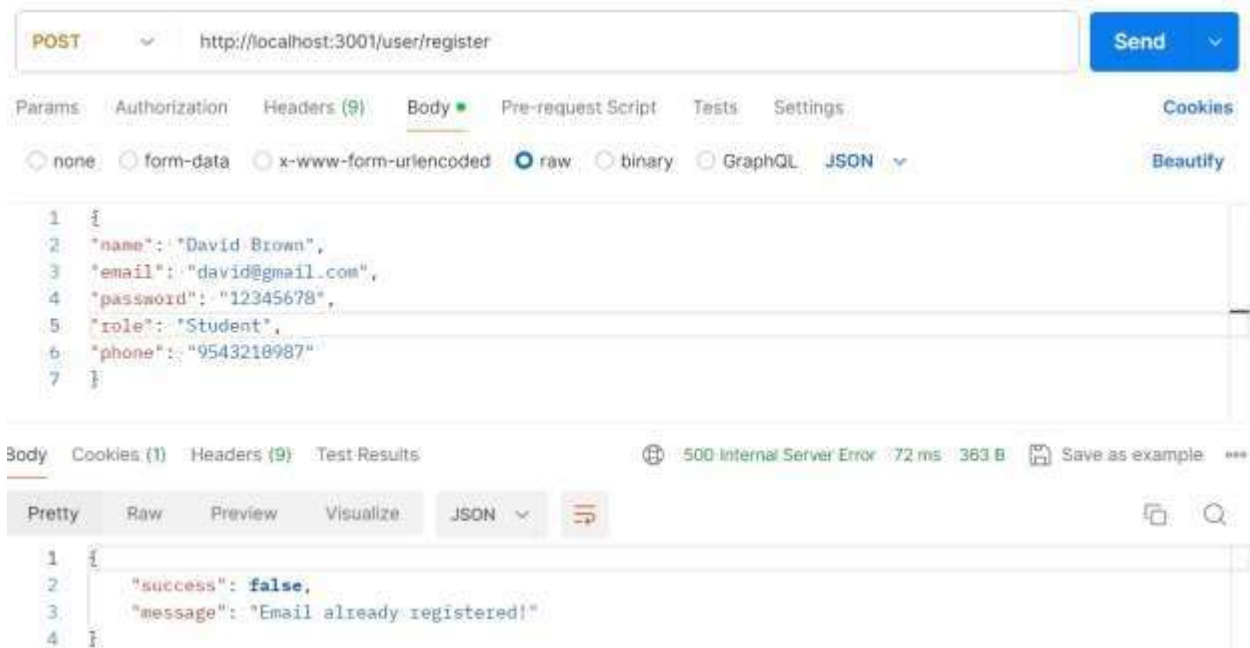
Test Case 3: Post Job module

Test Description: Testing to see if the admin can post job or not

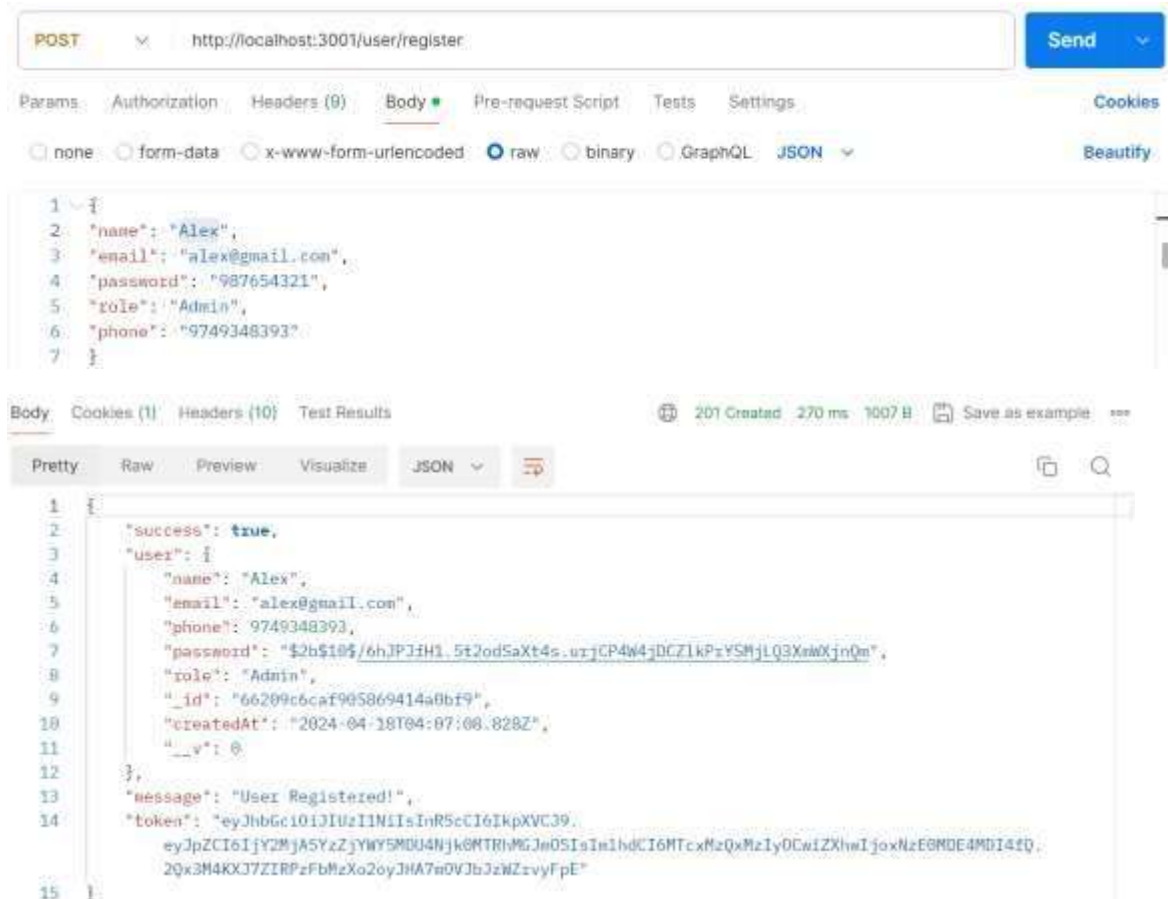
TEST INPUT	Expected Result	Actual Result	Pass/Fail
Posting a job as Student	Access denied to the route as Student	Error Message: "Student not allowed to access this resource."	Pass
Posting a job when some field is empty	Error on submission	Error: "Please provide full job details."	Pass
Posting a job with correct info	Job posted successfully	Success message: "Job Posted Successfully!"	Pass

Table 4.2.3

Registration Unsuccessful



Registration Successful



Login Successful

The screenshot shows a REST client interface with a POST request to `http://localhost:3001/user/login`. The request body is a JSON object with the following fields:

```
1 {
2   "email": "mayank@gmail.com",
3   "password": "mayank123",
4   "role": "Admin"
5 }
```

The response is a JSON object with the following fields:

```
1 {
2   "success": true,
3   "user": {
4     "id": "6611ba278f55c7bba7e72c03",
5     "name": "Mayank",
6     "email": "mayank@gmail.com",
7     "phone": "999999999",
8     "password": "$2b$10$81f5f0S0STPtby0x1X8wuo213hKoCGqTKV4J3yHAXZ1qc.ptnqUe",
9     "role": "Admin",
