SECP3204: Software Engineering WBL

System Requirements Specification (SRS)

MengajiOnetoOne Salary Management System

Version 1.0

09/06/2022

School of Computing, Faculty of Engineering

Prepared by: AppGuru

Table of Content

1. Introduction

1.1 Purpose

This system documentation will put a focus to explain the detail of the proposed salary management system. It will cover the details on the system purpose, system features. It will also explain how the system will work for each user type. Lastly, it will explain the constraints and conditions for the system to operate.

The intended audience of this document for both the client and the development team. It serves as a guideline for the development team to follow during the development phase. Then, it acts as a reference for the client to understand the development process and approach that will be taken by the development team.

1.2 Scope

In this project, the development team will develop a Salary management system that comes with some additional features for a local islamic education tutoring service provider, MengajiOnetoOne.com. The Salary management system is a web app that will come with a back-end relational database containing the data needed in the system.

The web salary management system is designed to assist the business admin to manage the teachers and students under their business. While it will help teachers to manage the class records and student feedback report. Teachers can insert class records and student feedback reports easily using the system. Besides, the admin will also have a more efficient way to manage and keep track of the salary payment for every teacher. Lastly, the students will be able to access their own feedback reports made by their teachers.

The goal of this system is to facilitate the business operation of our client, MenajiOnetoOne.com, It removes the need to use WhatsApp and Microsoft Word to keep track of their day-to-day business activity. It will provide a better UI and control for the admins, teachers and students to conduct their tasks. Therefore, they can finish their task in a more efficient and easier way. This system actually also expanded the potential of business as it allows our client to manage more teachers and clients. Last but not least, the database that comes with it will provide a more secure and reliable way to store and manage the data that the business owns.

1.3 Definitions, Acronyms and Abbreviation

Term	Descriptions
Client	MengajiOnetoOne.com
User	Refer to admins, teachers or students
Student feedback report	A document that contain detail descriptions of the student performance in his/her class
Invoice	A document that contain detail descriptions of the teacher's salary payment
CRUD	Create, Read, Update, Delete

1.4 Reference

https://www.slidesh55
are.net/KrishnasaiGudavalli/software-requirements-specification-17173967
https://www.chegg.com/homework-help/questions-and-answers/external-interface-requirements-1-user-interfaces-2-hardware-interfaces-3-software-interfa-q90133405
https://medium.com/trailblazer-of-salesforce/software-requirements-specification-srs-document-fd9ab103b18
https://nvlpubs.nist.gov/nistpubs/ams/NIST.AMS.300-2.pdf

1.5 Overview

The rest of the document will focus on the system requirement. The content can be separated into 3 main parts

1. The first part will cover the different interface requirements of the proposed system. For example, it will explain the user interfaces that is required to have in the system
2. The Second Part will include various diagrams like Domain Model diagram, Use case diagrams which depicts how the system should function. It will also include Sequence diagram which demonstrate the system workflow
3. The Last Part will explain the performance requirements, constraint in designing and also the attributes the software system needs to have

2. Specific Requirements

2.1 External Interface Requirements

2.1.1 User Interfaces

The user interface of the system will be developed using Bootstrap 5.0. The system is a mobile-first system, it will be responsive to different screen layouts or resolutions.

General

1. System users will first access a login page which requires users to keep in user ID and password
2. Login page will prompt users upon invalid user ID or password
3. Login page will provide “Remember user ID” and “Forgot Password” option
4. After logging, users will be presented a landing page that lists out various modules/functionalities that users have access to.
5. The landing page will also let every type of users to access “manage profile”, “manage password” and “logout” option
6. Whenever users make editing or deletion, the system will display a prompt message to confirm users’ action

Admin

1. Admin Landing page will let admins access “manage users”, “view class records”, “view student feedback report” and “manage teacher’s salary” modules.
2. For every module Admin access, the system will present a page that lists out every existing record that the module manages.
3. For “manage users” and “manage teacher’s salary” module, the page will have “create” buttons for admin to click and create new records
4. For “manage users” and “manage teacher’s salary” modules, every record will have a “EDIT”.
5. Only for “manage users” module, the page will have “DELETE” buttons at the end of rows

Teacher

1. The Teacher Landing page will let teachers access “manage class records”, “manage student feedback report” and “view teacher’s salary” modules.
2. For every module Teacher access, the system will present a page that lists out every existing record that the module manages.
3. For “manage class record” and “manage student feedback report” module, the page will have “create” buttons for Teacher to click and create new records
4. For “manage class record” and “manage student feedback report” module, every records will have a “EDIT” and “DELETE” buttons at the end of rows

Student

1. The Student Landing page will let students access the “view student feedback report” module.
2. This module allows students to view their student feedback report made by the teacher.

