**SOFTWARE ENGINEERING**



**Session: 2022 – 2026**

**Submitted by:**

Raveeha Mohsin 2022-CS-149

Tayyaba Afzal 2022-CS-134

**Supervised by:**

Sir Amjad Farooq

Department of Computer Science

**University of Engineering and Technology**

**Lahore , Pakistan**

**Contents**

[1 Presentation Layer 3](#_Toc178151413)

[1.1 Role 3](#_Toc178151414)

[1.2 Technologies 3](#_Toc178151415)

[1.3 Functionalities 3](#_Toc178151416)

[2 Business Logic Layer (Application Layer) 4](#_Toc178151417)

[2.1 Role 4](#_Toc178151418)

[2.2 Technologies 4](#_Toc178151419)

[2.3 Functionalities 5](#_Toc178151420)

[3 Data Access Layer (Database Layer) 7](#_Toc178151421)

[3.1 Role 7](#_Toc178151422)

[3.2 Technologies 7](#_Toc178151423)

[3.3 Database Tables 7](#_Toc178151424)

[3.4 ERD (Entity Relationship Diagram) 9](#_Toc178151425)

[3.5 Data Access Logic: 10](#_Toc178151426)

[4 Layered Communication Flow 11](#_Toc178151427)

[5 Example Breakdown for Career Path Navigator Features 12](#_Toc178151428)

[5.1 User Login (Authentication) 12](#_Toc178151429)

[5.2 Admin Dashboard 12](#_Toc178151430)

[5.2.1 View & Manage Users (Admin) 12](#_Toc178151431)

[5.2.2 View Feedback (Admin) 13](#_Toc178151432)

[5.3 Counselor Dashboard 13](#_Toc178151433)

[5.3.1 Manage Availability (Counselor) 13](#_Toc178151434)

[5.3.2 Schedule and Conduct Meetings (Counselor) 13](#_Toc178151435)

[5.3.3 View Feedback (Counselor) 14](#_Toc178151436)

[5.4 Student Dashboard 14](#_Toc178151437)

[5.4.1 Profile Management (Student) 14](#_Toc178151438)

[5.4.2 View all Recommendation & Courses 14](#_Toc178151439)

[5.4.3 View and Apply for Jobs (Student) 14](#_Toc178151440)

[5.4.4 Schedule Meetings with Counselors (Student) 15](#_Toc178151441)

[5.4.5 Provide Feedback (Student) 15](#_Toc178151442)

[6 Tools and Frameworks 16](#_Toc178151443)

**Layered Architecture Design for Career Path Navigator**

# Presentation Layer

## Role

This is the front-end layer responsible for interacting with users such as **students**, **counselors**, and **administrators**.

## Technologies

* **React.js** (for building dynamic user interfaces and components)
* **JavaScript** (for handling client-side logic and interactivity)
* **HTML/CSS** (for structuring the UI and applying styles)
* **Bootstrap** & **Tailwind CSS** (for responsive, utility-first design)
* **AJAX** (for asynchronous communication between client and server)
* **MERN stack** (MongoDB, Express, React, Node.js for full-stack functionality)

## Functionalities

 **Landing Page**: The landing page provides an intuitive and clean interface to navigate key sections like *Home*, *About*, *Career Paths*, and *Contact Us*.

 **Authentication**: Secure login for students, counselors, and admins using **JSON Web Tokens (JWT)** for session management.

 **Student Dashboard**:

* **Profile Setup**: Allows students to input their personal and academic details, career goals, and interests.
* **AI-Powered Career Recommendations**: Provides personalized career paths based on the student’s profile and industry trends.
* **Job and Internship Listings**: Displays career opportunities tailored to the student’s aspirations.
* **Career Path Visualization**: Offers a visual map of career progression steps, required skills, and job roles.
* **Learning Resources**: Recommends courses and certifications, which students can add to their **Library** for progress tracking.
* **Resume Builder**: Assists students in enhancing their resumes using AI suggestions.
* **Progress Tracker**: Monitors and displays completed tasks like courses and acquired skills.
* **Counselor Feedback**: Students can review and rate their counselors.
* **Counselor Meeting**: Schedule meetings with counselors, manage payments, and conduct meetings via Zoom.
* **Chatbot**: Offers quick career guidance through AI-powered interactions.
* **Notifications**: Real-time alerts on meetings, deadlines, and career resources.

