Project Software Requirement Specification On

"ResumeKraft"

Submitted For The

Degree Of

Bachelor Of Computer Application

By

Shiv Prakash Ojha (21201010052)

Aman Kumar Singh (21201010007)

Ritesh Kumar (21201010044)

Kunal Pandey (21201010027)

Asjad Khan (21201010034)

Under The Supervision Of

Mr. Gaurav Dwivedi

United University, Rawatpur, Prayagraj

Table of Contents

- 4.2 Security
- 4.3 Usability
- 4.4 Compatibility
- 4.5 Scalability
- 4.6 Reliability
- 4.7 Database Requirements
- 4.8 Documentation

5. User Interface
   - 5.1 User Interface Design
   - 5.2 User Interaction
6. System Interfaces
   - 6.1 External Interfaces
   - 6.2 API Specifications
7. Testing Requirements
   - 7.1 Unit Testing
   - 7.2 Integration Testing
   - 7.3 User Acceptance Testing
8. Maintenance and Support
   - 8.1 Maintenance Plan
   - 8.2 Support Plan
9. Appendices
   - 9.1 Glossary
   - 9.2 UML Diagrams
   - 9.3 Data Flow Diagrams
   - 9.4 Entity-Relationship Diagrams

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to define the functional and non-functional requirements for the development of "ResumeKraft," a web application that allows users to create, store, and download their resumes. This document serves as a reference for developers, testers, and stakeholders to ensure that the application is developed and delivered in accordance with the specified requirements.

1.2 Scope

ResumeKraft will be a web-based platform that enables users to create and manage their resumes through an interactive form. Users can register, log in, input their personal and professional details into the system, and generate professional-looking resumes. The application will store user data securely and allow users to download their resumes in various formats (e.g., PDF, Word).

1.3 Definitions, Acronyms, and Abbreviations

- **HTML: Hypertext Markup Language**
    - The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the meaning and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

- **CSS: Cascading Style Sheets**
    - CSS stands for **Cascading Style Sheets**. It is the language for describing the presentation of Web pages, including colours, layout, and fonts, thus making our web pages presentable to the users.

- **JavaScript: A programming language for web development**
    - JavaScript is a programming language that adds interactivity to your website. This happens in games, in the behavior of responses when buttons are pressed or with data entry on forms; with dynamic styling; with animation, etc. This article helps you get started with JavaScript and furthers your understanding of what is possible.

- **Django: A Python web framework**
    - Django is a Python-based web application framework that is free and open source. A framework is simply a collection of modules that facilitate development. They're grouped together and allow you to build apps or websites from scratch rather than starting from scratch.

- **MySQL: A relational database management system**
    - MySQL is an open-source relational database management system. As with other relational databases, MySQL stores data in tables made up of rows and columns. Users can define, manipulate, control, and query data using Structured Query Language, more commonly known as SQL.

1.4 References

ReportLab documentation :- https://docs.reportlab.com/

Django documentation :- https://docs.djangoproject.com/en/4.2/

1.5 Overview

The remainder of this document provides detailed information on the system, including its functionality, architecture, requirements, user interfaces, and more.

2. System Overview

2.1 System Description

ResumeKraft is a web application developed using HTML, CSS, JavaScript, the Django web framework, and a MySQL database. It allows users to create, store, and download professional resumes by filling out an interactive form. Users can register, log in, and manage their profiles.

2.2 System Architecture

The system follows a client-server architecture. The client-side consists of HTML, CSS, and JavaScript for the user interface, while the server-side is powered by Django, which handles user authentication, form submission, and resume generation. MySQL is used to store user data and resumes.

2.3 User Classes and Characteristics

- **Guest Users:** Users who visit the website without logging in.
- **Registered Users:** Users who have registered and logged into the system.
- **Administrators:** System administrators who have additional privileges for user management.

2.4 Operating Environment

The application is designed to run on web browsers that support HTML, CSS, and JavaScript. The server-side components run on a compatible web server environment.

2.5 Design and Implementation Constraints

- The application is developed using Django, which follows the Model-View-Controller (MVC) architectural pattern.
- The application will be responsive and accessible on various devices and screen sizes.

2.6 Assumptions and Dependencies

- It is assumed that users have basic internet connectivity and web browsing capabilities.
- The system relies on third-party libraries and frameworks (e.g., Django, MySQL) for its functionality.

