**Skill Matrix System**

**Major Project**

Contents

[1 Project Overview 3](#_Toc163829857)

[1.1 Brief introduction of the project. 3](#_Toc163829858)

[1.2 Purpose of the project. 3](#_Toc163829860)

[1.3 Goals of the project: 4](#_Toc163829861)

[1.3 Key features and functionalities of the web app. 5](#_Toc163829866)

[2 Architecture and Technology Stack 5](#_Toc163829874)

[2.1 Overall architecture of the application. 5](#_Toc163829875)

[2.2 Technology stack (e.g., programming languages, frameworks, libraries) 6](#_Toc163829894)

[3 Authentication and Authorization 7](#_Toc163829895)

[4 Data Model 10](#_Toc163829896)

[4.1.1 Entity Relationships: 10](#_Toc163829897)

[4.1.2 Data Flow Diagrams: 11](#_Toc163829898)

**5**  **Recommendation Model:……………………………………………………………………………………12**

# Project Overview

## Brief introduction of the project.

## The Skill Matrix System project aims to streamline the management of employee skills, certifications, and project experience within an organization. By providing a centralized platform, it enhances efficiency in assessing and leveraging the expertise of team members. Through intuitive user interfaces and robust functionalities, it empowers administrators and employees alike to contribute effectively to project success. This system serves as a dynamic tool for talent management, facilitating informed decision-making and fostering professional development opportunities. Ultimately, it strengthens the organization's capacity to align skills with project requirements, driving innovation and competitiveness.

## Purpose of the project.

The purpose of the Skill Matrix System project is to centralize and streamline the management of employee skills, certifications, and project experience within an organization. By providing a comprehensive platform, it aims to facilitate efficient allocation of resources, enhance transparency in talent distribution, ensure compliance with certification requirements, and foster continuous professional growth among employees. Ultimately, the project seeks to empower organizations to make informed decisions, optimize team compositions, and drive project success through effective utilization of internal expertise.

## 1.3 Goals of the project:

# Enhance transparency: Establish a clear overview of employees' skillsets, certifications, and project experiences to facilitate informed decision-making.

# Improve resource allocation: Enable efficient matching of employee skills with project requirements, optimizing team composition and project outcomes.

# Strengthen compliance: Ensure that employees maintain relevant certifications and adhere to project-specific requirements, reducing risks and enhancing quality assurance.

# Foster professional growth: Provide a platform for employees to showcase their expertise, identify skill gaps, and pursue targeted development opportunities, promoting continuous learning and career advancement.

## Key features and functionalities of the web app.

# Secure User Authentication: Implement robust login functionality to ensure only authorized personnel can access the system.

# User Profile Management: Allow administrators to create, edit, and manage user profiles, including basic information, roles, and contact details.

# Skill Tracking: Enable users to input and update their skills, certifications, and project experience, with options to specify proficiency levels and relevant details.

# Approval Workflow: Set up an approval process where designated approvers can review and approve users' certifications and project experience submissions.

# Email Notifications: Configure automated email notifications to alert users and approvers about pending approvals, updates, and other relevant actions within the system.

# Dashboard View: Provide intuitive dashboards for users and administrators to visualize skill profiles, certification statuses, pending approvals, and project experience summaries.

# Reporting and Analytics: Generate reports and analytics to track skill trends, certification compliance, project allocations, and other key metrics for informed decision-making.

# Architecture and Technology Stack

## Overall architecture of the application.

## Frontend (React, Material-UI, Bootstrap):

## React would serve as the core frontend library for building the user interface components.

## Material-UI and Bootstrap can be utilized for designing and styling the UI components, ensuring a visually appealing and responsive design.

## The frontend interacts with the backend server for data retrieval and updates using RESTful APIs.

## Node.js provides the runtime environment for running JavaScript on the server-side, allowing for efficient handling of asynchronous tasks.

## Express.js is a minimal and flexible Node.js web application framework, used for creating robust APIs to handle HTTP requests and responses.

## The backend integrates with the MongoDB database and Snowflake data warehouse for data storage, retrieval, and processing.

## Data Storage and Processing (MongoDB, Snowflake):

## MongoDB serves as the primary NoSQL database for storing user profiles, skill information, project details, and other application data.

## Snowflake data warehouse is utilized for storing and processing large volumes of structured and semi-structured data, providing scalability and performance for analytics and reporting purposes.

## Integration between MongoDB and Snowflake can be achieved using appropriate connectors or ETL (Extract, Transform, Load) processes.

## Analytics and Reporting (Power BI):

## Power BI is integrated for advanced data visualization, analytics, and reporting capabilities.

## Data from Snowflake and MongoDB can be extracted, transformed, and loaded into Power BI for creating interactive dashboards, reports, and visualizations.