10    "createdAt": "2024-04-06T21:09:59.685Z",
11    "_v": 0
12  },
13   "message": "User Logged In!",
14   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY2MTF5YTI3OGY1NWMyYmJhN2U3MmMwMyIsIm1hdCI6MTcxMzQxMjk2MSwiZXhwIjoxNzE0MDE3NzYxZQ. _cqYYBo_VdyP6EtQjjZ-zw_QCCdwL1qDDPK1GmNIOFU"
15 }
```

The response status is 201 Created, with a response time of 155 ms and a body size of 1010 B. The response is displayed in the JSON format.

Login Unsuccessful

The screenshot shows a REST client interface with a POST request to `http://localhost:3001/user/login`. The request body is a JSON object with the following fields:

```
1 {
2   "email": "mayank123@gmail.com",
3   "password": "mayank123",
4   "role": "Admin"
5 }
```

The response is a JSON object with the following fields:

```
1 {
2   "success": false,
3   "message": "Invalid Email Or Password."
4 }
```

The response status is 400 Bad Request, with a response time of 71 ms and a body size of 354 B. The response is displayed in the JSON format.

Job posted successfully

The screenshot shows a REST client interface with a POST request to `http://localhost:3001/job/post`. The request body is a JSON object representing a job posting. The response is a 200 OK status with a JSON body indicating success.

