**Software Requirements Specification**

**for**

**Project** PolyQuest

**Version 1.0 approved**

**Prepared by**

**Lam Yan Yee**

**Dallas Ng Zhi Hao**

**Yeo, Ruizhi Carwyn**

**Malcolm Fong Cheng Hong**

**Jerwin Lee Chu Hao**

**Khoo Ze Kang**

**NTU, TDDA Group 53**

**10 Nov 2024**

**Table of Contents**

[**1. Introduction 4**](#_heading=h.1fob9te)

[1.1 Purpose 4](#_heading=h.3znysh7)

[**1.2 Document Conventions 4**](#_heading=)

[**1.3 Intended Audience and Reading Suggestions 4**](#_heading=)

[**1.4 Product Scope 5**](#_heading=)

[**1.5 References 5**](#_heading=)

[**2. Overall Description 5**](#_heading=)

[2.1 Product Perspective 5](#_heading=h.2s8eyo1)

[**2.2 Product Functions 5**](#_heading=)

[**2.3 User Classes and Characteristics 6**](#_heading=)

[**2.4 Operating Environment 7**](#_heading=)

[**2.5 Design and Implementation Constraints 7**](#_heading=)

[**2.6 User Documentation 7**](#_heading=)

[**2.7 Assumptions and Dependencies 7**](#_heading=)

[**3. External Interface Requirements 8**](#_heading=)

[3.1 User Interfaces 8](#_heading=h.2jxsxqh)

[**3.2 Hardware Interfaces 8**](#_heading=)

[**3.3 Software Interfaces 8**](#_heading=)

[**3.4 Communications Interfaces 9**](#_heading=)

[4. Functional Requirements 10](#_heading=h.22mqdp5geyvm)

[**5. Nonfunctional Requirements 20**](#_heading=h.3whwml4)

[**5.1 Software Quality Attributes 22**](#_heading=)

[**5.2 Business Rules 23**](#_heading=)

[**6. Other Requirements 24**](#_heading=)

[**7. Use Case Descriptions 25**](#_heading=h.uimliok6sea7)

[I. Account Registration 25](#_heading=h.sn2c4m16t1zw)

[II. Login/Logout 27](#_heading=h.bivhz3h17q6o)

[III. Contact Us 29](#_heading=h.3y1aiwr6li21)

[IV. Activity 31](#_heading=h.mtuw015polu5)

[V. Taketest 34](#_heading=h.seym7iwito2q)

[VI. Dashboard 36](#_heading=h.6cgq1cmo6e8s)

[VII. Bookmarks 39](#_heading=h.hdrxkqj2yhi1)

[VIII. Settings 40](#_heading=h.ic1qudqlethi)

[**Appendix A: Data Dictionary 46**](#_heading=h.y1dvbfytifdz)

[**Appendix B: Analysis Models 46**](#_heading=h.d9xkftudidcz)

[**Appendix C: To Be Determined List 47**](#_heading=h.capnpelnoo64)

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| DALLAS NG  CARWYN YEO | 12/11/2024 | INITIAL SRS DOCUMENT | 1.0 |
| LAM YAN YEE | 12/11/2024 | DIAGRAM INPUTS | 1.0 |

# **Introduction**

## **Purpose**

This document specifies the software requirements for the application Project PolyQuest version Alpha.

Project PolyQuest is a web application which helps secondary school students entering polytechnics make more informed decisions about their tertiary educational paths.

The core feature of this application is a comprehensive personality quiz that matches students with suitable polytechnic courses based on their individuality, characteristics, interests, and aptitudes.

This will allow potential polytechnic students to better navigate through the myriad of courses available, ensuring their choices align with their true passions and strengths.

## **Document Conventions**

Priority of Requirements: The priority of higher-level requirements is assumed to be inherited by detailed requirements unless explicitly stated otherwise. The priority of all higher-level requirements unless explicitly stated otherwise.

**Level 1 Heading**: Font Family: Times, Font Size: 18, Font Weight: Bold

**Level 2 Heading**: Font Family: Times, Font Size: 14, Font Weight: Bold

**Level 3 Heading**: Font Family: Times, Font Size: 12

**Content**: Font Family: Arial, Font Size: 11

Throughout this document, ‘the app’ or simply ‘app’ refers to Project PolyQuest unless explicitly specified otherwise.

Other conventions and definitions of terms used in this document can be found in **Append A: Data**

**Dictionary**.

## **Intended Audience and Reading Suggestions**

This document is intended for users of the app, the software developers of the app, the

documentation writers, the project managers, the marketing staff, and testers.

This document details the description, use cases, functional and non-functional requirements,

interfaces, constraints of the app. It is intended to be read in sequence by all stakeholders involved

in this app.

## **Product Scope**

Project PolyQuest is a web application designed to support secondary students in making informed choices about polytechnic courses. By considering each student’s academic interests, achievements, and responses to a questionnaire, PolyQuest provides a straightforward interface that delivers personalised course recommendations, making the decision process easier and more insightful.

## **References**

1. Source Code (GitHub): https://github.com/softwarelab3/2006-TDDA-53
2. React: <https://react.dev/>
3. Firebase: <https://firebase.google.com/>
4. Google OAuth2.0: <https://developers.google.com/identity/protocols/oauth2>
5. PostgreSQL: <https://www.postgresql.org/docs/current/indexes.html>
6. GovData: <https://data.gov.sg/>
7. Mantine: <https://mantine.dev/>
8. Java Spring Boot: https://spring.io/projects/spring-boot/

# **Overall Description**

## **Product Perspective**

Project PolyQuest is a new app and is not part of a larger product family or product line.

## **Product Functions**

ProjectPolyQuest has the following main functions:

1. Account Registration: Users can create an account to personalize their experience and save progress.
2. Login/Logout: Secure access for users to log in and log out of their accounts.
3. Contact Us: A feature allowing users to reach out for support or inquiries.
4. Activity Tracking: The website tracks users’ interactions and activities to personalize recommendations.
5. Taketest (Personality Quiz): The primary feature allowing students to take a personality quiz to assess their interests and strengths.
6. Dashboard: A central location where users can view their profiles, quiz results, and recommended courses.
7. Bookmarks: Users can save and bookmark recommended courses and other resources for easy access.
8. Settings: Account settings to manage preferences and personal information.

