UTM UNIVERSITI TEKNOLOGI MALAYSIA	**Software Engineering Dept** **Faculty of Computing** **Universiti Teknologi Malaysia**

SDD Template

Software Design Description (SDD)

TBC TVPSS Program Management Information System

Prepared by: Milk Tea

Document No.	**:**	**<<Document ID>>**
WPC Ref. No.	**:**	**<<Work Package Contract Ref. No.>>**
Revision	**:**	**1.0**
Date Issued	**:**	**17.01.2025**

Prepared by		**Reviewed by**		**Approved by**	
1.  2.  3.  4.  5. 				<< signature >>	
Name(s)	Wajeeha Zeeshan Ahmed Zaki Ahmed Mohammed Al-Gabaly Azmain Mahpara Iqbal Tabinda Zeeshan Zuhayer Adnan Siddique	**Name**	Wajeeha Zeeshan	**Name**	Prof. Ts. Dr. Wan Mohd Nasir b. Wan Kadir
Position	Team Members	**Position**	Team Leader	**Position**	Lecturer

1. Frontispiece

a. Overview

This SDD will provide an overview about the TBC TVPSS Management Information System. The purpose, scope, design and implementation, everything is highlighted in this document. The user interfaces are also provided.

b. Issuing Organization / Team

Issuing Organization: Faculty of Computing, UTM

Team: SECJ3323 Project Team - Milk Tea

c. Authorship / Project Team Members

Names	Roles	Status
Wajeeha Zeeshan	Team Leader	Complete
Ahmed Zaki Ahmed Mohammed Al-Gabaly	Team Member	Complete
Azmain Mahpara Iqbal	Team Member	Complete
Tabinda Zeeshan	Team Member	Complete
Zuhayer Adnan Siddique	Team Member	Complete

d. Revision History

REVISION NO.	ISSUE DATE	DETAILS OF REVISION
01	17/01/2025	Initial submission

Note:
This System Documentation (SD) template is adapted from IEEE Recommended Practice for Software Requirements Specification (SRS) (IEEE Std. 830-1998), Software Design Descriptions (SDD) (IEEE Std. 10161998 1), and Software Test Documentation (IEEE Std. 829-2008) that are simplified and customized to meet the need of SCSJ2203 course at Faculty of Computing, UTM. Examples of models are from Arlow and Neustadt (2002) and other sources stated accordingly.

Table of Contents

2. Introduction

The following sections of the System Design Description (SDD) will be providing the details of the entire system design.

2.1 Purpose

This SD describes an overview of the architecture and design for the TBC TVPSS Program Management Information system. This document will be outlining all the interactions between the components of the system and will help in assuring that all the design decisions made are aligning with requirements of the stakeholders and answering their concerns.
The intended audience for this SD are the project managers, system architects, schools, PSS management team and all the other stakeholders who will be involved in the implementation and management of this system. The main purpose of the document is to ensure a clear understanding and communication between everyone involved.

2.2 Scope

The software product to be produced is the TBC TVPSS management information system, which is a web application that will be used to enhance the management of all the TVPSS programs organized for schools across Johor.

Features included:

- Centralized database that will manage all the necessary details including the video links, crew applications and program status.
- A Dashboard that will be used to monitor programs' progress and approvals.
- An automated process for requests that will facilitate the requests for upgrades.
- Search and filter to make it easy to filter out schools and programs
- A content management system that will facilitate the management of the creative content shared by the students

Capabilities:

- Data management and its retrieval will be more effective.
- Automated processes and dashboards will facilitate decision making.
- Data sharing will be integrated hence collaboration can be promoted.

Limitations:

- Management regarding the broadcasting of TVPSS content will be limited.
- Creative tools for content creation and editing will not be provided.

Application and Benefits

Manual workflows will be replaced by an automated system and platform that will provide

- Improvement and facilitation in data organization. .
- **I**mprovement in resource management
- More user-friendly interface
- Scalability according to the growth.

2.3 **Context**

This document is assembled to be used by several stakeholders to ensure that the system is effectively developed and implemented. This is also intended to be used in conjunction with the maintenance of the system. The targeted audience includes the JPN staff, PSS management team, school administrators and technical teams. This documentation aims to be a guide in understanding the system's design, functionality and workflows.

For the designers, this document will help in the selection, organisation and presentation of the design information. It is to ensure that the design process is aligned with the stakeholder requirements and follows the standards that have been established. This is to incorporate the best practices for the system's development process. Designers can produce clear, concise and consistent design artifacts to conduct a smooth development and implementation process of the system.

This document is designed to be practiced by all technical stakeholders and the managers involved in the TVPSS program. A clear framework is provided for design decisions while ensuring consistency and ease of communication. This will help all the teams to effectively understand and maintain the system.

2.4 **Summary**

This document provides a detailed description of the TVPSS Program Management information system's architecture and design. It identifies the stakeholders and ensures the system is designed in a way to meet the requirements.

The document also includes design viewpoints as mentioned in below:

- The context viewpoint highlights the system's interactions and showcases the use case diagram.
- The logical viewpoint uses the class diagram to explain the system's structure.
- The information viewpoint describes the design of the database with the ERD diagram.
- The interface viewpoint shows the wireframes for user interactions.
- The structure viewpoint highlights the behavioral and structural patterns.

- The interaction viewpoint focuses on dynamic workflows with sequence diagrams.
- The algorithm viewpoint uses decision tables to simplify system logic with the activity diagrams.

Concluding the document, the user interface design and appendices provide additional resources to ensure that the document is comprehensive and helps in development, implementation and maintenance of the system.

3. References

The following references were used:

- **GeeksforGeeks.** (2024, February 9). *Use case diagrams - Unified Modeling Language (UML).* GeeksforGeeks. Retrieved from https://www.geeksforgeeks.org/use-case-diagram/

- **Nicholas, J.** (2024, May 19). *Use case specification guideline - Best tips & guidance for 2024.* BusinessAnalystMentor.com. Retrieved from https://businessanalystmentor.com/use-case-specification-guidelines/

- **Kementerian Pendidikan Malaysia.** (2020, September 11). *KIT TVPSS MALAYSIA.* Ministry of Education, Malaysia. Retrieved from https://online.fliphtml5.com/nkvup/dfvt/#p=1

4. Glossary

Definitions of all terms, acronyms and abbreviations used are defined here.

- **TVPSS** - TV Pusat Sumber Sekolah
- **PSS** - Pusat Sumber Sekolah
- **MIS** - Management Information System
- **JPN** - Jabatan Pendidikan Negeri (State Education Department)
- **SRS** - Software Requirement Specification
- **DBMS** - Database Management System
- **HTTP** - Hypertext Transfer Protocol
- **HTTPS** - Hypertext Transfer Protocol Secure
- **SMTP** - Simple Mail Transfer Protocol
- **IP** - Internal Protocol

5. Design Body

This section of SDD contains the identified design stakeholders involved in the creation of TBC TVPSS Management information system , the design concerns and design viewpoints.

5.1 Design Stakeholders and Their Concerns

- PSS Management Team

  They are concerned with the access of data that should be easy in order to monitor the performances of schools and programs. The resource allocation process should also be easy, timely and exact. Lastly, they are also concerned with the functionality of the dashboard that will allow them to track for status and manage approvals.

- Project Managers

  They are concerned with completing the system and its functionality within the allocated time and budget and being able to fulfill all the requirements needed from this system.

- System Architects

  They are concerned with the scalability of the system along with the appropriate technologies that are to be used and cohesion with the standards of the industry.

- Students and teachers

  Their concerns are regarding the access of the system and the response it provides regarding the features that are specific to their own roles.

- School administrators

  Their concern is regarding the user friendliness of the system and the ease with which they will be able to access and filter out data that is relevant to their schools.

- JPN staff

  They are concerned with overviewing the program performances along with their statuses. They also have another concern that is regarding centralized data storage.

- Development team

  Their concern is regarding the feasibility of the designs proposed and the tools available to implement them along with the required technologies.

5.2 Context Viewpoint

The Context viewpoint depicts services provided by this design subject and is defined by reference to actors that include users and other stakeholders, which interact with the design subject in its environment.

5.2.1 **Design Concerns**

The purpose of the Context viewpoint is to identify a design subject's offered services, its actors (users and other interacting stakeholders), to establish the system boundary and to effectively delineate the design subject's scope of use and operation.

Primary Purpose:

- Identification of the services and functionality offered by the system
- Identification of the actors interacting with the system
- Establishment of boundaries for the respective system
- Detail regarding the operational context of the system

System features:

- Students can submit videos and apply for becoming members in the TVPSS crew
- Teachers can manage the dashboard or their schools, add and update the respective program details and request the version upgrades for their respective schools
- Pss Management team can view reports for respective programs, generate certificates, send notifications to participants, approve requests and upgrades and track statuses for all the programs and schools.

5.2.2 **Design View**

Figure 2.1 shows the Use Case Diagram (UCD) for TBC TVPSS Program MIS System. There are a total of three main actors involved in the diagram who are PSS management team, teacher and the student. This use case diagram comes along with 15 use cases in which each use case will specify the process involved when interacting with the system. The use cases are further structured into four main modules which are TVPSS Program and Crew Management Module, Reporting Module, Resource and Equipment Management Module and Monitoring School Status Module.

For the TVPSS Program and Crew Management Module, the student can submit TVPSS Program Video and apply to become TVPSS crew in their school through UC003 and UC005 respectively. Moreover, the teacher can manage their school program dashboard in UC009. Both UC010 and UC011 will be included respectively to allow the teacher adding and updating the program details.

Besides, for the Reporting Module, the PSS Management Team can view the program report that contains the program details in UC006 and generate e-certification for the participants through UC007. Hence, UC015 which is sending email and notification to the participants must be included in UC007.

On the other hand, the Resource and Equipment Management Module will have two use cases in which teachers are allowed to request TVPSS Version Upgrade through UC008 while the PSS management team will review the request and allocate the resources depending on the situation in UC004.

For the Monitoring School Status Module, the PSS management team can track the school progress such as program status in UC001 as they need to keep updated with the latest TVPSS program status in each school. Then, UC014 will be extended from UC001 to monitor user activity such as their engagement level with the system. Besides, teachers will request a version upgrade through UC002. Hence, the PSS management team can approve the school version upgrade in UC013 and provide feedback to the school when reviewing the version upgrade request in UC012.

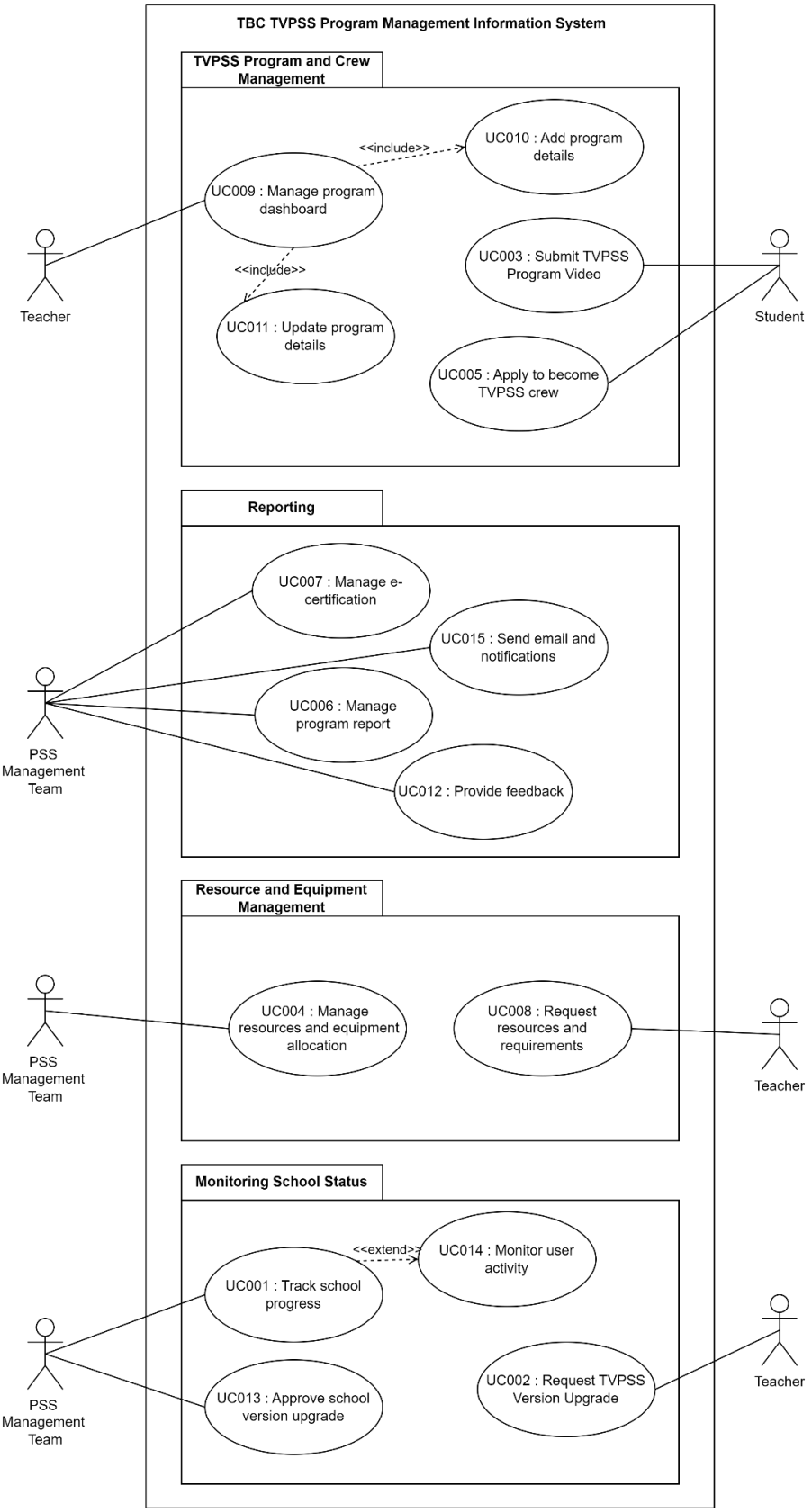


Figure 2.1: Use Case Diagram of TBC TVPSS Program MIS

5.2.2.1 UC001: Use Case Track school progress

Table 5.1 Use Case Specification for UC001 Track School Progress

Use Case ID	UC001
Use Case Name	Track School Progress
Description	This use case is used by the PSS management team to track the TVPSS program status in each school.
Actor(s)	PSS management team
Pre-condition(s)	PSS management team logged into the system.
Normal Flow(s)-NF	1. PSS management team clicks on the schools program list. 2. The system will redirect to the school program list page. 3. The system will display a list of schools participating in the TVPSS program. 4. PSS management selects a particular school. 5. The system retrieves the particular school's TVPSS program information from the database. 6. The system displays the school's TVPSS program details such as current school version, crew members, equipment status and any ongoing activities. 7. Use case ends.
Alternative Flow(s) - AF	**1. Cancel operation by user** 1.1. The system returns to the dashboard. 1.2. Use case is aborted.
Exception Flow(s) - EF	**1. System fail to retrieve data from database** 1.1. The system displays an error message. 1.2. NF4 will be executed again.
Post-condition(s)	The system will display the current progress of each school's TVPSS Program.

5.2.2.2 UC002: Use Case Request TVPSS Version Upgrade

Table 5.2 Use Case Specification for UC002 Request TVPSS Version Upgrade

Use Case ID	UC002
Use Case Name	Request TVPSS Version Upgrade
Description	This use case describes the process of a teacher requesting an upgrade of the TVPSS version through the TBC TVPSS MIS system.
Actor(s)	Teacher
Pre-condition(s)	The teacher is logged into the system and has access to the version upgrade interface.
Normal Flow(s)- NF	1. Teacher navigates to the version upgrade interface. 2. Teacher clicks on the version request button. 3. The system verifies if the upgrade is to version 3 and above. 4. If not, the system verifies criteria. 5. If criteria are met, the system generates an approval letter and displays it. 6. If criteria are not met, the system generates a rejection letter and displays it.
Alternative Flow(s) - AF	**1. Upgrade to Version 3 and Above** 1.1. The system notifies PSS staff for manual approval. 1.2. If the request is approved: 1.2.1. The system updates the TVPSS version status in the dashboard. 1.2.2. The system generates an approval letter. 1.3. If the request is not approved 1.3.1. The system generates a rejection letter.
Exception Flow(s) - EF	**1. Criteria Verification Error** 1.1. The system displays an error message indicating a verification error. 1.2. NF2 is executed again.

	2. Letter Generation Error 2.1. The system displays an error message indicating a generation error. 2.2. NF5 or NF6 will be executed again depending on the decision.
Post-condition(s)	1. The TVPSS version status is updated in the system.

5.2.2.3 UC003: Use Case Submit TVPSS Program Video

Table 5.3 Use Case Specification for UC003 Submit TVPSS Program Video

Use Case ID	UC003
Use Case Name	Submit TVPSS Program Video
Description	This use case allows a student to submit a video for the TVPSS program through the interface.
Actor(s)	Student
Pre-condition(s)	The student is logged into the system and has access to the video submission interface.
Normal Flow(s)-NF	1. Students click on the submit video option. 2. The system navigates to the video submission interface. 3. Student fills out the video description box. 4. Student attaches the video link. 5. Students click on the submit button. 6. The system verifies the submitted video. 7. If the submission is valid: 7.1. The system stores the video in the database. 7.2. If the video is successfully stored: 7.2.1. The system prompts a success message. 7.3. If the video is not successfully stored: 7.3.1. The system displays an error message.

Alternative Flow(s) - AF	**1. Invalid Submission** 1.1. The system displays an error message.
Exception Flow(s) - EF	**1. Database Error** 1.1. The system displays an error message.
Post-condition(s)	The submitted video is stored in the system database. The student receives a confirmation message.

5.2.2.4 UC004: Manage Resources and Equipment Allocation

Table 5.4 Use Case Specification for UC004 Manage Resources and Equipment Allocation

Use Case ID	UC004
Use Case Name	Manage Resources and Equipment Allocation
Description	This use case is used by the PSS Management Team to allocate resources and equipment for the TVPSS program.
Actor(s)	PSS Management Team
Pre-condition(s)	The PSS management team had received the resources and equipment request from the school.
Normal Flow(s)-NF	1. The PSS Management Team navigates to resources and equipment interfaces. 2. The system will display the list of schools from different districts. 3. The PSS Management Team selected the school that needed to allocate resources. 4. The system displays the allocation form. 5. The PSS Management Team enters the allocation details. 6. The system validates the allocation details. 7. If the details are valid, the system will update the resource allocation in the database. 8. Use case ends.

Alternative Flow(s) - AF	**1. Edit Allocation** 1.1. The PSS Management Team selects an existing allocation to edit. 1.2. The system displays the allocation form. 1.3. The PSS Management Team updates the details. 1.4. The system validates the form 1.4.1. IF form validation fails, EF 1 is executed. 1.5. Use case ends. **2. Delete Allocation** 2.1. The PSS Management Team selects an allocation to delete. 2.2. The PSS Management Team confirms the deletion. 2.3. The system deletes the allocation record
Exception Flow(s) - EF	**1. Allocation form validation failed** 1.1. The system displays an error message indicating the validation error. 1.2. NF 4 is executed again.
Post-condition(s)	Resources and equipment are allocated and recorded in the system.

5.2.2.5 UC005: Apply to Become TVPSS Crew

Table 5.5 Use Case Specification for UC005 Apply to Become TVPSS Crew

Use Case ID	UC005
Use Case Name	Apply to Become TVPSS Crew
Description	This use case is used by students to apply for a position in the TVPSS crew.
Actor(s)	Student
Pre-condition(s)	The student must be registered in the system. The application period must be open.
Normal Flow(s)-NF	1. The student navigates to the TVPSS crew application interface. 2. The system displays the TVPSS crew application form.

	3. The student fills out the application form with the required information. 4. The student submits the application form. 5. The system validates the application form. 6. If the application is valid, the system records the application in the database. 7. The system sends a confirmation email to the student. 8. Use case ends.
Alternative Flow(s) - AF	**1. Edit Application** 1.1. The student navigates to the previously submitted application. 1.2. The system displays the application form with the existing details. 1.3. The student updates the application details. 1.4. The student submits the updated application form. 1.5. The system validates the updated application form. 1.6. If the validation fails, EF 1 is executed. 1.7. Use case ends.
Exception Flow(s) - EF	**1. Application Form Validation Failed** 1.1. The system displays an error message indicating the validation error. 1.2. AF1 is executed.
Post-condition(s)	1. The student's application for the TVPSS crew is recorded in the system. 2. The student receives a confirmation email.