2.1.2 Hardware Interfaces

All server-side functions will be executed on the server. While for the client-side components, it will be executed on personal computers with display monitors or mobile phones through the browser.

2.1.3 Software Interfaces

For the front-end, the system will be developed using Bootstrap 5.0. While for back-end, since the system will be developed using .NET MVC 5.0, therefore the software interface should follow the Model-View-Controller(Model) for fetching and modelling data needed. For the data, the software interface will also connect to a SQL Server relational database. The Relation database will store data of users, class records, student’s feedback and also salary payment records. Lastly, the system will run on browsers. The recommended browsers for the system are Google Chrome, Safari and Microsoft Edge.

Bootstrap 5.0	***Mnemonic***	The front-end development tool of the system
	Specification Number	-
	Version Number	5.0
	Source	https://getbootstrap.com/
.NET MVC 5.0	***Mnemonic***	The back-end development tool of the system
	Specification Number	-
	Version Number	5.0
	Source	https://dotnet.microsoft.com/en-us/apps/aspnet/mvc
SQL Server	***Mnemonic***	Relational database to store and model data object
	Specification Number	15.0.18410.0
	Version Number	18.11.1
	Source	https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15
Browsers	***Recommended***	Google Chrome, Safari, Microsoft Edge

2.1.4 Communication Interfaces

The system will be a web-based system, therefore the data fetching process from the database must be done through HTTPS protocol. Besides, inside the system, different pages will use the .NET MVC session variables to communicate with each other.