Request:

```
1 {
2   "title": "VR Experience Designer",
3   "description": "Embark on an exciting journey as a Virtual Reality Experience Designer, crafting
4     immersive worlds and interactive experiences.",
5   "category": "Virtual Reality",
6   "country": "United States",
7   "city": "San Francisco",
8   "location": "Downtown San Francisco",
9   "fixedSalary": "80000"
10 }
```

Response:

```
1 {
2   "success": true,
3   "message": "Job Posted Successfully!",
4   "job": {
5     "title": "VR Experience Designer",
6     "description": "Embark on an exciting journey as a Virtual Reality Experience Designer, crafting
7       immersive worlds and interactive experiences.",
8     "category": "Virtual Reality",
9     "country": "United States",
10    "city": "San Francisco",
11    "location": "Downtown San Francisco",
12    "fixedSalary": 80000,
13    "expired": false,
14    "postedBy": "66209c6caf905869414a0bf9",
15    "_id": "66209de1af905869414a0c00",
16    "jobPostedOn": "2024-04-10T04:13:21.390Z",
17    "__v": 0
18  }
19 }
```

Job posting unsuccessful

The screenshot shows a REST client interface with a POST request to `http://localhost:3001/job/post`. The request body is a JSON object representing a job posting. The response is a 400 Bad Request status with a JSON body indicating failure.

Request:

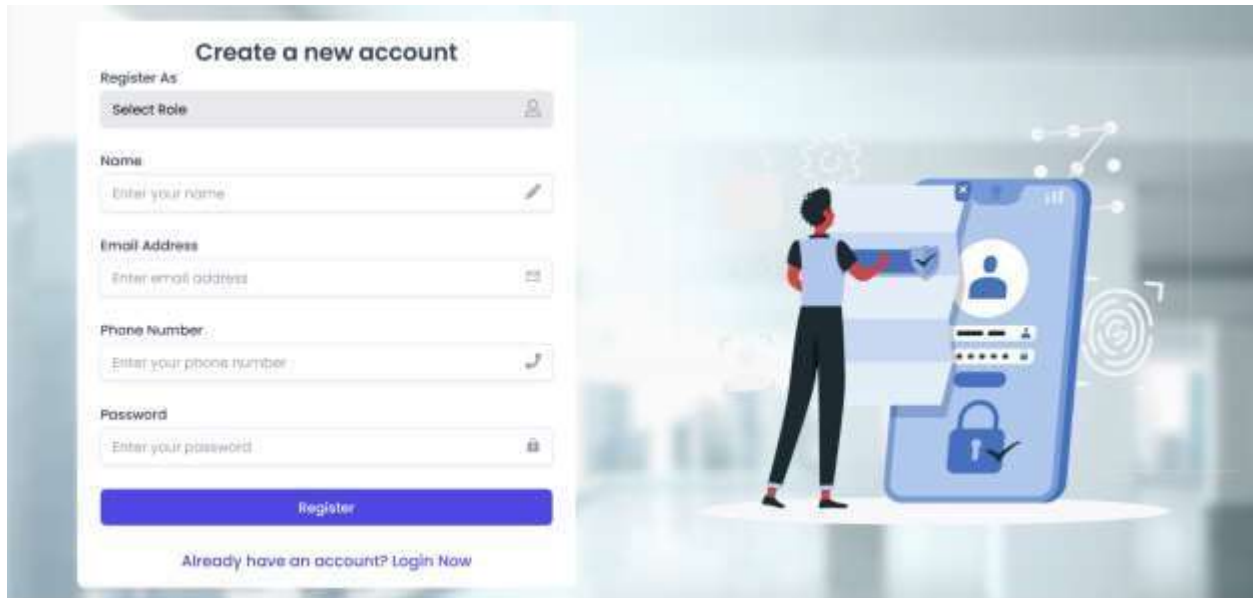
```
1 {
2   "title": "Video Animator",
3   "description": "Join our team as a video animator and bring stories to life.",
4   "category": "Video Animation",
5   "country": "Germany",
6   "city": "Berlin",
7   "fixedSalary": "65000"
8 }
```

Response:


```
1 {
2   "success": false,
3   "message": "Please provide full job details."
4 }
```