## **User Classes and Characteristics**

1. Students (Primary Use Case)

This group comprises non-tertiary students in Singapore who are looking for guidance on potential educational and career paths.

Characteristics:

* Age Group: Typically 15-18 years old.
* Frequency of Use: Moderate; students may use the platform occasionally during key decision points in their academic journey.
* Technical Expertise: Moderate to low; users may have limited technical expertise, so the platform should be intuitive and easy to navigate.
* Primary Functions Used: Account Registration, Login/Logout, Personality Quiz (Taketest), Activity Tracking, Dashboard, Bookmarks, and Settings.
* Educational Level: Secondary education.

Needs:

* Easy-to-use interface to navigate the quiz and view personalized course recommendations.
* Access to personality and interest assessments tailored to provide relevant insights.
* Ability to save and bookmark recommended courses for future reference.

This is the most important user class as they are the primary end-users. Meeting their needs is critical to the application's success.

1. Administrators (Support User Class)

Platform administrators responsible for managing the application, maintaining content, and supporting users.

Characteristics:

* Frequency of Use: Frequent; administrators manage user accounts, content, and system settings regularly.
* Technical Expertise: High; they are expected to have strong technical skills to handle system maintenance and support.
* Primary Functions Used: User Management, Content Management, System Settings, and Contact Us.
* Educational Level: Varies, though generally experienced in IT or content management.

Needs:

* Tools to manage user accounts, moderate content, and address technical issues.
* Security features to handle sensitive student data.
* Support for troubleshooting and resolving user inquiries.

Support user class. Essential for maintaining system operations, data security, and ensuring a smooth user experience for students and educators.

## **Operating Environment**

The ProjectPolyQuest web application runs on any valid internet browser, the application is only in English.

| ***Development Environment*** |
| --- |
| *Front-end: React, Mantine UI, TailwindCSS* |
| *Back-end: Firebase, PostgreSQL, Java Spring Boot, AWS* |

## **Design and Implementation Constraints**

I. The app uses PostgreSQL as a database

II. The app relies on Firebase for user login and registration. Hence, it is subject to

whether Firebase services are available. In cases where Firebase services as unreachable or

down, the app may not function as expected. The solution to this is to wait for the

Firebase services are to be available.

## **User Documentation**

A demonstration of how the app works along with a guide to install and run the app is provided in

the README.md file on the GitHub repository which contains the source code for this app (see

Section 1.5 References).

## **Assumptions and Dependencies**

I. The app relies on external datasets to get information such as the course information, career prospects and course intake.

II. It is assumed that the information obtained from these datasets is accurate

III. In cases where the API services become unavailable, the app may not meet some

requirements specified in this document.

IV. The User has to have a working internet connection to use the application.

V. The User has to use a valid internet browser to run the web application.

# **External Interface Requirements**

## **User Interfaces**

Please refer to the deliverables/lab5/ui\_mockups folder for User Interface Mockups.

## **Hardware Interfaces**

For PolyQuest, a web-based application, the primary hardware interface would be through standard internet-enabled devices such as desktops, laptops, tablets, and smartphones. The interface leverages web browsers, requiring no specialised hardware.

Key Characteristics:

Device Support: Any internet-enabled device capable of running modern web browsers (e.g., Chrome, Safari, Firefox).

Data and Control Interactions: The data exchanged includes HTTP requests and responses for application functionality, processed through backend APIs.

Communication Protocol: HTTP/HTTPS protocols ensure secure and reliable data transmission between client devices and the server.

As it’s a web-based application, the primary interface requirements centre around network connectivity and browser compatibility rather than direct hardware interaction.

## **Software Interfaces**

This application is built using a combination of front-end and back-end technologies that work together to provide a smooth and efficient user experience. Below is an overview of the different components, their roles, and the communication flows within the system.

#### 1. Front-End (React)

* Role: Provides user interface and handles user interactions.
* Communication: Sends HTTP requests to the back-end for data such as quiz submissions, user profiles, and recommendations.
* Key Functions: Registration, login, quiz interface, dashboards, bookmarks, and settings.

#### 2. Back-End (Spring Boot)

* Role: Manages business logic, data processing, and security.
* Communication: Exposes RESTful APIs for the front-end to send and receive data; communicates with the database via SQL queries.
* Key Functions: Handles authentication, processes quiz results, retrieves user data, and manages recommendations.

#### 3. Database (PostgreSQL)

* Role: Stores persistent data such as user profiles, quiz results, and course recommendations.
* Communication: Receives SQL commands from the back-end to store and retrieve data.
* Key Data: User accounts, quiz responses, bookmarks, and activity logs.

#### 4. Data Flow Summary

* Front-End to Back-End: Sends user data (e.g., login, quiz responses) via HTTP; back-end returns processed data (e.g., recommendations, user profile).
* Back-End to Database: Stores and retrieves data as needed, enforcing data integrity and security.

#### 5. Implementation Constraints

* Security: Uses token-based authentication (e.g., JWT) to secure user sessions.
* Data Sharing: Managed solely through the back-end to prevent unauthorized access.

## **Communications Interfaces**

For PolyQuest, the primary communication interface requirements are centered around client-server interactions over HTTP/HTTPS protocols, providing a secure and responsive web application experience.

Key Communication Requirements:

Protocols:

HTTP/HTTPS: Ensures secure communication between clients and the server for all requests and responses.

RESTful API: The backend is exposed via REST API endpoints, providing structured data interchange through JSON for front-end interactions.

Message Formatting:

JSON: Standard data format for all requests and responses between the frontend and backend.

Status Codes: HTTP status codes (e.g., 200 OK, 404 Not Found) provide standardised responses for request outcomes, aiding in error handling and client feedback.