5.2.2.6 UC006: Manage Program Report

Table 5.6 Use Case Specification for UC006 Manage Program Report

Use Case ID	UC006
Use Case Name	Manage Program Report
Description	This use case is used by the PSS Management Team to view reports on the TVPSS program.
Actor(s)	PSS Management System
Pre-condition(s)	The system has existing program data. The user is authenticated and authorized to view reports.
Normal Flow(s)-NF	1. The PSS Management Team navigates to the program reports interface. 2. The system displays the list of available reports. 3. The PSS Management Team selects a report to view. 4. The system generates and displays the selected report. 5. Use case ends.
Alternative Flow(s) - AF	**1. Download Report** 1.1. The PSS Management Team selects the option to download the report. 1.2. The system generates the report in the selected format (e.g., PDF, Excel). 1.3. The system prompts the user to save the report file. 1.4. Use case ends. **2. Filter Report Data** 2.1. The PSS Management Team selects filtering options such as date range and school district. 2.2. The system applies the filters and regenerates the report with filtered data. 2.3. The system displays the filtered report. 2.4. Use case ends.
Exception Flow(s) - EF	**1. Report Generation Fail** 1.1. The system displays an error message indicating the issue. 1.2. NF 2 will be invoked again.
Post-condition(s)	The selected report is displayed to the PSS Management Team. The report data can be downloaded or filtered if necessary.

5.2.2.7 UC007: Generate E-Certification

Table 5.7 Use Case Specification for UC007 Generate E-Certification

Use Case ID	UC007
Use Case Name	Manage E-Certification
Description	This use case is used by the PSS Management Team to generate electronic certifications for students or teachers who have completed the TVPSS program.
Actor(s)	PSS management team
Pre-condition(s)	The student or teacher had completed a TVPSS program.
Normal Flow(s)-NF	1. The system displays the school list associated with the program name. 2. The PSS management team selects the school that needs the generation of e-certification. 3. The system retrieves program details from the database. 4. The system will display a sample e-certification for confirmation. 5. The system generates the e-certification. 6. The system will send the e-certification to the participants. 7. Concurrently, the system will invoke UC015. 8. Use case ends.
Alternative Flow(s) - AF	**1. Edit Certification Details** 1.1. The PSS Management Team edits the certification details. 1.2. The system displays the certification form. 1.3. The PSS Management Team updates the details. 1.4. The system validates the form. 1.4.1. IF the form validation fails, EF1 is executed. 1.5. The system updates the certification details. **2. Cancel operation by user** 2.1. The PSS Management Team cancels the operation. 2.2. The system redirects users to the last accessed page. 2.3. Use case ends.
Exception Flow(s) - EF	**1. Validation Failed**

	1.1. The system displays an error message indicating the validation error. 1.2. AF 1 is invoked again. **2. Certificate Generation Error** 2.1. The system displays an error message indicating a generation error. 2.2. The system logs the error for further investigation. 2.3. NF 4 is executed again.
Post-condition(s)	An e-certification is generated and sent to the participants.

5.2.2.8 UC008: Request Resource and Requirements

Table 5.8 Use Case Specification for UC008 Request Resource and Requirements

Use Case ID	UC008
Use Case Name	Request Resources and Requirements
Description	This use case allows a teacher to request resources and support through the system.
Actor(s)	Teacher
Pre-condition(s)	The teacher is logged into the system and has access to the "Resource Support" section.
Normal Flow(s)-NF	1. Teacher navigates to the "Resource Support" section. 2. Teacher fills out and submits a Resource Support request. 3. The system categorizes the request. 4. The system checks resource and equipment availability. 5. If resources are available: 5.1. The system generates a request report. 5.2. The system sends a notification to the PSS Management Team. 6. The PSS Management Team views the request report on the dashboard. 7. The PSS Management Team clicks the approve button accordingly. 8. IF staff assistance is needed:

	8.1. The system displays the staff assistance schedule. 9. IF equipment is needed: 9.1. The system displays the equipment lending letter. 10. IF no staff or equipment is needed: 10.1. The system displays the resource support letter
Alternative Flow(s) - AF	**1. Request Not Categorizable** 1.1. The system displays an unavailability message.
Exception Flow(s) - EF	**1. Resource Unavailability** 1.1. The system displays an unavailability message. 1.2. The process is halted.
Post-condition(s)	The resource support request is updated in the system. The teacher receives a confirmation message.

5.2.2.9 UC009: Manage Program Dashboard

Table 5.9 Use Case Specification for UC009 Manage Program Dashboard

Use Case ID	UC009
Use Case Name	Manage Program Dashboard
Description	This use case is used by the Teacher and PSS Management Team to manage the program dashboard, including viewing and updating program information.
Actor(s)	Teacher
Pre-condition(s)	Teacher is logged into the system
Normal Flow(s)-NF	1. The system displays the program dashboard interface. 2. The user views the program information. 3. The user selects an option. 4. IF the teacher selects to add a program. 4.1. THEN UC010 will be invoked. 5. ELSE IF the teacher selects to update program details. 5.1. THEN UC011 will be invoked. 6. The system will validate the details. 7. IF the details are valid, the system updates the program dashboard. 8. Use case ends.

Alternative Flow(s) - AF	**1. Delete Program Details** 1.1. The user selects a program to delete. 1.2. The user confirms the deletion. 1.3. The system deletes the dashboard detail (EF 2).
Exception Flow(s) - EF	**1. Validation Failed** 1.1. The system displays an error message indicating the validation error. 1.2. NF 3 is invoked again. **2. Network Error** 2.1. The system displays an error message indicating network error has occurred. 2.2. The system will attempt to reconnect. 2.3. NF3 is executed again.
Post-condition(s)	The program dashboard is updated and saved in the system.

5.2.2.10 UC010: Add Program Details

Table 2.10 Use Case Specification for UC010 Add Program Details

Use Case ID	UC010
Use Case Name	Add program details
Description	This use case is used by the teacher to add new program details to the system.
Actor(s)	Teacher
Pre-condition(s)	Teacher is logged into the system.
Normal Flow(s)-NF	1. The system displays the add program details interface. 2. The teacher enters the new program details. 3. The system validates the entered details. 4. IF the details are valid, the system will save the new program details. 5. Use case ends.
Alternative Flow(s) - AF	**1. Edit Program Details** 1.1. The teacher edits the new program details before saving. 1.2. The system displays the edit interface. 1.3. The teacher updates the details. 1.4. The teacher confirms the update.

	1.5. The system validates the form. 1.6. The system updates the dashboard details.
Exception Flow(s) - EF	**1. Database Insertion Error** 1.1. The system displays an error message indicating a database insertion error. 1.2. The system will log the error. 1.3. NF2 is executed again. **2. Network Timeout** 2.1. The system displays an error message indicating the network timeout. 2.2. The system attempts to reconnect and update the dashboard again. 2.3. NF2 is executed again.
Post-condition(s)	New program details are added and stored in the database.

5.2.2.11 UC011: Update Program Details

Table 5.11 Use Case Specification for UC011 Update Program Details

Use Case ID	UC011
Use Case Name	Update program details
Description	This use case is used by the teacher to update existing program details in the system.
Actor(s)	Teacher
Pre-condition(s)	1. Teacher is logged into the system. 2. There are existing program details in that school.
Normal Flow(s)-NF	1. The system displays the update program details interface. 2. The teacher selects the program details to update. 3. The system displays the current details. 4. The teacher updates the details. 5. The system validates the updated details. 6. IF the program details are valid 6.1. The system saves the updated program details. 7. Use case ends.

Alternative Flow(s) - AF	**1. Delete Program Details** 1.1. The teacher selects program details to delete. 1.2. The teacher confirms the deletion. 1.3. The system deletes the program details (EF 2). **2. Cancel Operation** 2.1. The PSS Management Team cancels the operation. 2.2. The system redirects users to the last accessed page. 2.3. The use case ends.
Exception Flow(s) - EF	**1. Validation Failed** 1.1. The system displays an error message indicating the validation error. 1.2. NF 4 is executed again. **2. Network Timeout** 2.1. The system displays an error message indicating the network timeout. 2.2. The system attempts to reconnect. 2.3. The system prompts the users to update the details. 2.4. NF4 is executed again.
Post-condition(s)	Program details are updated and stored in the system.

5.2.2.12 UC0012: Provide Feedback or Suggestions

Table 5.12 Use Case Specification for UC0012 Provide Feedback or Suggestions

Use Case ID	UC012
Use Case Name	Provide Feedback or Suggestions
Description	This use case is used by the PSS Management Team to provide feedback or suggestions for the TVPSS program.
Actor(s)	PSS Management Team
Pre-condition(s)	The user is authenticated and authorized to provide feedback or suggestions.
Normal Flow(s)-NF	1. The PSS Management Team navigates to the feedback interface. 2. The system displays the feedback form. 3. The PSS Management Team fills out the feedback form with the required information.

	4. The PSS Management Team submits the feedback form. 5. The system validates the feedback form. 6. If the feedback is valid, the system records the feedback in the database. 7. The system sends a confirmation notification to the user. 8. Use case ends.
Alternative Flow(s) - AF	**1. Edit Feedback** 1.1. The PSS Management Team navigates to previously submitted feedback. 1.2. The system displays the feedback form with existing details. 1.3. The PSS Management Team updates the feedback details. 1.4. The PSS Management Team submits the updated feedback form. 1.5. The system validates the updated feedback form. 1.6. If the validation fails, EF 1 is executed. 1.7. Use case ends. **2. Delete Feedback** 2.1. The system displays the feedback details. 2.2. The PSS Management Team selects the option to delete the feedback. 2.3. The system confirms the deletion action. 2.4. Use case ends.
Exception Flow(s) - EF	**Feedback Form Validation Failed** 1. The system displays a warning message indicating the validation error. 2. NF 3 is invoked again.
Post-condition(s)	The feedback or suggestion is recorded in the system. The PSS Management Team receives a confirmation notification.

5.2.2.13 UC013: Approve School Version Upgrade

Table 5.13 Use Case Specification for UC013 Approve School Version Upgrade

Use Case ID	UC013
Use Case Name	Approve School Version Upgrade
Description	This use case is used by the PSS Management Team to approve a school's request for a version upgrade of the TVPSS program.
Actor(s)	PSS management team
Pre-condition(s)	The teacher has submitted a version upgrade request
Normal Flow(s)-NF	1. The system forwards version upgrade request to PSS Management Team. 2. The PSS management team reviews the request. 2.1. IF request to upgrade version 3 or higher 2.1.1. System sends requests for inspection. 2.1.2. District PSS Staff inspect the studio and program content. 2.1.3. The PSS management team approves the upgrade. 2.2. ELSE 2.2.1. System automatically validates the version upgrade request. 3. The system updates the latest school version. 4. The system generates a version approval letter. 5. The system sends the notification to the teacher. 6. The use case ends.
Alternative Flow(s) - AF	**1. Reject Version Upgrade** 1.1. The system prompts the PSS Management Team to provide feedback. 1.2. The system will generate a rejection letter. 1.3. The teacher will be notified of the rejection. **2. Request More Information** 2.1. The PSS management team requests additional information. 2.2. System will send notification to the teacher for additional information requests. 2.3. The teacher provides additional information. 2.4. NF3 is invoked again.
Exception Flow(s) - EF	**1. System updates fail** 1.1. The system displays an error message.

	1.2. NF3 is invoked again.
Post-condition(s)	The school's TVPSS version is upgraded and the TVPSS School Information Database is updated.

5.2.2.14 UC014: Monitor User Activity

Table 5.14 Use Case Specification for UC014 Monitor User Activity

Use Case ID	UC014
Use Case Name	Monitor User Activity
Description	This use case is used by the PSS management team to monitor the user activities within the system.
Actor(s)	PSS management team
Pre-condition(s)	PSS management team is logged into the system.
Normal Flow(s)-NF	1. The system displays user activity logs. 2. The PSS management team reviews the logs. 3. The PSS management team chose to execute AF1 or AF2. 4. The use case ends.
Alternative Flow(s) - AF	**1. Filter User Activity** 1.1. The PSS management team selects filter criteria. 1.2. The system displays the filtered activity logs. 1.3. The PSS management team sorts the data based on certain criteria. 1.4. The system displays the filtered and sorted activity logs. **2. Export User Activity** 2.1. The PSS management team selects export options such as in the PDF version. 2.2. The system generates the activity log report. **3. Cancel Operation** 3.1. The PSS management team canceled the operation. 3.2. The system redirects users to the last accessed page. 3.3. Use case ends.
Exception Flow(s) - EF	**1. System fail to fetch user activities from database**

	1.1. The system displays an error message. 1.2. NF1 will be invoked again.
Post-condition(s)	The system will display the user activity logs and enable users to export the activity logs as PDF files.

5.2.2.15 UC015: Send Email and Notifications

Table 2.15 Use Case Specification for UC015 Send Email and Notifications

Use Case ID	UC015
Use Case Name	Send Email and Notifications
Description	This use case is used by the PSS Management Team to send emails and notifications to teachers and students regarding the TVPSS program.
Actor(s)	PSS Management Team
Pre-condition(s)	1. PSS Management Team is logged into the system. 2. An e-certification has been generated or a version upgrade request has been processed.
Normal Flow(s)-NF	1. The system displays the email interface. 2. The PSS Management Team selects the recipients. 3. The PSS Management Team composes the email. 4. The system will attach the related e-certification to the email. 5. The PSS Management Team sent the email to the participants. 6. The use case ends.
Alternative Flow(s) - AF	**1. Edit Message Content** 1.1. The PSS Management Team edits the message content before sending. 1.2. The system displays the edit form. 1.3. The PSS Management Team modifies the email content. 1.4. The PSS Management Team confirms the update.
Exception Flow(s) - EF	**1. Email fail to be sent** 1.1. The system displays an error message indicating the email server is down. 1.2. The system queues the email or notification to be sent later.
Post-condition(s)	Emails and notifications are sent to the participants.

5.3 Logical Viewpoint

Logical viewpoint is there to elaborate existing and designed types and their implementations as classes and interfaces with their structural static relationships.

5.3.1 Design Concerns

The purpose of the logical viewpoint is to elaborate the implementation of classes with their relationship. Classes and interfaces highlight all the attributes and methods associated with a specific class. Relationships highlight how all the classes and interfaces interact with each other.

This system is divided into four modules.

- TVPSS Program and crew management which manages programs, applications and dashboards.
- Reporting which manages the certificates, feedbacks, notifications and reports
- Resource and equipment management which handles the requests and allocation of resources
- Monitoring school status which manages the monitoring of school status and version upgrades along with the tracking of user activities

The classes include the core classes which define the main attributes and functions and the controller classes which are used for the interaction between the user and the system. The relationships included are off association and composition.

5.3.2 Design View

The following is the class diagram of the system.

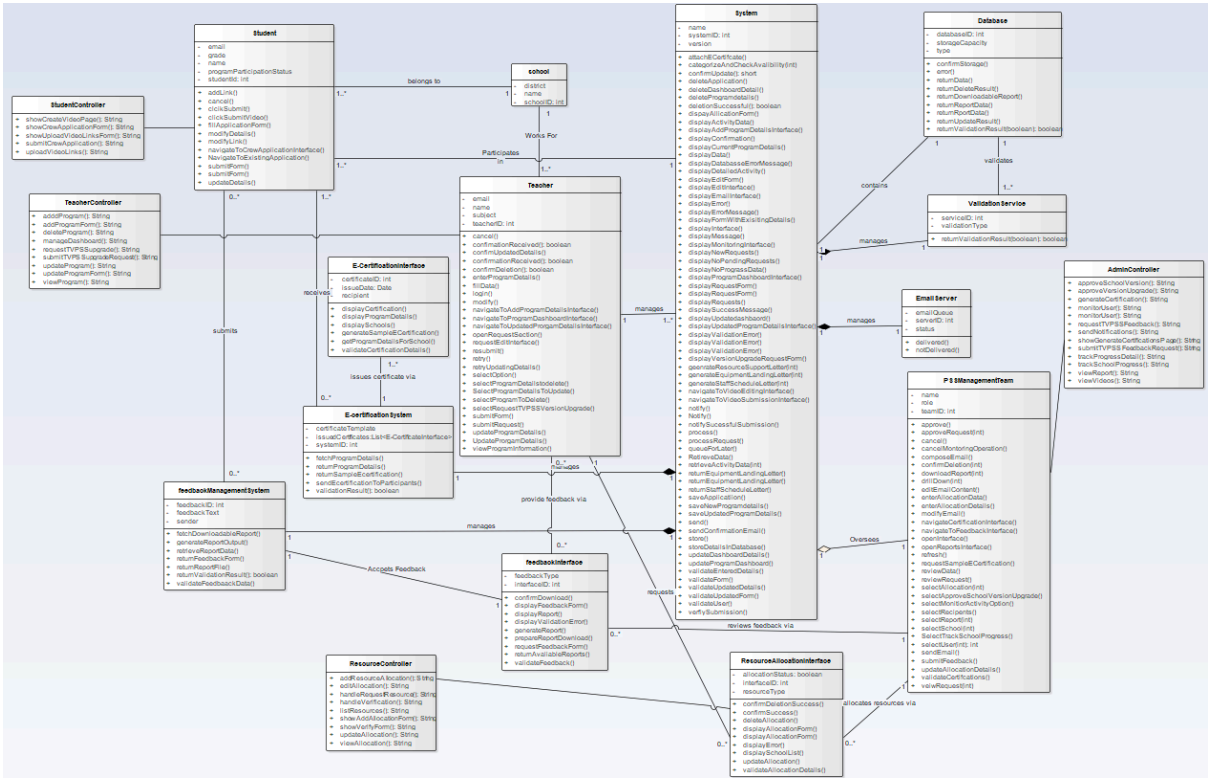


Figure 4.3: Class diagram for TBC TVPSS Management information system

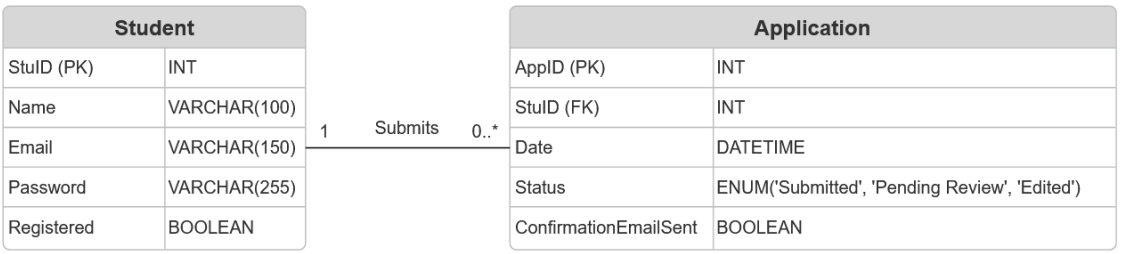
5.4 Information Viewpoint (UC005)

5.4.1 Design Concerns

The Information viewpoint applies when a design involves significant persistent data. Key concerns include data structure, content, management, access, and metadata. The system's information domain is transformed into data structures by identifying major entities and their relationships, mapped to tables in a relational database. The primary entities, Student and TVPSSApplication, are stored and organized in the StudentApplicationDB database.

Entity Name	Description
Student	A student, in all their individuality, who may submit applications. Contains the student's particulars: name, email, password, and registration status.
TVPSSApplication	An application, borne of a student's endeavor, submitted with details such as the date of submission, its status, and the state of the confirmation email.

5.4.2 Design View



Relationships:

Student → TVPSSApplication

- **Relationship Name**: **"Submits"**
- **Cardinality**:
  - A **Student** can submit **zero or many applications** (1 → 0..*).
  - A **TVPSSApplication** must belong to **exactly one student** (0..* → 1).

Database Design Logic:

1. **Data Integrity**: Foreign key constraints ensure that every TVPSSApplication references a valid Student.
2. **Scalability**: The structure allows multiple applications per student without duplicating data.
3. **Validation**:
   - Status ensures consistency by restricting values to predefined application states.
   - Registered ensures only valid users can submit applications.
4. **Automation**:
   - Auto-increment for StudentID and ApplicationID simplifies record creation.
   - ConfirmationEmailSent helps track whether a follow-up has been made.