CHAPTER-5 **INTERFACE DESIGN**


Register





Create a new account

Register As
Select Role 

Name
Enter your name 

Email Address
Enter email address 

Phone Number
Enter your phone number 

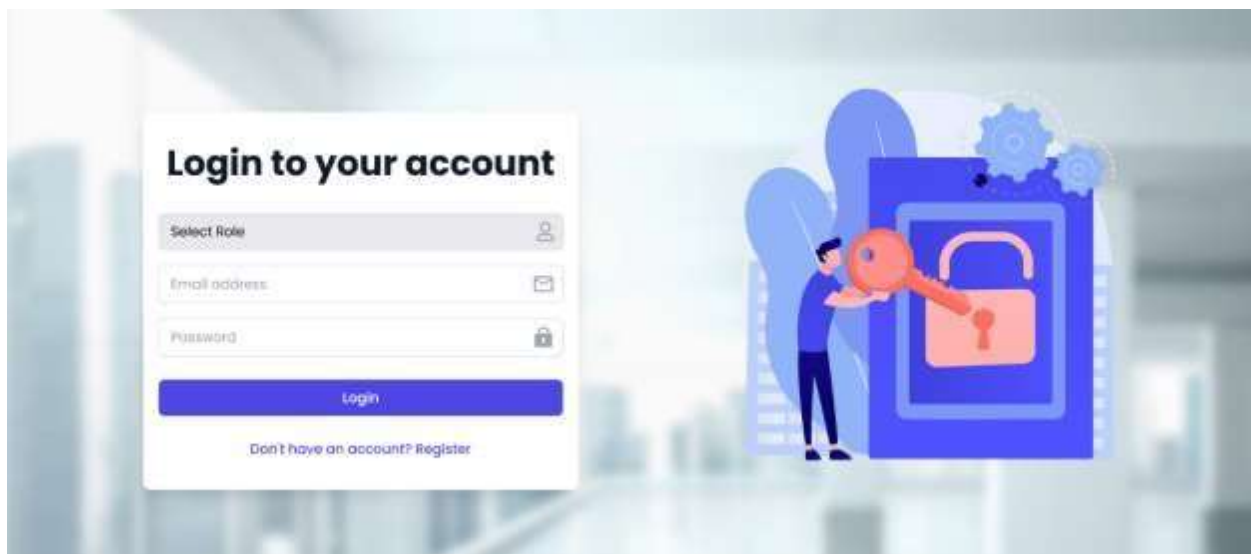
Password
Enter your password 

Register


Already have an account? [Login Now](#)


The illustration on the right shows a person standing next to a large, stylized smartphone. The phone screen displays a user profile with a checkmark, a password field with dots, and a lock icon. A fingerprint sensor is also visible on the phone. The background is a blurred cityscape.


Login



Login to your account

Select Role 

Email address 


Password 

Login

Don't have an account? [Register](#)

The illustration on the right shows a person standing next to a large, stylized smartphone. The phone screen displays a large red padlock with a keyhole. The background is a blurred cityscape.


Admin Homepage

**Placement Office**


[All Jobs](#) [Interview Ques](#) [Post Ques](#) [Post Job](#) [My Jobs](#) [Student's Applications](#) [Logout](#)

Find a job that suits your interests and skills

Embark on a journey where your skills meet purpose, transforming passion into profession. Make everyday an opportunity to excel and inspire.




HOW PLACEMENT OFFICE WORKS




Create Account

Embark on your journey with just a few clicks. Unlock a world of opportunities.



Find a Job/Post a Job


Discover the perfect job or the ideal candidate. A seamless search awaits.




Apply for Job/Recruit Suitable Candidates

Bridge your ambitions with opportunities. Apply or recruit with ease and precision.


POPULAR CATEGORIES




Graphics & Design




Mobile App Development




Frontend Web Development




MERN STACK Development




Account & Finance



Artificial Intelligence




Video Animation




Game Development


OUR MAJOR RECRUITERS




[Visit Website](#)




[Visit Website](#)




[Visit Website](#)




[Visit Website](#)




[Visit Website](#)




[Visit Website](#)




[Visit Website](#)



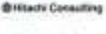
[Visit Website](#)




[Visit Website](#)




[Visit Website](#)




[Visit Website](#)




[Visit Website](#)



[Visit Website](#)






[Visit Website](#)



[Visit Website](#)

Placement Office

Your go-to platform for all placement related activities.


Follow Us

Contact Us

Email: info@placementtrackerapp.com
Phone: +1 (23) 456-7890

© 2024 Placement Office App. All rights reserved.