Security and Encryption:

HTTPS: Encrypts all data in transit to prevent eavesdropping and unauthorised access.

Authentication: OAuth2.0 and other authentication protocols ensure that only authorised users access secure areas of the application.

Data Protection: Sensitive data, such as user credentials, are encrypted and handled securely.

Data Transfer Rates and Synchronization:

Asynchronous Requests: Ensures fast and responsive client-side experience by allowing multiple requests without blocking the user interface.

Real-time Updates (if applicable): WebSocket or similar protocols could be used to enable live updates for certain dynamic features, minimising delays.

Standards and Compliance:

PolyQuest’s communication architecture follows established web standards like REST and JSON API specifications for ease of integration and maintainability.

# Functional Requirements

### Account Registration

* 1. Users shall be able to register an account manually through their email.
     1. The system must validate emails during registration.
        1. The validation shall use a simplified non-exhaustive regular expression (regex) to ensure the email format is valid (e.g., [username@domain.com](mailto:username@domain.com)), belongs to an existing email domain.
           1. The system must enforce if there is 1 character before “@”
           2. The system must enforce if there is 1 character after “@”
           3. The system must enforce if there a singular “@”
           4. The system must enforce if the address ends with a valid top-level domain (TLD).
        2. The system must check if the email is already associated with an existing account.
        3. If the email is already associated with an existing account, the interface must indicate that the user has to choose another email because it has been taken.
        4. If the email is valid and not associated with an existing account, it must be stored in a database.
     2. Each account registered by the user must have a password.
        1. The system must enforce a minimum password complexity of at least 8 characters.
        2. The system must enforce a minimum password complexity of 1 upper-case letter.
        3. The system must enforce a minimum password complexity of 1 lower-case letter.
        4. The system must enforce a minimum password complexity of 1 symbol character.
        5. The system will display an error toast message if the password complexity criteria are not fulfilled. (refer to 1.1.2.1 to 1.1.2.4)
        6. The password created must be stored in a database.
  2. Users shall be able to register their accounts using social media authentication methods.
     1. The system must support social media authentication methods.
        1. The system must be able to support Google.
     2. The system shall retrieve the user’s email and basic profile information from the social media account.
        1. The system must request only the essential information from the user's social media profile.
           1. The system must request the user’s profile picture from their social media profile.
           2. The system must request the user’s full name from their social media profile.
           3. The system must request the user’s email from their social media profile.
  3. The user shall be able to delete their account.

### User Authentication

* 1. Users shall be able to log in using their email and password.
     1. The system must encrypt passwords using a secure hashing algorithm, before storing them.
        1. The system must enforce a minimum password complexity. (refer to 1.1.2.1)
        2. The system must implement rate limiting to prevent brute force attacks.
           1. The system must impose a rate limit at 5 attempts per 30 minutes.
     2. The system must allow users to log in using their social media accounts if they registered using this method. (refer to 1.2.1)
        1. The system must verify the user's social media credentials before granting access to the user.
        2. If the user provides an unregistered email address, a toast message must appear to prompt the user to register an account, and the user cannot proceed to log in.
        3. The system must synchronize any profile changes made in the social media account with the user's profile.
  2. Users shall be able to reset their passwords.
     1. The system must provide a "Forgot Password" option on the login page.
        1. The system must prompt users to enter their registered email to initiate the password reset process.
        2. The system must send a password reset link to the user’s registered email.
        3. The system must notify users if the entered email is not associated with any account.
     2. The system must allow users to set a new password that meets the defined complexity requirements. (refer to 1.1.2)
        1. The system must enforce minimum password length and character variety (refer to 1.1.2).
        2. The system must display a toast message if the password criteria are not fulfilled during the reset process. (refer to 1.1.2.5)
        3. The system must prevent users from reusing their existing password.
           1. If the user reuses their existing password, display a toast message that says “ You cannot change your password to your existing password ”.
  3. Users shall be able to log out of their accounts.
     1. The system must provide a logout function on every page once the user is logged in.
     2. The system must terminate the user session immediately after the "Logout" button is clicked.
     3. The system must redirect the user to the login page immediately after logout.
     4. The system must display a toast message "You have successfully logged out" upon successful logout.

### Academic Interest & Performance Input

* 1. Users shall be able to input their academic interests and performance.
     1. The system must allow users to enter their grades for key secondary academic subjects relevant to polytechnic courses.
        1. The system must support various subjects, including compulsory and elective subjects as mentioned below.
           1. Compulsory Subjects

English

Elementary Mathematics

* + - * 1. Elective Subjects

Principles of Accounts

Literature in Chinese/Malay/Tamil

History

Geography

Literature

Art

Music

Drama

Physical Education

Physics

Chemistry

Biology

Combined Science

Combined Humanities with Social Studies

Nutrition and Food Science

Electronics

Design and Technology

Computing

Additional Mathematics

Higher Chinese/Malay/Tamil

Mother Tongue

* + - 1. The system must validate the grades entered by the user.
         1. The system validates either through letter grades (A1 to F9).
      2. The system must allow users to update their grades.
    1. The system shall provide an interface for users to specify their interest in various academic subjects.
       1. The system must offer a list of predefined academic fields for users to choose from. (refer to 3.1.1.1)
       2. The system must allow users to select their interest in each academic field.
       3. The system must suggest related academic fields based on the user’s indicated interests in 3.1.2.

### Situational Test

* 1. The system shall present a situational test users can take to assess users' interests, strengths, and personality traits.
     1. The test must include Likert scale Situational questions.
        1. The test must have 48 questions covering different aspects of personality and interests.
        2. The Likert scale must have 5 ranking options:
           1. The first option shall display “Strongly Disagree”.
           2. The second option shall display “Disagree”.
           3. The third option shall display “Neutral”.
           4. The fourth option shall display “Agree”.
           5. The fifth option shall display “Strongly Agree”.
        3. The system must randomise the questions to reduce bias and ensure a unique experience for each user.
        4. The system must allow users to skip questions and return to them later.
        5. The system must display the submit button only after the user finishes all 48 questions.
     2. The system must offer an option for the user to pause and resume the test at a later time.
        1. The system must include a resume button to allow the user to continue from their saved progress.
        2. The system must save user progress automatically to prevent data loss in case of interruptions.
     3. The system must offer an option for the user to retake the test at any time, with each attempt stored separately.
        1. The system must track each test attempt and its corresponding results for the user.
     4. The system must provide a summary of the results immediately after test completion on a “Results” Page.
        1. The system will present a bar chart that indicates the most suitable fields.
           1. The bar chart consists of the following:

Manufacturing.

