SOFTWARE REQUIREMENTS SPECIFICATION

for

Design and Implementation of a Lightweight Education Data Bay Area (E-DBA)

Version 1.0 Approved

Prepared by:
Taian Yu (FULL) Yolanda Yin (FULL)
Justin Zhang (FULL) Aaron Liu (FULL)
Steven Luo (FULL) Younger Yang (FULL)

Group C01

February 26, 2025

Contents

Chapter 1

Introduction

All external database interfaces mentioned in the document (such as the student information database, the degree thesis database, the bank account database, etc.) should be integrated and invoked according to the interfaces configured by the providers.

1.1 Purpose

This project aims to design and implement a lightweight **Educational Data Platform (E-DBA)** to enable secure and efficient data-sharing services among registered higher education institutions. The platform's core objectives include providing convenient and secure mechanisms for data provisioning and consumption, supporting student identity authentication and thesis-sharing services, integrating online payment functionality, and facilitating course information exchange. Additionally, it offers optional data warehousing services for institutions lacking independent database maintenance capabilities. Operating under compliance with data-sharing policies and data sovereignty regulations, the E-DBA platform enables authorized data providers and consumers to exchange educational resources while safeguarding institutional data rights, ultimately fostering collaborative innovation and service optimization across academic organizations.

1.2 Document Conventions

This document adheres to the following standardized conventions:

1.2.1 Formatting and Typography

The documentation maintains consistent typographic practices, with body text in a uniform font and size. Headings are bolded and organized in a hierarchical structure through section divisions to clarify content organization.

1.2.2 Terminology Guidelines

Key terms are consistently defined and formatted for clarity. For example, **data providers** and **data consumers** refer to user roles that supply data to the system and retrieve data from the system, respectively.

1.2.3 Modular Structure

Content is structured according to the system's functional components and modules, such as user role management, data-sharing services, payment systems, and other operational divisions, ensuring logical separation and ease of navigation.

1.3 Intended Audience and Reading Suggestions

This document is intended for the following audience groups:

Project Managers and Developers: Tailored to clarify the project's overarching objectives, system functionalities, and technical implementation requirements.
System Administrators: Provides comprehensive details on administrator privileges, configuration workflows, and system management procedures.
End Users (e.g., students, faculty, and staff): Offers step-by-step operational guidance for accessing and utilizing services within the E-DBA platform.

Readers are advised to begin with the overview section to grasp the system's holistic architecture and capabilities before diving into specific functional modules, which cover role-based workflows, technical specifications, and user-centric tutorials aligned with their responsibilities or interests.

1.4 Project Scope

The development of the E-DBA platform's user interface, backend management system, and associated service modules, alongside the implementation of role-based access control for diverse user categories such as administrators, organizational coordinators, and data providers. Core functionalities include data provisioning and consumption mechanisms covering student identity verification, degree thesis access, and course information sharing, as well as the integration of a multi-payment-method online transaction system. Additionally, the platform will offer data warehousing capabilities to enable dataset export and collaborative sharing. Emphasis is placed on ensuring cross-institutional compatibility, robust data security protocols, and intuitive user experience design to streamline data interoperability and service delivery across participating educational organizations.

1.5 References

Foundational insights are drawn from the Eclipse Dataspace Connector (EDC) framework documentation, including its technical specifications and implementation guidelines for secure data exchange. Architectural principles align with the reference models and interoperability standards published by the International Data Spaces Association (IDSA). Additionally, compliance considerations incorporate relevant educational data governance frameworks, cross-institutional data-sharing

policies, and sector-specific regulatory requirements. Technical implementation details are informed by industry-standard development protocols, API integration best practices, and security benchmarks for distributed systems, ensuring alignment with modern data infrastructure paradigms.