3. Functional Requirements

3.1 User Registration and Authentication

1. **User Registration**: Users can register by providing a valid email address, and password.
2. **User Login**: Registered users can log in using their email address and password.
3. **Forgot Password**: Users can request a password reset if they forget their password.
4. **User Profile**: Users can update their profile information, including name, contact details, and profile picture.

3.2 Resume Creation

5. **Interactive Form**: Users can fill out a form with sections for personal information, education, work experience, skills, and other relevant resume sections.
6. **Real-time Validation**: The system should provide real-time validation and feedback to assist users in completing the form.
7. **Preview**: Users can preview their resume before finalizing it.

3.3 Resume Storage

8. **User-specific Storage**: Each user's resume data is stored securely and associated with their account.
9. **Data Security**: User data is encrypted and protected against unauthorized access.

3.4 Resume Retrieval

10. **Resume List**: Users can view a list of their saved resumes.
11. **Resume Details**: Users can view and edit the details of a specific resume.

3.5 Form to Resume Conversion

12. **Generate Resume**: Users can convert the data entered in the form into a professional-looking resume format.
13. **Download Options**: Users can choose to download their resumes in various formats (e.g., PDF, Word).

3.6 Resume Download

14. **Download Resume**: Users can download their resumes to their local devices.
15. **Download History**: Users can view a history of their resume downloads.

4. Non-Functional Requirements

4.1 Performance

16. **Response Time**: The system should respond to user actions within an acceptable time frame.
17. **Scalability**: The system should be scalable to handle a growing user base and resume data.

4.2 Security

18. **User Authentication**: User authentication must be secure and protect against unauthorized access.
19. **Data Encryption**: User data, including passwords, should be stored securely with encryption.
20. **Data Privacy**: User data should not be shared with third parties without user consent.

4.3 Usability

21. **User-Friendly Interface**: The user interface should be intuitive and easy to use.
22. **Accessibility**: The application should be accessible to users with disabilities.

4.4 Compatibility

23. **Browser Compatibility**: The application should be compatible with commonly used web browsers.
24. **Mobile Responsiveness**: The application should be responsive on mobile devices.

4.5 Scalability

25. **Database Scalability**: The database should be designed for scalability as the user base grows.

4.6 Reliability

26. **Availability**: The application should be available and reliable for users.

4.7 Database Requirements

27. **Data Backup**: Regular backups of user data and resumes should be maintained.
28. **Database Maintenance**: Database maintenance tasks should be scheduled and documented.

4.8 Documentation

29. **User Documentation**: Provide user documentation explaining how to use the application.
30. **Code Documentation**: Ensure code is well-documented for maintenance and future development.

5. User Interface

5.1 User Interface Design

The user interface should have a clean and modern design with a focus on user experience. It should follow best practices in web design.

5.2 User Interaction

User interactions should be intuitive, with clear instructions and feedback during form filling, resume generation, and other activities.

6. System Interfaces

6.1 External Interfaces

The application may integrate with external services for features such as email notifications and authentication.

6.2 API Specifications

Specify any APIs used for integration with third-party services.

7. Testing Requirements

7.1 Unit Testing

Unit tests should be developed to ensure the functionality of individual components, including forms, models, and views.

7.2 Integration Testing

Integration tests should be conducted to verify the interaction between different components of the system.

7.3 User Acceptance Testing

Stakeholders and end-users should participate in user acceptance testing to validate that the system meets their requirements.

8. Maintenance and Support

8.1 Maintenance Plan

A maintenance plan should be established to address bug fixes, updates, and improvements after the initial deployment.

8.2 Support Plan

A support plan should be in place to provide assistance to users and address their inquiries or issues.