Business.

Technology.

Healthcare.

Education.

Engineering.

Creative Arts.

Public Service.

Sales and Marketing.

Construction.

Hospitality and Tourism.

Agriculture and Natural Resources.

* + - 1. The system will display the top 3 course recommendations.
         1. The system will display the top 3 most popular career paths for each recommendation
  1. The test results must be stored in the user’s profile for future reference.
     1. The system must store test results in a database.
        1. The system must display an option to view attempt history.
        2. The system must allow users to delete their test results.
        3. The system must timestamp each test attempt to allow users to view their attempt history.

### Test Result Analysis

* 1. The system shall analyse the results of the personality test and academic input.
     1. The analysis must identify key traits, strengths, and interests based on the test responses.
        1. The system must categorise traits and strengths into defined personality types or profiles.
        2. The system must present the analysis in a bar chart.
     2. The system must consider academic performance when analysing results to provide accurate recommendations.
        1. The system must compare academic performance with test results to identify aligned strengths.
     3. The analysis must be repeatable and consistent, providing similar results for similar inputs.
        1. The system must ensure the algorithm used for analysis is deterministic and based on consistent rules.
        2. The system must provide an option for users to give feedback on the accuracy of the analysis.

### Polytechnic & Diploma Recommendations

* 1. The system shall provide course recommendations based on test results and academic input.
     1. Recommendations must be tailored to align with the user’s identified interests, strengths, and inputted academic performance.
        1. The system must use a matching algorithm to align test results with relevant courses.
        2. The system must provide a ranking of recommended courses based on the compatibility of the match.
        3. The system must allow users to filter recommendations.
           1. The system must allow users to filter recommendations by the specific polytechnic.
     2. Each recommendation must include a detailed description.
        1. The recommendation must include the course name.
        2. The recommendation must include 3 examples of career prospects from the course.
        3. The recommendation must include the key skills required for the course.
        4. The system must obtain course descriptions from an up-to-date database of polytechnic programs.
        5. The system must highlight the key skills developed in each course.
        6. The system must provide an option for the user to save the recommended course for future perusal.
  2. The system shall offer alternative course recommendations if the user’s first choice does not match their strengths or interests.
     1. The system must suggest 3 alternative courses related to the user’s top choice.
     2. The system must allow users to explore alternative courses in-depth, with the same level of detail as the top recommendations. (refer to 6.1.2)

### Progress Tracking and Feedback

* 1. Users shall be able to track their progress through the application.
     1. The system must have a page that displays the user's past test results. (refer to 4.1.3)
     2. The system must have a page that displays course recommendations.
     3. The system must have a page that displays the user’s saved courses.
     4. The system must have a page that has a feedback option. (refer to 7.3)
     5. The system must update in real-time as users complete the test or save courses.
  2. The system must allow users to revisit and revise their test results and course recommendations.
     1. The system must store historical test results and recommendations for comparison.
        1. The system must timestamp each set of test results and recommendations.
        2. The system must allow users to delete old test results if desired.
  3. Users shall be allowed to provide feedback on the application
     1. The feedback will include:
        1. Likert Scale for each category
           1. User Experience
           2. Test Quality
           3. Recommendation Accuracy
        2. Comment Section
     2. The feedback will be sent from the email associated with the registered account.

# **Nonfunctional Requirements**

### Security & Privacy

* 1. The system shall ensure the security and privacy of user data.
     1. The system must encrypt all personal particulars and passwords stored in the database.
        1. The system must use industry-standard encryption methods for data at rest.
        2. The system must ensure data in transit is encrypted using Transport Layer Security (TLS).

### Usability

* 1. The system shall be easy to use for users with different levels of technical proficiency.
     1. The user interface must be intuitive and user-friendly, with clear navigation.
        1. The system must use consistent design patterns throughout the application.
        2. The system must provide tooltips or help icons to explain complex features.
        3. The system must ensure that all interactive elements are easily accessible.
        4. The system must use simple language and provide examples where necessary to clarify questions.
        5. The system must offer an option to review instructions before starting the test.

### Performance

* 1. The system shall be responsive and performant under normal usage conditions.
     1. The system must load pages within 3 seconds under normal network conditions.
        1. The system must optimize images and media files to reduce page load times.
        2. The system must use caching strategies to improve load times for repeat visitors.
        3. The system must minimize the use of third-party scripts that could slow down page loading.
        4. The system must provide a loading indicator if page content takes longer than 2 seconds to load.
     2. The system shall handle up to 5 concurrent users without significant degradation in performance.
        1. The system must use a content delivery network (CDN) to improve performance for users in different regions.
     3. The system must optimize database queries to ensure quick data retrieval.
        1. The system must use indexed queries for frequently accessed data.
        2. The system must minimize the number of database calls by using efficient query structures.
        3. The system must implement database caching for read-heavy operations.

### Scalability

* 1. The system shall be scalable to accommodate a growing number of users.
     1. The architecture must support horizontal scaling to handle increased traffic.
        1. The system must separate application components to scale them independently.
     2. The system must be able to integrate with cloud services to dynamically scale based on demand.
        1. The system must optimize storage and compute resources based on real-time demand analysis.
        2. The system must use cloud-native databases to scale data storage and retrieval.
     3. The system must support global distribution to serve users from different geographic regions.
        1. The system must use a global CDN to deliver static assets efficiently.

## **Software Quality Attributes**

The following quality attributes are essential to ensure that the application meets the expectations of both the users and developers.