## Power BI dashboards provide insights into employee skills, certification statuses, project experiences, and other relevant metrics for informed decision-making.

## ETL Processes (dbt - Data Build Tool):

## dbt serves as the data modeling and transformation layer, providing a SQL-based workflow for building data pipelines and transformations.

## dbt can be used to perform transformations on raw data from MongoDB and Snowflake, creating structured datasets optimized for reporting and analytics in Power BI.

## Technology stack (e.g., programming languages, frameworks, libraries)

* + **React**: Utilized for building dynamic and responsive user interfaces, React provides a component-based architecture and virtual DOM for efficient rendering.
  + **MongoDB**: Employed as the database solution for its flexibility with unstructured data, MongoDB stores user profiles, skill information, and project details in JSON-like documents.
  + **Node.js**: Used as the backend runtime environment, Node.js enables server-side logic execution with its non-blocking I/O model, integrating seamlessly with JavaScript on both frontend and backend.
  + **Express**: Chosen as the web application framework for Node.js, Express simplifies the process of building robust APIs with its minimalist design and middleware support.
  + **Bootstrap**: Employed for frontend development to ensure rapid prototyping and consistent UI design across different devices, Bootstrap offers a plethora of pre-designed components and responsive layouts.
  + **Power BI**: Integrated for data visualization and analytics, Power BI empowers users to create interactive reports and dashboards, extracting insights from the Skill Matrix System's data.
  + **Material-UI**: Utilized for UI component libraries, Material-UI provides pre-built React components following Google's Material Design guidelines, facilitating the creation of visually appealing interfaces.
  + **DBT (Data Build Tool):** Incorporate DBT into your technology stack for modeling, transforming, and testing your data in Snowflake. DBT's SQL-based approach enables you to define data transformations, manage dependencies, write tests, and generate documentation within your data warehouse environment.
  + **Python**: To generate fake data with Python, utilize libraries like Faker or Random for user profiles, skills, certifications, and project experiences. Define functions to generate data based on predefined templates. Then, establish a connection to MongoDB using the pymongo library with appropriate parameters like host, port, database name, and credentials if needed. Finally, insert the generated data into MongoDB collections using pymongo functions tailored to each collection.

# Authentication and Authorization

**1. User Roles:**

1. **Admin:**
   * **Responsibilities:**
     + Create new user accounts.
     + Approve or reject certificates and project experiences.
     + Manage system settings and configurations.
   * **Permissions:**
     + Full access to all administrative functionalities.
     + Can access and modify any user's profile data.
2. **User:**
   * **Responsibilities:**
     + Maintain their own profile, including skills and project experiences.
     + Submit certificates and project experiences for approval.
   * **Permissions:**
     + View and update their own profile data.
     + Submit certificates and project experiences for approval.
     + View approved certificates and project experiences.
3. **Approver:**
   * **Responsibilities:**
     + Review and approve/reject certificates and project experiences submitted by users.
     + Ensure the accuracy and validity of submitted data.
   * **Permissions:**
     + - Access to approve/reject functionalities for certificates and project experiences.
       - View details of submitted certificates and project experiences awaiting approval.

**2. Authentication:**

• Describe the authentication process used to verify user identities, such as:

• JWT (JSON Web Tokens): Explain how users obtain JWT tokens upon successful login.

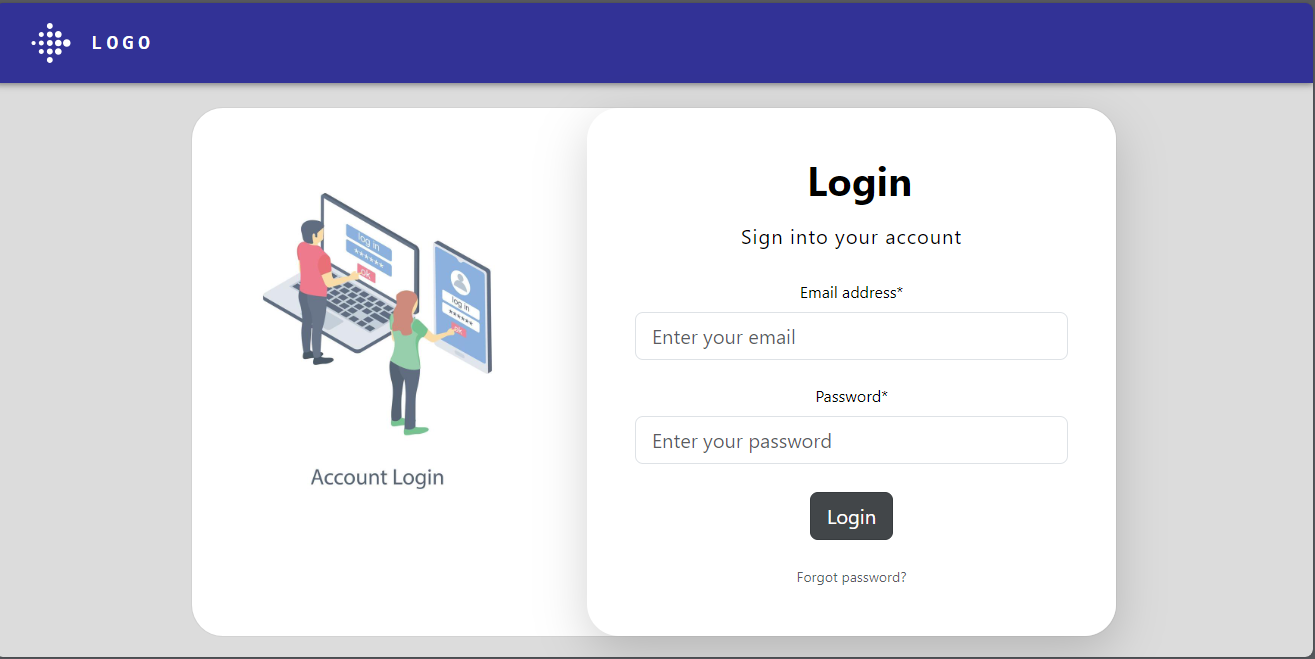
• Authentication API: Detail the endpoints and methods used for user authentication.