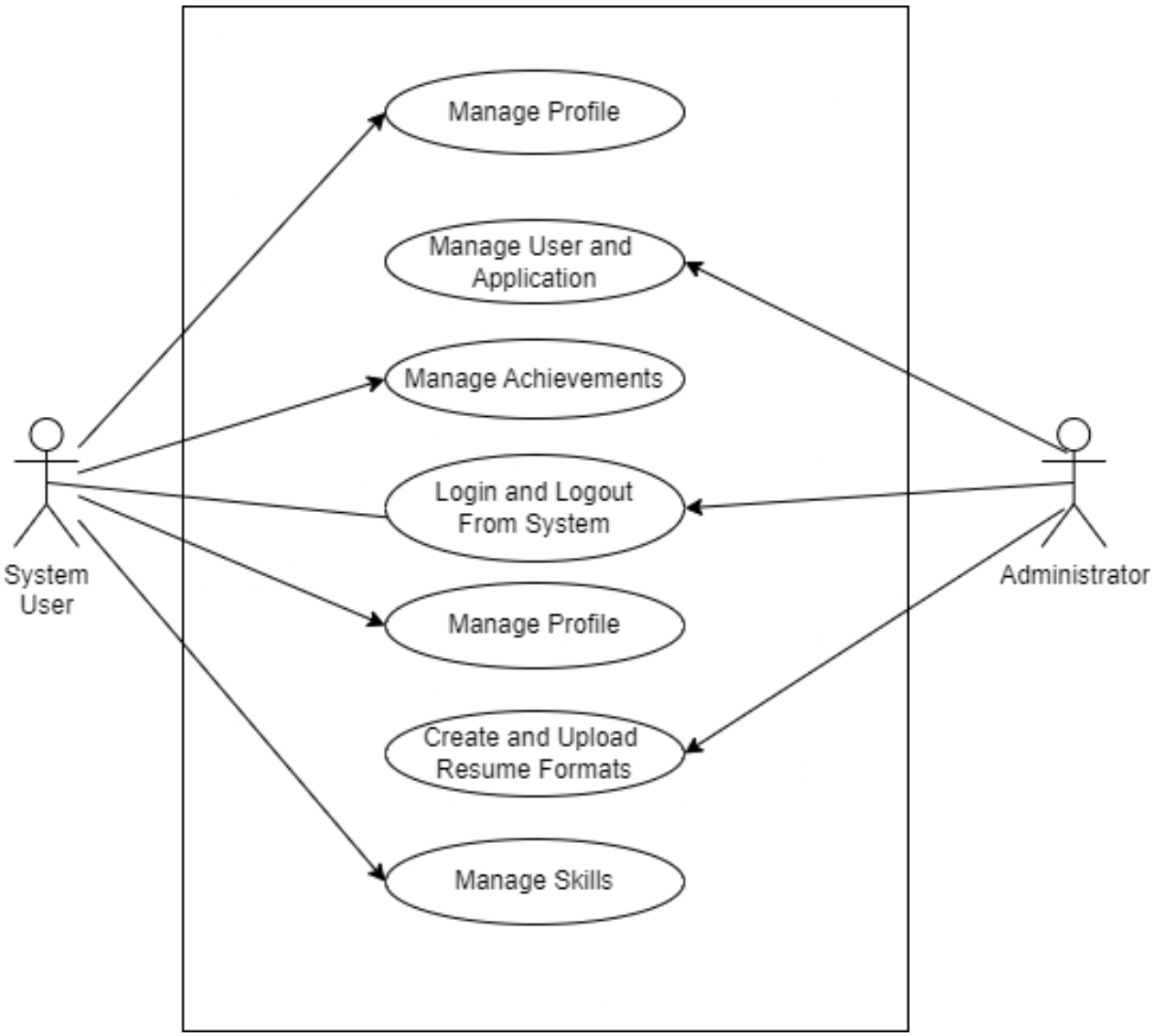
9. Appendices

9.1 Glossary

API :- Application Programming Interface
SRS: Software Requirements Specification
HTML: Hypertext Markup Language
CSS: Cascading Style Sheets
JavaScript: A programming language for web development
Django: A Python web framework
MySQL: A relational database management system