 **Counselor Dashboard**:

* **Graphical View**: Visual representation of student progress and feedback.
* **Meeting Management**: Schedule and manage meetings with students.
* **Student List**: Access detailed student profiles and track their career progress.
* **Reviews**: View and provide feedback to students based on interactions.

 **Admin Dashboard**:

* **Manage Counselors**: Approve or reject counselor applications, view profiles, and track performance.
* **Graphical Dashboard**: View data on counselor activities, student progress, and meetings.
* **Manage Users**: View, track, and manage student and counselor accounts and meeting histories.

# Business Logic Layer (Application Layer)

## Role

This server-side layer is responsible for processing business logic, interacting with the **database**, and **handling API requests**, providing responses to the front-end (Presentation Layer).

## Technologies

* **Node.js** (for server-side logic and API creation)
* **Express.js** (for building RESTful APIs and managing routes)
* **MySQL** (relational database for managing structured data)
* **JSON** (for data interchange between client-side React and server-side APIs)
* **External APIs** (for fetching job listings from platforms like **LinkedIn**, **Indeed**, etc.)

## Functionalities

* + **User Authentication & Authorization**:
    - **JWT-Based Authentication**: Handle secure login and token-based session management, ensuring that users are authenticated and authorized to access specific resources.
    - **Role-Based Access Control (RBAC)**: Provide different functionalities for students, counselors, and administrators.
      * **Student**: Access personalized career guidance, recommendations, and resources.
      * **Counselor**: Manage student profiles, schedule meetings, and provide feedback.
      * **Admin**: Manage user accounts, approve/reject counselors, and oversee system usage.
  + **AI-Powered Career Recommendations**:
    - **Career Path Generation**: Based on student profiles (skills, interests, academic background), generate personalized career paths using machine learning algorithms.
    - **Learning Resources and Job Listings**:
      * **LinkedIn/Indeed Integration**: Fetch real-time job and internship listings through APIs from platforms like LinkedIn or Indeed. Job listings will be tailored based on the student’s career interests and skill sets.
  + **Profile Management**:
    - **Student Profile Setup and Updates**: Logic to handle profile creation and updates, including interests, skills, and career goals.
    - **Counselor Profile Management**: Allow counselors to update their profiles, manage their schedule, and track students’ progress.
  + **Career Progress Tracking**:
    - **Progress Monitoring**: Track student progress on recommended learning resources, completed courses, and acquired skills.
    - **Career Path Visualization**: Provide a backend service to generate visual representations of career paths, showing progression steps and required skills.
  + **Counselor-Student Meeting Management**:
    - **Scheduling System**: Enable students to request meetings, counselors to accept or reject meeting requests, and integration with third-party services like **Zoom** for conducting meetings.
    - **Payment Processing**: Handle logic for processing payments for counselor meetings (e.g., set a time limit for payment completion and verify payment before scheduling).
  + **Job and Internship API Integration**:
    - **External Job Listings**: Fetch jobs and internships using APIs from **LinkedIn**, **Indeed**, or other career platforms. Display relevant opportunities based on student profiles and AI analysis.
  + **Chatbot Integration**:
    - **AI-Powered Chatbot**: Handle real-time communication between students and the AI chatbot to provide instant guidance on career paths, skills development, and academic decisions.
  + **Notifications**:
    - **Real-Time Notification System**: Logic to send notifications to students and counselors about meeting requests, feedback reminders, and important deadlines.
  + **Feedback and Reviews**:
    - **Counselor Feedback Management**: Allow students to submit feedback on their counseling sessions.
    - **Counselor Reviews**: Enable counselors to review students' progress and provide constructive feedback.
    - **Admin Review System**: Admins can monitor and review the feedback given to counselors to ensure quality.
  + **Data Validation & Sanitization**:
    - **Input Validation**: Validate and sanitize all user inputs (e.g., login credentials, profile details, and course feedback) to prevent data breaches or unauthorized access.
    - **Data Integrity**: Ensure the integrity of data stored in the **MySQL** database, especially related to user profiles, career recommendations, and payment processing.