**3. Authorization:**

• Specify the authorization rules for different endpoints and actions, including:

• Only admins can access endpoints for creating new users, approving certificates, and project experiences.

• Users can only access and manage their own profile data.

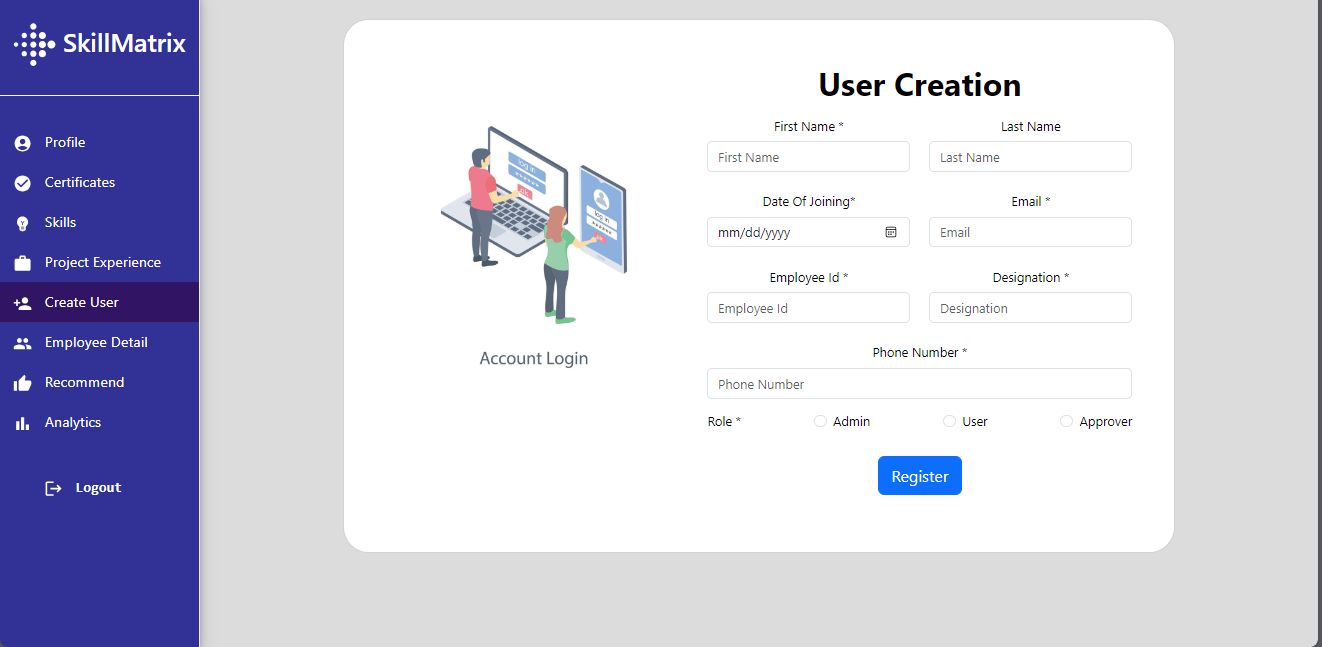


**4. New User Creation:**

• Outline the process for creating a new user account:

• Only admins can create new user accounts through a dedicated admin interface or API endpoint.

• Upon creation, a temporary password is generated and emailed to the new user.



**5. Password Reset:**

• Describe the process for resetting forgotten passwords:

• Users can request a password reset by providing their email address.