Data Dictionary

Entity: *Student*

Column Name	Data Type	Constraints	Description
StudentID	INT	Primary Key, Auto-increment	Unique identifier for each student
Name	VARCHAR(100)	NOT NULL	Full name of the student
Email	VARCHAR(150)	NOT NULL, UNIQUE	Student's email address
Password	VARCHAR(255)	NOT NULL	Encrypted password
Registered	BOOLEAN	NOT NULL	Indicates if the student is registered

Entity: *TVPSSApplication*

Column Name	Data Type	Constraints	Description
ApplicationID	INT	Primary Key, Auto-increment	Unique identifier for each application
StudentID	INT	Foreign Key (Student.StudentID), NOT NULL	Links the application to a student
SubmissionDate	DATETIME	NOT NULL	Date and time the application was submitted
Status	ENUM	NOT NULL, Values: ('Submitted', 'Pending Review', 'Edited')	Status of the application
ConfirmationEmailSent	BOOLEAN	NOT NULL	Indicates if a confirmation email was sent

Interface Viewpoint

5.4.3 Design Concerns

The interface viewpoint covers user interfaces, hardware interfaces and software interfaces descriptions. This viewpoint will focus on the user's interaction with the system, the details of data flows between the system components and the feedback that will be given to the users. The goal is to design a user friendly interface to enhance usability and support the system functionalities. This viewpoint also ensures that all the external interfaces and internal interfaces are well documented so the system can be understood easily. This section will highlight the details about user interfaces and communication interfaces.

5.4.4 Design View

UI wireframe link:

https://www.figma.com/design/XAEYES57OU9Lw2EzgK4edM/RESM-Project?node-id=0-1&t=n1Dwqt5jF746A0iN-1

User Interfaces

The goal of this section is to define how the system will interact with its users.

Logical characteristics:

- All the pages will have a uniform and consistent layout in order to ensure ease of use
- Application logo is present in the header section
- Navigation bar is present on the side
- Report have drop down menu to select for particular school
- Each dashboard has a menu for respective user based on role
- Messages are there to indicate completion or failure of action
- text fields and drop down menus are standardized

Optimizing the interface:

- Submit and cancel buttons are placed logically
- A user is able to navigate from dashboard to another page with just one click
- Success messages are there to inform users about the actions

Do's:

- Button styles and positions are same
- Indicator for all users
- Simple language

Don'ts:

- Extra information on screens is avoided
- Avoided using error codes without explanations

Verification requirements:

- A student can perform the respective functions after 15 mins of training
- Dashboard loads within 5 secs
- All input fields should be validated with test cases

Hardware interfaces:

The logical characteristics for each interface are listed below

Devices:

- PC and laptop
  Devices using windows, macOS and linux are all supported. Wifi connectivity is required.
- Mobile devices
  Android and iOS both are supported but internet connectivity is required.
- Keyboard and Mouse
  Need for data input and navigation

Configuration connectivity:

- Wifi is required for connectivity

Software Interfaces

Software interfaces are listed below.

Required software products:

Development environment

- **Name:** Spring Tool Suite
- **Mnemonic:** STS
- **Specification number:** DEV-01
- **Version number:** 4.27.0
- **Source:** https://spring.io/tools

Database

- **Name:** MySQL
- **Mnemonic:** SQL
- **Specification number:** DB-01
- **Version number:** 9.1.0

- **Source:** https://www.mysql.com/

Frontend

- **Name:** HTML, CSS, JavaScript
- **Mnemonic:** Hypertext Markup, Cascading Styles, Just Scripts
- **Specification number:** DEV-01
- **Version number:** Standard W3C compliant
- **Source:** https://www.w3.org/

Interface Details

Purpose:

The development and deployment of the system is through these frameworks listed above

Communication format:

- MySql queries have been used in order to conduct database actions
- The frontend and backend communicate through JSON format

Communication Interfaces

Communication Interfaces used are listed below.

Network Communication

HTTP/HTTPS Protocols:

The web based access is possible through a connection that is secure.

Local Communication

Wi-fi:

TCP/IP protocols are used and access is granted within local networks.

5.5 Structure Viewpoint

5.5.1 Design Concerns

Structure viewpoint informs user and the developer about the interaction between the classes / packages in the system as it focuses on the internal components and the organization of the design subject. A clear representation of the organisation and interactions of classes, packages and modules is highlighted. This viewpoint ensures clarity in the system's design and implementation. Behavioral Patterns and Structural patterns are incorporated in this viewpoint.

Behavioral Patterns focuses on the dynamic interactions between components of the system and also signifies how the system reacts to different events and achieves its intended

functionalities. While the Structural Patterns focuses on the organization of classes and packages, showcasing their interconnections to support system functionalities.

5.5.1.2 UC003: Submit TVPSS Program Video

5.5.1.2.1 Structural Pattern : Composite Pattern

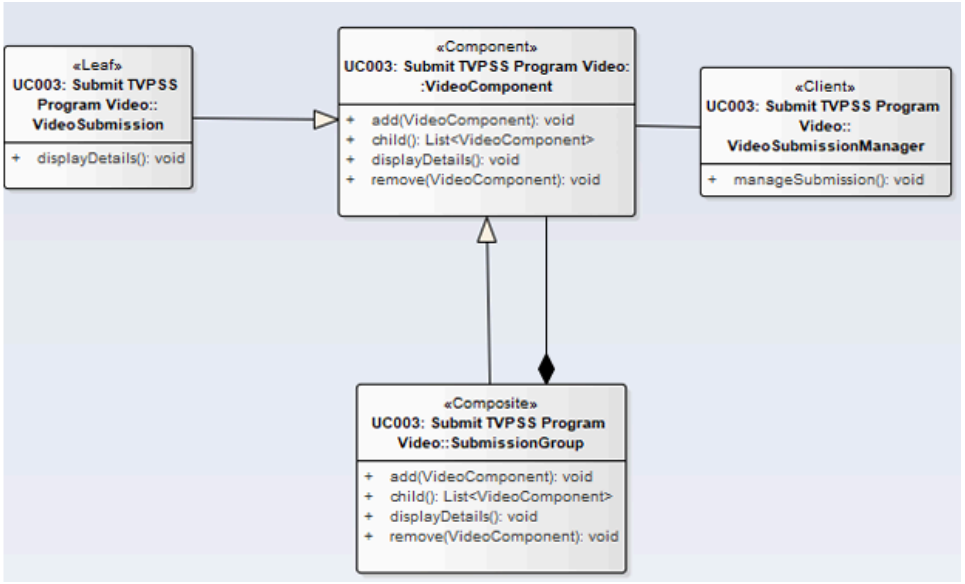


Figure 5.5.1.2.1 Class Diagram for Composite Pattern in submit TVPSS program Video

Code Snippet:

```
// Component Interface
interface VideoComponent {
    void add(VideoComponent component);
    void remove(VideoComponent component);
    void displayDetails();
    List<VideoComponent> child();
}

// Leaf
class VideoSubmission implements VideoComponent {
    @Override
    public void add(VideoComponent component) {
        throw new UnsupportedOperationException("Adding to leaf not successful.");
    }

    @Override
    public void remove(VideoComponent component) {
```

```
        throw new UnsupportedOperationException("Removing from a leaf not successful.");
    }

    @Override
    public void displayDetails() {
        System.out.println("Video submission details of single video.");
    }

    @Override
    public List<VideoComponent> child() {
        throw new UnsupportedOperationException("No children of leaf nodes");
    }
}

// Composite
class SubmissionGroup implements VideoComponent {
    private List<VideoComponent> components = new ArrayList<>();

    @Override
    public void add(VideoComponent component) {
        components.add(component);
    }

    @Override
    public void remove(VideoComponent component) {
        components.remove(component);
    }

    @Override
    public void displayDetails() {
        System.out.println("Details of different video submissions");
        for (VideoComponent component : components) {
            component.displayDetails();
        }
    }
```

```
    @Override
    public List<VideoComponent> child() {
        return components;
    }
}

// Client: Manages video submissions
class VideoSubmissionManager {
    private VideoComponent root;

    public VideoSubmissionManager(VideoComponent root) {
        this.root = root;
    }

    public void manageSubmission() {
        System.out.println("Managing video submissions:");
        root.displayDetails();
    }
}

// Main Class
public class VideoSubmissionSystem {
    public static void main(String[] args) {
        // Create individual
        VideoComponent video1 = new VideoSubmission();
        VideoComponent video2 = new VideoSubmission();
        VideoComponent video3 = new VideoSubmission();

        // Create group
        SubmissionGroup group1 = new SubmissionGroup();
        group1.add(video1);
        group1.add(video2);

        // Create new group and its sub group
        SubmissionGroup mainGroup = new SubmissionGroup();
        mainGroup.add(group1);
```

```
        mainGroup.add(video3);

        // Display submissions
        VideoSubmissionManager manager = new VideoSubmissionManager(mainGroup);
        manager.manageSubmission();
    }
}
```

5.5.1.2.2 Behavorial Pattern : Observer Pattern

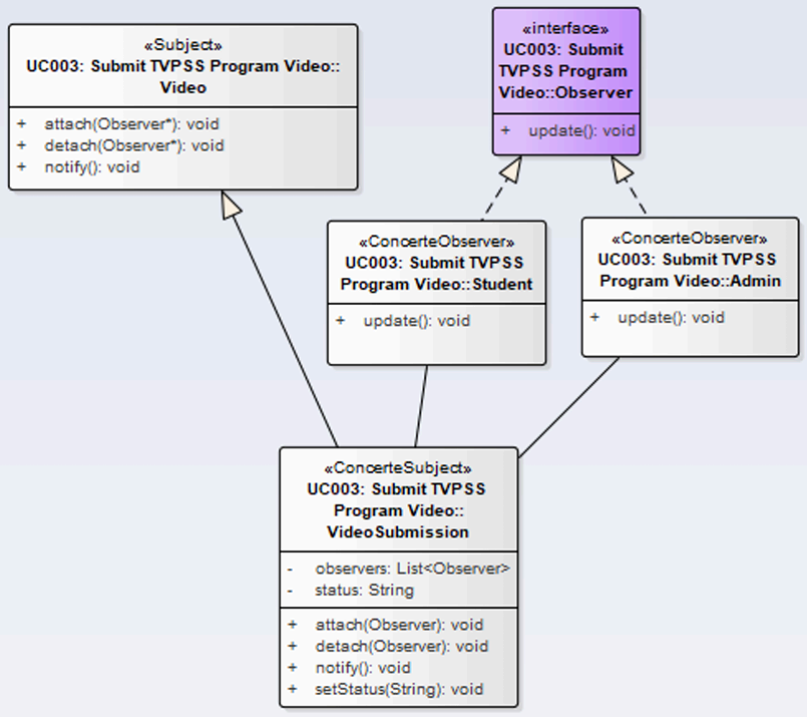


Figure 5.5.1.2.2 Class Diagram for Observer Pattern in submit TVPSS program Video

Code Snippet:

```
// Observer Interface
interface Observer {
    void update(String status);
}

// StudentObserver
class Student implements Observer {
    @Override
    public void update(String status) {
```

```
        System.out.println("Student received update: " + status);
    }
}

// AdminObserver
class Admin implements Observer {
    @Override
    public void update(String status) {
        System.out.println("Admin received update: " + status);
    }
}

// Subject Interface
interface Subject {
    void attach(Observer observer);
    void detach(Observer observer);
    void notify();
}

// ConcreteSubject: VideoSubmission
class VideoSubmission implements Subject {
    private List<Observer> observers = new ArrayList<>();
    private String status;

    @Override
    public void attach(Observer observer) {
        observers.add(observer);
    }

    @Override
    public void detach(Observer observer) {
        observers.remove(observer);
    }

    @Override
    public void notify() {
```

```
        for (Observer observer : observers) {
            observer.update(status);
        }
    }

    public void setStatus(String status) {
        this.status = status;
        notify();
    }
}

// Main Class
public class VideoSubmissionSystem {
    public static void main(String[] args) {
        // Create VideoSubmission
        VideoSubmission submission = new VideoSubmission();

        // Create Observers
        Observer student = new Student();
        Observer admin = new Admin();

        // Attach Observers
        submission.attach(student);
        submission.attach(admin);

        // Status Change
        System.out.println("Changing status to 'Pending Review'...");
        submission.setStatus("Pending Review");

        System.out.println("Changing status to 'Approved'...");
        submission.setStatus("Approved");
    }
}
```

5.5.2 **Design View**

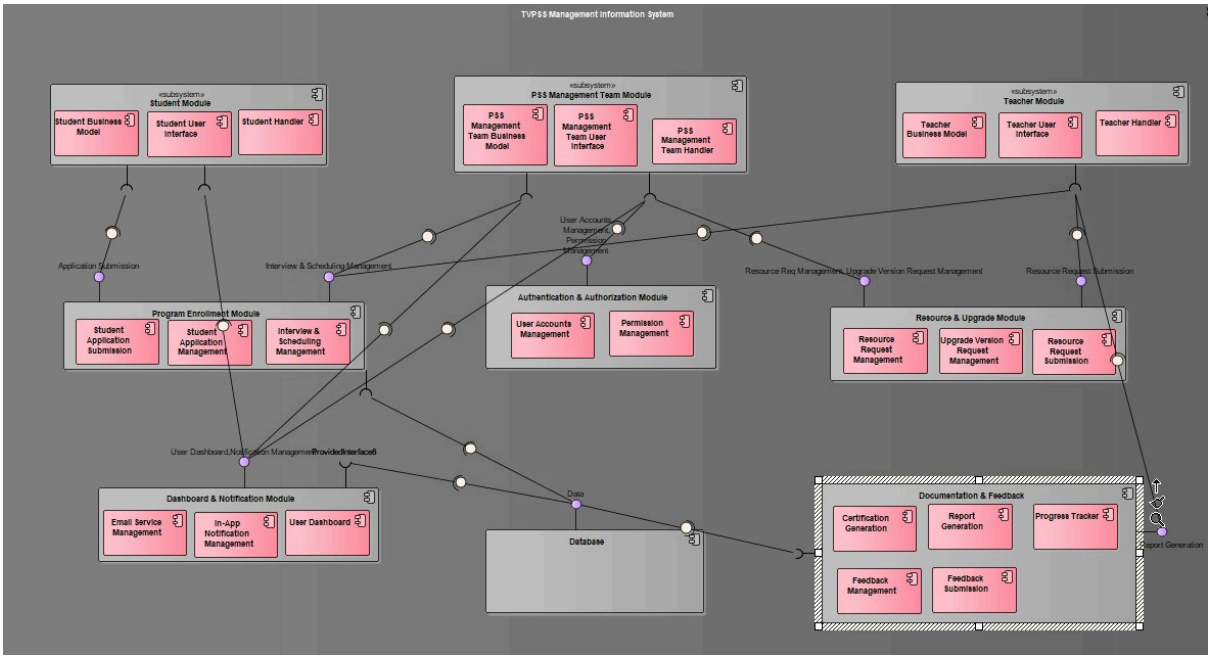


Figure 4.1: Package Diagram for TBC TVPSS Management Information system

5.6 **Interaction Viewpoint**

5.6.1 **Design Concerns**

The functionalities of the system are given by the help of class and sequence diagrams. Moreover, it defines strategies for interaction among entities addressing why; certain action occurs, purposes of interactions, where; the locations of systems where the interactions are held, how; the method of function calls, and when; the timing of the interactions. The class diagrams are used to represent structural relationships and sequence diagrams are used for dynamic behaviours and collaboration among the objects.

5.6.1.1 UC003: Submit TVPSS Program Video

5.6.1.1.1 Creational Pattern: Factory Method

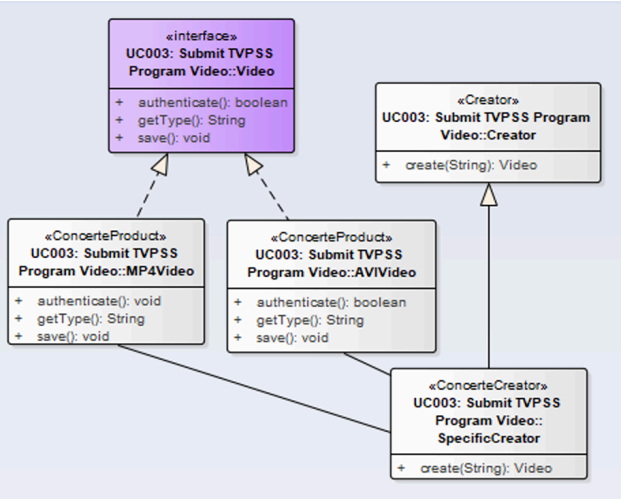


Figure 5.6.1.1.1 Class Diagram for Factory Method in submit TVPSS program Video

Code Snippet:

```
// Video Interface
public interface Video {
    boolean authenticate();
    String getType();
    void save();
}

// ConcreteProduct Class MP4Video
public class MP4Video implements Video {
    @Override
    public boolean authenticate() {
        System.out.println("Authenticating...");
        return true;
    }

    @Override
    public String getType() {
        return "MP4";
    }

    @Override
    public void save() {
```

```
        System.out.println("Saving...");
    }
}

// ConcreteProduct Class AVIVideo
public class AVIVideo implements Video {
    @Override
    public boolean authenticate() {
        System.out.println("Authenticating...");
        return true;
    }

    @Override
    public String getType() {
        return "AVI";
    }

    @Override
    public void save() {
        System.out.println("Saving...");
    }
}

// Creator Abstract Class
public abstract class Creator {
    public abstract Video create(String type);
}

// ConcreteCreator Class
public class SpecificCreator extends Creator {
    @Override
    public Video create(String type) {
        if ("MP4".equalsIgnoreCase(type)) {
            return new MP4Video();
        } else if ("AVI".equalsIgnoreCase(type)) {
            return new AVIVideo();
```

```
        } else {
            throw new IllegalArgumentException("Video type not supported: " + type);
        }
    }
}

// Factory Method
public class Main {
    public static void main(String[] args) {
        Creator creator = new SpecificCreator();

        // MP4 video
        Video mp4Video = creator.create("MP4");
        System.out.println("Video Type: " + mp4Video.getType());
        mp4Video.authenticate();
        mp4Video.save();

        // AVI video
        Video aviVideo = creator.create("AVI");
        System.out.println("Video Type: " + aviVideo.getType());
        aviVideo.authenticate();
        aviVideo.save();
    }
}
```

5.6.2 **Design View**.