#### 1. Usability

* Goal: The application must be intuitive and easy to navigate, particularly for non-tertiary students with limited technical skills.
* Measurement: Users should be able to complete the primary tasks (e.g., taking a quiz, viewing recommendations) within three clicks, with task completion success rates above 95% in usability tests.

#### 2. Reliability

* Goal: The application must operate smoothly with minimal downtime, ensuring a dependable experience.
* Measurement: Target uptime of 99.9% and error rates below 0.1% per transaction. Regular testing and monitoring should verify this.

#### 3. Performance

* Goal: Fast load times and responsiveness are crucial, especially on the quiz interface and dashboard.
* Measurement: Pages should load within 2 seconds on average for a typical user profile, and API responses should be under 500ms.

#### 4. Maintainability

* Goal: The codebase must be easy to update and modify for future enhancements.
* Measurement: Code should follow standard conventions, be modular, and maintain a high level of documentation. Target code readability scores of at least 80% using static analysis tools.

#### 5. Security

* Goal: User data, especially personal and quiz information, must be secure.
* Measurement: Implement token-based authentication (e.g., JWT) and ensure data encryption at rest and in transit. Conduct quarterly security audits.

#### 6. Interoperability

* Goal: The system should integrate smoothly with other educational or reporting systems if needed in the future.
* Measurement: Ensure RESTful APIs follow standard protocols and are well-documented for future integrations.

#### 7. Scalability

* Goal: The application should handle increased load as more students use the platform.
* Measurement: System should scale to support a 200% increase in user traffic with minimal performance degradation.

## **Business Rules**

The following business rules govern how different user roles interact with the application, defining permissions, restrictions, and specific operational guidelines.

#### 1. User Registration and Access

* Only non-tertiary students are allowed to create accounts and access the quiz functionality.

#### 2. Quiz Participation and Recommendations

* Users must complete the quiz fully before receiving any course recommendations.
* Course recommendations are based on quiz results, academic performance, and stated interests, ensuring they align with each user's profile.

#### 3. User Roles and Permissions

* Students: Can access personal profiles, take quizzes, view recommendations, bookmark courses, and adjust settings. Students cannot access other users' information or view administrative data.
* Administrators: Have full access to all data and system settings. They can manage users, moderate content, troubleshoot issues, and perform system maintenance.

#### 4. Data Privacy and Security

* All user data is confidential and can only be accessed by the user or administrators under specific conditions (e.g., technical support or compliance requests).
* Personal data must be encrypted in storage and transmission, adhering to data protection

#### 6. Account Management

* Users can request account deletion, which will permanently remove their data from the system.
* Administrators have the right to suspend or terminate accounts if there is evidence of misuse, such as impersonation or data abuse.

#### 7. Activity Logging and Monitoring

* User interactions, including quiz attempts, profile updates, and course bookmarks, are logged to provide personalized recommendations and for audit purposes.
* Access logs are maintained for security and compliance, ensuring that only authorized access is permitted.

# **Other Requirements**

This section defines additional requirements for the application that are not covered in previous sections, including database, internationalization, legal, and other project-specific considerations.

#### 1. Database Requirements

* Data Integrity: The database must enforce data integrity constraints, such as unique keys for user accounts and foreign key relationships between tables (e.g., user and quiz results).
* Data Backup: Regular automated backups are required to prevent data loss, with daily incremental backups and weekly full backups.
* Data Retention: User data should be retained for a minimum of five years, with the option for users to request data deletion if they no longer wish to use the platform.

#### 2. Internationalization (I18N) Requirements

* Language Support: The application must support English as the primary language. If the product is expanded for use outside Singapore, additional language support (e.g., Mandarin, Malay, Tamil) may be considered.

#### 3. Legal and Compliance Requirements

* Data Protection Compliance: The application must comply with Singapore’s Personal Data Protection Act (PDPA) to safeguard user data and privacy.
* User Consent: Users must provide consent for data collection, particularly for personal and quiz data, through a clear terms-of-use agreement upon registration.

#### 4. Performance and Scalability Requirements

* Scalability: The system should be designed to support up to 10,000 concurrent users without significant performance degradation.
* Load Testing: Regular load testing is required to ensure the application can handle peak usage times, particularly during school counselling sessions or examination periods.

#### 5. Reuse Objectives

* Component Reuse: Code components, particularly UI elements and API services, should be designed for reuse to facilitate future development and maintenance.
* Modular Architecture: The application should adopt a modular architecture, allowing easy integration of new features (e.g., additional quizzes or counseling tools) without extensive refactoring.

#### 7. Future Integration

* External APIs: The application should be designed to allow future integration with external APIs (e.g., government education portals or career guidance tools) to provide enhanced functionality for users.
* Data Export: Allow for the export of user data in standardized formats (e.g., CSV, JSON) for reporting and analysis by authorized users (e.g., counselors or educators).

# 7. Use Case Descriptions

## Account Registration

### Register Account

| Use Case ID: | #1-1 | | |
| --- | --- | --- | --- |
| Use Case Name: | RegisterAccount | | |
| Created By: | Lam Yan Yee | Last Updated By: | Lam Yan Yee |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user registers an account using an email / social media account |
| Preconditions: | The user must already have a valid email / social media account |
| Postconditions: | There will be a toast message indicating account registration was a success  The user will also receive a confirmation email to check the validity of the email |
| Priority: | High |
| Frequency of Use: | High |
| Flow of Events: | 1. The user clicks on “Sign Up” on the initial landing page 2. The user enters their email 3. The user enters their desired password that meets the minimum security requirement 4. The user successfully creates an account |
| Alternative Flows: | The User opts to sign up using their Social Media Credentials instead:   1. The user clicks on Google/Facebook 2. The user links their Google/Facebook account 3. (continue to step 3 in the initial flow of events) |
| Exceptions: | * If the user provides an invalid email address, a toast message will appear and the user cannot proceed to register an account. * If the user’s password does not meet the minimum security requirement, a toast message will appear and the user cannot proceed to register an account * If the user’s Google/Facebook account is invalid, the Google/Facebook third-party API will prompt the user that the account is invalid and they cannot proceed to register an account |
| Includes: | * Login |
| Special Requirements: | The email address has to be valid and the password has to hit the minimum security criteria before the user can create an account |
| Assumptions: | The user has a pre-existing email  OR  The user has a pre-existing Google account  OR  The user has a pre-existing Facebook account |
| Notes and Issues: | None |