• An email containing a password reset link is sent to the user's email address.

• The link expires after a set period or upon successful password reset.

**6. Approval Workflow:**

• Explain the workflow for approving certificates and project experiences:

• Certificates and project experiences submitted by users are marked as pending approval.

• Admins or designated approvers review and approve/reject pending submissions.

• Upon approval, the certificates and project experiences become visible and accessible to users.

**7. Email Notifications:**

• Detail the email notifications sent to users and admins for important events, including:

• New user creation: Notify users of their account creation and provide instructions to change their temporary password.

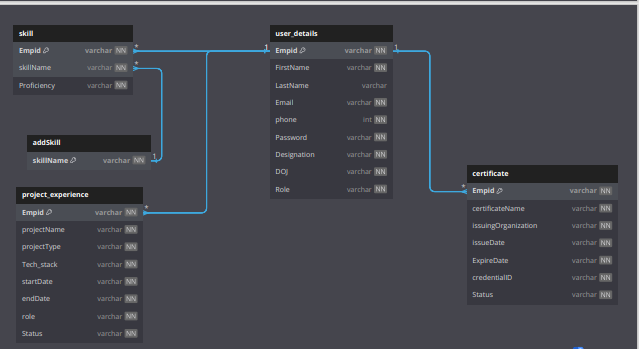
• Password reset: Notify users of their password reset request and provide a link to reset their password.

• Certificate and project experience approval: Notify users when their submissions are approved or rejected.



# Data Model

### Entity Relationships:



* **User\_Detail Table:**
  + Attributes: Emp\_id (Primary Key), username, email, password, role
* **Certificates Table:**
  + Attributes: Emp\_id , certificate\_name, issuer, date\_earned,status
  + Relationship: Many-to-One with User\_Detail (One User can have Multiple Certificates)
* **Skills Table:**
  + Attributes: Emp\_id , user\_id (Foreign Key), skill\_name, proficiency\_level
  + Relationship: Many-to-One with User\_Detail (One User can have Multiple Skills)
* **Project\_Experience Table:**
  + Attributes: project\_name, project\_description, start\_date, end\_date,Status
  + Relationship: Many-to-One with User\_Detail (One User can have Multiple Project Experiences)
* **New\_Skill Table:**
  + Attributes: (Primary Key), skill\_name,
  + Relationship: No direct relationship with User\_Detail (directly related through Skills)

### Data Flow Diagrams:

### 

1. **Recommendation Model:**

**1 Overview:**

The recommendation model is designed to suggest suitable employees for project assignments based on their skills, certifications, and proficiency levels. Leveraging text preprocessing techniques and machine learning algorithms, the model identifies employees whose profiles align closely with the user's specified requirements.

**2 Data Preparation:**

The model assumes the existence of a dataset, merged\_data, containing comprehensive information about employees, including their certificates, skills, and proficiency levels. This dataset is transformed into a pandas DataFrame (df) to facilitate further processing and analysis.

**3 Text Preprocessing:**

To enhance the effectiveness of text analysis, a stem\_text function is defined to apply stemming to the combined text of certificate names, skill names, and tech stacks. Stemming reduces words to their root form, thereby standardizing text representations. The resulting stemmed text is stored in a new column, stemmed\_text, within the DataFrame.

**4 Defining Required Tech Stack and Proficiency:**

The recommendation model incorporates user-defined requirements regarding the desired tech stack and proficiency levels. These requirements are converted to lowercase and concatenated into a single string, representing the user's criteria for candidate selection.

**5 Feature Extraction with TF-IDF:**

Text data is converted into numerical vectors using the TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer from the scikit-learn library. The fit\_transform method is applied to the stemmed text data (df['stemmed\_text']) to generate feature vectors (X) representing each employee profile. TF-IDF assigns weights to terms based on their frequency in a document and inverse frequency across documents, capturing their importance in distinguishing profiles.

1. **Calculating Similarity:**

Cosine similarity is computed between the feature vectors of employee profiles and the feature vector representing the user's specified requirements. This metric quantifies the similarity between two vectors by measuring the cosine of the angle between them. Higher cosine similarity scores indicate greater alignment between employee profiles and user preferences.

1. **Ranking Employees:**

The calculated cosine similarity scores are added as a new column (similarity\_score) to the DataFrame. Employees are then ranked based on their similarity scores, with the top 10 candidates selected as recommendations for the user. This ranking ensures that employees whose profiles closely match the user's requirements are prioritized for consideration.

1. **Skill Training Recommendation:**

From the top 10 recommended employees, the model extracts their skill names and identifies the most common skill among them. This skill is recommended to the user as a potential area for training or skill development, facilitating strategic workforce planning and talent development initiatives.

**Appendix Title**

Document Title