Chapter 2

Overall Description

2.1 Product Perspective

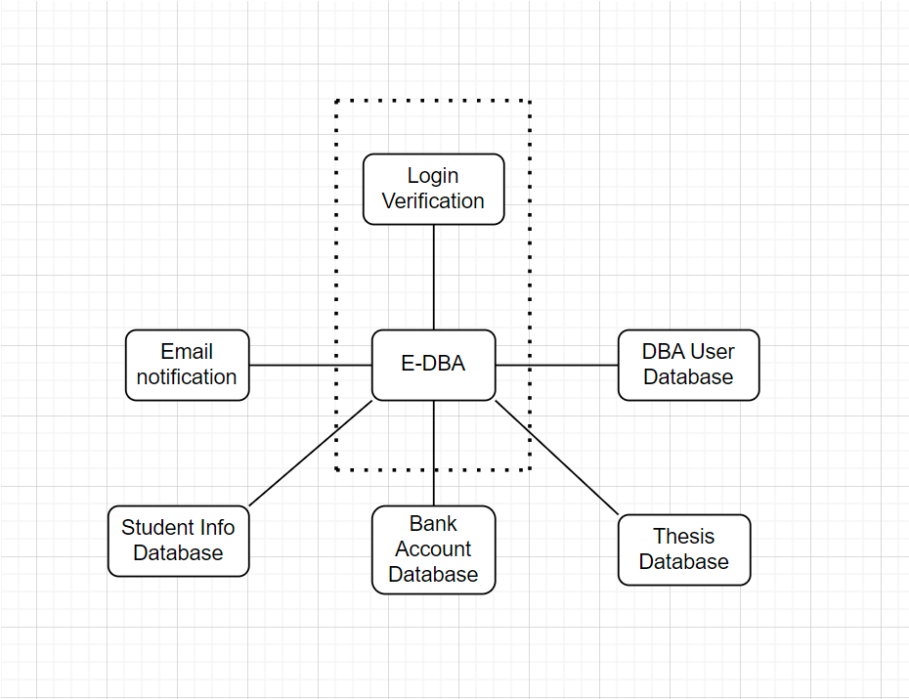

Figure 2.1: Product Perspective

2.2 Product Functions

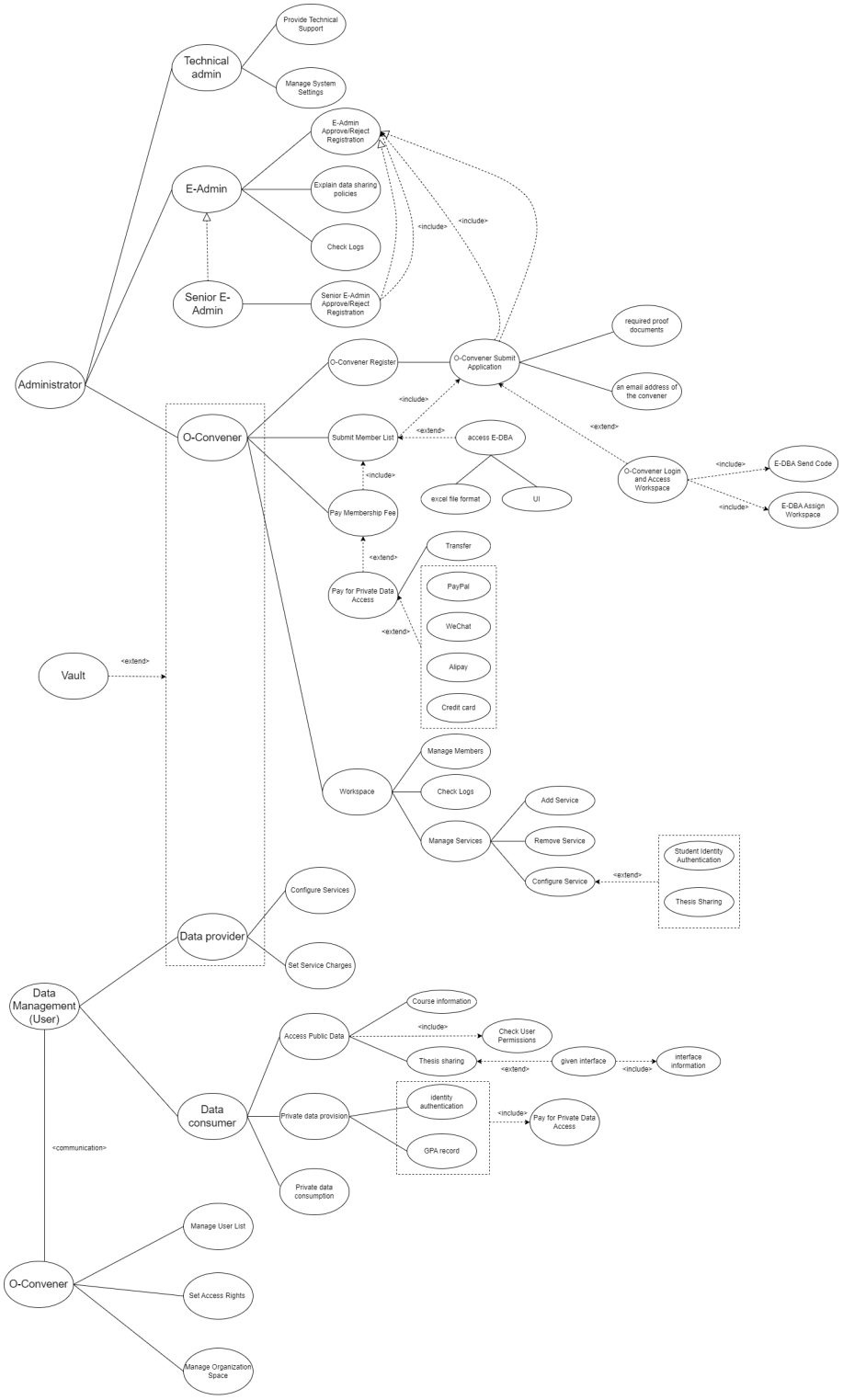

Figure 2.2: Product Function

The registration process initiates when the **O-Convener** completes the registration form (Step 1), providing contact information, institutional affiliation, and required documentation. Subsequently, the **E-Admin** or **Senior E-Admin** conducts

a comprehensive review (Step 2), including material verification, authenticity validation, and background screening. Upon approval, system validation occurs (Step 3), generating a unique convener ID and activating service access privileges. Finalization (Step 4) triggers automated credential distribution and system onboarding initiation.

Registration Rejection: In cases of administrative rejection, the system issues an automated notice detailing rejection rationale and appeal procedures, while preserving records for 30 days per GDPR compliance.
Incomplete Submission: System-level validation identifies missing elements through automated checks, flagging incomplete fields and initiating follow-up protocols requiring convener remediation before process continuation.

2.3 User Classes and Characteristics

- **Admin**:
  - **T-admin**: Responsible for overseeing and managing system functionality. Has access to the full suite of administrative controls.
  - **E-admin**: Handles user and system management, including configuration of system features and settings.
  - **Senior E-admin**: Manages both system and user configurations at an advanced level.
- **O-Convener**: Focuses on managing user lists, workspace organization, and service configurations for the system.
- **Data Provider and Data Consumer**:
  - **Data Provider**: Supplies the system with data for processing and consumption.