## Login/Logout

### Login

| Use Case ID: | #2-1 | | |
| --- | --- | --- | --- |
| Use Case Name: | Login | | |
| Created By: | Lam Yan Yee | Last Updated By: | Lam Yan Yee |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user logs in using an email / social media account |
| Preconditions: | The user has to have already created an account beforehand.  The user is not logged in to the web application |
| Postconditions: | The user will log into the web application successfully and be redirected to the web application’s home page |
| Priority: | High |
| Frequency of Use: | High |
| Flow of Events: | 1. The user arrives on the initial “Login to Your Account” landing page 2. The user keys in their email 3. The user keys in their password 4. The user arrives at the main home page of the web application |
| Alternative Flows: | 1. The User can click on the “Continue with Google” button and type in their Google Account details   OR   1. The User can click on the “Continue with Facebook” button and type in their Facebook Account details 2. The User arrives at the main home page of the web application |
| Exceptions: | * In step 2, If the user provides an unregistered email address, a toast message will prompt the user to register an account and the user cannot proceed to log in * If the user’s Google/Facebook account is unregistered, a new account will be registered and the user will be able to proceed and arrive at the home page of the web application. |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | The user has an existing account stored in the system’s database |
| Notes and Issues: | None |

### Logout

| Use Case ID: | #2-2 | | |
| --- | --- | --- | --- |
| Use Case Name: | Logout | | |
| Created By: | Lam Yan Yee | Last Updated By: | Lam Yan Yee |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user logs out of the web application |
| Preconditions: | The user is logged into the web application |
| Postconditions: | The user will successfully log out of the web application, and be redirected to the initial “Login to Your Account” landing page |
| Priority: | Medium |
| Frequency of Use: | Medium |
| Flow of Events: | 1. The user clicks on “Logout” at the bottom left of the NavBar 2. The user is logged out of the web application and arrives at the Initial “Login to Your Account” landing page |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | None |
| Notes and Issues: | None |

## Contact Us

### Contact Us

| Use Case ID: | #3-1 | | |
| --- | --- | --- | --- |
| Use Case Name: | ContactUs | | |
| Created By: | Dallas Ng Zhi Hao | Last Updated By: | Dallas Ng Zhi Hao |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user can use this feature to send feedback to the web application developers and also find the developers’ email address |
| Preconditions: | The user is logged into the web application. |
| Postconditions: | The user will be directed to a “Thank you for your feedback” page. |
| Priority: | Medium |
| Frequency of Use: | Medium |
| Flow of Events: | 1. The user clicks on “Contact Us” at the bottom left of the screen 2. The system provides the admin’s email address 3. The user is able to email the admin regarding any inconveniences |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | None |
| Notes and Issues: | None |

### Providing General Feedback

| Use Case ID: | #3-2 | | |
| --- | --- | --- | --- |
| Use Case Name: | Feedback | | |
| Created By: | Jerwin Lee Chu Hao | Last Updated By: | Jerwin Lee Chu Hao |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user can use this feature to send feedback to the web application developers and also find the developers’ email address |
| Preconditions: | The user is logged into the web application. |
| Postconditions: | The user will be directed to a “Thank you for your feedback” page. |
| Priority: | Low |
| Frequency of Use: | Low |
| Flow of Events: | 1. The user clicks on “Contact Us” at the bottom left of the screen 2. The user will be redirected to the feedback page 3. The user answers “User Experience” related feedback questions 4. The user answers “Test Quality” related feedback questions 5. The user answers “Recommendation Accuracy” related feedback questions 6. The user keys in any comments at the end 7. Feedback will be sent out successfully |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | The user has feedback to provide |
| Notes and Issues: | None |

## Activity

### Activity

| Use Case ID: | #4-1 | | |
| --- | --- | --- | --- |
| Use Case Name: | Activity | | |
| Created By: | Dallas Ng Zhi Hao | Last Updated By: | Dallas Ng Zhi Hao |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user can access their past activity via this feature |
| Preconditions: | The user is logged into the web application |
| Postconditions: | None |
| Priority: | Medium |
| Frequency of Use: | Medium |
| Flow of Events: | 1. The user clicks on “Activity” on the NavBar 2. The user is directed to the Activity page 3. The user can see their past test results/delete their past test results on this page |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | None |
| Notes and Issues: | None |

### View Past Test Results

| Use Case ID: | #4-2 | | |
| --- | --- | --- | --- |
| Use Case Name: | ViewPastResults | | |
| Created By: | Jerwin Lee Chu Hao | Last Updated By: | Jerwin Lee Chu Hao |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user can view their past test results using this feature |
| Preconditions: | The user must be logged in to the web application. |
| Postconditions: | None |
| Priority: | Medium |
| Frequency of Use: | Medium |
| Flow of Events: | 1. The user clicks on “Activity” on the nav bar on the left-hand side of the screen 2. The user will be redirected to the Activity page 3. The user can view their past test results on that page |
| Alternative Flows: | None |
| Exceptions: | If the user has not done a test before, the page will be empty |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | None |
| Notes and Issues: | None |