a) SD001: Sequence diagram for Track School Progress

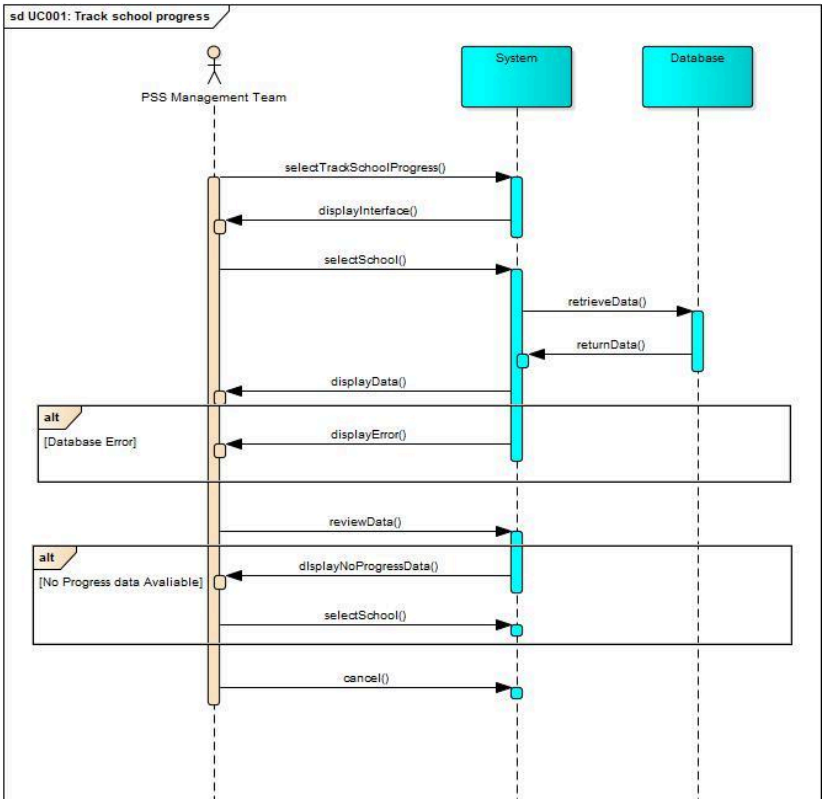


Figure 4.4: Sequence Diagram for <Track School Progress>

b) SD002: Sequence diagram for Request TVPSS Version Upgrade

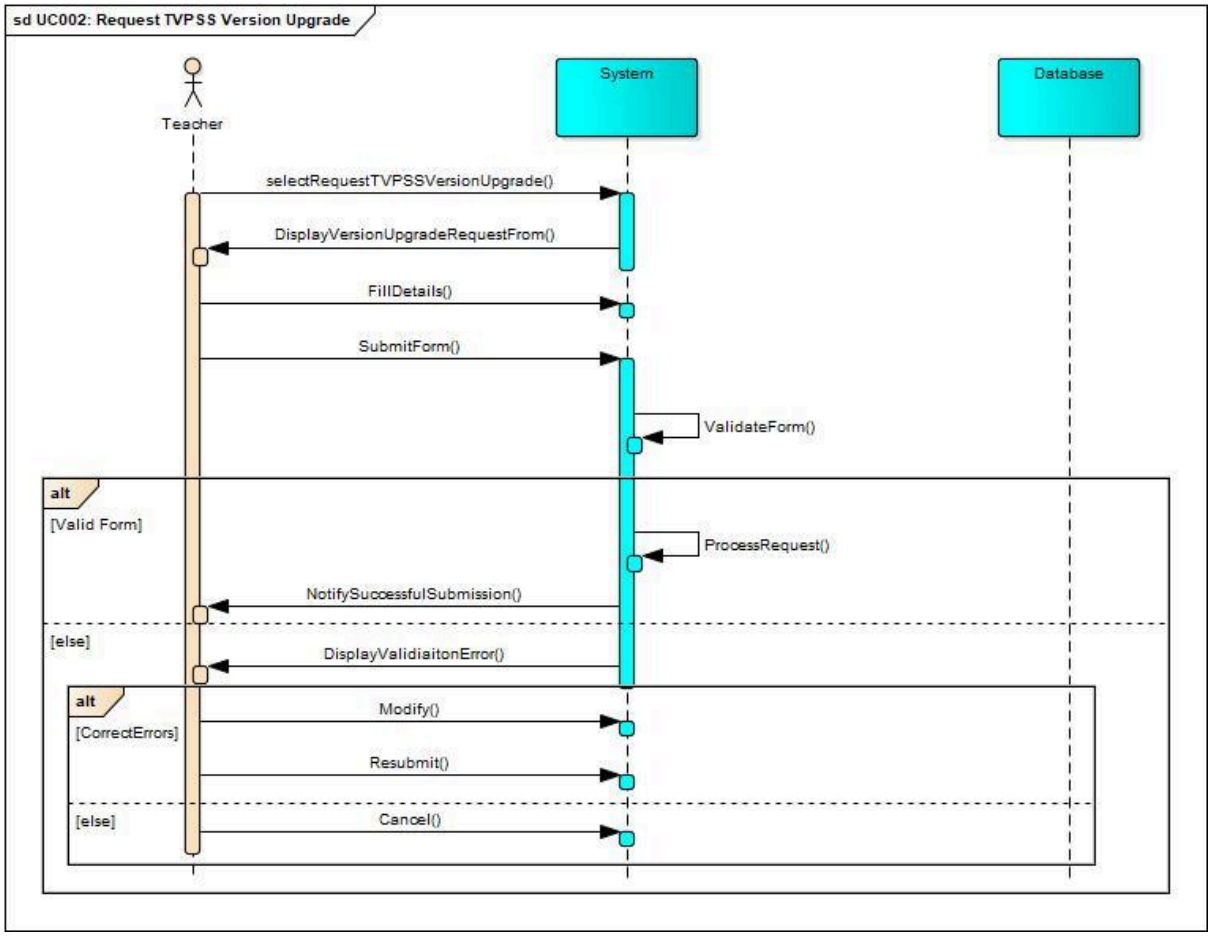


Figure 4.5: Sequence Diagram for <Request TVPSS Version Upgrade >

c) SD003: Sequence diagram for Submit TVPSS Program Video

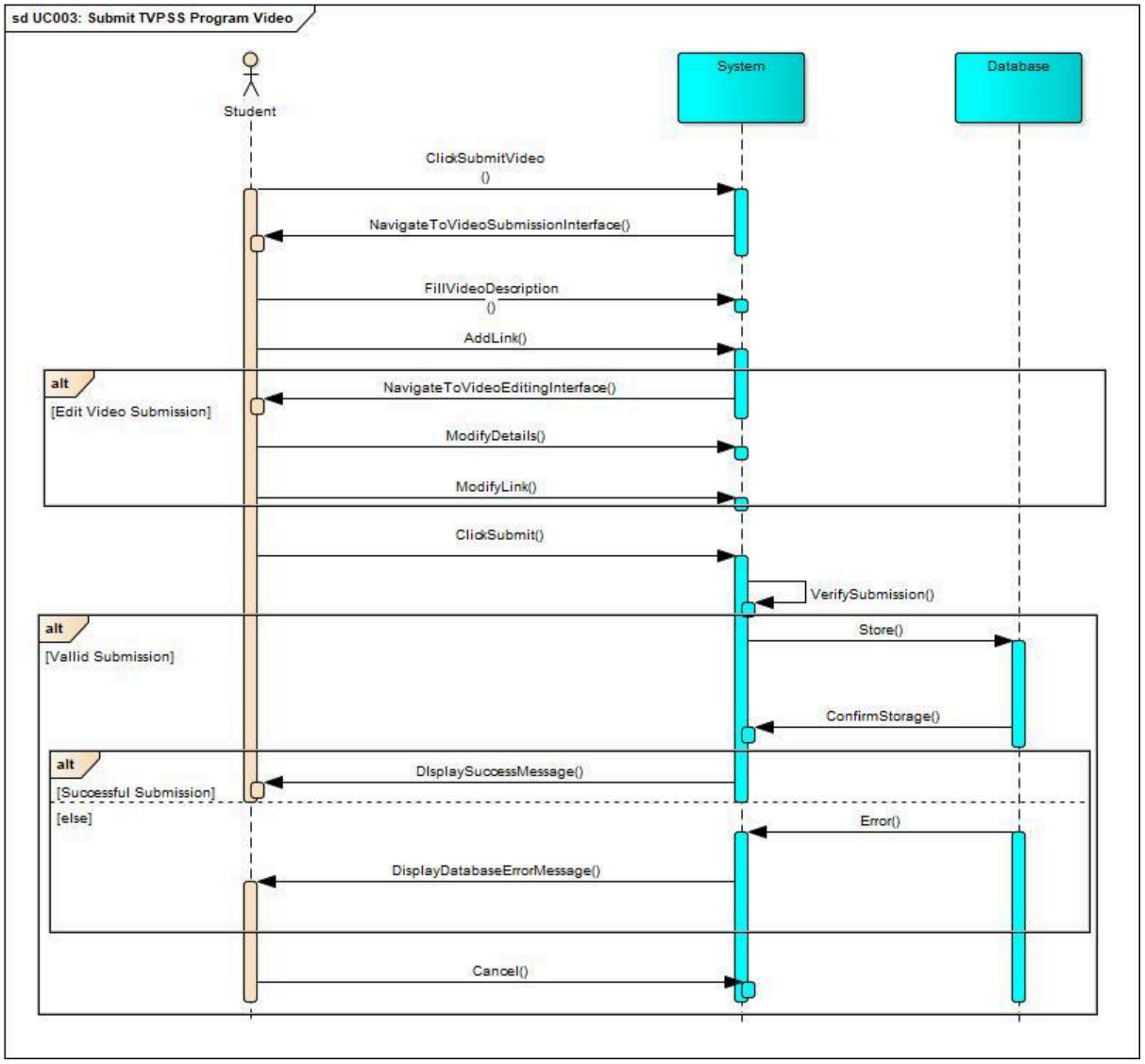


Figure 4.6: Sequence Diagram for UC003 Submit TVPSS Program Video

d) SD004: Sequence diagram for Manage Resources and Equipment Allocation

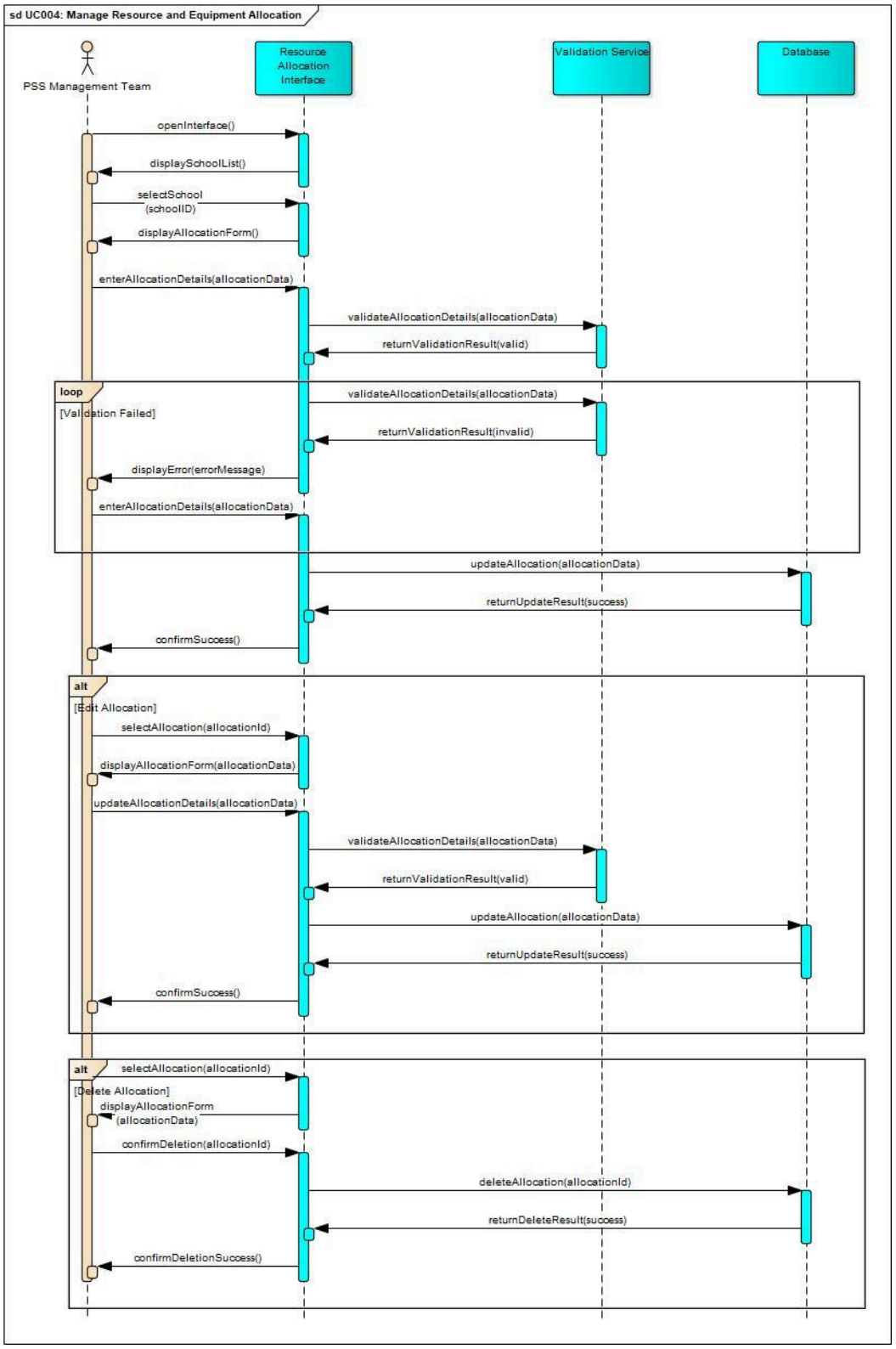


Figure 4.7: Sequence Diagram for UC004 Manage Resources and Equipment Allocation

e) SD005: Sequence diagram for Apply to Become TVPSS Crew

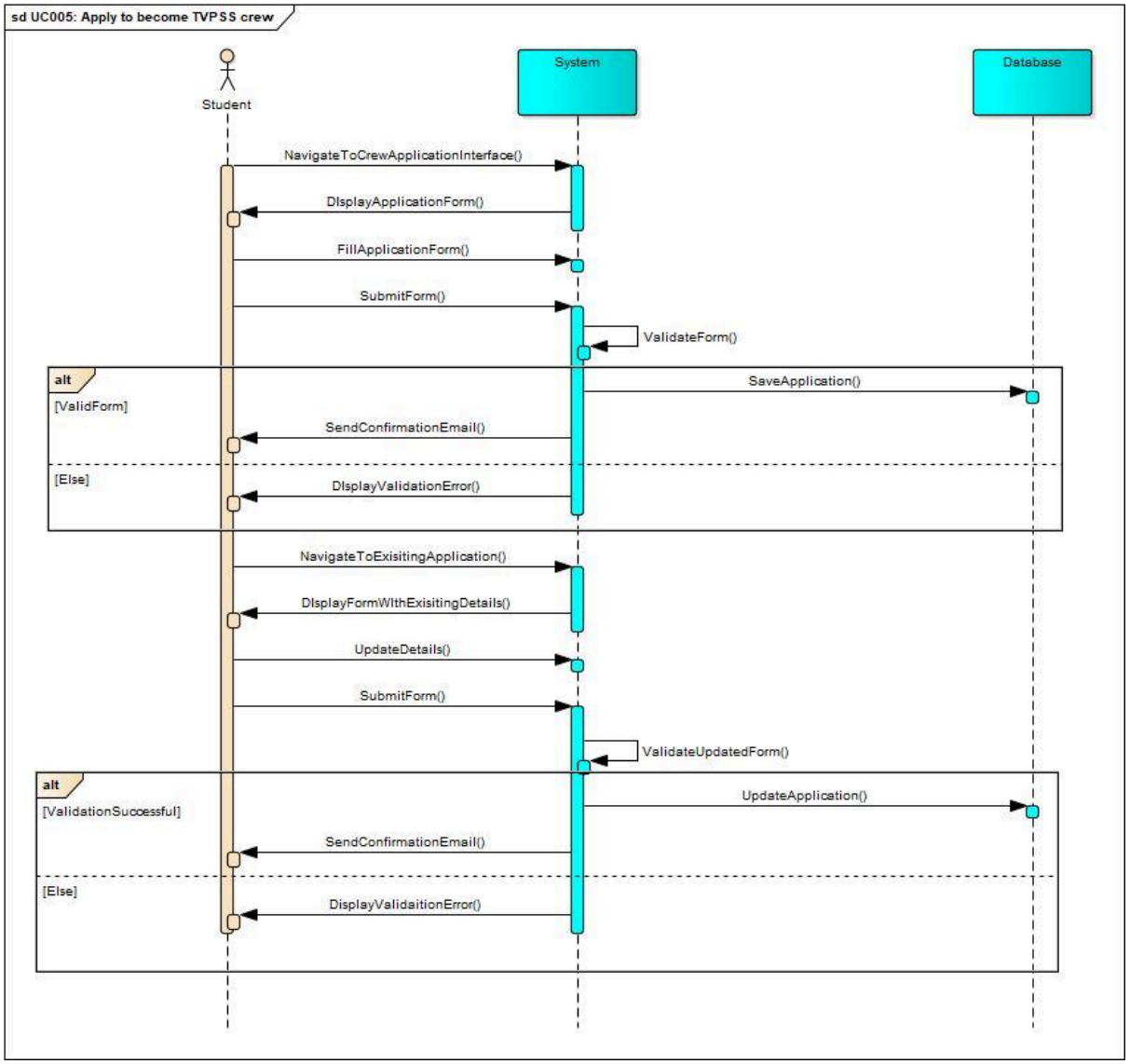


Figure 4.8: Sequence Diagram for UC005 Apply to Become TVPSS Crew

f) SD006: Sequence diagram for Manage Program Report

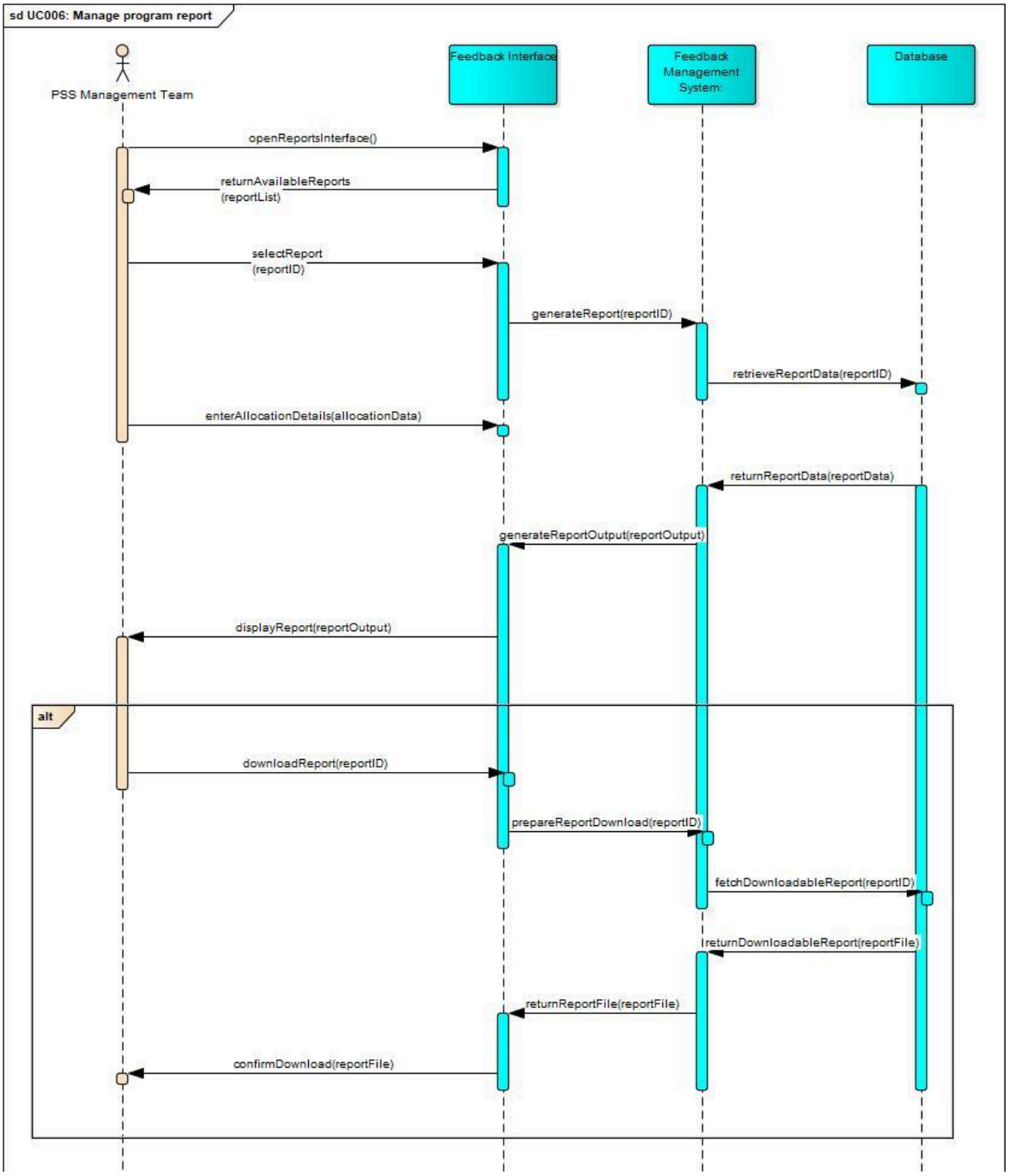


Figure 4.9: Sequence Diagram for UC006 Manage Program Report

g) SD007: Sequence diagram for Generate E-Certification

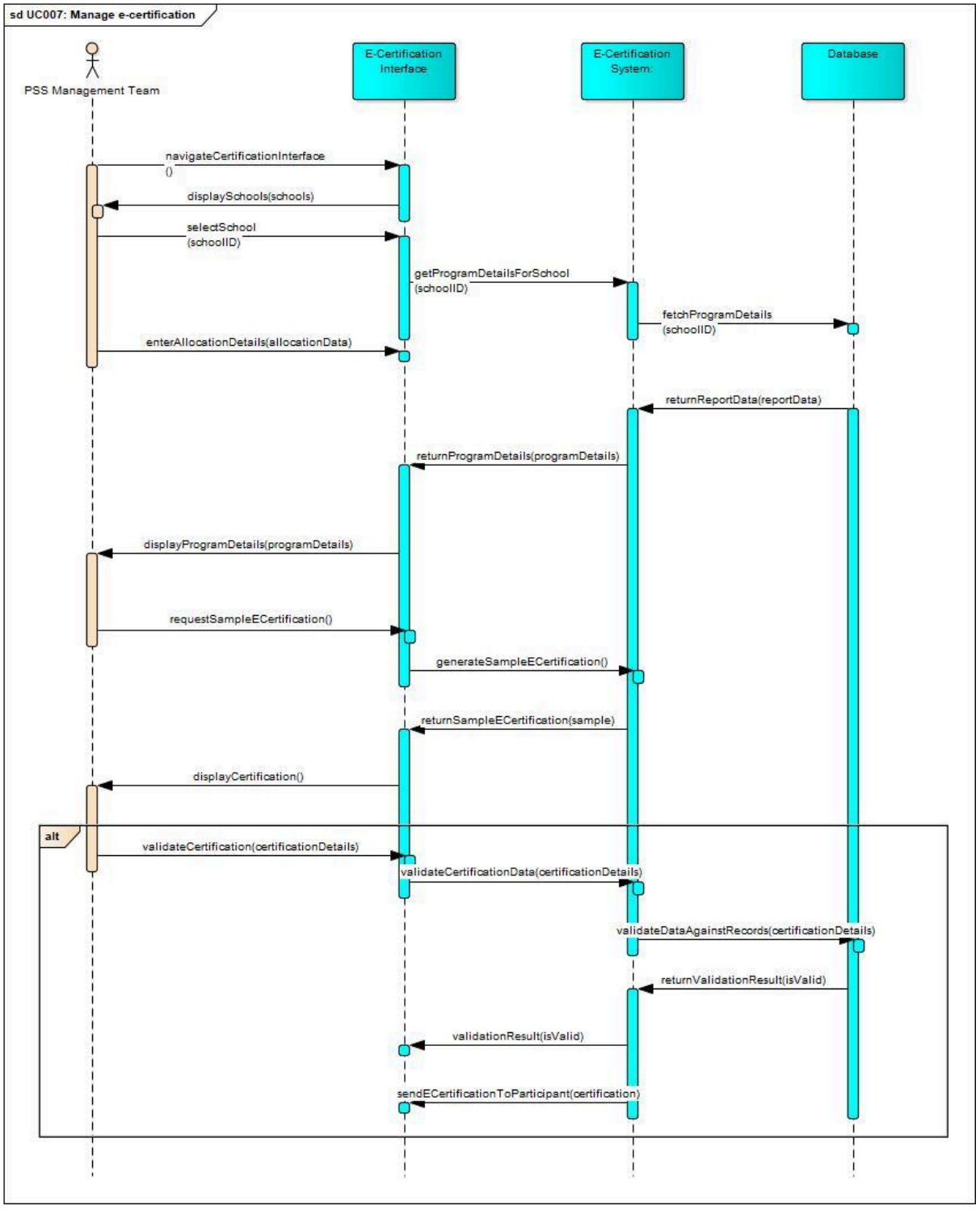


Figure 4.10: Sequence Diagram for UC007 Generate E-Certification

h) SD008: Sequence diagram for Request Resource and Requirements

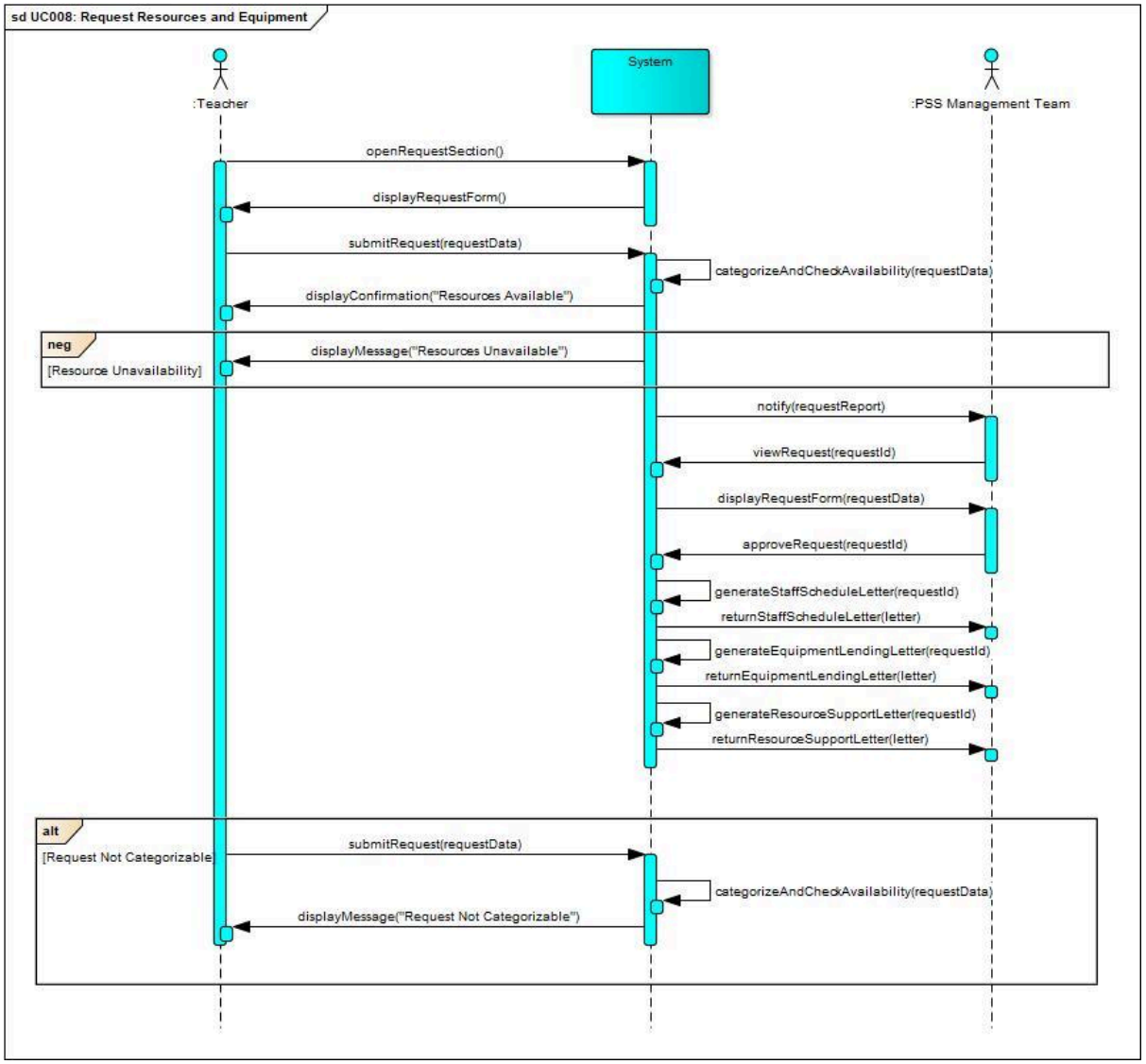


Figure 4.11: Sequence Diagram for UC008 Request Resource and Requirements

i) SD009: Sequence diagram for Manage Program Dashboard

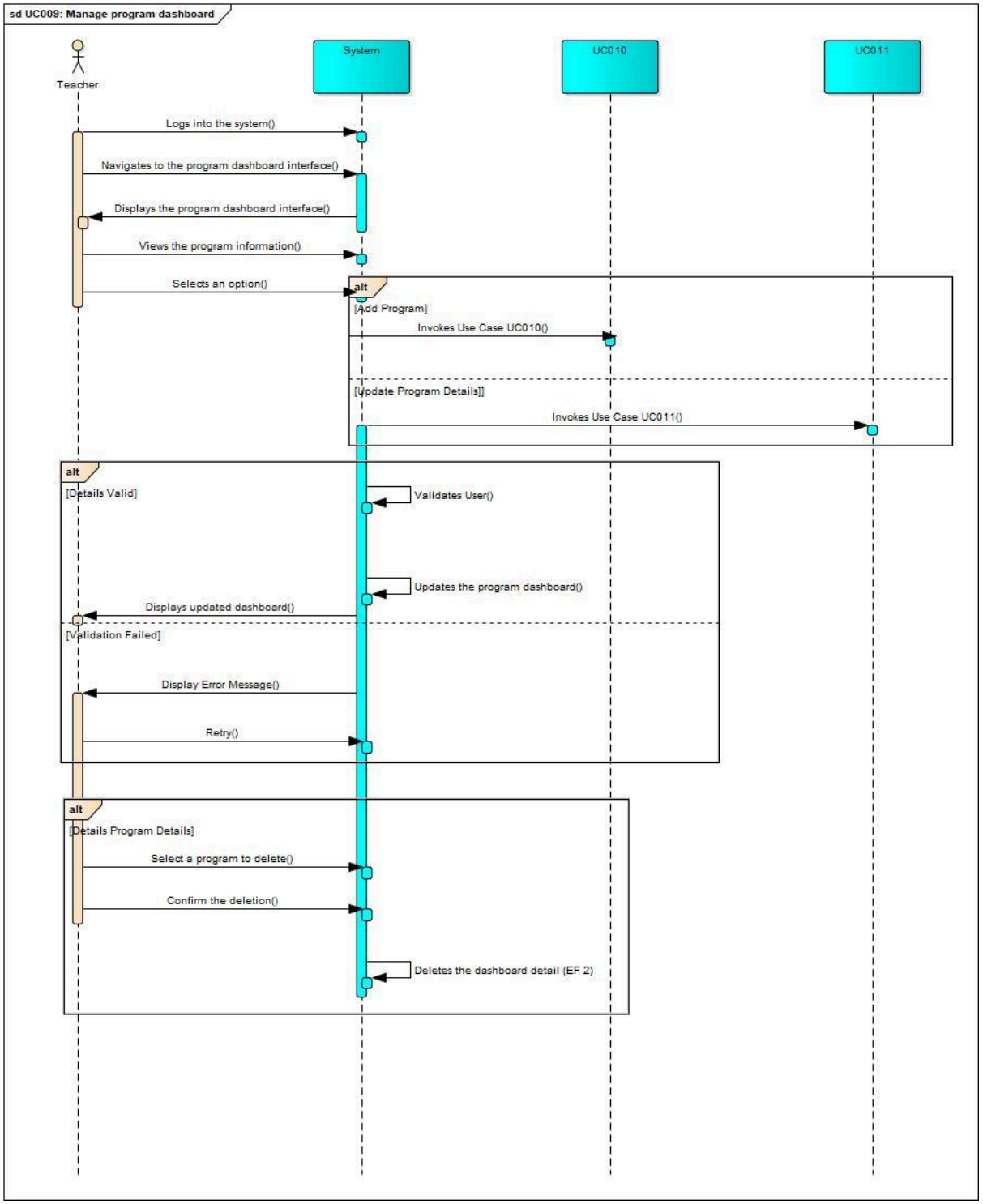


Figure 4.12: Sequence Diagram for UC009 Manage Program Dashboard

j) SD010: Sequence diagram for Add Program Details

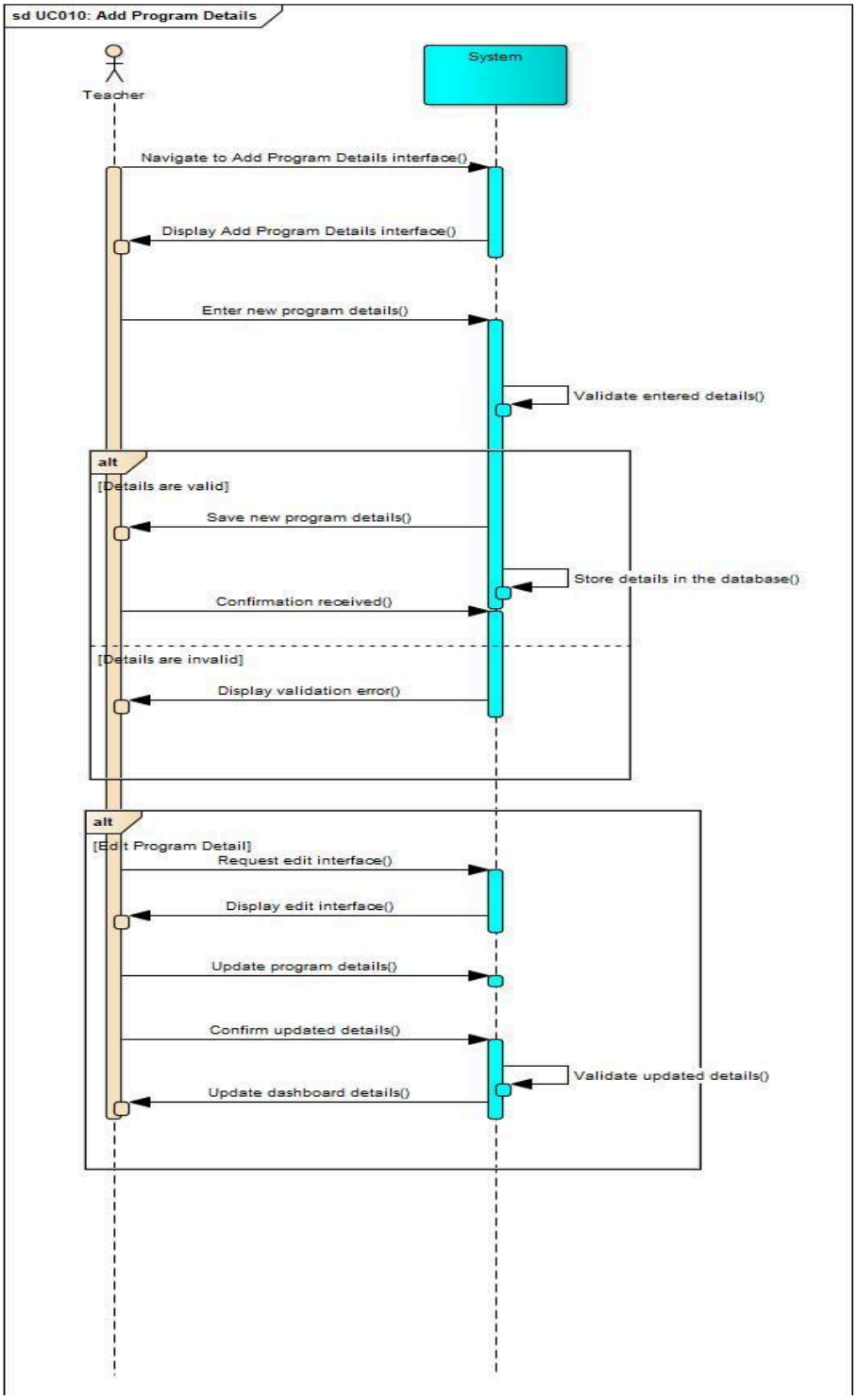


Figure 4.13: Sequence Diagram for UC010 Add Program Details

k) SD011: Sequence diagram for Update Program Details

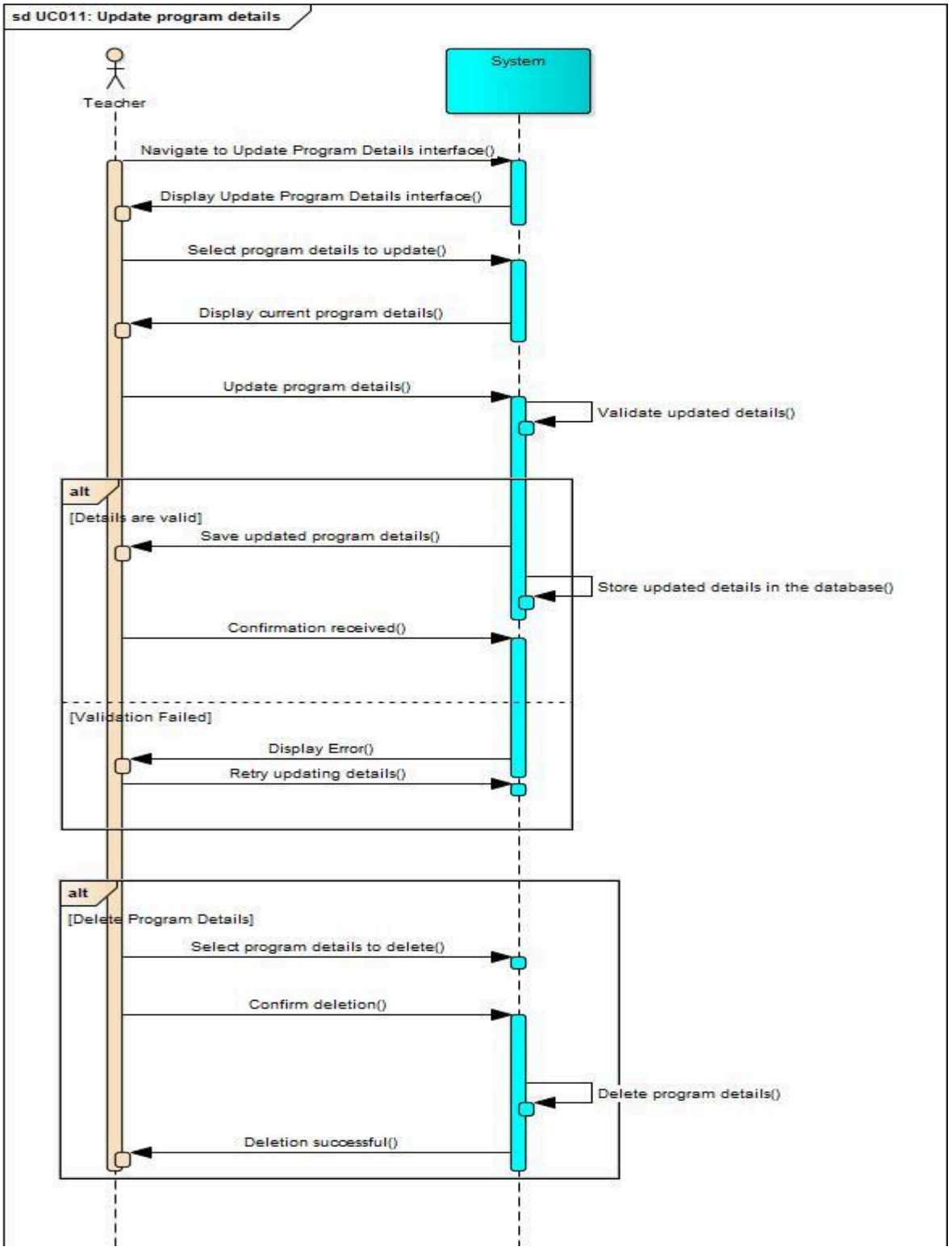


Figure 4.14: Sequence Diagram for UC011 Update Program Details

l) SD012: Sequence diagram for Provide Feedback or Suggestions