# Data Access Layer (Database Layer)

## Role

This layer is responsible for communicating with the **MySQL** database, performing operations like fetching, inserting, updating, and deleting records as per the application's business logic.

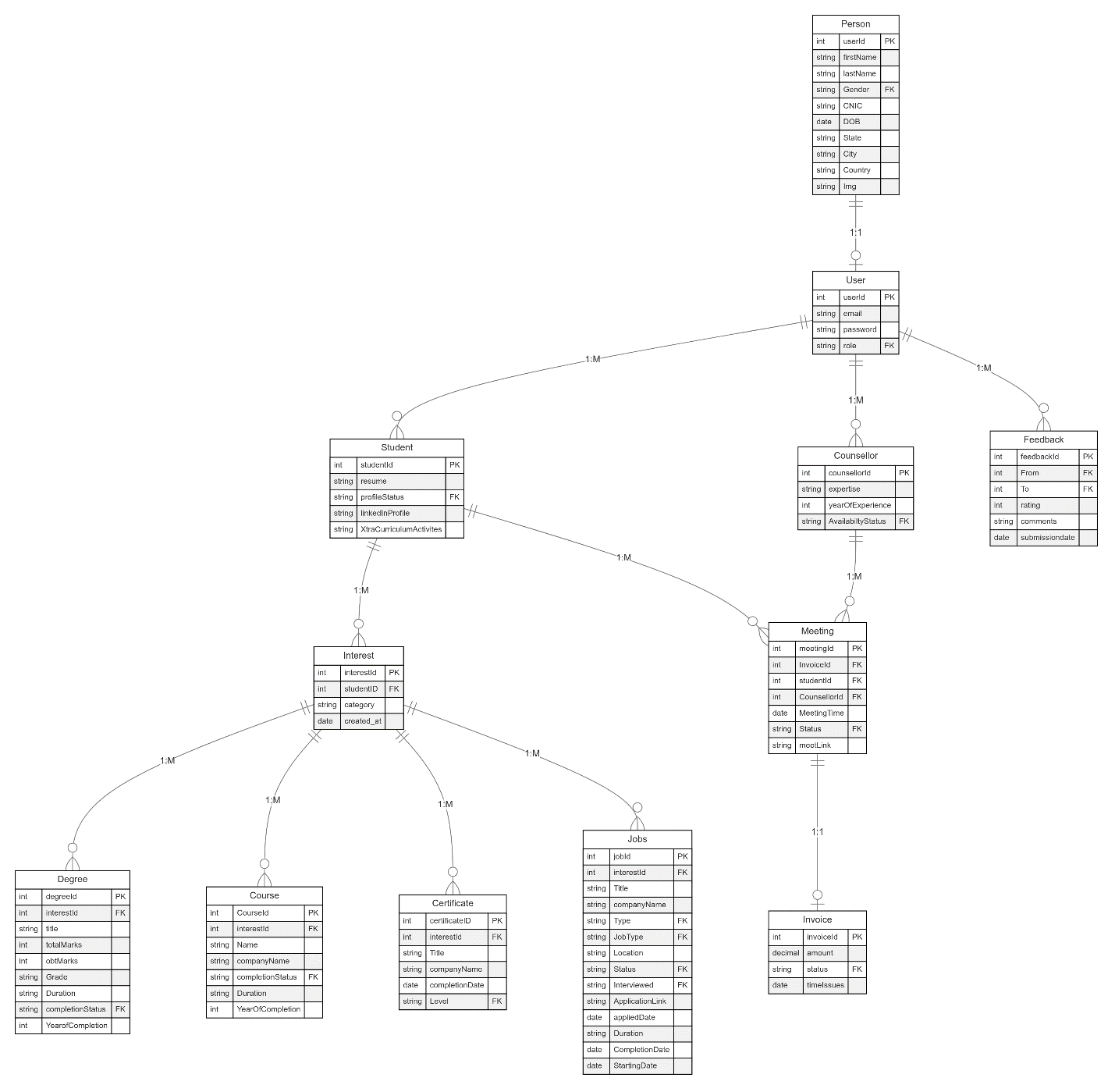
## Technologies

* **MySQL** (Relational Database Management System)
* **Node.js** and **Express.js** (to interact with the MySQL database through queries and API requests)
* **Axios** (to fetch external API data, such as job listings from platforms like LinkedIn)