2.2 System Features

Use Case Diagrams

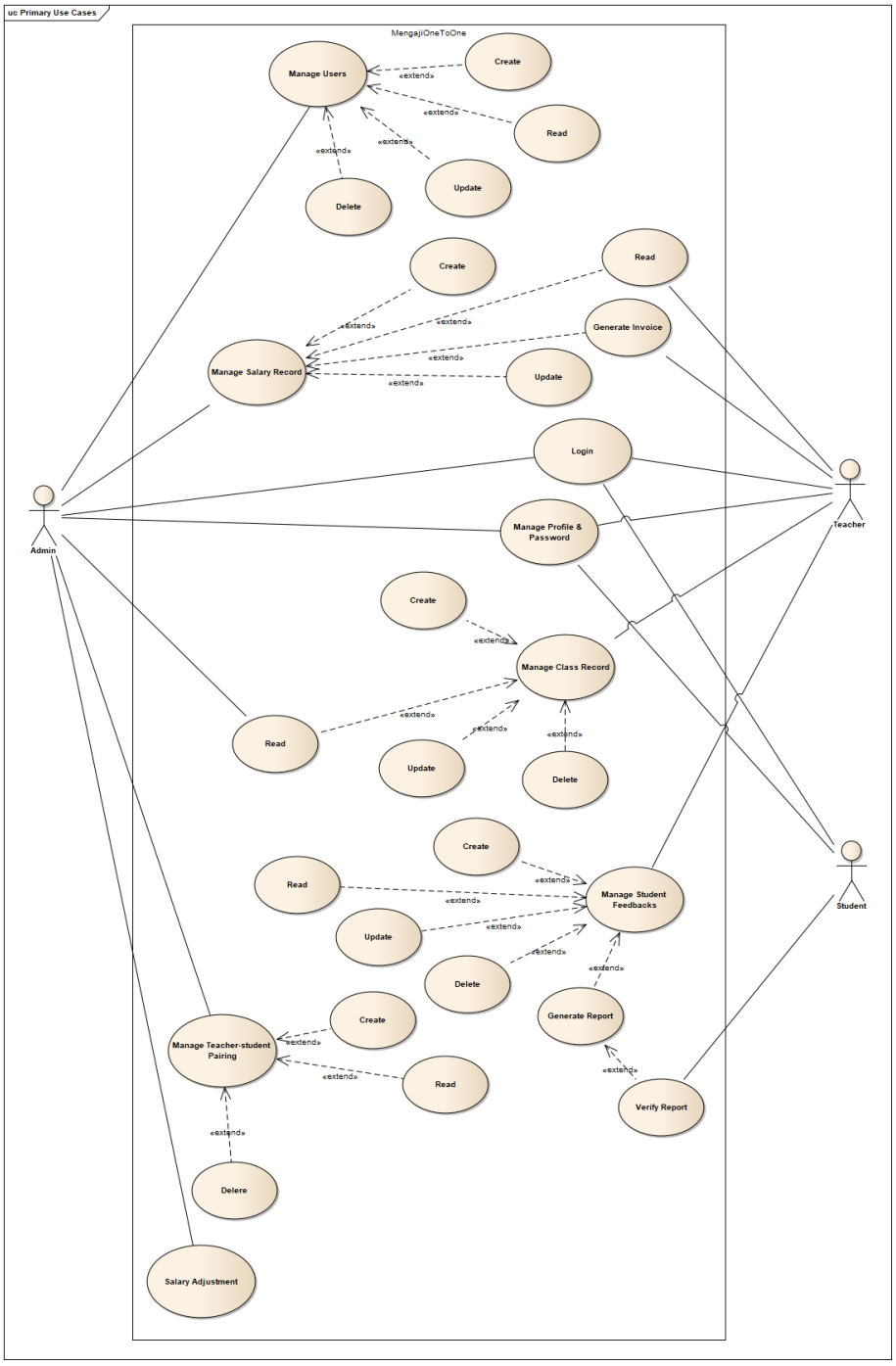

Figure 2.1: Use Case Diagram for MengajiOnetoOne Salary Management System

Activity Diagram