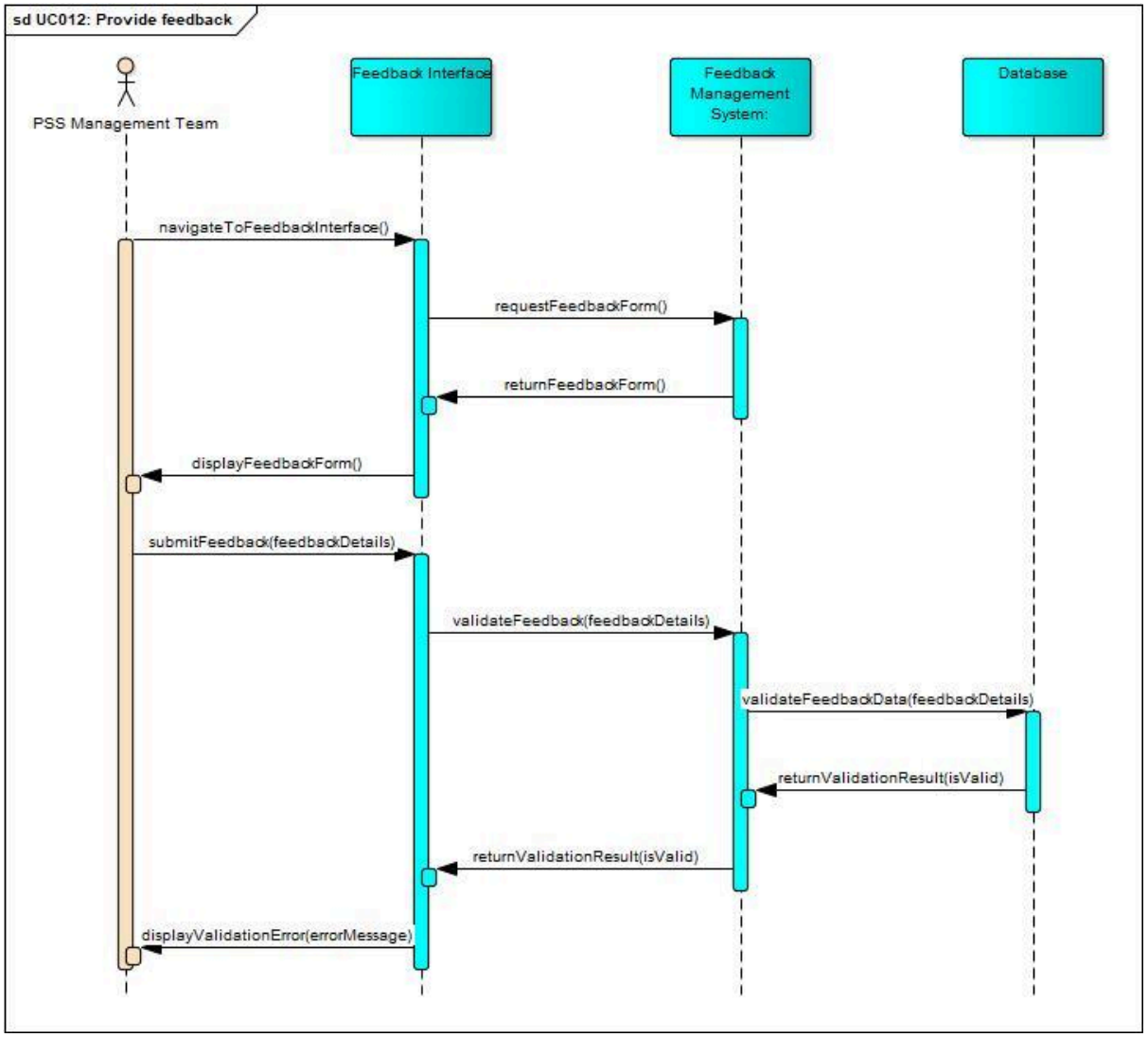


Figure 4.15: Sequence Diagram for UC012 Provide Feedback or Suggestions

m) SD013: Sequence diagram for Approve School Version Upgrade

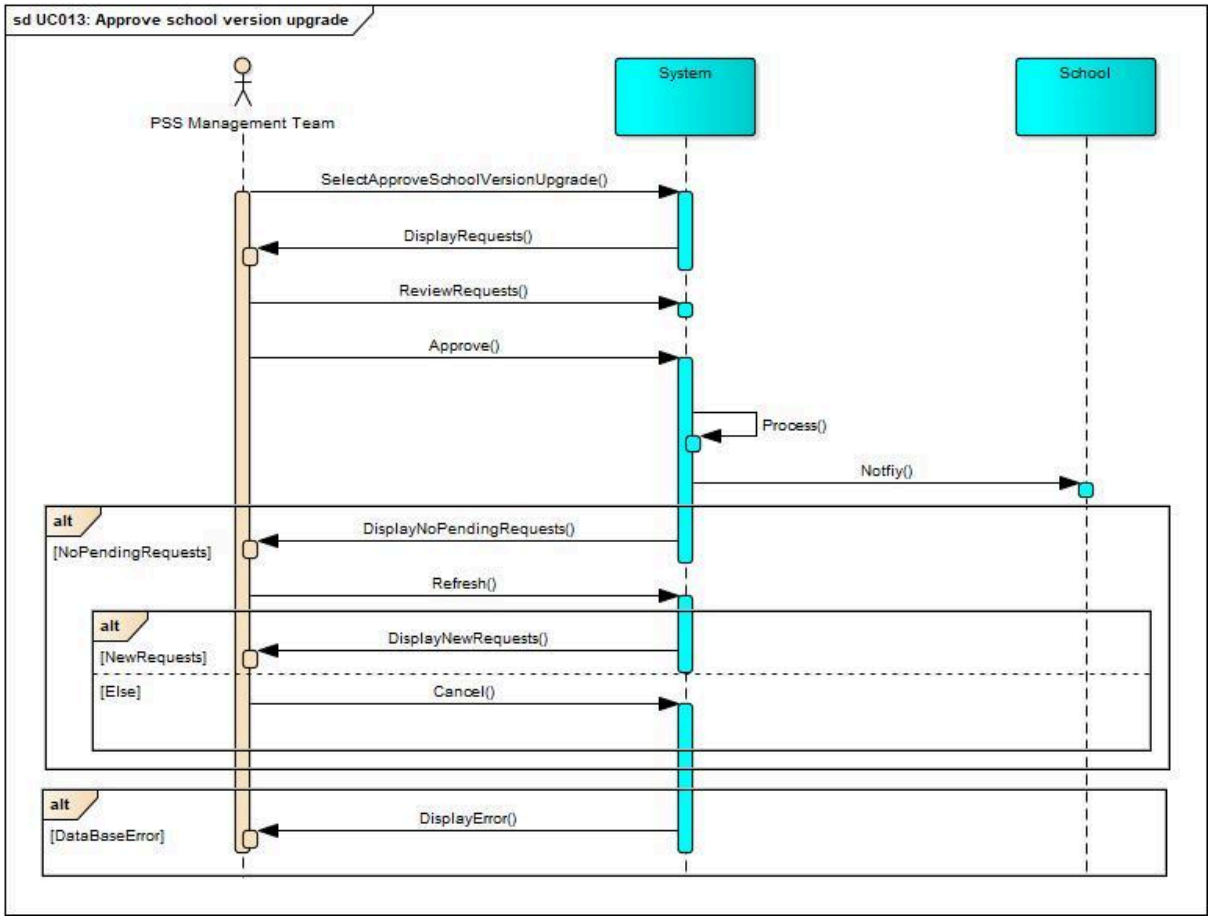


Figure 4.16: Sequence Diagram for UC013 Approve School Version Upgrade

n) SD014: Sequence diagram for Monitor User Activity

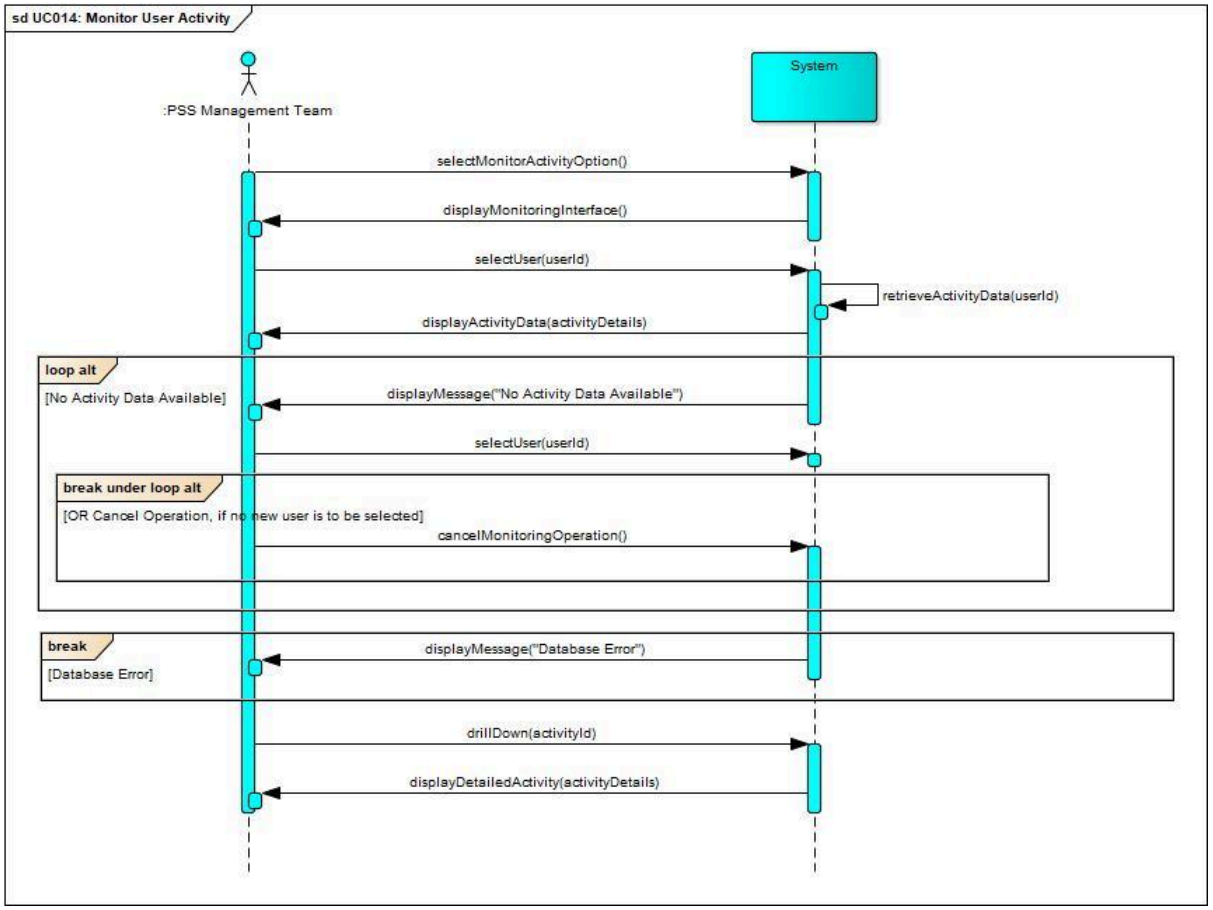


Figure 4.17: Sequence Diagram for UC014 Monitor User Activity

o) SD015: Sequence diagram for Send Email and Notifications

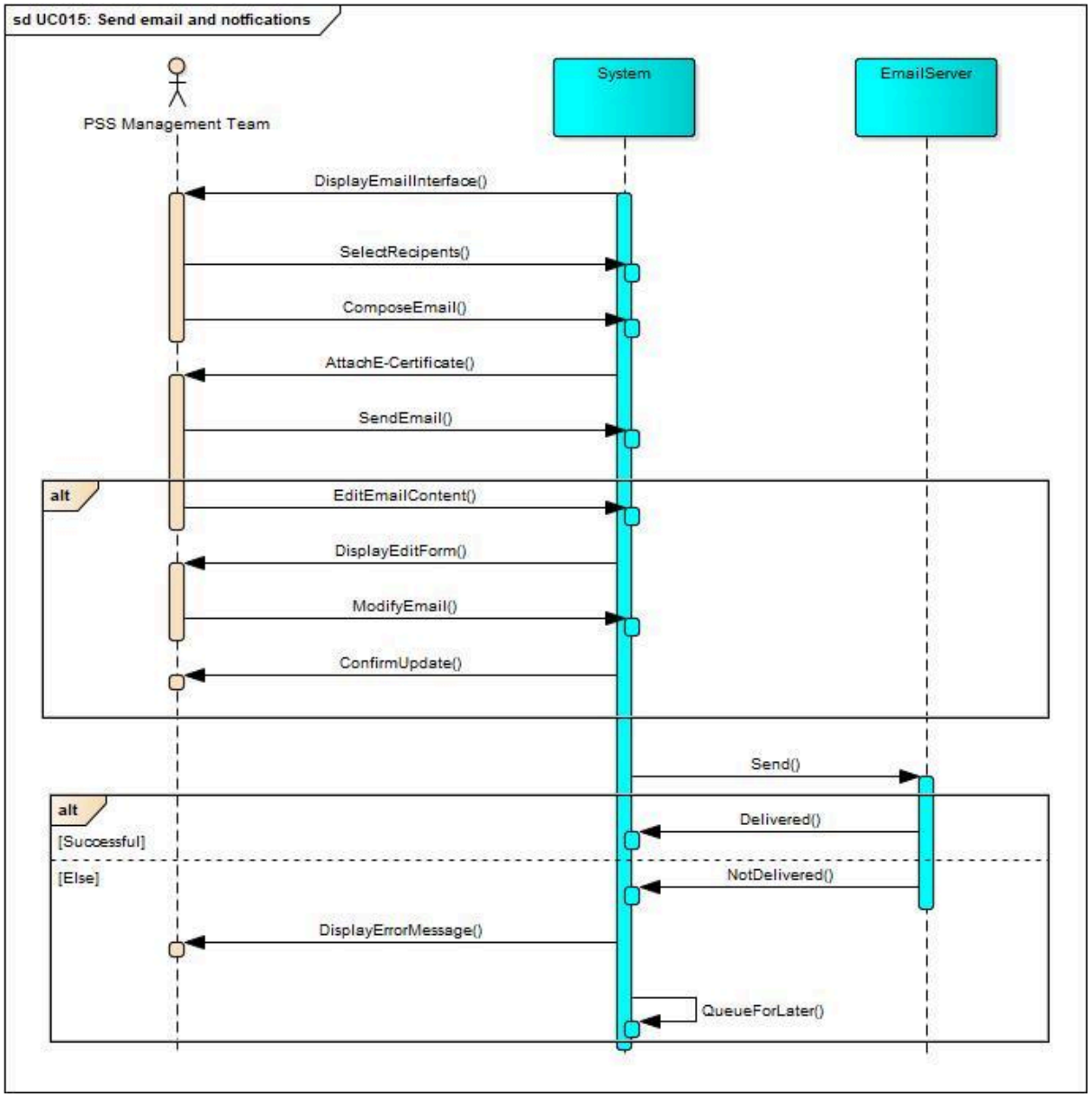


Figure 4.18: Sequence Diagram for UC015 Send Email and Notifications

5.7 Algorithm Viewpoint (UC005)

5.7.1 Design Concerns

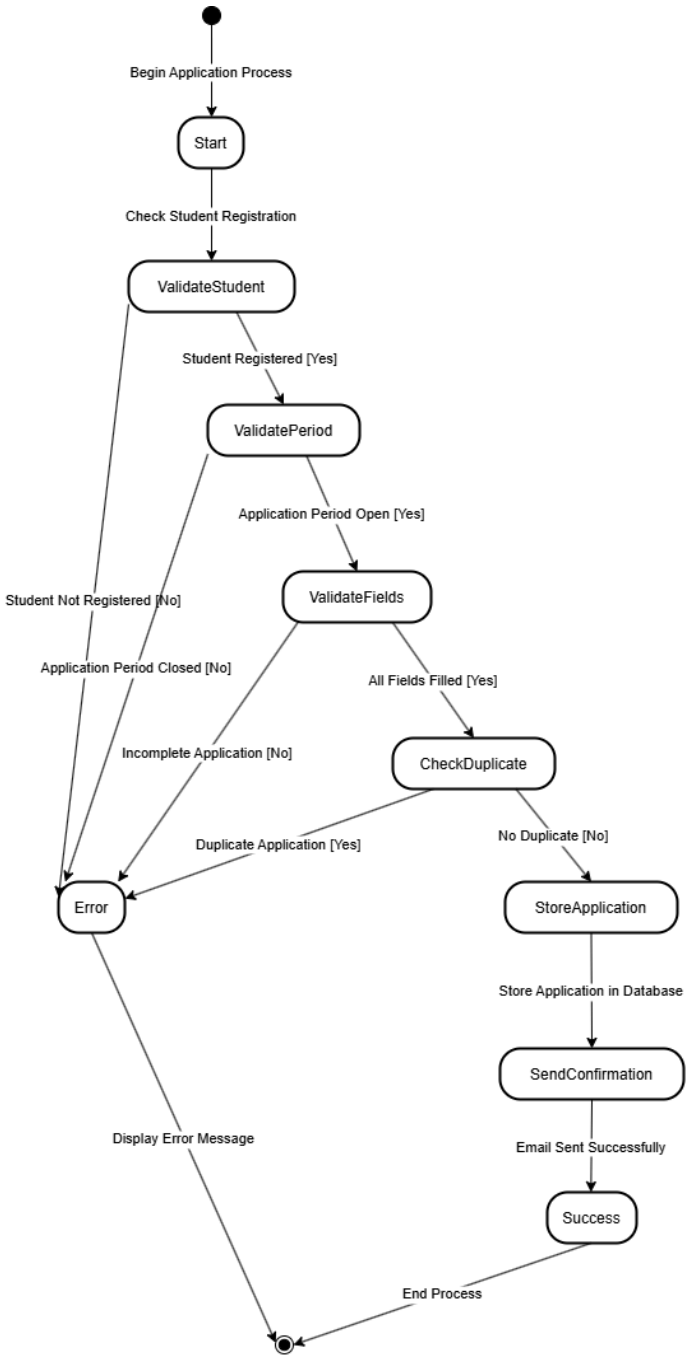
The Algorithm viewpoint provides details needed by programmers, analysts of algorithms in regard to time-space performance and processing logic prior to implementation, and to aid in producing unit test plans.

For UC005 our design concerns for the algorithm viewpoint consists of -

- **Data Validation:** Ensure that the application form is complete, accurate, and does not contain invalid entries.
- **Eligibility Check:** Confirm the student is registered in the system and that the application period is open.
- **Database Consistency:** Properly log the application into the database without errors or duplication.
- **Feedback Mechanism:** Provide immediate and meaningful feedback to the student upon submission or error.
- **Notification Delivery:** Send a confirmation email to the student upon successful submission.
- **Error Handling:** Handle exceptions such as invalid form input, database errors, or email service failures gracefully.

5.7.2 **Design View**

5.7.2.1 **State Diagram of UC005**



5.7.2.2 **Decision Table of UC005**

Condition/Action		**1**	**2**	**3**	**4**	**5**
Condition	Student Registered	Y	Y	Y	Y	N
	Application Period Open	Y	Y	Y	N	-