## Database Tables

* **Users: (**userId, email, password, role**)**
* Stores basic information and login credentials for all users, including their role, which defines whether the user is a student, counselor, or admin.
* **Person: (**userId, firstName, lastName, Gender, CNIC, DOB, State, City, Country, Img**)**
* Contains detailed personal information about each user, linked to the Users table.
* **Students:(**studentId, resume, profileStatus, linkedInProfile, XtraCurriculumActivites**)**
* Stores student-specific information, including their resume, LinkedIn profile, extracurricular activities, and profile status.
* **Counsellor: (**counsellorId, expertise, yearOfExperience, AvailabiltyStatus**)**
* Holds details about counselors, including their expertise, years of experience, and availability status.
* **Meeting: (**meetingId, InvoiceId, studentId, CounsellorId, MeetingTime, Status, meetLink**)**
* Contains information about meetings between students and counselors, including the meeting status, time, and link.
* **Invoice: (**invoiceId, amount, status, timeIssues**)**
* Stores invoice data for meetings between students and counselors, including payment status and timestamps.
* **Interest: (**interestId, studentID, category, created\_at**)**
* Contains data on the different interests a student has, such as career preferences or learning interests.
* **Degree: (**degreeId, interestId, title, totalMarks, obtMarks, Grade, Duration, completionStatus, YearofCompletion**)**
* Stores information about the academic degrees a student has earned, including their marks, grade, duration, and completion status.
* **Course: (**CourseId, interestId, Name, companyName, completionStatus, Duration, YearOfCompletion**)**
* Stores information about courses related to a students interest, including the name of the course, the company offering it, and completion details.
* **Certificate: (**certificateID, interestId, Title, companyName, completionDate, Level**)**
* Holds information about the certificates a student has obtained, including the level and date of completion.
* **Jobs: (**jobId, interestId, Title, companyName, Type, JobType, Location, Status, Interviewed, ApplicationLink, appliedDate, Duration, CompletionDate, StartingDate**)**
* Stores information about jobs or internships, including job type, company details, application status, and job duration.
* **Feedback: (**feedbackId, rating, comments, submissiondate, From, To**)**
  + Contains feedback left by users (students or counselors), including ratings and comments about meetings or guidance.

## ERD (Entity Relationship Diagram)



## Data Access Logic:

 **Queries to Fetch User Data for Login:**

* Fetch user information from the Users table based on the email and password provided during login.
* Identify the user’s role (student, counselor, or admin) and retrieve their detailed personal information from the Person table.

 **API Fetching for Job Listings and Recommendations:**

* Use external APIs such as LinkedIn Jobs API to fetch job listings and update the Jobs table with relevant job details for the students, including Title, CompanyName, Job Type, etc.

 **CRUD Operations:**

* **Student Records:**
  + Insert, update, or delete student data from the Students table.
  + Manage student's resume, LinkedIn profile, and extracurricular activities.
* **Counselor Records:**
  + Insert, update, or delete counselor data in the Counselor table.
  + Manage counselor expertise, experience, and availability status.
* **Meetings:**
  + Schedule, update, and cancel meetings between students and counselors using the Meeting table.
  + Update meeting statuses and generate or retrieve the meeting link (e.g., Zoom or Google Meet).
* **Invoices:**
  + Create and update invoices for meetings in the Invoice table.
  + Manage payment status and timestamps related to each session.

 **Interest and Degree Management:**

* Insert records for student interests in the Interest table.
* Manage academic degree details, such as marks, grade, and completion status, in the Degree table.