  - **Data Consumer**: Engages with the system to access and use provided data, including processing payments for data consumption.
- **End Users**: Individuals who interact with the system on a more basic level to access or utilize specific functionalities. End users are the primary consumers of the system's services, with access tailored to their specific needs.

2.4 Operating Environment

In most probability, we will be using either a Windows or Linux operating system in our environment. Windows is widely used in consumer and business contexts due to its user-friendly interface and broad application support. Linux, on the other hand, is popular in server environments and among developers because of its open-source nature, stability, and flexibility. Both operating systems have distinct advantages depending on the specific requirements, such as Windows for graphical user interfaces and compatibility with mainstream software, and Linux

for high-performance, security, and customization.

2.5 Design and Implementation Constraints

- The system must adhere to a high level of security to protect sensitive data and user privacy.
- The system must be capable of connecting to and interacting with a database management system to store and retrieve data efficiently.

2.6 User Documentation

- **User Manuals**: Comprehensive guides tailored to each user class (admin, end user, etc.), providing step-by-step instructions for system setup and usage.
- **On-line Help**: An integrated help system within the software that offers real-time support to solve user problems.
- **Tutorials**: Both video and text-based tutorials designed to help users understand how to operate the system effectively.
- **Release Notes**: A summary of updates and changes made in each version of the system to inform users of new features, fixes, or enhancements.