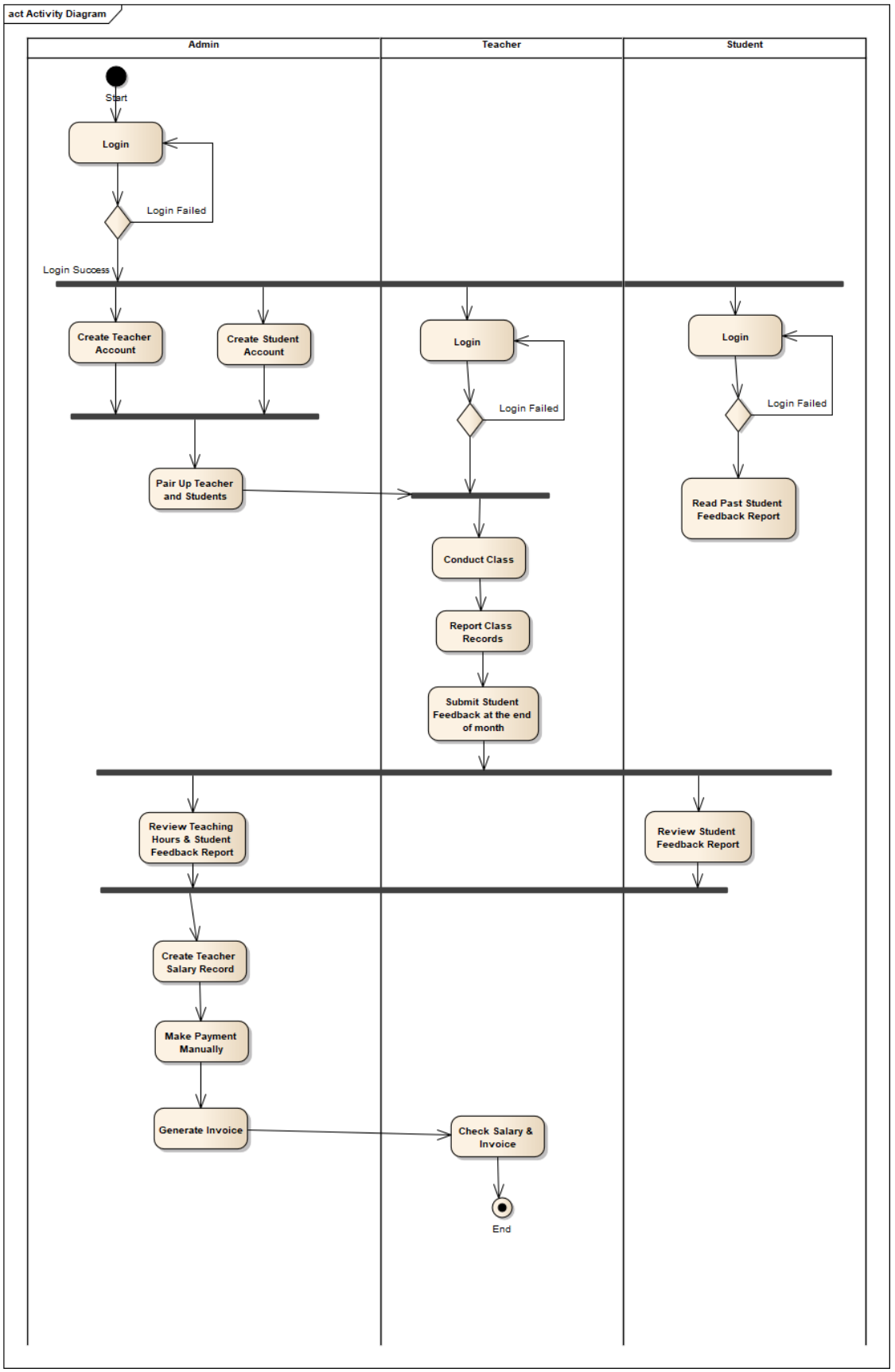

Figure 2.2: Activity Diagram for MengajiOnetoOne Salary Management System

Domain Model Diagram

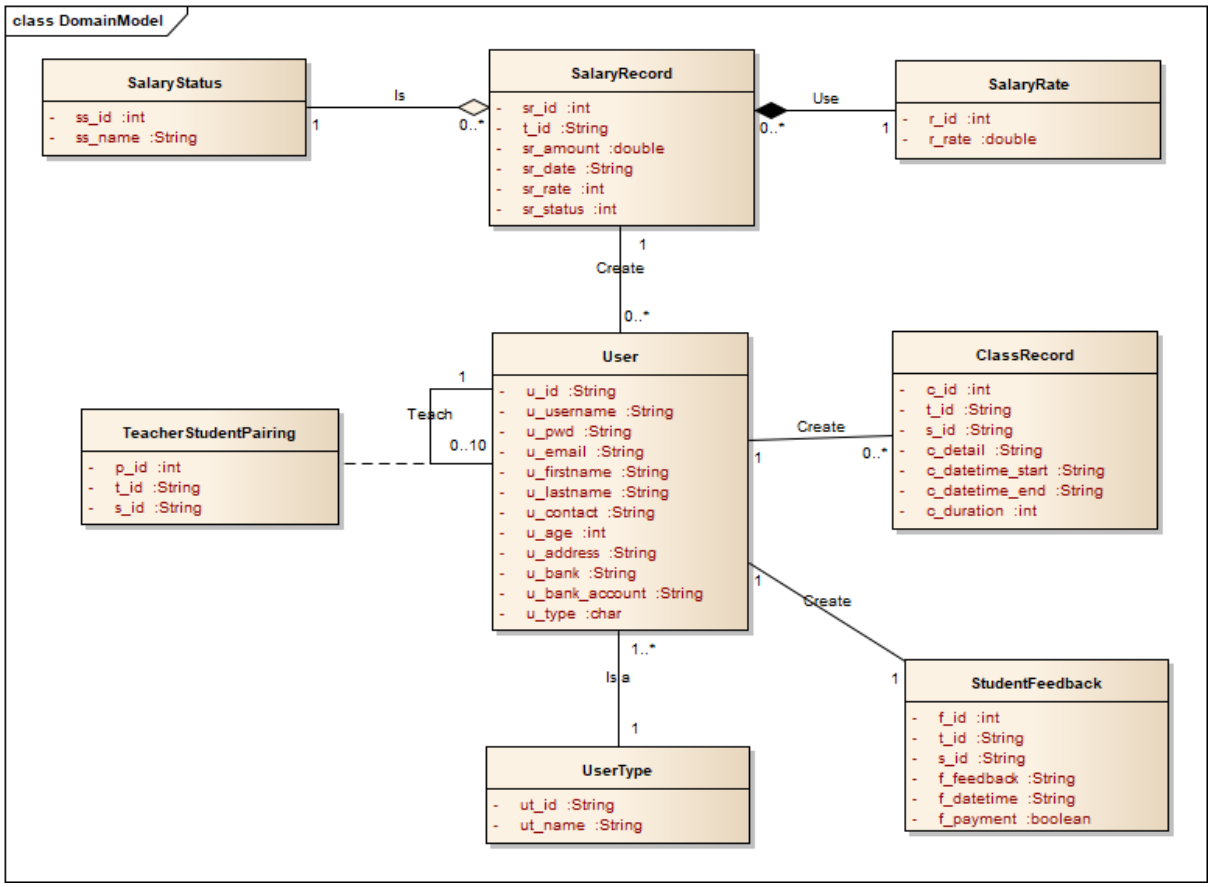

