SkillCorp Group(Employee Skill Management System)

Major Project

Contents

1 Project Overview

1.1.1 Brief introduction of the project.

The Skill Matrix System project aims to streamline the management of employee competencies, certifications, and project involvement within the organization. It includes features such as a secure login page, user profile creation, skill and project data input, automated email notifications, and a comprehensive dashboard view. Throughout the development process, key functionalities have been implemented to ensure efficiency, security, and user friendliness.

1.1.2 Purpose of the project.

The goal of this project is to empower employees and enhance their professional experience by providing them with a robust skill matrix management solution. This solution aims to streamline the process of tracking employees' skills, certifications, and project experiences, ensuring transparency, efficiency, and career development opportunities within the organization.

The key goals of the project include:

- Develop a secure login portal for authorized access to the Skill Matrix System.
- Enable administrators to create user profiles with essential details such as name, role, and contact information.
- Allow users to input their skill sets, certifications, and project experiences, including proficiency levels.
- Implement a mechanism for assigning approvers to review and approve users' certifications and project engagements.
- Establish an automated email notification system to keep users and approvers informed about pending evaluations and system updates.
- Provide a comprehensive dashboard interface for users and administrators to monitor skill profiles, certification statuses, and project involvements.

1.1.3 Goals of the project:

- Enhance Employee Empowerment: The primary goal is to empower employees by providing them with a robust skill matrix management system, enabling them to take ownership of their professional development and career progression within the organization.

- Improve Talent Management: The project aims to enhance talent management practices by facilitating comprehensive tracking of employee skills, certifications, and project experiences, thereby enabling more strategic decision making related to resource allocation, training initiatives, and project staffing.

- Increase Organizational Efficiency: By implementing an efficient skill matrix system, the project seeks to optimize workforce utilization, streamline skill matching for project assignments, and reduce administrative overhead associated with manual skill tracking processes, ultimately leading to improved operational efficiency.

- Ensure Skill Alignment with Organizational Goals: A key goal of the project is to ensure that employees' skills and competencies align with organizational objectives and strategic initiatives, enabling the organization to leverage its workforce effectively to drive business success.

- Promote Continuous Learning and Development: The project aims to foster a culture of continuous learning and development within the organization by providing employees with access to training resources, skill assessment tools, and personalized development plans tailored to their career aspirations and organizational needs.

- Enhance Decision Making with Data Insights: By collecting and analyzing data on employee skills and proficiency levels, the project aims to provide decisionmakers with valuable insights for workforce planning, succession management, and talent retention strategies, enabling data driven decision making processes.

- Facilitate Compliance with Industry Standards: The project seeks to ensure compliance with industry standards and best practices related to talent management and skill tracking, aligning with regulatory requirements and organizational policies to maintain high standards of talent management practices.

- Support Employee Engagement and Satisfaction: Ultimately, the project aims to support employee engagement and satisfaction by providing transparent career development opportunities, recognizing and rewarding skills and achievements, and fostering a sense of value and belonging within the organization.

2 Project Scope

2.1.1 Key features and functionalities of the web app.

Key features and functionalities of the web app are:

1. User Authentication and Authorization: Implement secure user authentication and authorization mechanisms to ensure that only authorized personnel can access the system, with role based permissions for different user roles such as administrators, managers, and employees.

2. User Profile Management: Provide users with the ability to create and manage their profiles, including personal information, contact details, and professional credentials such as skills, certifications, and project experience.

3. Skill Tracking and Proficiency Levels: Enable users to input their skills and indicate proficiency levels for each skill, allowing for comprehensive skill tracking and assessment within the organization.

4. Certification Management: Allow users to record their certifications, including certification name, issuing authority, expiration date, and any relevant documentation, facilitating easy verification and tracking of certification statuses**.**

5. Project Experience Tracking: Enable users to input details of their project experience, including project names, roles, duration, and key responsibilities, providing a comprehensive overview of their professional experience within the organization**.**

6. Approver Assignment and Evaluation: Implement a workflow for assigning approvers to evaluate and approve users' certifications and project experiences, ensuring accuracy and reliability of the data recorded in the system.

7. Notification System: Set up automated email notifications to notify users about pending evaluations, expiring certifications, or updates to their profiles, keeping them informed and engaged with the system.

8. Dashboard and Reporting: Provide users with a dashboard interface where they can view summaries of their skill profiles, certification statuses, and project involvements, as well as generate customizable reports for further analysis and decision making.

9. Search and Filtering Capabilities: Incorporate search and filtering functionalities to allow users to quickly find and access relevant information, such as searching for employees with specific skills or certifications.

10. Integration with Training Resources: Integrate with external training resources or learning management systems to provide users with access to relevant training materials and courses based on their skill gaps or career aspirations.

By incorporating these key features and functionalities, the web app aims to provide users with a comprehensive and user friendly platform for managing their skills, certifications, and project experiences, ultimately facilitating talent management and career development within the organization.

3 Architecture and Technology Stack

3.1.1 Overall architecture of the web app.

The overall architecture of the web app follows a modern and scalable approach, designed to ensure robustness, flexibility, and security. Here's an overview of the architecture:

Client Side (Frontend):

- The client side of the web app is built using modern frontend technologies such as HTML5, CSS3, and JavaScript frameworks like React.js.
- It provides a responsive and intuitive user interface for interacting with the application.
- Client side routing is implemented to manage navigation within the app without full page reloads, providing a smoother user experience.