	All Fields Filled	Y	Y	N	-	-
	Duplicate Application	N	Y	-	-	-
Action	Validate Application	√	√	-	-	-
	Store in Database	√	-	-	-	-
	Send Confirmation Email	√	-	-	-	-
	Display Error Message	-	√	√	√	√

6. User Interface Design

6.1 Overview of User Interface

The system's main interface, for an admin, includes a sidebar with navigation buttons such as dashboard, profile, activity list, progress tracker, resource allocation, monitor activity, approve school version upgrade, program report, e-certificate, feedback and suggestions, send email and notifications. The specific buttons will direct to the respective page for each button.

For the teacher, the system's main interface sidebar has buttons for the main dashboard, add program, update program, crew list, activity list, application review, version upgrade and request resource.

For the student, the system's main interface sidebar can direct to the dashboard, profile, activity list, crew application and video studio.

The user interfaces for admin, student and teacher are designed based on the tasks to be done according to the respective roles. The interfaces also include forms for students and teachers to input data and approval interfaces for the admins to approve or reject requests. The users login with their respective credentials and access their dashboards.

Regarding the feedback information, success and errors messages display upon form submission, approval and rejections. Visual indicators such as progress bars are used to reflect the progress.The UI wireframes show the placement of all the elements.

The system will also be integrated with other software systems such as databases for interaction with centralized DBMS for data retrieval and storage. The system will use REST APIs to manage communication between frontend and backend components. Third party tools such as email services like SMTP will be integrated for notifications.