### Delete Past Test Results

| Use Case ID: | #4-3 | | |
| --- | --- | --- | --- |
| Use Case Name: | DeletePastResults | | |
| Created By: | Jerwin Lee Chu Hao | Last Updated By: | Jerwin Lee Chu Hao |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user can delete their past test results using this feature |
| Preconditions: | The user must be logged in to the web application |
| Postconditions: | The result that the user have deleted will not be present in the future |
| Priority: | Low |
| Frequency of Use: | Medium |
| Flow of Events: | 1. The user clicks on “Activity” on the NavBar 2. The user clicks on the trashcan icon on the top right of the past test result 3. A toast message “Are you sure?” will appear, 4. The user clicks on “Delete Quiz” 5. The selected past test result will be deleted 6. The user will be redirected to the “Activity” page |
| Alternative Flows: | In step 3, if the user decides they do not want to delete their account:   1. Continuing from step 3, The user will click on the “X” on the top right of the toast message 2. The user will be redirected to the “Activity” page without any changes to past test results |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | The user must have taken a test before to access this feature. |
| Assumptions: | The user has taken a test before. |
| Notes and Issues: | None |

## Taketest

### Undertaking of the Test

| Use Case ID: | #5-1 | | |
| --- | --- | --- | --- |
| Use Case Name: | TakeTest | | |
| Created By: | Malcolm Fong Cheng Hong | Last Updated By: | Malcolm Fong Cheng Hong |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user uses the Situational Test which is the main feature of our web application. |
| Preconditions: | The user must be logged in to the web application. |
| Postconditions: | The user will be shown a list of Likert Scale questions to answer.  Upon Completion, they will be redirected to the results page. |
| Priority: | High |
| Frequency of Use: | High |
| Flow of Events: | 1. The user clicks on “Take a Test” on the left-hand side of the screen, 2. The user will be redirected to the test page where they can answer the Likert Scale questions 3. The user will be redirected to the results page, which shows them their recommended courses and the industry sectors they are likely suited for |
| Alternative Flows: | If the user is taking the test for the first time, they will be able to take the test directly from the dashboard that they are directed to upon logging in for the first time:   1. The user logs in for the first time 2. The user is directed to the Dashboard page 3. The user clicks on “Let’s Start” 4. The user needs to input their Academic Results 5. The user needs to choose their Academic Interest 6. Then the user will be able to take the test |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | If the user is taking the test for the first time:  They need to fill in their academic performance + academic interests. |
| Assumptions: | None |
| Notes and Issues: | The user will be notified that they can change their Academic Results and Academic Interests in the settings. |

## Dashboard

### Dashboard

| Use Case ID: | #6-1 | | |
| --- | --- | --- | --- |
| Use Case Name: | Dashboard | | |
| Created By: | Jerwin Lee Chu Hao | Last Updated By: | Jerwin Lee Chu Hao |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | Dashboard |
| Preconditions: | The user must be logged in to the web application |
| Postconditions: | None |
| Priority: | High |
| Frequency of Use: | Medium |
| Flow of Events: | The user can select the following few options:   1. Recommended courses will be shown once the user has entered the “Dashboard” page 2. If the user clicks on the filter icon, the user will then choose which polytechnic to filter out what course that specific polytechnic offers 3. The user can also bookmark the recommended courses that they like by pressing the bookmark icon on the top right of the recommended course |
| Alternative Flows: | 1. By Default, The first time the user logs into the web application, they will arrive at the dashboard 2. Also, They will be able to Take a Test from the Dashboard as a First-time user |
| Exceptions: | If the user has not taken any test, no recommended course will be shown on the “Dashboard” |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | The user has already taken a test |
| Notes and Issues: | None |

### Filter Course

| Use Case ID: | #6-2 | | |
| --- | --- | --- | --- |
| Use Case Name: | FilterCourse | | |
| Created By: | Jerwin Lee Chu Hao | Last Updated By: | Jerwin Lee Chu Hao |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | FIlterCourse |
| Preconditions: | The user must be logged in to the web application |
| Postconditions: | The system will filter accordingly to the polytechnic of the user’s choice |
| Priority: | High |
| Frequency of Use: | Medium |
| Flow of Events: | 1. The user clicks on the filter icon, the user will then choose which polytechnic to filter out what course that specific polytechnic offers 2. The user will be able to see the recommended courses according to the filter |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | The user must have taken a test before accessing this feature |
| Assumptions: | The user wants to filter the courses by polytechnics |
| Notes and Issues: | None |

### Bookmark Courses

| Use Case ID: | #6-3 | | |
| --- | --- | --- | --- |
| Use Case Name: | BookmarkCourses | | |
| Created By: | Jerwin Lee Chu Hao | Last Updated By: | Jerwin Lee Chu Hao |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user can bookmark courses that they are interested in |
| Preconditions: | The user must be logged in to the web application |
| Postconditions: | None |
| Priority: | High |
| Frequency of Use: | Medium |
| Flow of Events: | 1. The user clicks on “Dashboard” on the NavBar 2. Once on the “DashBoard’ page, the user can see the recommended course that the system has suggested 3. Then, the user can bookmark a course by pressing the bookmark icon on the top right of the recommended course |
| Alternative Flows: | None |
| Exceptions: | If the user does not do any test, no recommended course will appear and the user will not be able to bookmark it. |
| Includes: | None |
| Special Requirements: | The user must have taken a test before, to access this feature. |
| Assumptions: | None |
| Notes and Issues: | None |

## Bookmarks

| Use Case ID: | #7-1 | | |
| --- | --- | --- | --- |
| Use Case Name: | Bookmarks | | |
| Created By: | Dallas Ng Zhi Hao | Last Updated By: | Dallas Ng Zhi Hao |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user’s saved recommended courses will appear here |
| Preconditions: | The user is logged into the web application |
| Postconditions: | None |
| Priority: | High |
| Frequency of Use: | Medium |
| Flow of Events: | 1. The user clicks on “Bookmarks” on the nav bar 2. The user is directed to the Bookmarks page 3. The user can access the courses that they have bookmarked |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | None |
| Notes and Issues: | None |