Server Side (Backend):

- The server side of the web app is built using a scalable and efficient backend framework such as Node.js or Django, depending on the specific requirements and preferences.
- The backend serves as the logic layer of the application, handling user requests, business logic, and data processing.
- It interacts with the database to retrieve and manipulate data, as well as external APIs for additional functionality or integration with other systems.

API Layer:

- A RESTful API (Application Programming Interface) is implemented to facilitate communication between the frontend and backend components of the web app.
- The API endpoints are responsible for handling various CRUD (Create, Read, Update, Delete) operations related to user profiles, skills, certifications, projects, authentication, and authorization.
- JSON (JavaScript Object Notation) is used as the data interchange format for communication between the client and server.

Authentication and Authorization:

- The web app implements secure authentication and authorization mechanisms to ensure that only authenticated users can access protected resources.
- Techniques such as JSON Web Tokens (JWT) or session based authentication are employed to authenticate users and manage user sessions securely.
- Role based access control (RBAC) is implemented to enforce fine grained access permissions based on user roles and privileges.

Security:

- In addition to encryption, consider using salted hashing algorithms like bcrypt for storing passwords securely. These algorithms add a random salt to each password before hashing, making it more difficult for attackers to crack passwords using rainbow tables or brute force attacks.
- Enforce strong password policies, including minimum length requirements, complexity rules (e.g., uppercase letters, numbers, special characters), and password expiration periods. Educate users about the importance of creating strong, unique passwords and encourage them to use password managers.

Database Management System (DBMS):

- OLTP Database (MongoDB): Continue to utilize MongoDB as the OLTP database for storing transactional data such as employee profiles, skills, certifications, and project experiences. MongoDB's document based model and flexibility are well suited for handling transactional workloads.

- OLAP Database (Snowflake): Maintain Snowflake as the OLAP database for analytical processing. Snowflake's cloud based data warehouse architecture is ideal for scalable analytics and reporting on large datasets.

Data Modelling and Analysis Tools:

- OLTP Data Modelling (MongoDB): Continue designing MongoDB collections to store transactional data efficiently. Consider indexing and schema design to optimize CRUD operations and ensure data integrity.
- OLAP Data Modelling (Snowflake + dbt): Utilize dbt (Data Build Tool) to implement OLAP data modelling in Snowflake. Define dimensional models such as star schemas or snowflake schemas using dbt's modelling capabilities, ensuring that the analytical models support complex reporting and analysis requirements.

Business Intelligence and Analytics Platforms:

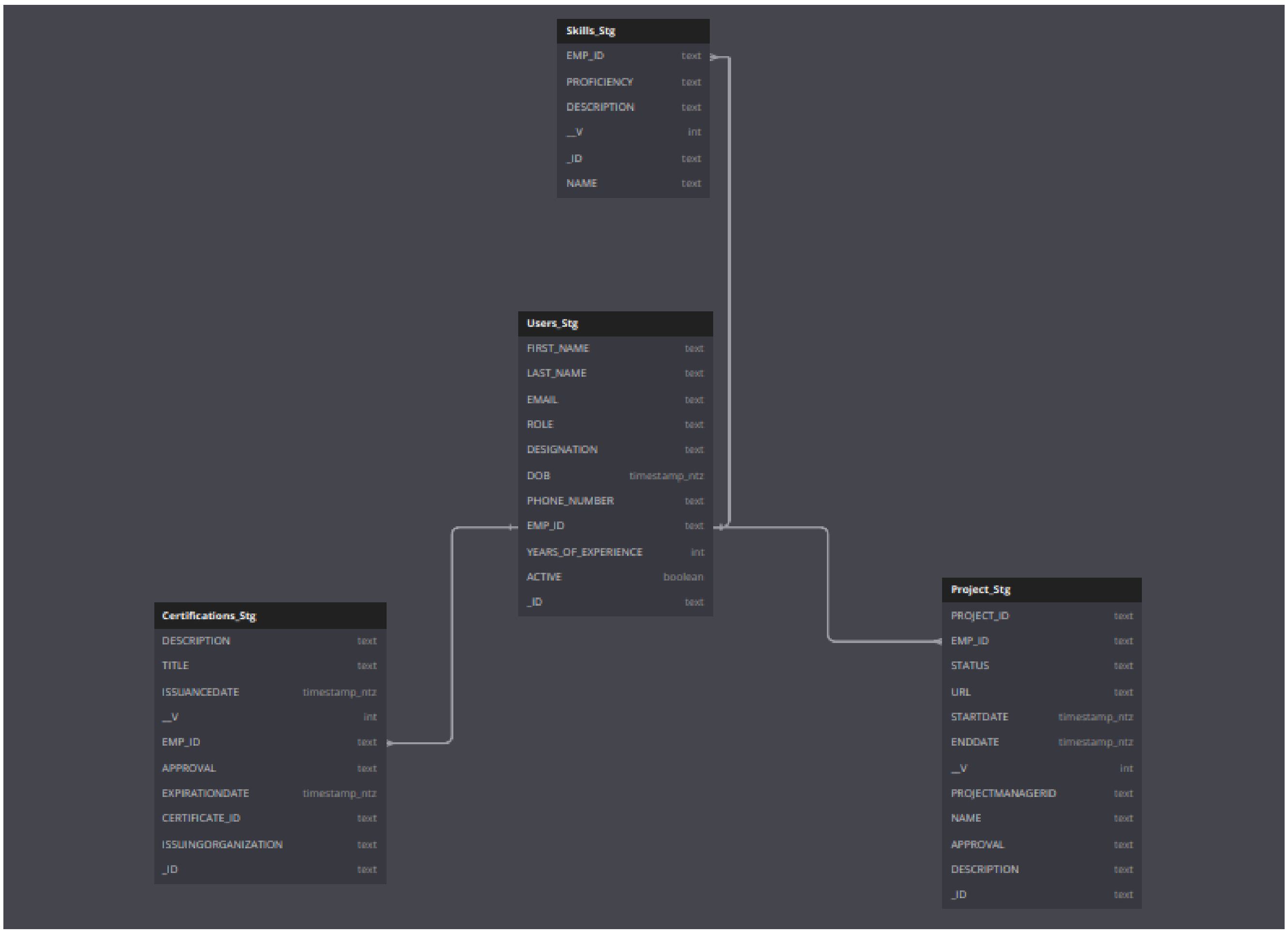
- OLTP Reporting (Power BI): Maintain Power BI for operational reporting on MongoDB data. Create reports and dashboards to monitor employee profiles, skill data, and project information in real time.