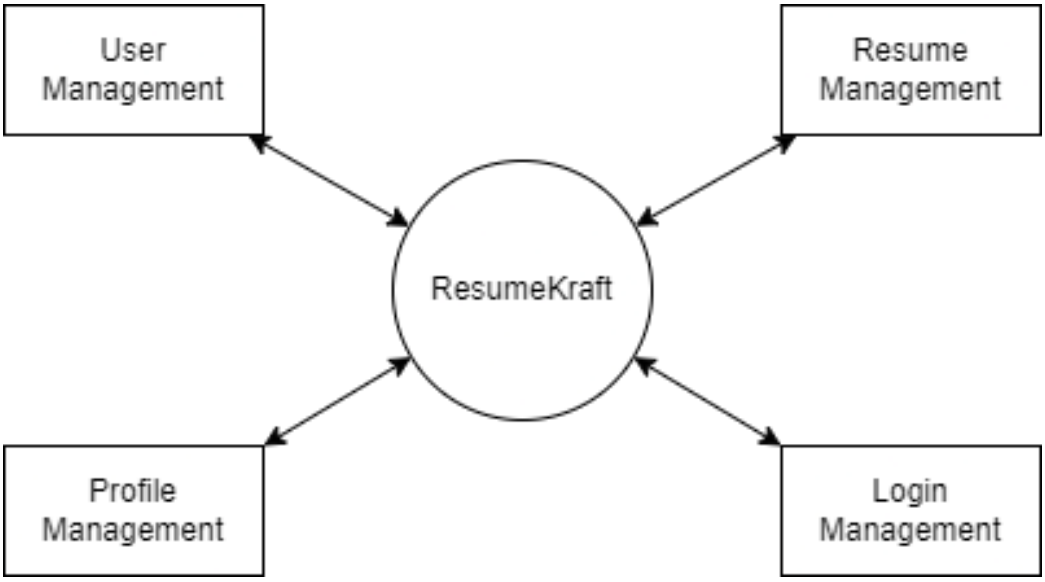
9.2 UML Diagrams

9.2.1 Use Case Diagram

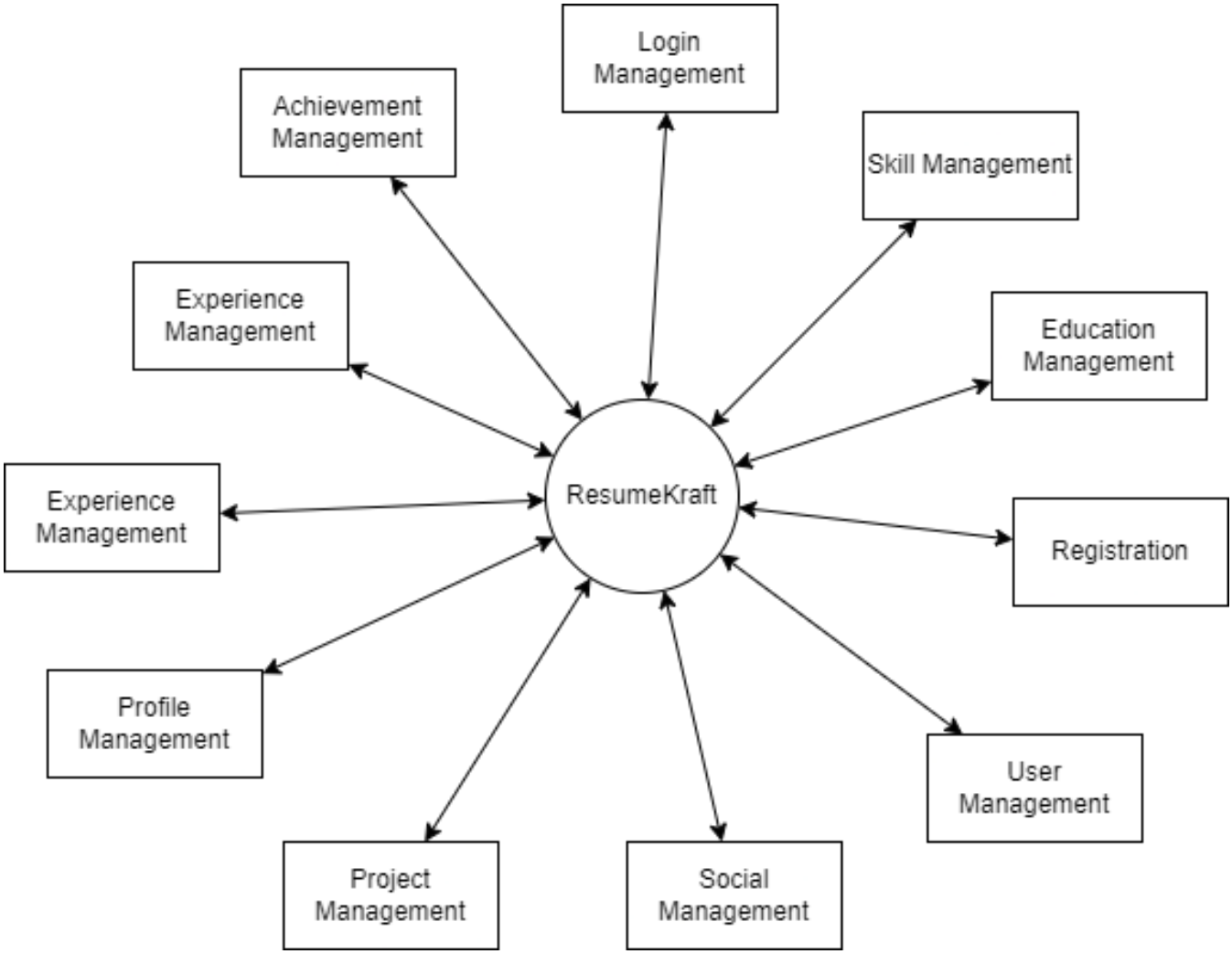