## Settings

### Settings

| Use Case ID: | #8-1 | | |
| --- | --- | --- | --- |
| Use Case Name: | Settings | | |
| Created By: | Yeo, Ruizhi Carwyn | Last Updated By: | Yeo, Ruizhi Carwyn |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user can either delete their account, change their particulars (Profile Pic & Password), or update their Academic Results and Interests. |
| Preconditions: | The user has to be logged into the web application. |
| Postconditions: | The user can choose to delete their account, change personal particulars, or change Academic Results and Interest. |
| Priority: | High |
| Frequency of Use: | Low |
| Flow of Events: | 1. The user clicks on “Settings” on the sidebar on the right-hand side of the screen 2. If the user clicks on “ Deletion of Account”, the user can delete their account 3. If the user clicks on “Personal Info”, the user can change their password or profile picture 4. If the user clicks on “Academic Results and Interest”, the user can edit their Academic result or Interest |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | None |
| Notes and Issues: | None |

### Deletion of Account

| Use Case ID: | #8-2 | | |
| --- | --- | --- | --- |
| Use Case Name: | AccountDeletion | | |
| Created By: | Yeo, Ruizhi Carwyn | Last Updated By: | Yeo, Ruizhi Carwyn |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user will be able to delete their account via this option. |
| Preconditions: | The user has to be logged in to the web application. |
| Postconditions: | The user’s account information will be erased and returned to the initial log-in page. |
| Priority: | Medium |
| Frequency of Use: | Low |
| Flow of Events: | 1. The user clicks on “Settings” 2. The user clicks “Delete Account” 3. A toast message will ask the user “Are you sure?” 4. The user clicks “Delete Account” 5. The user’s information is erased from the system 6. The system returns the user to the initial starting log-in page |
| Alternative Flows: | In step 4, if the user does not want to delete their account:   1. The user will click on the “X” on the top right of the toast message 2. The system will redirect the user back to the settings page without the user account being deleted |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | None |
| Notes and Issues: | None |

### Change Password

| Use Case ID: | #8-3 | | |
| --- | --- | --- | --- |
| Use Case Name: | ChangePassword | | |
| Created By: | Yeo, Ruizhi Carwyn | Last Updated By: | Yeo, Ruizhi Carwyn |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user can change their password using this feature. |
| Preconditions: | The user has to be logged in to the web application. |
| Postconditions: | The user’s password will be updated to the new password the user has just set. |
| Priority: | High |
| Frequency of Use: | Low |
| Flow of Events: | 1. The user clicks on “Settings” 2. The user clicks on “Personal info” 3. The user clicks on ‘Change password” 4. The user inputs a valid new password 5. A toast message will appear “Password has been changed successfully” 6. The user will be redirected to the “Settings” page |
| Alternative Flows: | None |
| Exceptions: | If the user types in a password that does not meet the password requirements, it will prompt the user to enter another password |
| Includes: | None |
| Special Requirements: | The password keyed in has to meet the minimum security requirement of 8 characters, consisting of 1 uppercase letter, 1 lowercase letter, and 1 symbol |
| Assumptions: | None |
| Notes and Issues: | None |

### 

### Input of Academic Performances (Subjects & Grades)

| Use Case ID: | #8-4 | | |
| --- | --- | --- | --- |
| Use Case Name: | ChangeAcademicResults | | |
| Created By: | Malcolm Fong Cheng Hong | Last Updated By: | Malcolm Fong Cheng Hong |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user will input their new academic grades into the system |
| Preconditions: | The user must be logged into the web application. |
| Postconditions: | None |
| Priority: | High |
| Frequency of Use: | Low |
| Flow of Events: | 1. The user clicks on “Change Academic Results and Interest” 2. The user will then update respective grades for the subjects that the user has scored differently. |
| Alternative Flows: | None |
| Exceptions: | At step 2, If the user keys in a grade that is not valid i.e. (Not A1 - F9), then there will be a toast error message prompting the user to enter a letter grade (A1 - F9). |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | The user is a student and going to take ‘O’ levels. |
| Notes and Issues: | None |

### Input of Academic Interests

| Use Case ID: | #8-5 | | |
| --- | --- | --- | --- |
| Use Case Name: | AcademicInterests | | |
| Created By: | Malcolm Fong Cheng Hong | Last Updated By: | Malcolm Fong Cheng Hong |
| Date Created: | 9th September 2024 | Date Last Updated: | 9th September 2024 |

| Actor: | User |
| --- | --- |
| Description: | The user will input their academic interests into the system. |
| Preconditions: | The user must be logged in to the web application. |
| Postconditions: | The system will save their academic interests in the database. |
| Priority: | High |
| Frequency of Use: | Low |
| Flow of Events: | 1. The user will choose their academic interests by clicking on the academic subjects they resonate strongly with |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | None |
| Special Requirements: | None |
| Assumptions: | None |
| Notes and Issues: | None |

# Appendix A: Data Dictionary

| Term | Definition |
| --- | --- |
| User | An individual who interacts with the system, creating an account, logging in, and using the features provided. |
| Account | A digital identity created by a user to access the system's services. It includes information such as email, password, and user profile. |
| Dashboard | A user interface component where users can view their test results, course recommendations, and other personal data. |
| Course | A structured set of educational lessons or lectures is provided to users, which may include exams, assignments, and interactive content. |
| Test | An assessment tool used to evaluate a user’s knowledge, skills, interests, or personality traits. tests can have various question formats. |
| Academic Performance | The measurement of a user's grades, skills, and interests in various academic fields, which can be used to recommend courses or further educational paths. |
| Recommendation | A suggestion provided to the user, often based on test results and academic performance, aimed at guiding their educational or career choices. |
| Compulsory Subjects | A subject that a student is required to take and pass to graduate |
| Elective Subjects | A subject that a student takes because you want to rather than to fill a particular requirement, although you still get credit for it. |

# Appendix B: Analysis Models

<https://github.com/softwarelab3/2006-TDDA-53>

*< Please Refer to all the diagrams attached in Lab5, they are all the finalised versions of the Analysis Models used for our Project>  
These Diagrams include:*

*Class Diagram, Key Boundary Classes & Control Classes, Dialog Map, Sequence Diagrams, Architecture Diagrams and Use Case Diagram.*

# 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.>*

*(TBD)*

Source: http://www.frontiernet.net/~kwiegers/process\_assets/srs\_template.doc