All Jobs Page

**Placement Office**

[All Jobs](#) [Interview Ques](#) [Post Ques](#) [Post Job](#) [My Jobs](#) [Student's Applications](#) [LOGOUT](#)

ALL AVAILABLE JOBS

Frontend Web Developer
Frontend Web Development
Canada
[Job Details](#)

Data Entry Operator
Data Entry Operator
Australia
[Job Details](#)

AI Engineer
Artificial Intelligence
United Kingdom
[Job Details](#)

Marketing Manager
Account & Finance
India
[Job Details](#)


Placement Office
Your go-to platform for all placement related activities.

Follow Us
[f](#) [t](#) [in](#)

Contact Us
Email: info@placementtrackerapp.com
Phone: +1 (123) 456-7890

© 2024 Placement Office App. All rights reserved.

Post Job Page

**Placement Office**

[All Jobs](#) [Interview Ques](#) [Post Ques](#) [Post Job](#) [My Jobs](#) [Student's Applications](#) [LOGOUT](#)

Post New Job

Job Title

Select Category

Country

City

Location

Select Salary Type

Job Description

Create Job

Placement Office
Your go-to platform for all placement related activities.

Follow Us
[f](#) [t](#) [in](#)

Contact Us
Email: info@placementtrackerapp.com
Phone: +1 (123) 456-7890

© 2024 Placement Office App. All rights reserved.

My Jobs Page


Placement Office

[Home](#)
[About Us](#)
[Services](#)
[Contact Us](#)
[My Jobs](#)
[My Profile](#)
[My Account](#)
[My Settings](#)
[My Notifications](#)

[Logout](#)

Your Posted Jobs

Graphic Designer

Job ID

Graphic Designer

Location

London, UK

Salary

£30,000 - £35,000

Experience

1-3 years

Education

Graduate

Job Type

Full Time

Job Description

Join our team as a creative graphic designer to work on exciting projects.

Apply Now

Frontend Web Developer

Job ID

Frontend Web Developer

Location

London, UK

Salary

£25,000 - £30,000

Experience

1-3 years

Education

Graduate

Job Type

Full Time

Job Description

Join our team as a frontend web developer to work on exciting projects.

Apply Now

Data Entry Operator

Job ID

Data Entry Operator

Location

London, UK

Salary

£15,000 - £20,000

Experience

1-3 years

Education

Graduate

Job Type

Full Time

Job Description

Join our team as a data entry operator to work on exciting projects.

Apply Now

AI Engineer

Job ID

AI Engineer

Location

London, UK

Salary

£40,000 - £50,000

Experience

1-3 years

Education

Graduate

Job Type

Full Time

Job Description

Join our team as an AI engineer to work on exciting projects.

Apply Now

Video Animator

Job ID

Video Animator

Location

London, UK

Salary

£20,000 - £25,000

Experience

1-3 years

Education

Graduate

Job Type

Full Time

Job Description

Join our team as a video animator to work on exciting projects.

Apply Now

Marketing Manager

Job ID

Marketing Manager

Location

London, UK

Salary

£30,000 - £35,000

Experience

1-3 years

Education

Graduate

Job Type

Full Time

Job Description

Join our team as a marketing manager to work on exciting projects.

Apply Now

Placement Office
 Your go-to platform for employment search and recruitment.


Follow Us




Contact Us
 Email: info@placementoffice.co.uk
 Phone: +44 (0)20 1234 5678

© 2024 Placement Office. All rights reserved.

Interview Questions Page

 **Placement Office**

All JobsInterview QuesPost QuesPost JobMy JobsStudent's ApplicationsLOGOUT

Prepare for Interviews

Filter by subject: All

What is the time complexity of the quicksort algorithm?

Subject: Algorithms

The average and best-case time complexity of quicksort is $O(n \log n)$, while the worst-case time complexity is $O(n^2)$.

EditDelete

What is a hash table, and how does collision resolution work in it?

Subject: Data Structures

A hash table is a data structure that stores key-value pairs. Collision resolution techniques like chaining or open addressing handle situations where multiple keys hash to the same index by storing them in the same location.

EditDelete

What is the purpose of virtual memory in an operating system?

Subject: Operating Systems

Virtual memory allows the system to use disk space as an extension of RAM. It enables processes to use more memory than is physically available and provides memory protection and efficient memory allocation.

EditDelete

What is the difference between TCP and UDP?