2.7 Assumptions and Dependencies

Assumptions:

- The stability and reliability of third-party services integrated with the system.

Dependencies:

- The project will rely on a database management system for development and data storage.

Chapter 3

System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

3.1 System Feature 1

<Don't really say "System Feature 1." State the feature name in just a few words.>

3.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

3.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

3.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1: REQ-2:

3.2 System Feature 2 (and so on)

Chapter 4

External Interface Requirements

4.1 User Interfaces

TBD

4.2 Hardware Interfaces

TBD

4.3 Software Interfaces

TBD

4.4 Communications Interfaces

1 Email addresses and electronic forms, including proof documents required by the O-convener for registration.

2 Excel files required by a registered organization to authenticate student identity through batch input.

Chapter 5

Other Nonfunctional Requirements

5.1 Performance Requirements

- **Response Time**: To ensure a good user experience, the system should minimize delays, especially in critical operations such as real-time data access and identity verification.
- **Concurrency Handling**: During peak periods, such as the start of the school term or thesis submission deadlines, the system may experience a large number of concurrent users. The system must have strong concurrency handling capabilities to manage multiple user requests simultaneously.
- **Scalability**: The system should be scalable to accommodate future growth. New features should be easily added, and more users should be supported without affecting existing functionalities.
- **Data Consistency and Reliability**: For key operations such as payments and identity verification, data loss or inconsistencies must be prevented to avoid serious consequences. The system should have robust transaction management and error recovery mechanisms.
- **Resource Consumption**: Developers should optimize the code and architecture to ensure efficient use of resources, avoiding excessive consumption of server resources during operation.

5.2 Safety Requirements

- **Data Breach Risk**: The system must prevent privacy violations and identity theft by encrypting data transmission and implementing strict access control mechanisms. Unauthorized access must be prohibited, and all user activities should be monitored.
- **Payment Security Risk**: The system should mitigate financial transaction risks by using multi-factor authentication and adopting certified secure payment gateways, ensuring the safety and reliability of payment channels.
- **System Stability Risk**: To prevent service interruptions, data loss, or corruption due to system instability, important data must be backed up regularly, and performance metrics should be continuously monitored.

5.3 Security Requirements

- **Data Sharing Security**: The system must ensure that data is shared securely and transparently between different organizations. Data transmission must be encrypted, and only authorized parties should be able to access specific data.
- **Online Payment Security**: All payment operations must be secure to prevent financial fraud. Strict security measures must be taken to protect users' payment information, especially when handling personal financial details.
- **Non-tamperable Log Records**: System logs should be automatically categorized based on user activity and be non-modifiable to ensure the integrity of log data.

5.4 Software Quality Attributes

5.4.1 For Users

- **Usability**: The system's user interface should be easy to use for individuals in different roles, with detailed guidance to help them get started quickly.
- **Reliability**: The system should run stably, minimizing the risk of failures and ensuring a quick recovery if issues arise.
- **Efficiency**: The system should respond quickly to user actions, whether browsing public information or querying private data.

5.4.2 For Developers

- **Maintainability**: Code structure should be clear, following best practices, to ensure easier updates and iterations.
- **Compatibility**: The system should function correctly across various operating systems and browsers.
- **Flexibility**: The system should have flexible service interfaces for easy integration with external systems and support customizable configurations based on organizational needs.
- **Testability**: A comprehensive testing framework should be established, covering unit tests, integration tests, stress tests, and others.
- **Compliance**: The system should adhere to relevant laws and regulations, particularly those relating to personal data protection, following strict privacy guidelines.

5.5 Business Rules

- **Roles and Permissions**: Different users and organizations have specific access rights and permissions. Only authorized users can perform critical operations like identity verification and payment processing.
- **Data Privacy**: The system must ensure that sensitive personal information is protected according to privacy regulations, restricting access to authorized personnel only.

Chapter 6

Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

6.1 Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

6.2 Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

6.3 Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>