 **Course and Certification Management:**

* Store and retrieve information about courses a student has completed using the Course table.
* Manage certification records in the Certificate table, including the completion date and level of the certification.

 **Job Applications and Updates:**

* Insert and update job application status in the Jobs table, including job title, company, application link, interview status, and job duration.
* Fetch job details from external APIs and populate the Jobs table with new job opportunities.

 **Feedback System:**

* Insert feedback into the Feedback table after meetings or interactions, allowing students and counselors to rate and comment on their experiences.

 **Report Generation:**

* Generate comprehensive reports on students’ academic history, certifications, job applications, and meeting schedules by querying data from relevant tables (Degree, Course, Jobs, and Meeting).

# Layered Communication Flow

* **Front-end (React + JavaScript + HTML)** sends a request (e.g., user login or fetching job listings) to the **Node.js server** (via Axios or Fetch API).
* **Node.js (Business Logic Layer)** verifies the user’s credentials and checks their role (e.g., student, counselor, or admin) by querying the **MySQL (Data Layer)** for the relevant user data.
* If the user authentication is successful, **Node.js** fetches additional data (such as personalized career recommendations, job listings, or student dashboard information) from MySQL or external APIs (like LinkedIn for job listings).
* **Node.js** sends the appropriate data back to the **Front-end (React)**, where it's rendered as per the user role (e.g., student dashboard, counselor appointments, or admin controls).
* For operations like booking counselor meetings or updating student profiles:
  + **React (JavaScript)** sends an API request (e.g., scheduling a meeting or submitting profile changes) to **Node.js**.
  + **Node.js** processes the request, performs data validation and business logic, then updates the necessary records in **MySQL**.
  + **Node.js** responds with a success or failure message, which is displayed on the **Front-end**.
* For external operations such as fetching job listings:
  + **React (JavaScript)** sends a request to **Node.js**.
  + **Node.js** uses Axios or Fetch to get job data from external APIs (e.g., LinkedIn or other job platforms).
  + The response is stored in the **MySQL** database and sent back to the **Front-end** for display in the student dashboard.

# Example Breakdown for Career Path Navigator Features

## ****User Login (Authentication)****

* **Presentation Layer:**
  + Login form built with **React** for dynamic rendering and **Bootstrap** for responsive design.
  + Form validation with React hooks or JavaScript (client-side validation).
* **Business Logic Layer:**
  + Backend in **Node.js** (Express) handles login requests, validates user credentials, checks roles (Admin, Student, Counselor), and creates a session using **JWT** (if valid).
* **Data Access Layer:**
  + MySQL queries to retrieve user details (email, password, and role) from the **Users** table.

## ****Admin Dashboard****

### ****View & Manage Users (Admin)****

* **Presentation Layer:**
  + Built with **React** and **Bootstrap** for UI components like tables, pagination, and user management options (edit/delete buttons).
  + React for dynamic data binding and filtering/search functionalities.
* **Business Logic Layer:**
  + Node.js fetches user data (students and counselors), processes update/delete requests, and communicates with the MySQL database.
* **Data Access Layer:**
  + MySQL queries retrieve user data from the **Users** and **Person** tables.
  + View queries for managing users.

### ****View Feedback (Admin)****

* **Presentation Layer:**
  + **React** and **Bootstrap** for the feedback interface. Feedback dynamically loads based on user input or filters (rating, date).
* **Business Logic Layer:**
  + Node.js retrieves feedback data (ratings and comments) between students and counselors.
* **Data Access Layer:**
  + MySQL queries fetch feedback data from the **Feedback** table.

## ****Counselor Dashboard****

### ****Manage Availability (Counselor)****

* **Presentation Layer:**
  + Form built with **React** and **Bootstrap** for updating counselor availability, with form validation using React hooks.
* **Business Logic Layer:**
  + Node.js processes availability status updates and updates the counselor's availability.
* **Data Access Layer:**
  + MySQL updates availability status in the **Counselor** table.