Subject: Networking

TCP (Transmission Control Protocol) provides reliable, connection-oriented communication, ensuring data delivery and ordering. UDP (User Datagram Protocol) is connectionless and offers unreliable, datagram-oriented communication with no guarantees of delivery or order.

EditDelete

What is a binary search tree, and what operations can be performed on it?

Subject: Data Structures

A binary search tree is a data structure that allows for efficient searching, insertion, and deletion operations. Operations include searching for a key, inserting a key, deleting a key, finding the minimum and maximum keys, and traversing the tree.

EditDelete

What is machine learning, and what are the three main types of machine learning algorithms?

Subject: Artificial Intelligence

Machine learning is a subset of artificial intelligence that enables systems to learn from data. The three main types of machine learning algorithms are supervised learning, unsupervised learning, and reinforcement learning.

EditDelete

What is normalization in the context of database design?

Subject: Database Management Systems

Normalization is the process of organizing data in a database efficiently. It involves reducing redundancy and dependency by dividing large tables into smaller tables and defining relationships between them.

EditDelete

What is the difference between black-box testing and white-box testing?

Subject: Software Engineering

Black-box testing focuses on testing the functionality of a system without knowledge of its internal code structure. White-box testing, on the other hand, tests the internal structures or workings of a system, including its code.

EditDelete

What is the OSI model, and what are its seven layers?

Subject: Networking

The OSI (Open Systems Interconnection) model is a conceptual framework that standardizes the functions of a telecommunication or computing system into seven abstraction layers. The layers are: Physical, Data Link, Network, Transport, Session, Presentation, and Application.

EditDelete




What is ray tracing, and how does it differ from rasterization?

Subject: Computer Graphics

Ray tracing is a rendering technique that simulates the way light interacts with objects in a scene by tracing the path of light rays. Rasterization, on the other hand, converts 3D objects into 2D images by projecting them onto a 2D plane.

EditDelete


Placement Office
Your go-to platform for all placement related activities.

Follow Us
  

Contact Us
Email: info@placementtrackerapp.com
Phone: +1 (123) 456-7890

© 2024 Placement Office App. All rights reserved.

Post Ques Page

 **Placement Office**

[All Jobs](#) [Interview Ques](#) [Post Ques](#) [Post Job](#) [My Jobs](#) [Student's Applications](#) [LOGOUT](#)

Post New Interview Question

Post Question


Placement Office
Your go-to platform for all placement related activities.

Follow Us
[f](#) [t](#) [in](#)

Contact Us
Email: info@placementtrackerapp.com
Phone: +1 (123) 456-7890

© 2024 Placement Office App. All rights reserved.


Student Applications Page

 **Placement Office**

[All Jobs](#) [Interview Ques](#) [Post Ques](#) [Post Job](#) [My Jobs](#) [Student's Applications](#) [LOGOUT](#)

Applications From Students

Name: David
Email: david@gmail.com
Phone: 9347394933
Address: 1234 Tech Avenue, Silicon Valley, San Francisco, CA
CoverLetter: I am writing to express my interest in the Graphic Designer position at your company, as advertised. With a keen eye for design and a passion for creativity, I am excited about the opportunity to contribute to your team and help bring your projects to life. Having worked in the field of graphic design for over 5 years, I have developed a strong portfolio that showcases my ability to create visually compelling designs across various platforms. My experience ranges from branding and logo design to print collateral and digital media, allowing me to adapt to the diverse needs of clients and projects.




Placement Office
Your go-to platform for all placement related activities.

Follow Us
[f](#) [t](#) [in](#)

Contact Us
Email: info@placementtrackerapp.com
Phone: +1 (123) 456-7890

© 2024 Placement Office App. All rights reserved.


Student Dashboard

 **Placement Office**


[All Jobs](#) [Interview Ques](#) [My Applications](#) [Contact](#) [Placement Highlights](#) [LOGOUT](#)

Find a job that suits your interests and skills


Embark on a journey where your skills meet purpose, transforming passion into profession. Make everyday an opportunity to excel and inspire.




HOW PLACEMENT OFFICE WORKS

**Create Account**

Embark on your journey with just a few clicks. Unlock a world of opportunities.


**Find a Job/Post a Job**


Discover the perfect job or the ideal candidate. A seamless search awaits.


**Apply For Job/Recruit Suitable Candidates**


Bridge your ambitions with opportunities. Apply or recruit with ease and precision.