- OLAP Analytics (Power BI + Snowflake): Continue leveraging Power BI for advanced analytics and reporting on Snowflake data. Connect Power BI to Snowflake to perform complex analytical queries, visualize data trends, and generate insights from OLAP data stored in Snowflake.

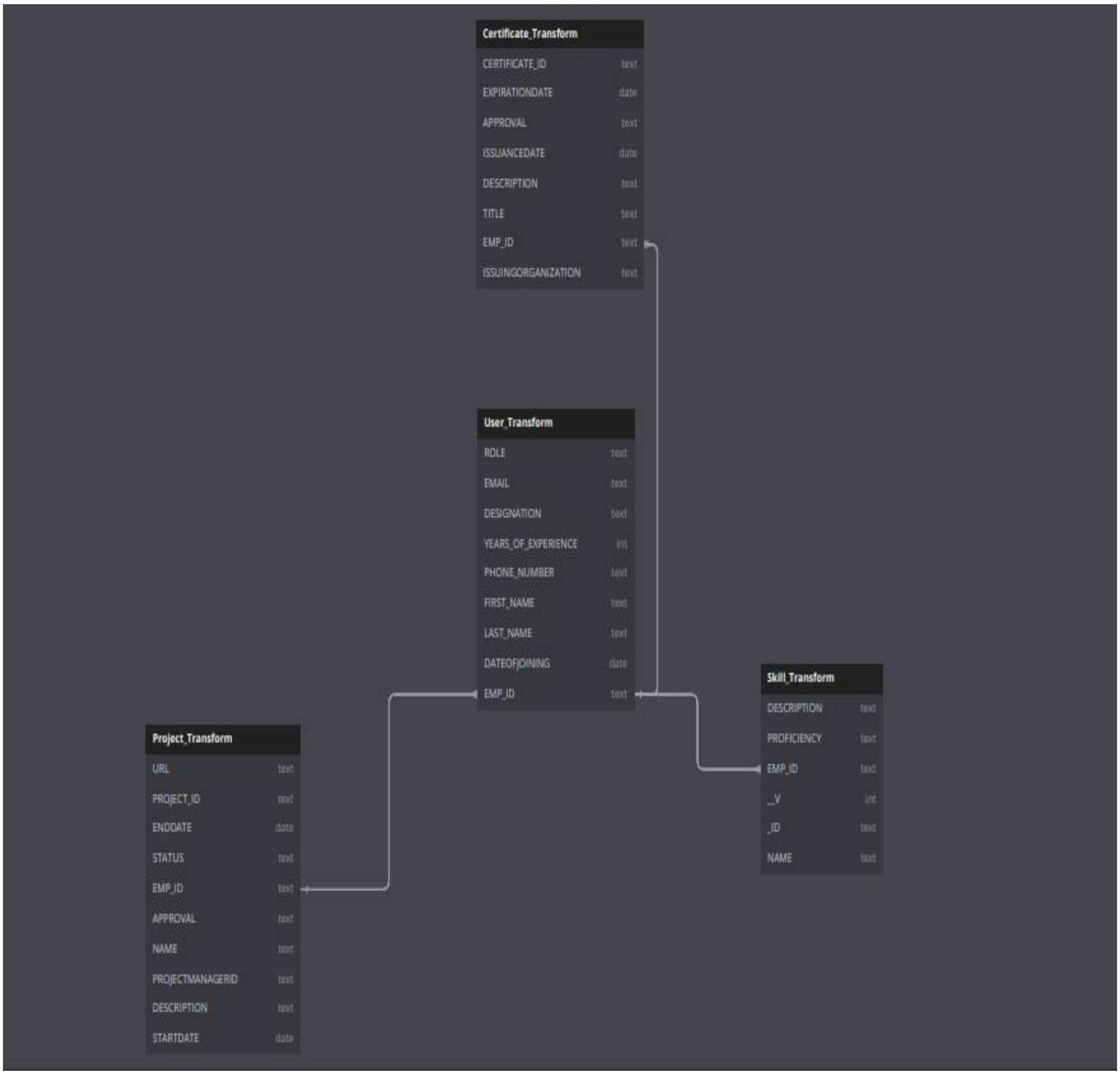
3.1.2 Architecture

DBT Layer Architecture:

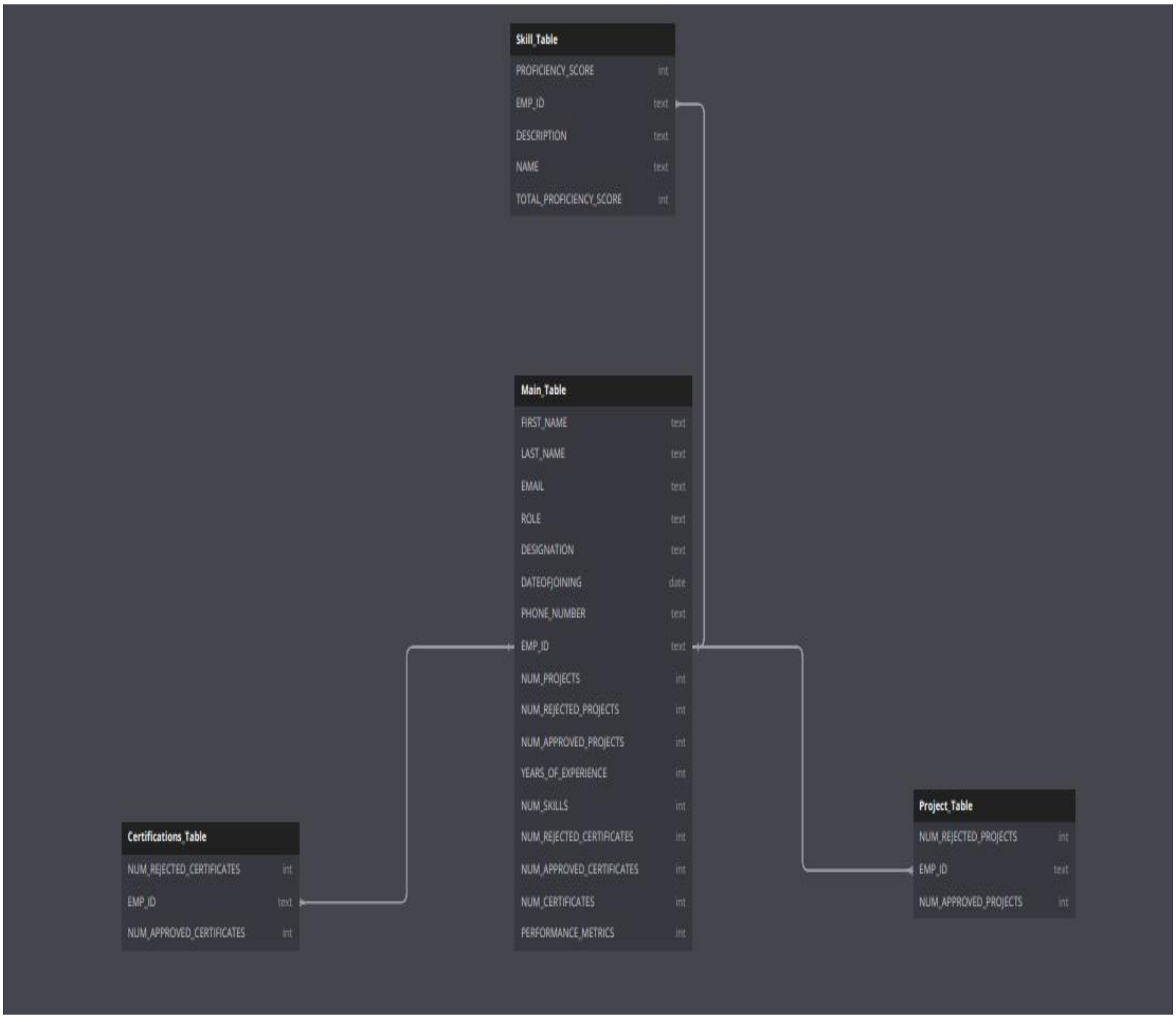
Staging-



Transformation-



Mart-



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

1. Frontend:

- **HTML**: Used for structuring the web pages and defining their content.
- **CSS:** Employed for styling the HTML elements, including layout, colors, fonts, and visual effects.
- **React**: Utilized as the frontend JavaScript library for building interactive user interfaces and managing the application state efficiently.
- **JavaScript** (JS): Used for clientside scripting, enabling dynamic behavior and interactions within the web pages.
- **Axios**: A JavaScript library used for making HTTP requests from the browser to the server or external APIs, facilitating data retrieval and manipulation.

2. Backend(Machine Learning):

- **Python**: Chosen as the backend programming language, primarily for machine learning tasks and serverside logic.

3. Machine Learning:

- **Python**: Utilized for implementing machine learning algorithms, data preprocessing, model training, and evaluation.
- **Scikitlearn, TensorFlow**: Python libraries used for machine learning tasks, offering a wide range of algorithms and tools for building and deploying ML models.

4. Data Transformation and Modelling:

- **dbt (Data Build Tool):** Used for data transformation, Modelling, and analysis in the backend. dbt leverages SQL for defining transformations and building analytical models.
- **SQL (Structured Query Language):** SQL queries are written within dbt to transform raw data into analytics ready datasets, perform data Modelling, and generate reports.

5. Database Management:

- **Snowflake (Data Warehouse):** Used as the backend data storage and analytics platform. Snowflake provides scalable cloud based data warehousing solutions and supports SQL for querying and data manipulation.

6. Other Tools and Libraries:

- **CORS (Cross Origin Resource Sharing):** Used to enable cross origin requests from the frontend to the backend server. CORS is essential for allowing resources on one domain to be requested from another domain securely.

This refined technology stack reflects your usage of Python for ML tasks and SQL within dbt for data transformation and Modelling. It also includes the necessary frontend and backend technologies for building a modern web application, along with the tools and libraries commonly used in ML and data transformation processes.