### ****Schedule and Conduct Meetings (Counselor)****

* **Presentation Layer:**
  + **React** displays scheduled meetings. Links to join meetings (e.g., via Zoom) with real-time updates on the meeting status.
  + Real-time meeting status changes handled by React and WebSocket’s (or polling).
* **Business Logic Layer:**
  + Node.js fetches the counselor's scheduled meetings, generates meeting links, and updates the meeting status (pending, completed).
* **Data Access Layer:**
  + MySQL retrieves meeting data from the **Meeting** table and updates meeting status.

### ****View Feedback (Counselor)****

* **Presentation Layer:**
  + **React** interface for counselors to view feedback left by students after a session.
* **Business Logic Layer:**
  + Node.js fetches feedback related to the counselor.
* **Data Access Layer:**
  + MySQL queries retrieve feedback data from the **Feedback** table for the counselor.

## ****Student Dashboard****

### ****Profile Management (Student)****

* **Presentation Layer:**
  + **React** form for students to update personal details (resume, LinkedIn, extracurricular activities), styled using **Bootstrap**.
  + React handles file uploads and form validation.
* **Business Logic Layer:**
  + Node.js processes updates to the student's profile (e.g., resume, LinkedIn profile).
* **Data Access Layer:**
  + MySQL updates the **Students** table with the student’s profile data (resume, profile status, LinkedIn).

### ****View all Recommendation & Courses****

* **Presentation Layer:**
  + **React** interface for viewing all career recommendations and courses.
  + **Bootstrap** ensures responsive design for listings.
* **Business Logic Layer:**
  + Node.js fetches recommendations from external APIs (e.g., LinkedIn API) and processes job applications (stores data, updates application status).
* **Data Access Layer:**
  + MySQL stores job listings fetched from APIs in the **Jobs** table.
  + Application data and status updates are stored in MySQL.

### ****View and Apply for Jobs (Student)****

* **Presentation Layer:**
  + **React** interface for viewing job opportunities with filtering options (date, company, etc.). Data fetched from real-time APIs.
  + **Bootstrap** ensures responsive design for job listings.
* **Business Logic Layer:**
  + Node.js fetches job listings from external APIs (e.g., LinkedIn API) and processes job applications (stores data, updates application status).
* **Data Access Layer:**
  + MySQL stores job listings fetched from APIs in the **Jobs** table.
  + Application data and status updates are stored in MySQL.

### ****Schedule Meetings with Counselors (Student)****

* **Presentation Layer:**
  + **React** form for students to schedule meetings with counselors. Availability is checked in real-time.
  + JavaScript dynamically validates and displays available time slots.
* **Business Logic Layer:**
  + Node.js handles meeting requests, checks counselor availability, and schedules the meeting.
* **Data Access Layer:**
  + MySQL inserts meeting data into the **Meeting** table (studentId, counselorId, meeting time).

### ****Provide Feedback (Student)****

* **Presentation Layer:**
  + **React** form for students to submit feedback after counseling sessions.
  + **Bootstrap** styled feedback form, with dynamic rating system using React state management.
* **Business Logic Layer:**
  + Node.js processes feedback submissions, stores the data, and triggers any notifications.
* **Data Access Layer:**
  + MySQL stores feedback in the **Feedback** table (rating, comments, submission date).

# Tools and Frameworks

* **Backend Framework:** Use **Node.js** with **Express.js** for managing routing, middleware, and business logic efficiently.
* **Frontend Framework:** Use **React.js** along with **Bootstrap** and **Tailwind CSS** for building interactive, responsive user interfaces and ensuring a smooth user experience.
* **Database:** Use **MySQL** for database management, including **MySQL Workbench** for database design and management.
* **API Integration:** Use **Axios** or **Fetch API** in React to interact with external APIs such as **LinkedIn** for fetching job listings and other external data sources.
* **Authentication:** Implement **JSON Web Tokens (JWT)** for secure user authentication and session management across roles like Students, Counselors, and Admins.

This layered architecture allows for clear separation of concerns, with each layer responsible for a distinct part of the school system. It ensures that changes in one layer (e.g., UI or database) don’t significantly impact others