6.2 Screen Images

6.2.1 Module 1: TVPSS Program and Crew Management

6.2.1.1 UC009: Manage Program Dashboard

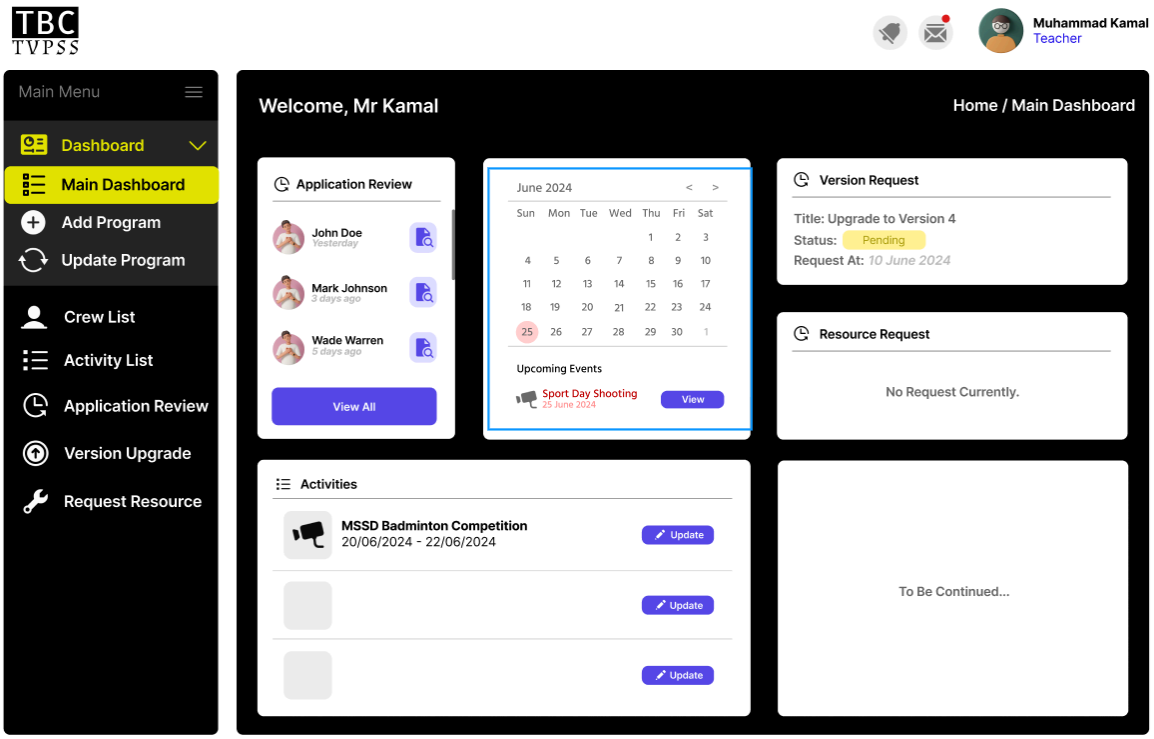


Figure 6.2.1.1 shows the User Interface for managing program dashboard from teacher's role

6.2.1.2 UC010: Add a Program

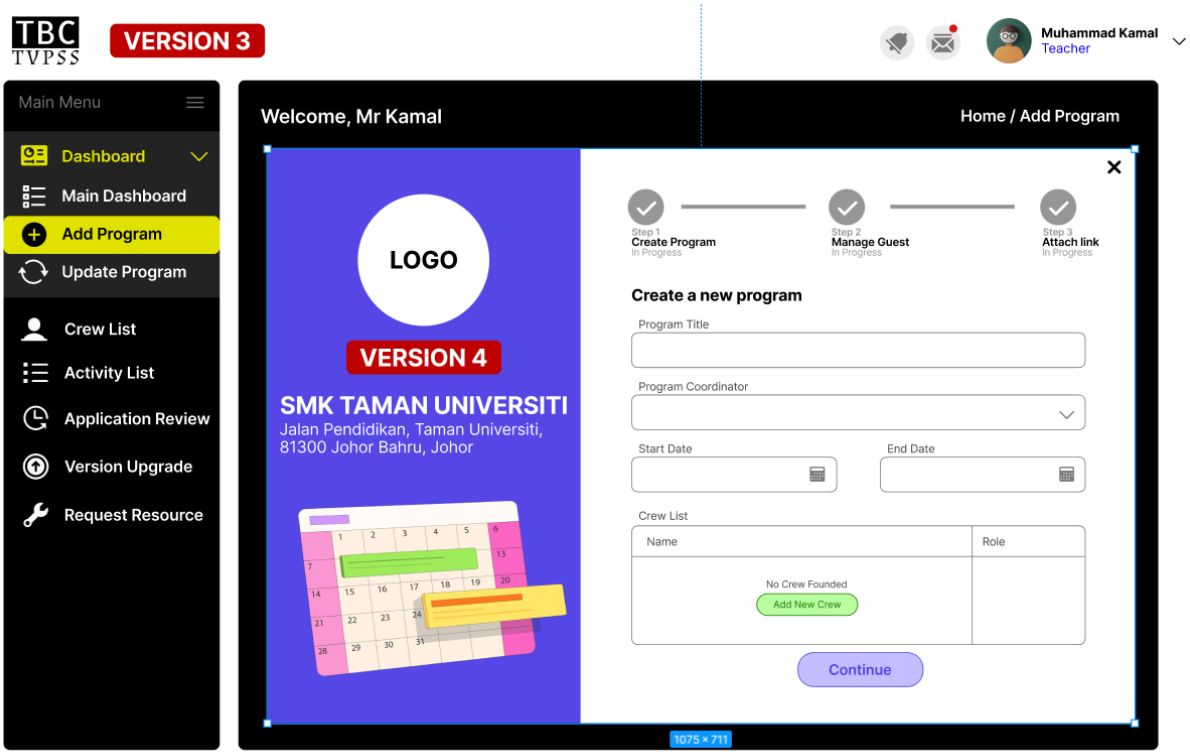


Figure 6.2.1.2 shows the User Interface for adding a program from teacher's role

6.2.1.3 UC011: Update Program Details

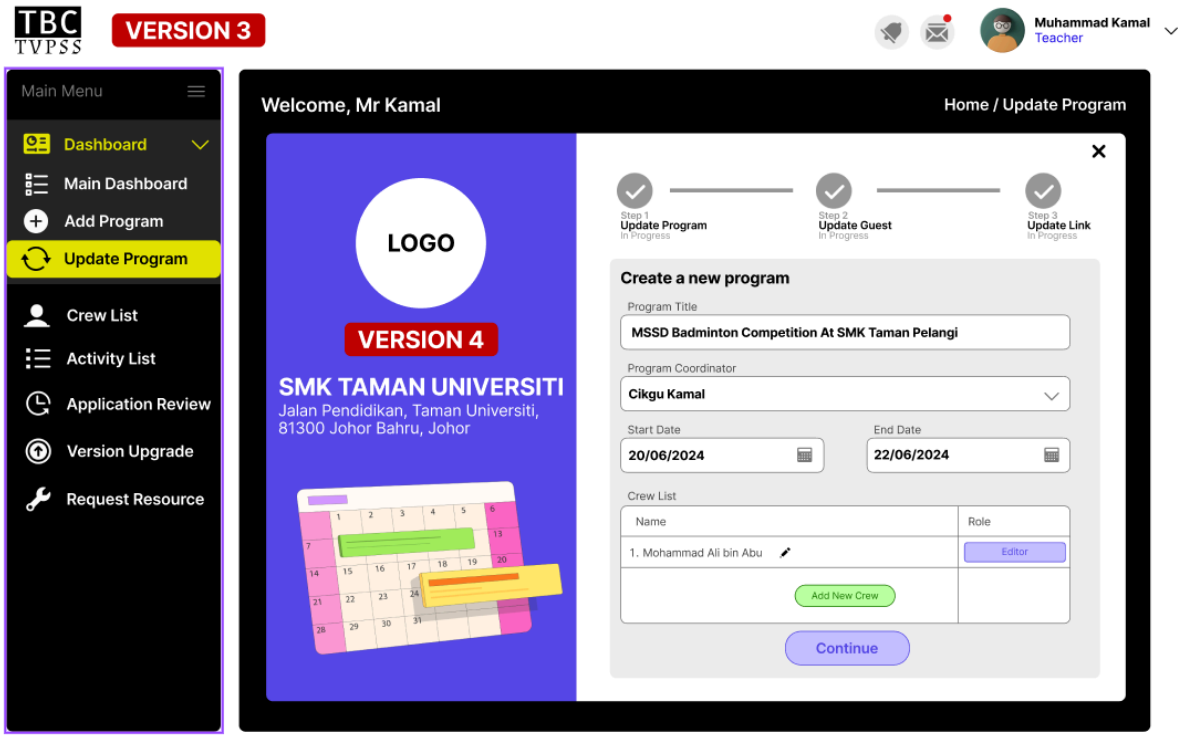


Figure 6.2.1.3 shows the User Interface for updating program details from teacher's role

6.2.1.4 UC003: Submit TVPSS Program Video

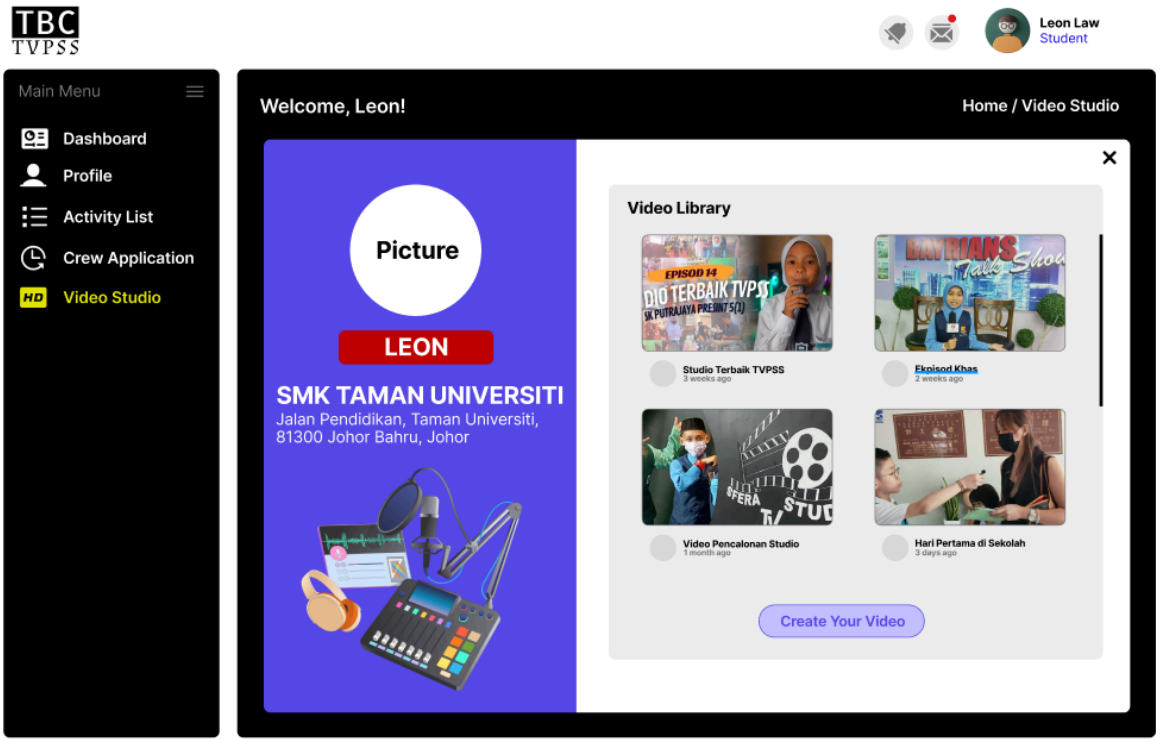


Figure 6.2.1.4 shows the User Interface for submitting program video from student's role

6.2.1.5 UC005: Apply to become TVPSS crew

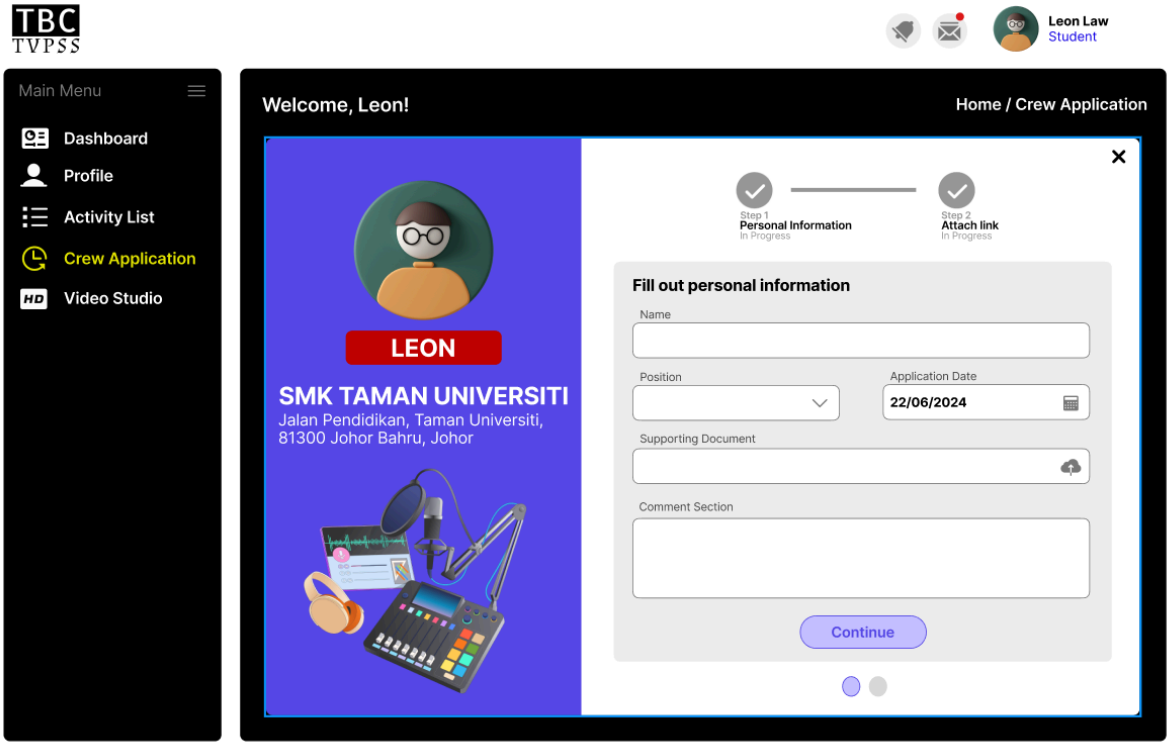


Figure 6.2.1.4 shows the User Interface for applying to become TVPSS crew from student's role

6.2.2 Module 2: Reporting

6.2.2.1 UC007: Manage e-certification

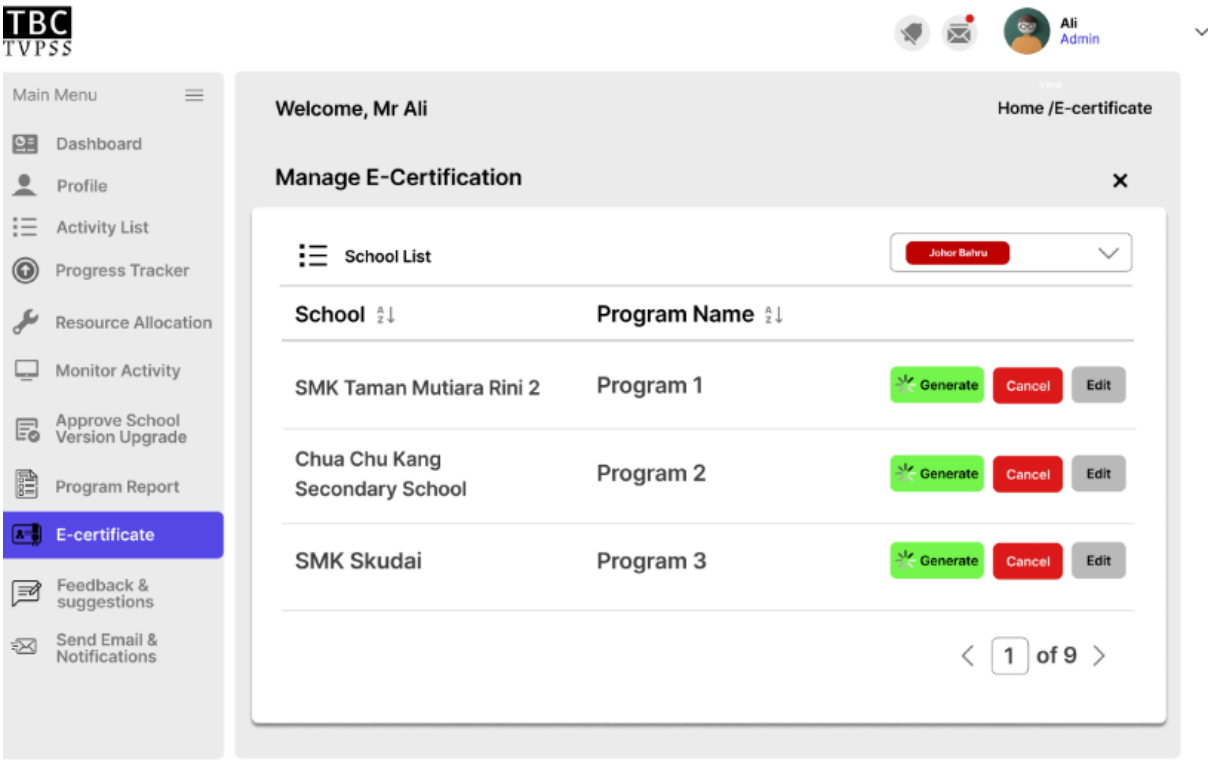


Figure 6.2.2.1 shows the User Interface for managing e-certificates from admin's role

6.2.2.2 UC015: Send email and notifications

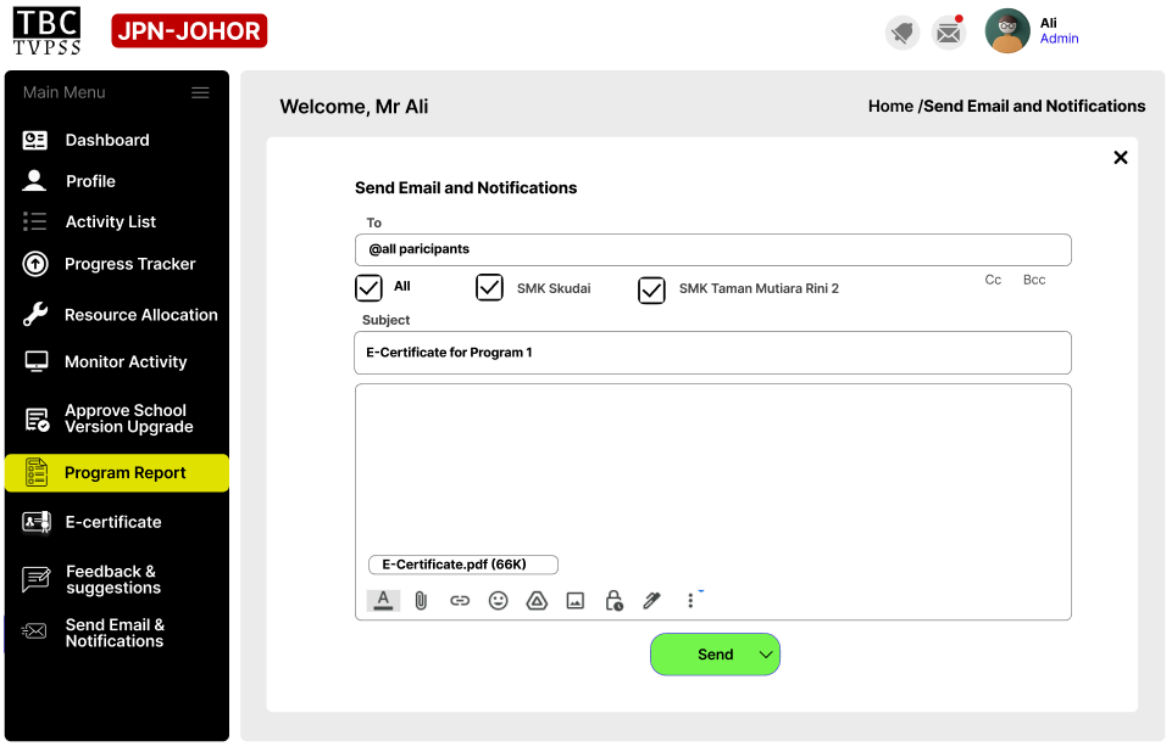


Figure 6.2.2.2 shows the User Interface for sending email and notifications from admin's role

6.2.2.3 UC006: Manage Program Report

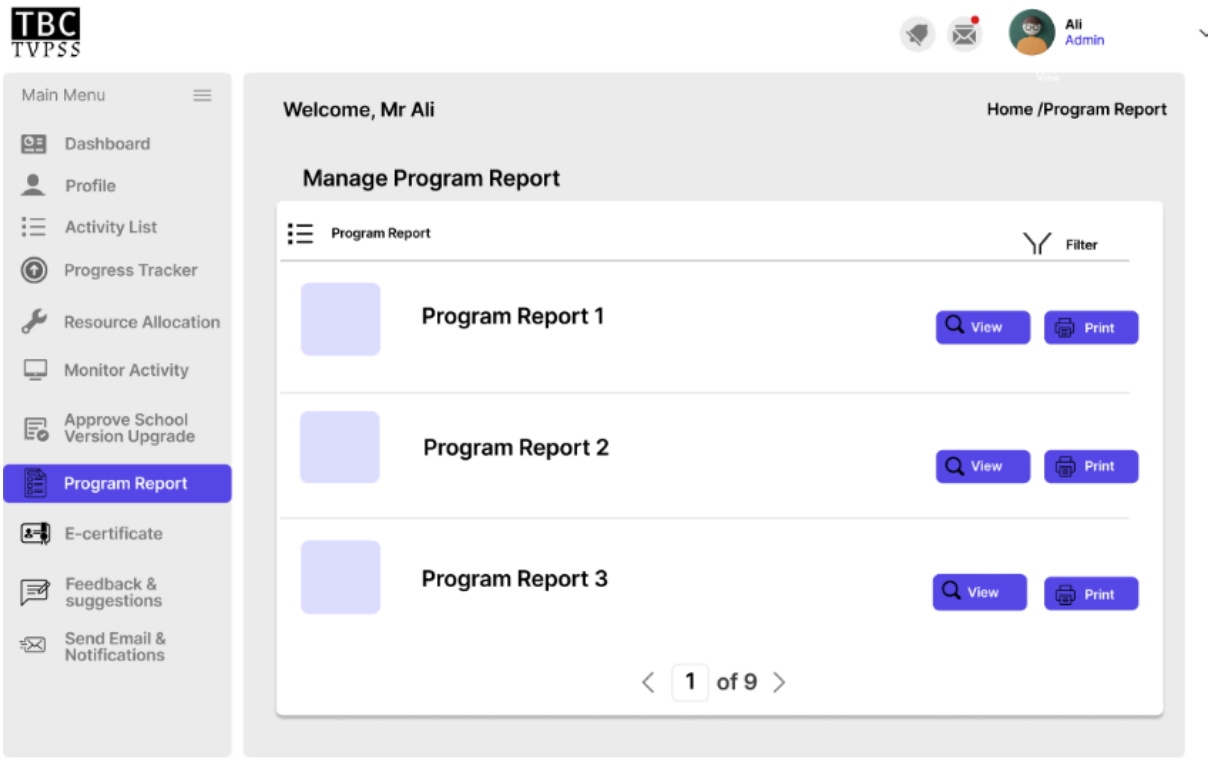


Figure 6.2.2.3 shows the User Interface for managing program reports from admin's role