3.1.4 Rationale behind the chosen technology stack.

The rationale behind the chosen technology stack for your project, which involves employee skill management and likely includes machine learning (ML) components, can be summarized as follows:

1. Python for Backend and Machine Learning:

- **Versatility**: Python is renowned for its versatility and ease of use, making it an excellent choice for both backend development and machine learning tasks. Its extensive library ecosystem, including Flask and Django for web development and scikitlearn, TensorFlow, and PyTorch for ML, provides robust support for various aspects of your project.
- **Data Science Ecosystem**: Python's dominance in the data science ecosystem ensures seamless integration with machine learning libraries and frameworks, allowing for efficient data preprocessing, model training, and evaluation.
- **Community Support**: Python boasts a vibrant and active community, offering abundant resources, tutorials, and third party packages to streamline development and troubleshooting.

2. React for Frontend:

- **ComponentBased Architecture**: React's component based architecture enables modular development, making it easier to manage complex user interfaces and ensure code reusability.
- **Virtual DOM**: React's virtual DOM efficiently updates only the necessary parts of the UI when data changes, resulting in improved performance and a smoother user experience.

- **Large Ecosystem**: React has a large ecosystem of libraries and tools (such as Redux for state management) that complement its core functionalities, providing flexibility and extensibility for frontend development.

3. dbt and SQL for Data Transformation and Modelling:

- **Data Pipeline Efficiency:** dbt's use of SQL for defining data transformations and Modelling enables a straightforward and efficient data pipeline, leveraging the familiarity and power of SQL for analytics and reporting tasks.
- **Version Control and Collaboration:** dbt's focus on version control and collaboration streamlines the development process, facilitating team collaboration and ensuring consistency and reproducibility in data transformations.
- **Integration with Snowflake**: The compatibility of dbt with Snowflake, a scalable cloud based data warehouse, allows for seamless integration of data transformation and Modelling workflows with the backend data storage and analytics platform.

4. Snowflake for Data Warehousing:

- **Scalability and Performance**: Snowflake's cloud based architecture provides scalability and elasticity, enabling the storage and analysis of large volumes of data without the need for infrastructure management.
- **Concurrent Access and Workload Isolation**: Snowflake's multi cluster architecture ensures concurrent access to data and workload isolation, optimizing performance and resource utilization for analytical workloads.
- **Security and Compliance:** Snowflake offers robust security features, including encryption, access controls, and auditing, ensuring data security and compliance with regulatory requirements.

By selecting this technology stack, you leverage the strengths of each component to build a scalable, efficient, and feature rich solution for employee skill management. Python and React enable seamless integration of backend logic, machine learning capabilities, and frontend user interfaces, while dbt and Snowflake streamline data transformation, Modelling, and analytics workflows, ensuring data accuracy, consistency, and accessibility. Overall, the chosen technology stack aligns with the project's requirements and objectives, providing a solid foundation for success.

4 Web app Components

4.1.1 Main components of the web app.

1. Frontend Components:

Components:

- AdminPage
- ApproverDesk
- ApproverPage
  CerificatePage
- DataAnalysis
  ForgetPassword
  Home
  LoginPage
- Logout
- ProjectPage
  ResetPasswordPage
- SignUp
- SkillPage
- UserDataAnalysis
- UserDetails
- UserManagement
- UserPage

2. Backend Components:

Controllers:

- authcontroller
- userController

Models:

- certificateModel
- projectModel
- skillModel
- userModel

Routes:

- authRouter
- userRouter

Utils:

- email
- seeder

3. Database Components:

- users
- projects
- certificates
- skills

4.1.2 Purpose of each component.

1. Frontend Components:

- **AdminPage**: Interface for administrative tasks and management.
- **ApproverDesk**: Dashboard for approvers to review and approve certifications or project involvements.
- **ApproverPage**: Page for individual approvers to manage certification or project approval tasks.
- **CertificatePage**: Interface for viewing and managing certifications.
- **DataAnalysis**: Component for analyzing data, possibly including visualizations and reports.
- **ForgetPassword**: Functionality for users to reset forgotten passwords.
- **Home**: Landing page or homepage of the web app.
- **LoginPage**: Login interface for users to access the web app.
- **Logout**: Functionality to log out and end the user session.
- **ProjectPage**: Interface for viewing and managing projects.
- **ResetPasswordPage**: Page for resetting user passwords.
- **SignUp**: User registration page.
- **SkillPage**: Interface for viewing and managing skills.
- **UserDataAnalysis**: Component for analyzing userspecific data.
- **UserDetails**: Page for viewing and managing user details.
- **UserManagement**: Interface for managing users, possibly including roles and permissions.
- **UserPage**: User profile page.

2. Backend Components:

Controllers:

- **authcontroller**: Handles authenticationrelated logic, such as login, logout, and password reset.
- **userController**: Manages userrelated operations, such as user creation, deletion, and updates.

Models:

- **certificateModel**: Defines the structure and behavior of certification data.
- **projectModel**: Defines the structure and behavior of project data.
- **skillModel**: Defines the structure and behavior of skill data.
- **userModel**: Defines the structure and behavior of user data.

Routes:

- **authRouter**: Defines routes for authenticationrelated endpoints.
- **userRouter**: Defines routes for userrelated endpoints.

Utils:

- **email**: Utility functions for sending emails, such as password reset instructions.
- **seeder**: Utility for seeding initial data into the database.