POPULAR CATEGORIES


**Graphics & Design**


**Mobile App Development**


**Frontend Web Development**

**MERN STACK Development**


**Account & Finance**


**Artificial Intelligence**


**Video Animation**


**Game Development**


OUR MAJOR RECRUITERS


[Visit Website](#)


[Visit Website](#)


[Visit Website](#)


[Visit Website](#)


[Visit Website](#)


[Visit Website](#)


[Visit Website](#)


[Visit Website](#)


[Visit Website](#)


[Visit Website](#)

[Visit Website](#)

[Visit Website](#)




[Visit Website](#)

[Visit Website](#)

[Visit Website](#)

Placement Office

Your go-to platform for all placement related activities.


Follow Us
  

Contact Us

Email: info@placementtrackerapp.com
Phone: +1 (23) 456-7890

© 2024 Placement Office App. All rights reserved.

My Applications

 **Placement Office**


[All Jobs](#) [Interview Ques](#) [My Applications](#) [Contact](#) [Placement Highlights](#) [LOGOUT](#)

MY APPLICATIONS

Name: David

Email: david@gmail.com
Phone: 9347394933
Address: 1234 Tech Avenue, Silicon Valley, San Francisco, CA
CoverLetter: I am writing to express my interest in the Graphic Designer position at your company, as advertised. With a keen eye for design and a passion for creativity, I am excited about the opportunity to contribute to your team and help bring your projects to life. Having worked in the field of graphic design for over 5 years, I have developed a strong portfolio that showcases my ability to create visually compelling designs across various platforms. My experience ranges from branding and logo design to print collateral and digital media, allowing me to adapt to the diverse needs of clients and projects.

[Delete Application](#)




Placement Office
Your go-to platform for all placement related activities.

Follow Us
[f](#) [t](#) [in](#)

Contact Us
Email: info@placementtrackerapp.com
Phone: +1 (123) 456-7890


© 2024 Placement Office App. All rights reserved.

Contact us

 **Placement Office**

[All Jobs](#) [Interview Ques](#) [My Applications](#) [Contact](#) [Placement Highlights](#) [LOGOUT](#)

Let's connect....



Name

Email

Subject

Message

[Send](#)

Placement Office
Your go-to platform for all placement related activities.

Follow Us
[f](#) [t](#) [in](#)

Contact Us
Email: info@placementtrackerapp.com
Phone: +1 (123) 456-7890

© 2024 Placement Office App. All rights reserved.

Placement Highlights



Application form

Placement Office All Jobs Interview Ques My Applications Contact Placement Highlights **Logout**

Application Form

Enter your Name

Enter your Email

Enter your Phone Number

Enter your Address

Enter cover letter

Upload Resume

Browse... No file selected.

Submit Application

Placement Office
Your go-to platform for all placement related activities.

Follow Us
f t in

Contact Us
Email: info@placementtrackerapp.com
Phone: +1 (123) 456-7890

© 2024 Placement Office App. All rights reserved.

CHAPTER - 6

FUTURE SCOPE, SUMMARY, CONCLUSION

6.1 FUTURE SCOPE

While our project has reached a mature stage, continuous improvement remains a key principle. Several features can be integrated to further enhance the software's reliability and user experience:

- ❖ **Optimized Resource Requirements:** Originally, the project focused on minimizing hardware and software requirements to support a broader user base.
- ❖ **Enhanced Search Functionality:** Strengthening the search capabilities is imperative, ensuring that placement officers can swiftly access previous data.
- ❖ **Graphical Improvements:** A refined approach with enhanced graphics can be implemented to elevate the overall visual appeal of the project.
- ❖ **Backup Procedures:** Incorporating robust backup procedures becomes crucial to safeguard the integrity of the database.
- ❖ **Real-time Communication:** Facilitating recruiter access at any time through the application ensures seamless communication with placement officers.
- ❖ **Efficient Registration Process:** Simplifying the placement application process for students by introducing a streamlined registration form.

6.2 SUMMARY

Through a meticulous analysis of strengths and limitations, it is evident that the product stands as a highly efficient GUI-based component. Its adaptability to various systems is a notable feature, coupled with user-friendly interfaces. Addressing the challenges faced by Placement Officers in manual information management, this software emerges as a comprehensive solution.

6.3 CONCLUSION

The culmination of our efforts is the 'Placement Office' web application, meticulously designed to facilitate faculty and students. By providing access to previous placement records and streamlining the application process, this software addresses critical challenges faced by academic institutions. Its structured framework contributes to an efficient and effective placement management system.