9.3 Data Flow Diagrams

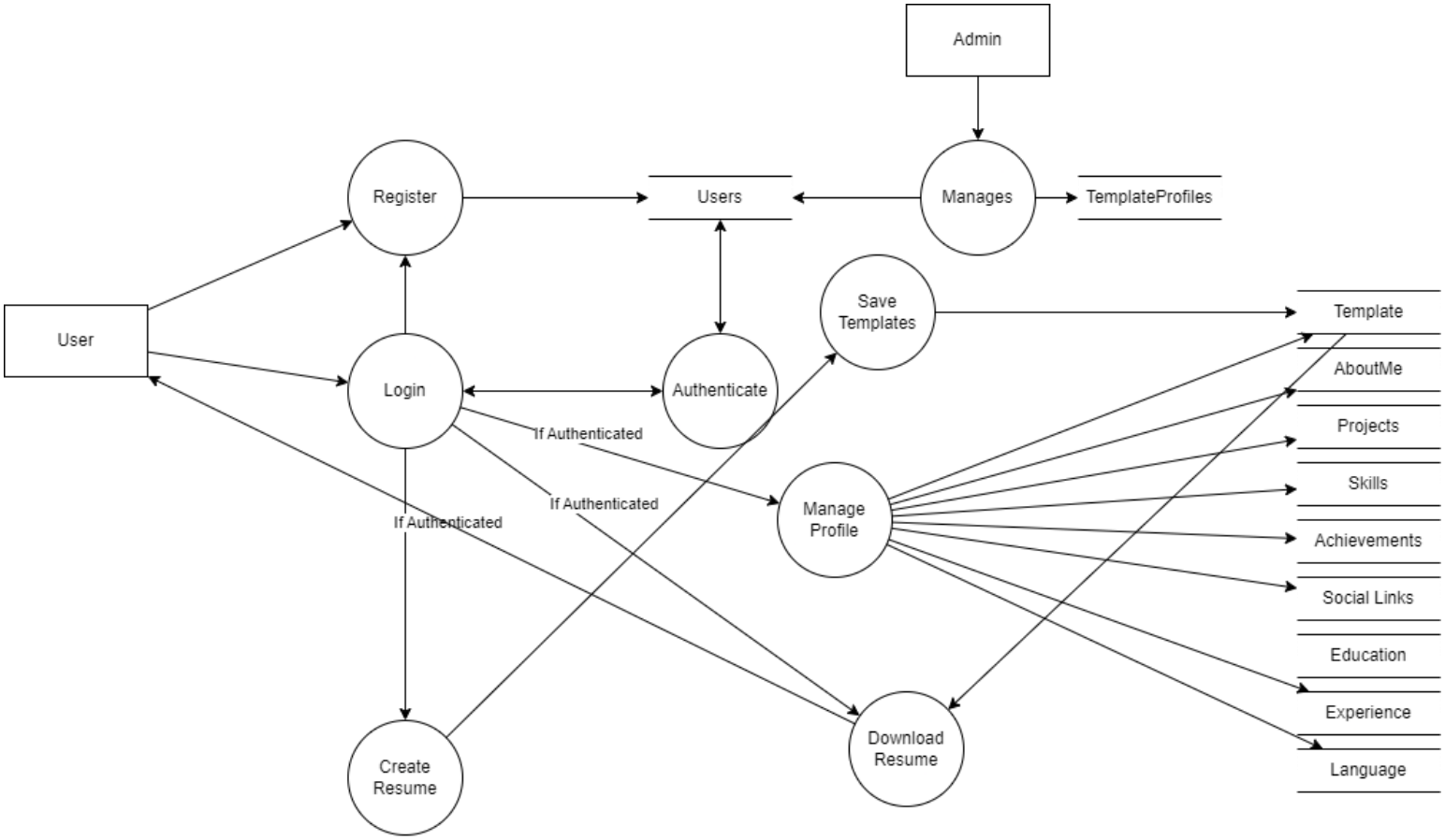
9.3.1 DFD 0: Context Diagram



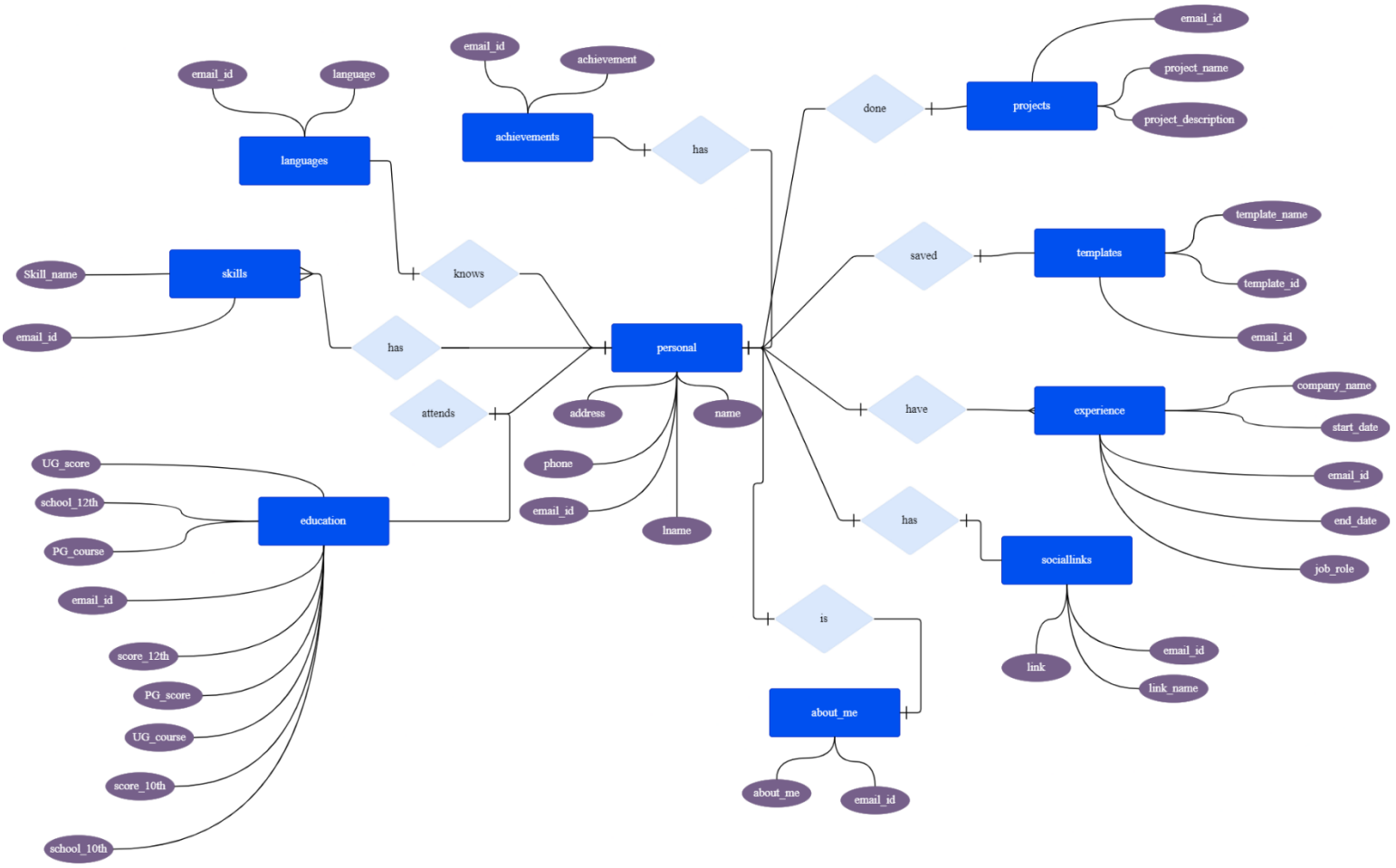
9.3.2 DFD 1: Process Decomposition



9.3.2 DFD 2: Deeper Dives



9.4 Entity-Relationship Diagrams



9.5 DataBase Diagrams