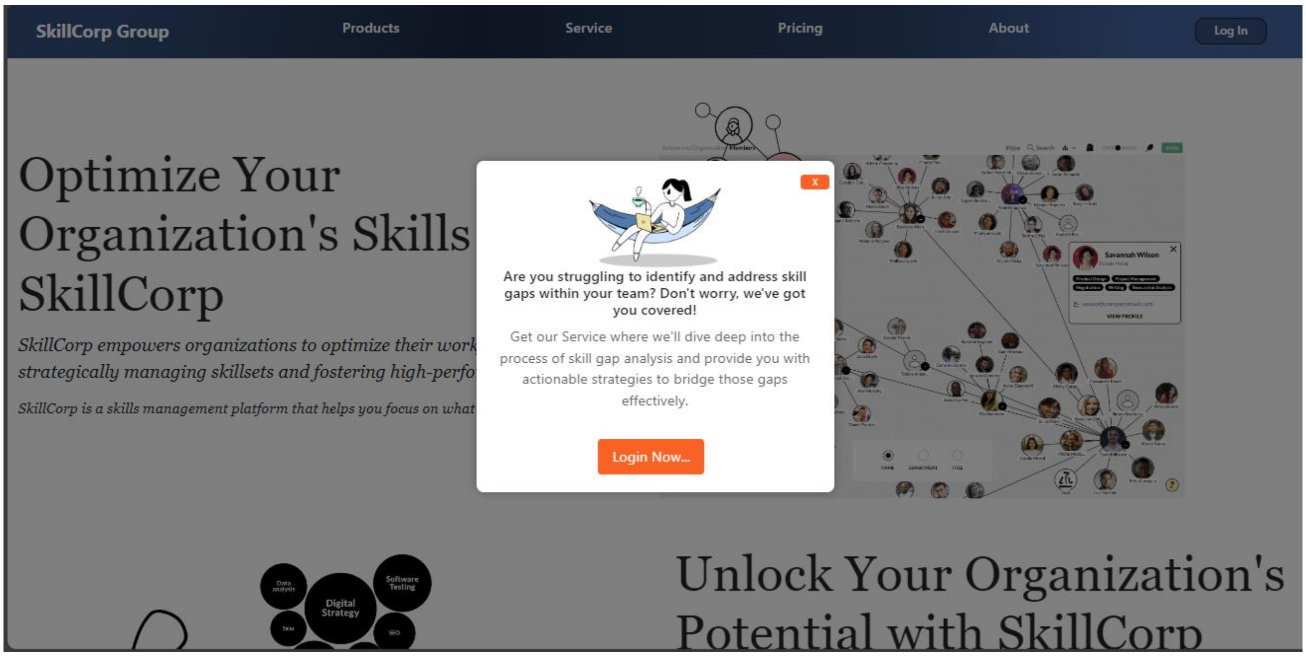
3. Database Components:

- **users**: Database table for storing user information.
- **projects**: Database table for storing project information.
- **certificates**: Database table for storing certification information.
- **skills**: Database table for storing skill information.

Each component serves a specific purpose within the web app, contributing to its overall functionality and user experience.

5 User Interface(UI) design approach:

Home Page:



SkillCorp Group Products Service Pricing About Log In

Optimize Your Organization's Skills SkillCorp

SkillCorp empowers organizations to optimize their work strategically managing skillsets and fostering high-perfo

SkillCorp is a skills management platform that helps you focus on what

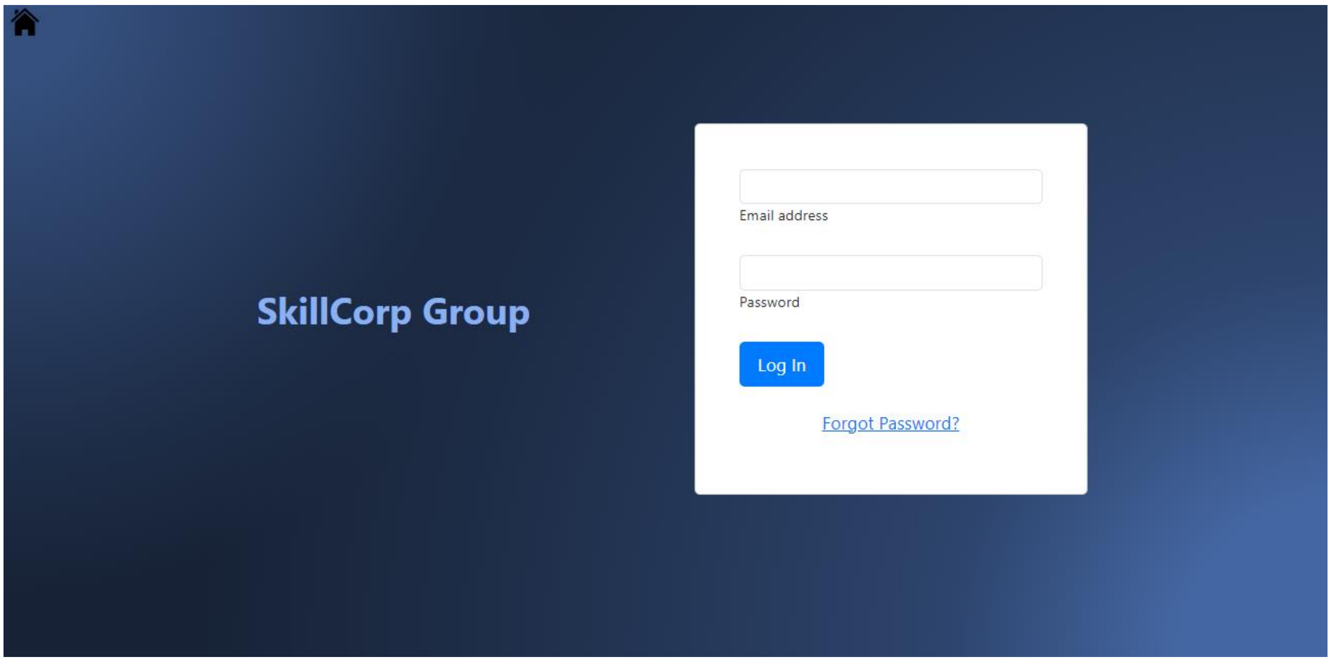
Are you struggling to identify and address skill gaps within your team? Don't worry, we've got you covered!