Figure 2.3: Domain Model for MengajiOnetoOne Salary Management System

State Machine Diagrams

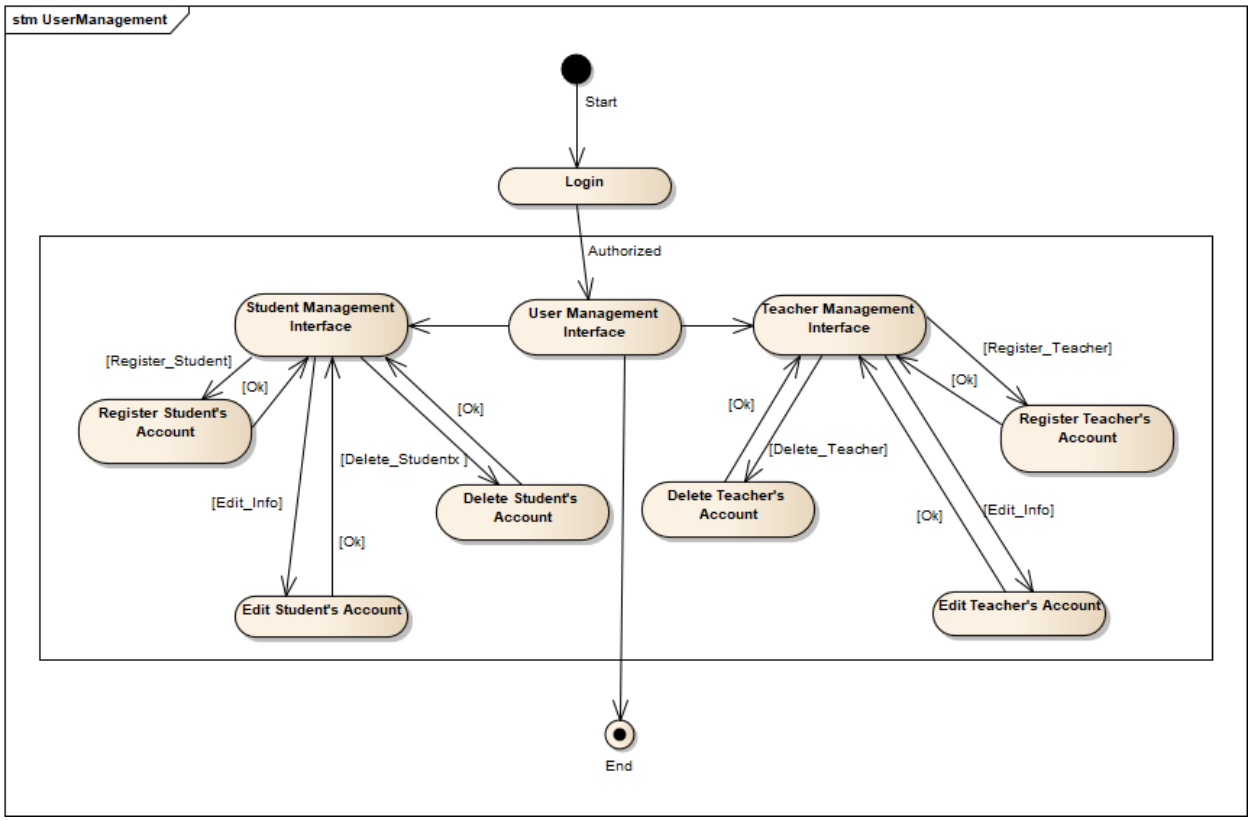

Figure 2.4: State Machine Diagram for User Class