6.2.2.4 UC012: Provide feedback

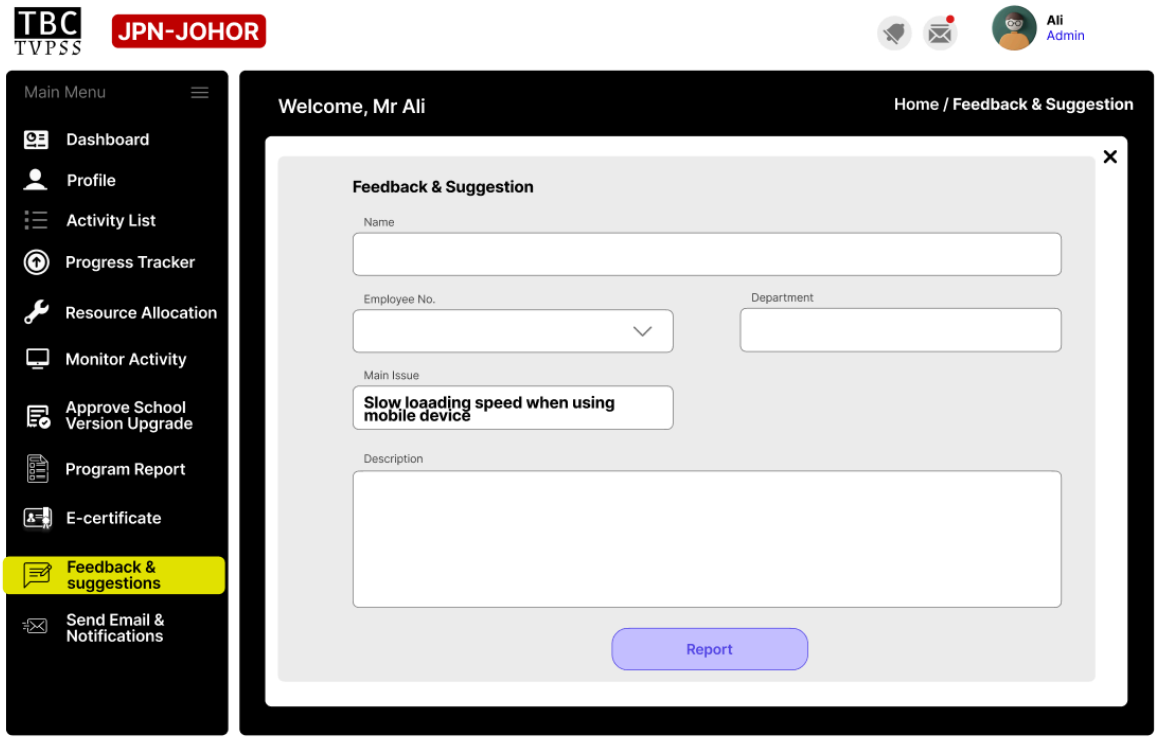


Figure 6.2.2.4 shows the User Interface for providing feedback from admin's role

6.2.3 Module 3: Resource and Equipment Management

6.2.3.1 UC004: Manage resources and equipment allocation

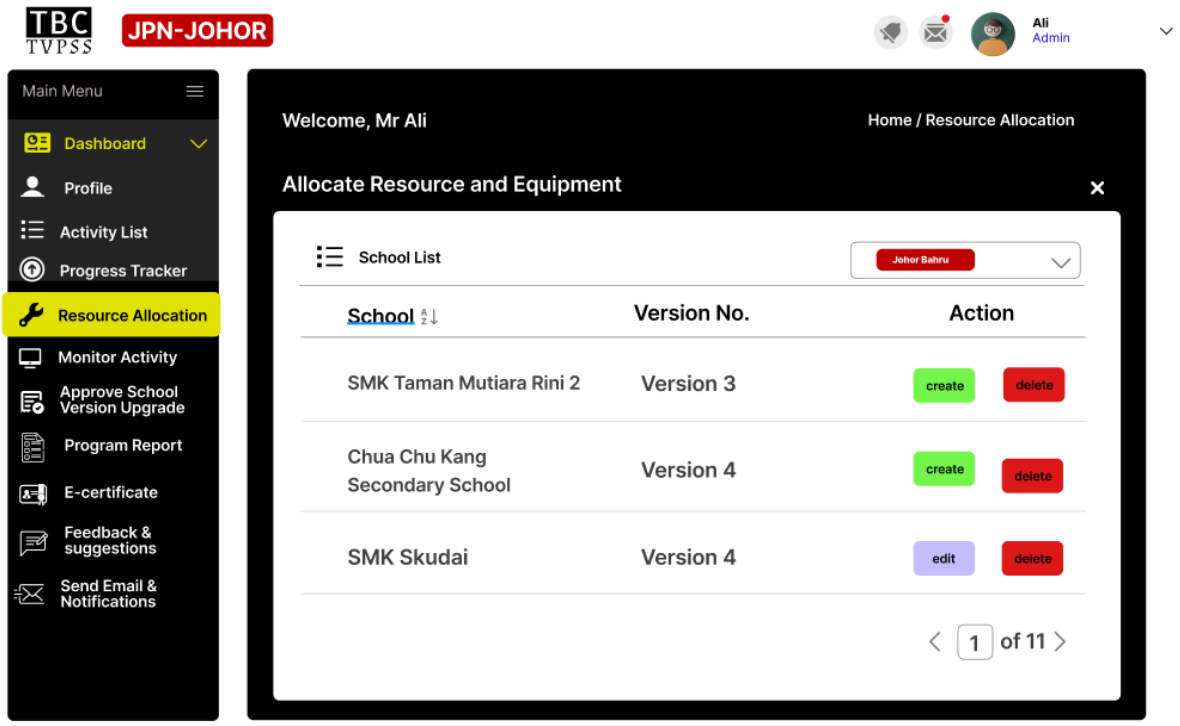


Figure 6.2.3.1 shows the User Interface for managing resources from admin's role

6.2.3.2 UC008: Request Resources and Requirements

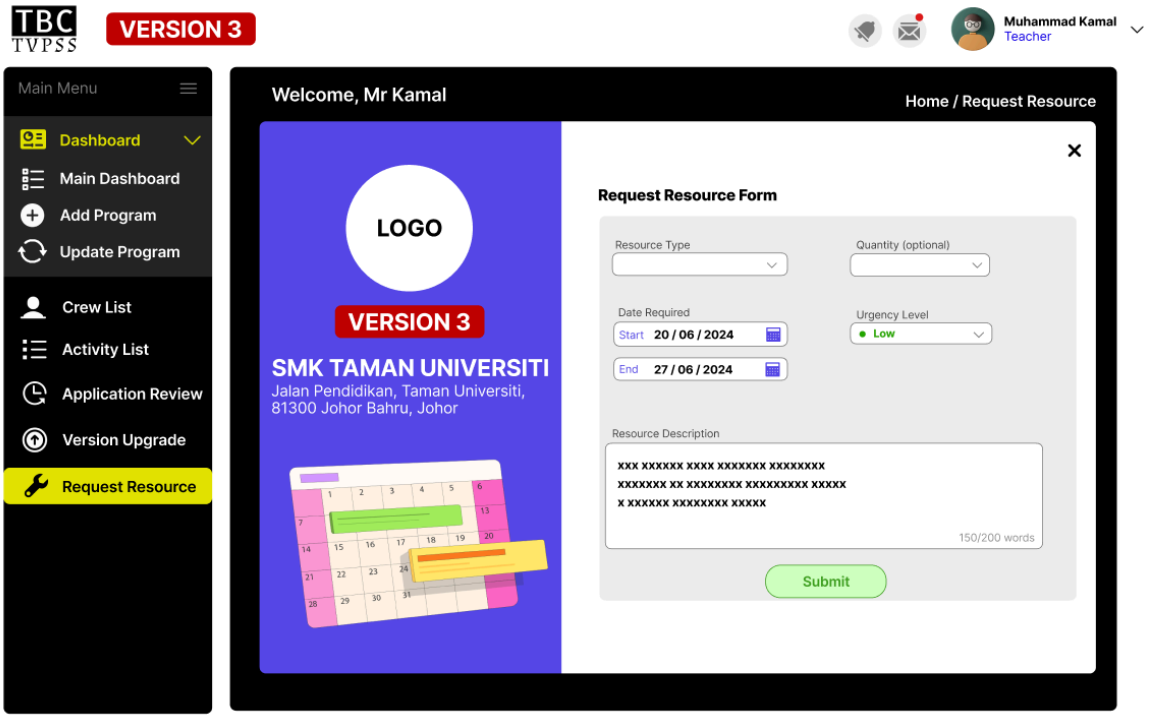


Figure 6.2.3.2 shows the User Interface for requesting resources from teacher's role

6.2.4 Module 4: Monitoring School Status

6.2.4.1 UC001: Track School Progress

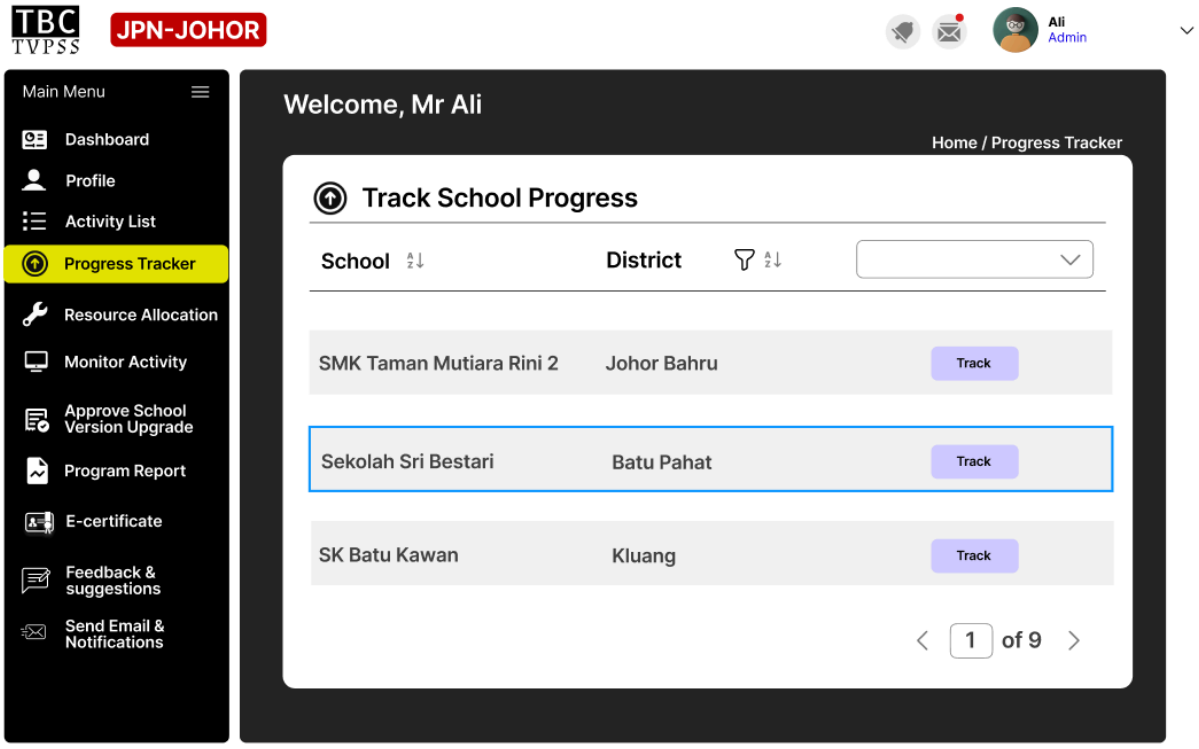


Figure 6.2.4.1 shows the User Interface for tracking school progress from admin's role

6.2.4.2 UC014: Monitor user activity

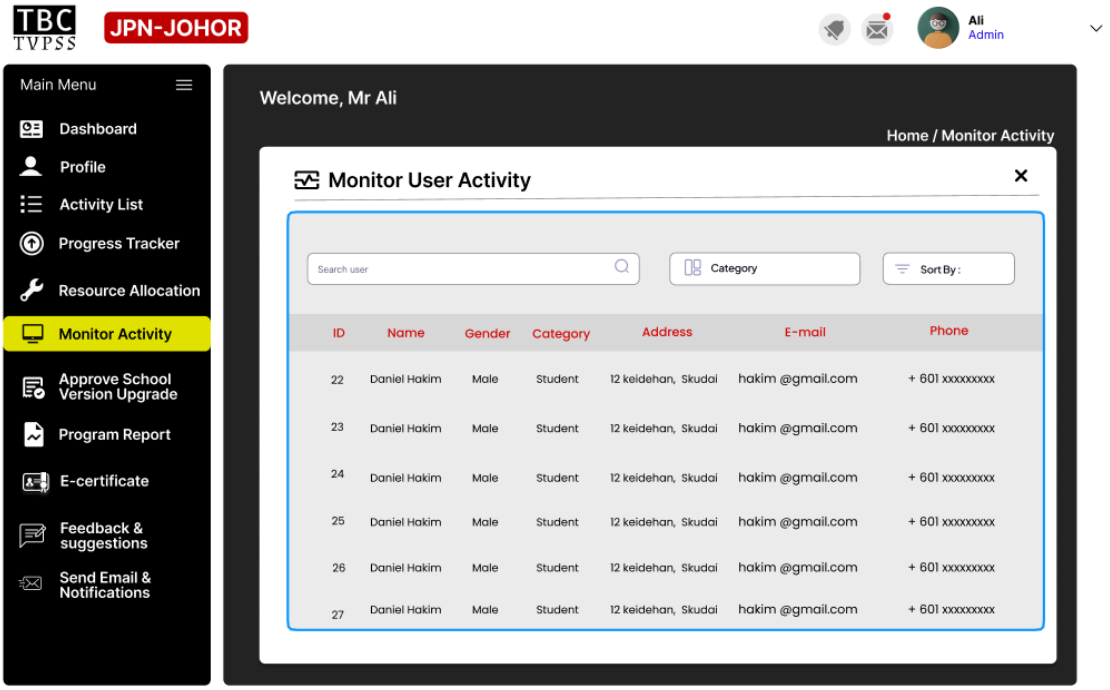


Figure 6.2.4.2 shows the User Interface for monitoring user activity from admin's role

6.2.4.3 UC013: Approve School Version Upgrade

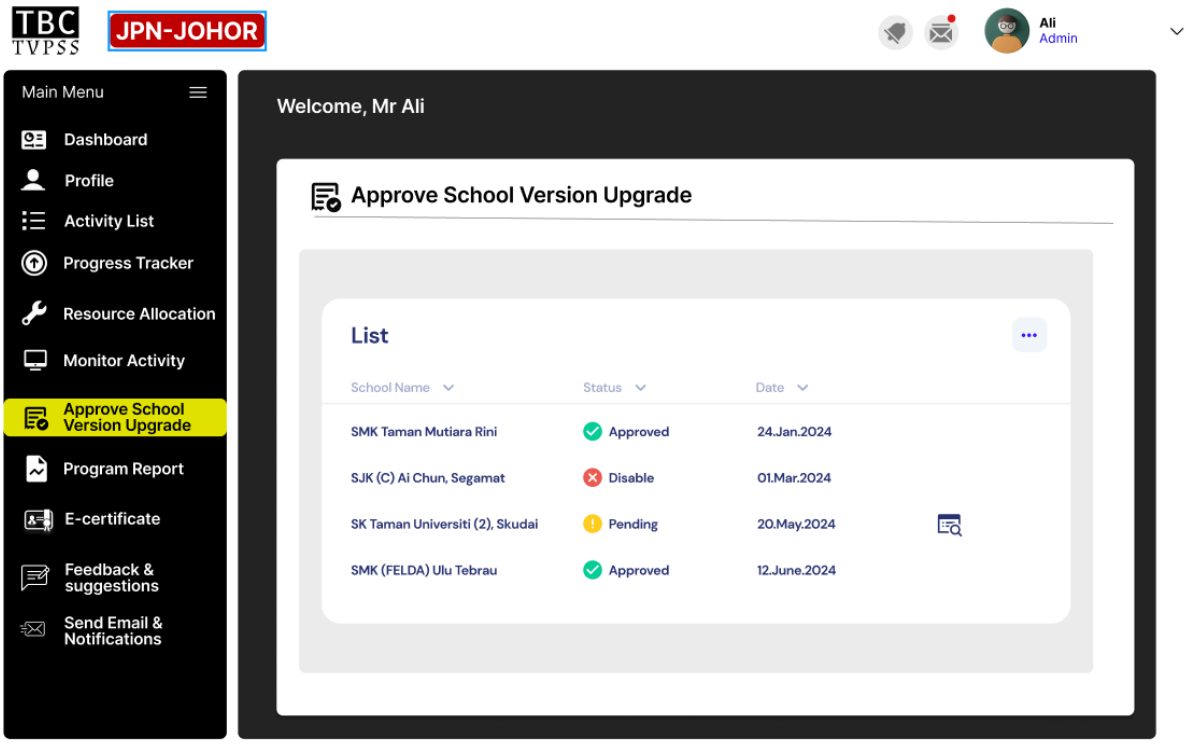


Figure 6.2.4.3 shows the User Interface for approving school version upgrade from admin's role

6.2.4.4 UC002: Request TVPSS Version Upgrade

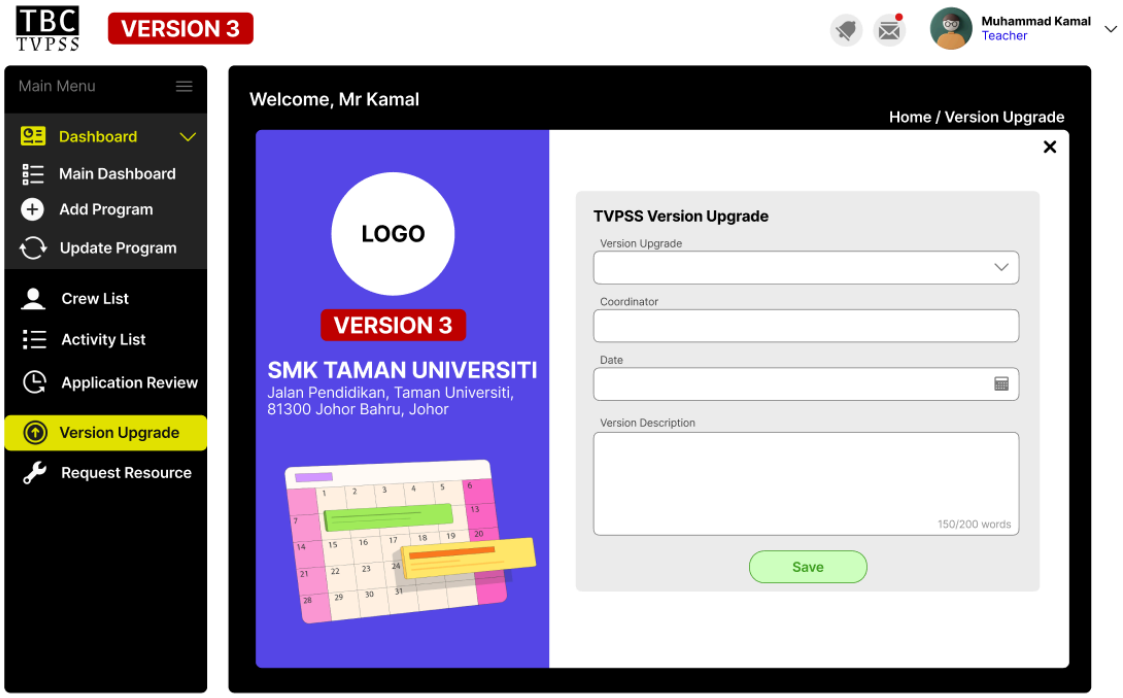


Figure 6.2.4.4 shows the User Interface for requesting TVPSS version upgrade from teacher's role

7. Appendices