Get our Service where we'll dive deep into the process of skill gap analysis and provide you with actionable strategies to bridge those gaps effectively.

Login Now...

Unlock Your Organization's Potential with SkillCorp

Login Page:



SkillCorp Group

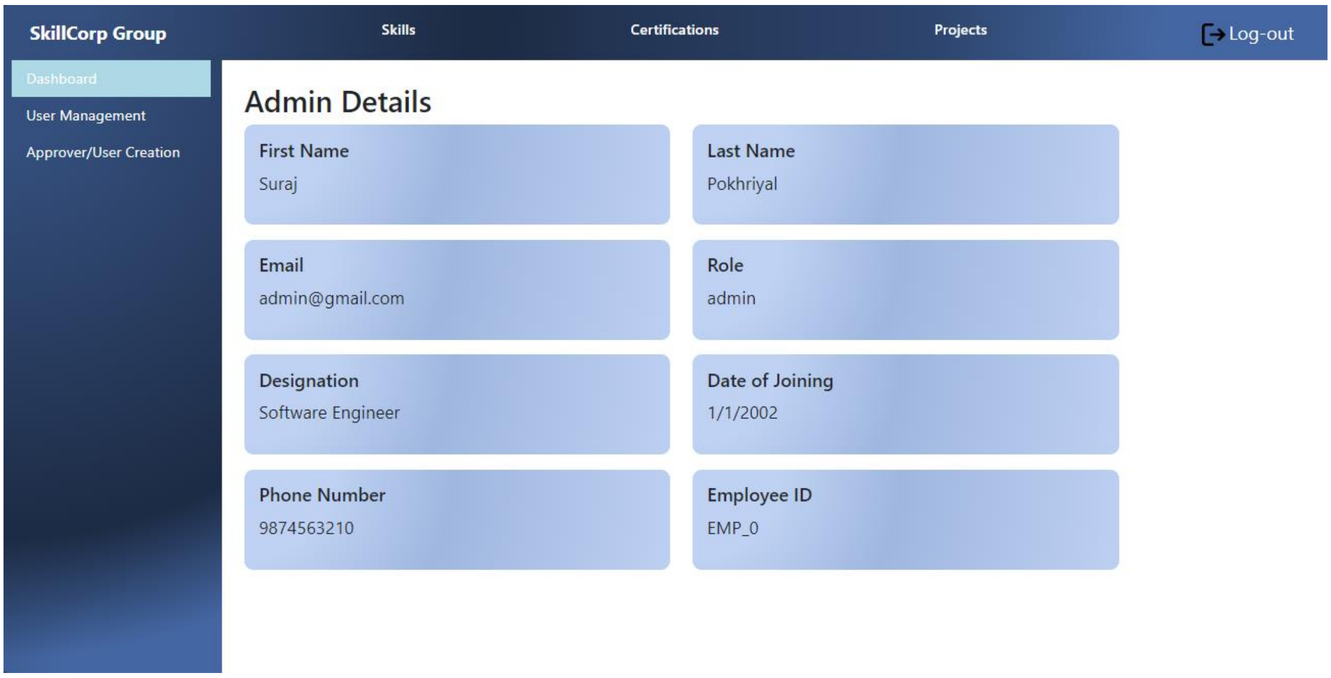
Email address

Password

Log In

Forgot Password?

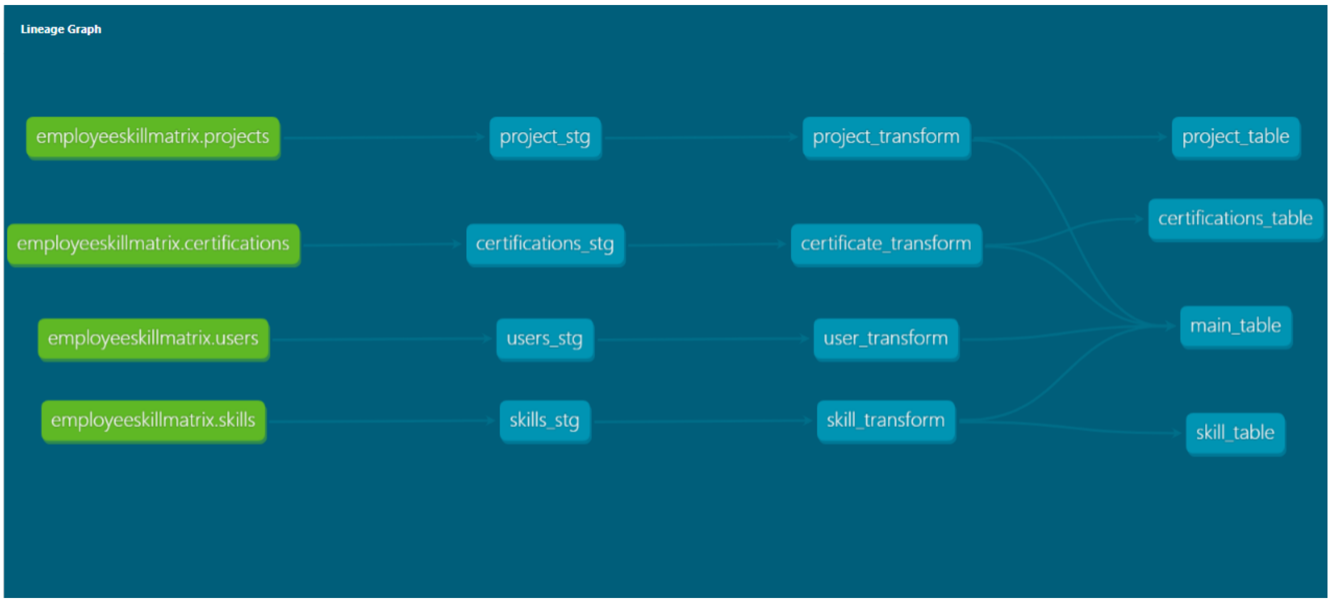
Admin/User/Approver Page:



Admin Details

First Name Suraj	Last Name Pokhriyal
Email admin@gmail.com	Role admin
Designation Software Engineer	Date of Joining 1/1/2002
Phone Number 9874563210	Employee ID EMP_0

dbt Lineage Graph:



6 Testing and Quality Assurance

For your website, the testing approach and types of testing to be performed are critical to ensure its quality, reliability, and functionality. Here's how you can apply the testing approach and various types of testing to your employee skill management web app:

6.1.1 Testing approach for the web app.

1. **Requirement Analysis:**

   Understand the requirements of your employee skill management web app thoroughly, including its functionality, user roles, and data management needs.

2. **Test Planning:**

   Develop a comprehensive test plan outlining testing objectives, scope, environments, test cases, and techniques.

   Determine the types of testing to be performed, including functional testing, usability testing, system testing, retesting, regression testing, and compatibility testing.

3. **Test Environment Setup:**

   Set up test environments that closely mimic the production environment, including databases, servers, and client devices.

   Ensure compatibility across different browsers, devices, and operating systems to cover a wide range of user scenarios.

4. **Functional Testing:**

   Verify that all features and functionalities of the web app meet the specified requirements.

   Test user authentication, skill management, certification validation, and project assignment functionalities.

5. **Usability Testing:**

   Assess the user interface for consistency, responsiveness, and ease of use.

   Evaluate navigation, intuitiveness, learnability, and error frequency and severity to optimize user experience.

6.1.2 Types of testing to be performed..

1. **Functional Testing:**

   Ensure each function of the application behaves as expected, validating inputs, outputs, and expected outcomes.

   Verify that users can perform actions such as signing up, logging in, managing skills, and viewing project details without errors.

2. **Usability Testing:**

Evaluate the web app's interface across different devices, screen sizes, and resolutions to identify compatibility issues.

Measure user satisfaction, navigation ease, learnability, and error frequency to enhance user experience.

3. System Testing:

Test the entire system from end to end to validate compliance against specified requirements.

Cover scenarios involving multiple components interacting with each other, including user authentication, data processing, and database management.

4. Retesting:

Execute previously failed tests against new software versions to ensure identified issues have been resolved.

Verify that fixes have been implemented correctly and that the application operates as intended.

5. Regression Testing:

Perform regression testing to verify that new changes or fixes do not adversely impact existing functionality.

Ensure stability and reliability of existing features, especially after significant updates or enhancements.

6. Compatibility Testing:

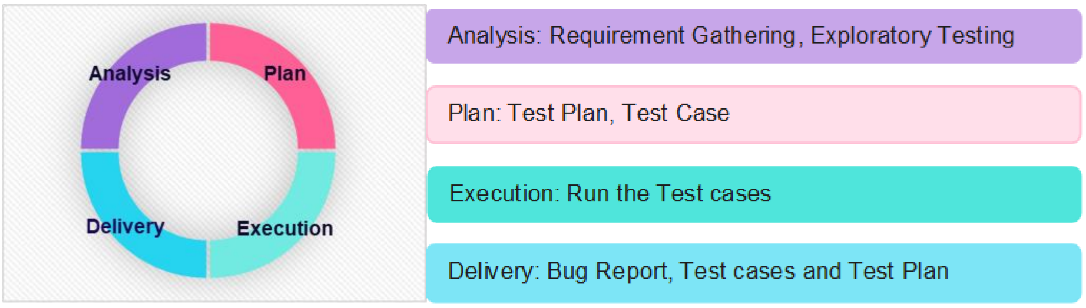
Test the web app's compatibility across different browsers, devices, and operating systems to ensure consistent functionality.

Identify and resolve compatibility issues early to enhance user satisfaction and market acceptance.

By following this testing approach and conducting various types of testing, you can ensure that your employee skill management web app meets quality standards, provides a seamless user experience, and delivers value to users and stakeholders alike.

6.1.3 Quality assurance processes and tools to ensure app reliability.

The objective of the test is to define the goals and purpose of the testing effort. It aims to provide a comprehensive and focused statement of what is to be accomplished through testing. The test objectives serve as a guiding principle for the testing activities and ensure that they are aligned with the overall objectives of the project



Analysis: Requirement Gathering, Exploratory Testing

Plan: Test Plan, Test Case

Execution: Run the Test cases

Delivery: Bug Report, Test cases and Test Plan

7 Risks and Mitigation Strategies

7.1.1 Potential risks and challenges associated with the project.

S.NO	Risk / Challenges	Impact	Mitigation Plan
1	Delay in API Development	Project deliverables will be delayed	Ensure all APIs are finalized and documented before development begins. Collaboration between development and API teams to streamline the process and prioritize critical APIs. Regular communication and updates to manage expectations.