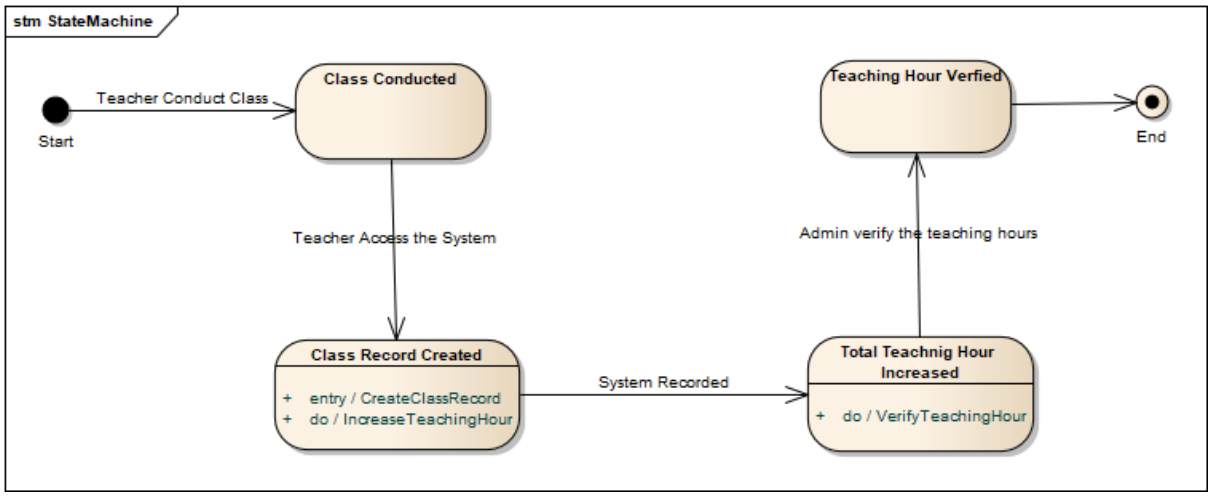

Figure 2.5: State Machine Diagram for Class Record Class

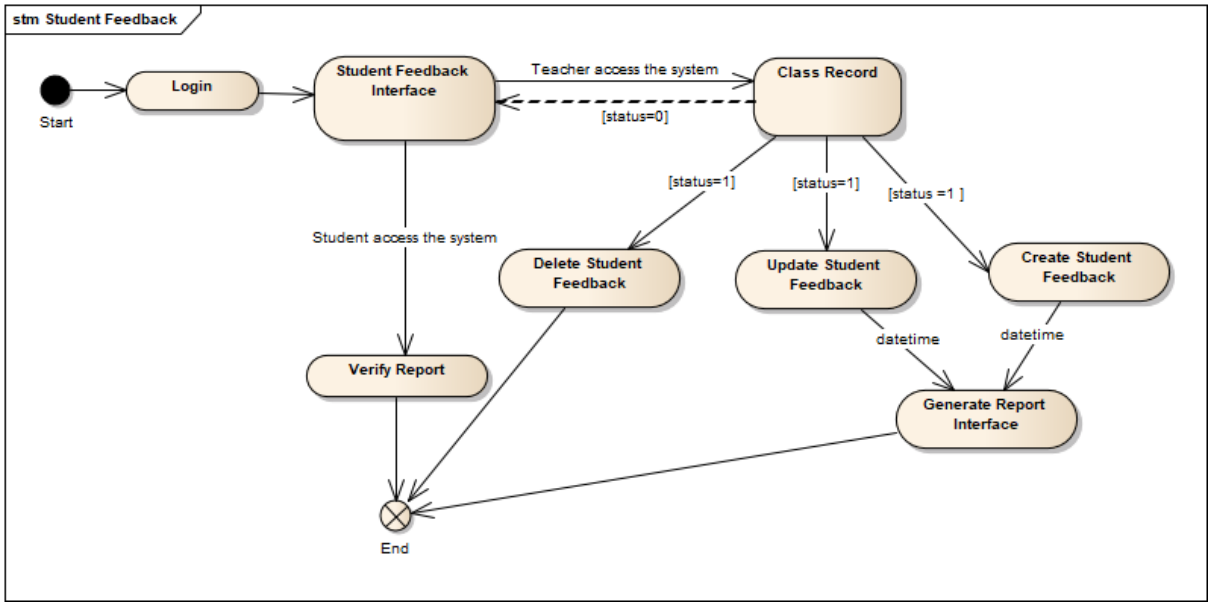

Figure 2.6: State Machine Diagram for Student Feedback Class.

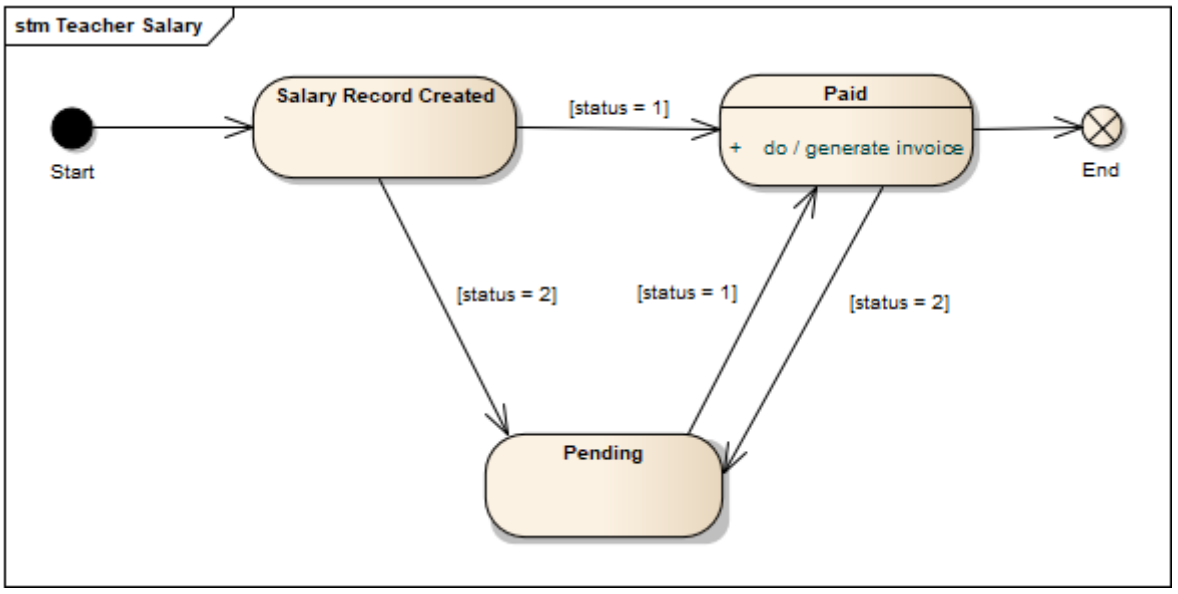

Figure 2.7: State Machine Diagram for Salary Record Class

2.2.1 UC001: Use Case <Create User>

Use case: Create New Users
ID: UC001
Actors: Admin
Preconditions: 1. A valid admin is logged on to the system.
Flow of events: 1. Admin selects the “Manage User” module from the navbar. It will be dropdowned another 2 options 2. If admin selects “Teacher” a. It will go to the teacher management page. b. Admin select the “Create” option to create a new account for teachers. 3. Else admin selects “Students” a. It will go to the student management page. b. Admin select the “Create” option to create a new account for students. 4. Admin save the account creation.
Postconditions: The new account info has been updated and saved.
Alternative flow *1*: Admin can leave the user creation screen as long as not press the save button.
Postconditions: Changes discarded after a few minutes.

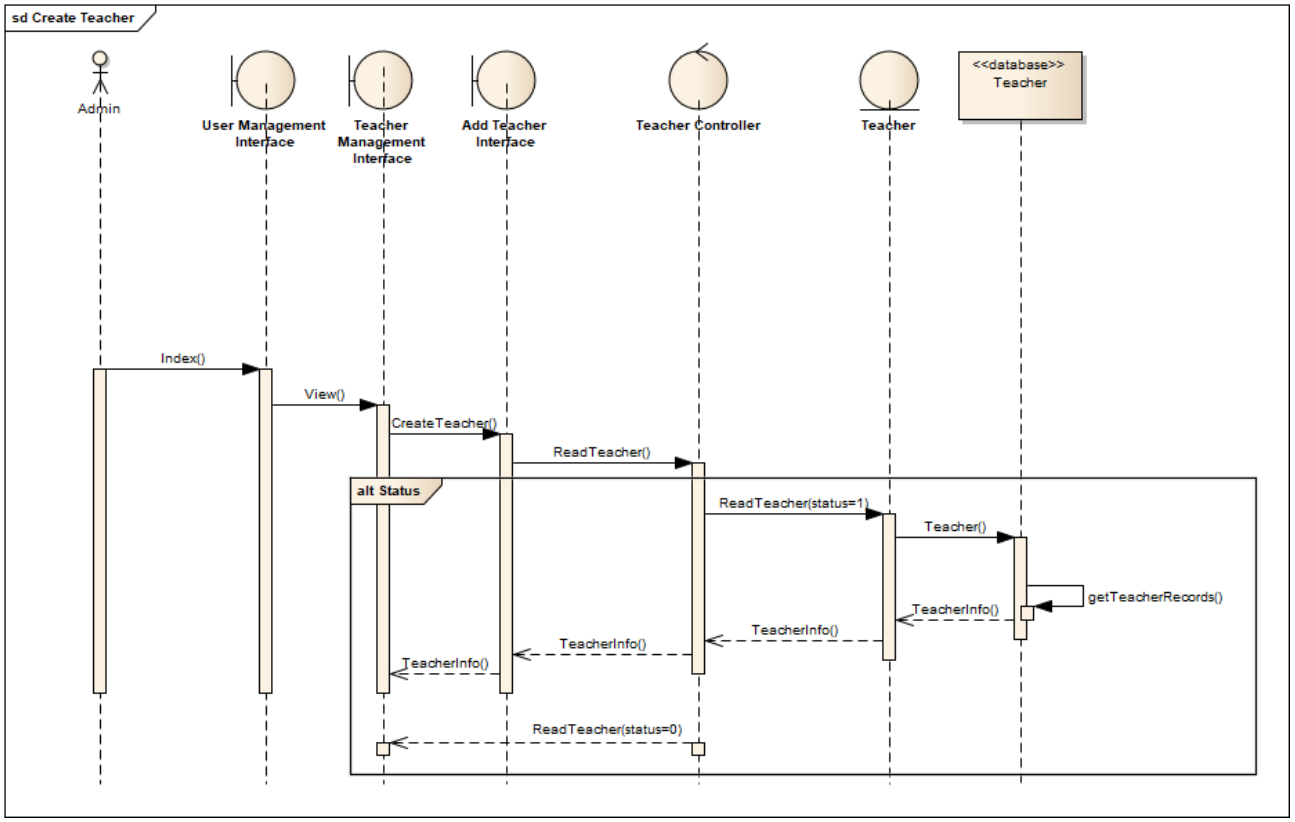

Figure 2.8: Sequence Diagram for Create User (Teacher)

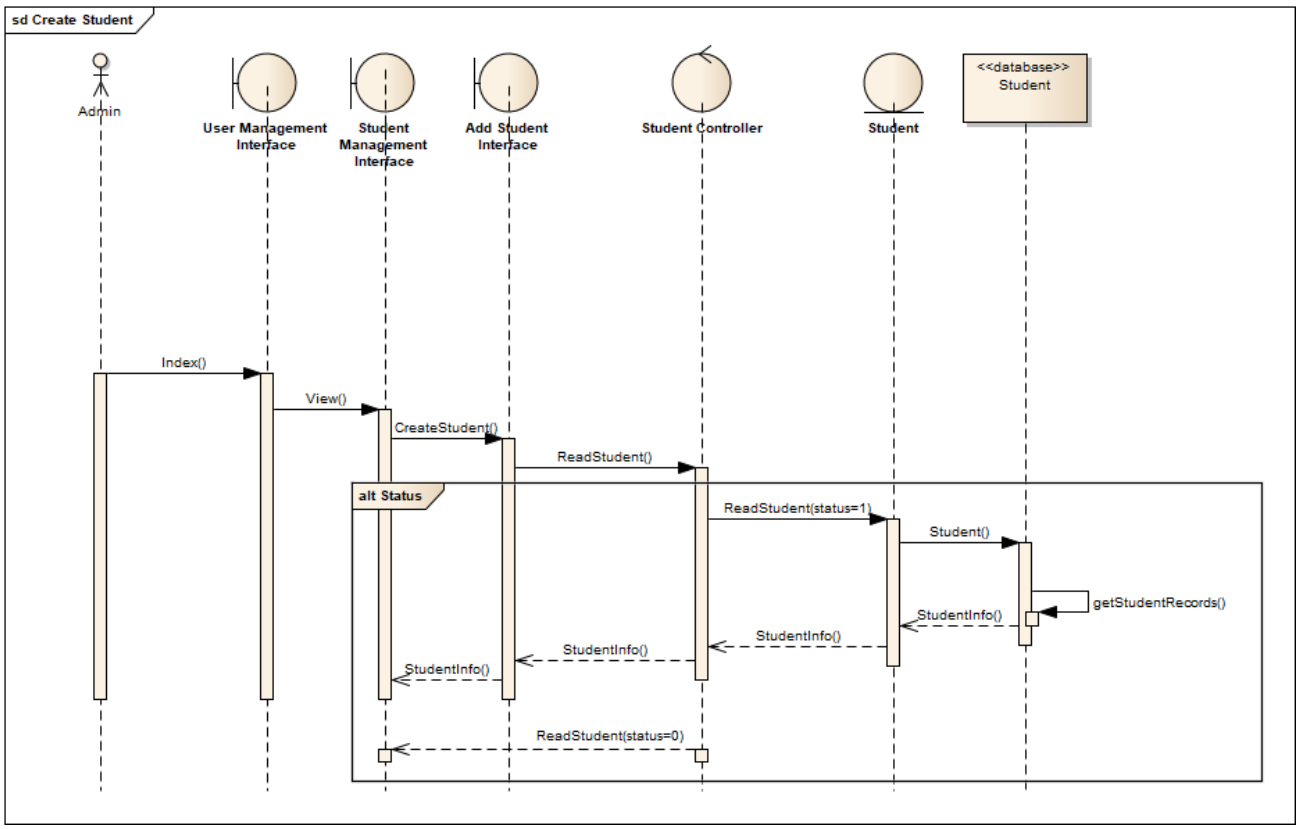

Figure 2.9: Sequence Diagram for Create User (Student)

2.2.2 UC002: Use Case <Update and View User Profile>

Use case: Update and View User's Profile
ID: UC002
Actors: Admin
Preconditions: 1. A valid admin is logged on to the system.
Flow of events: 1. Admin selects the "Manage User" module from the navbar. It will be dropdowned another 2 options 2. If admin selects "Teacher" a. It will go to the teacher management page. b. Admin select the teachers from the list or search their id c. Admin view teacher's profile d. Admin click on "Edit" to update the teacher's profile 3. Else admin selects "Students" a. It will go to the student management page. b. Admin select the students from the list or search their id c. Admin view student's profile d. Admin click on "Edit" to update the student's profile 4. Admin save the account update.
Postconditions: The edited account info has been updated and saved.
Alternative flow *1*: Admin can leave the screen as long as not press the save button.
Postconditions: Changes discarded after a few minutes.

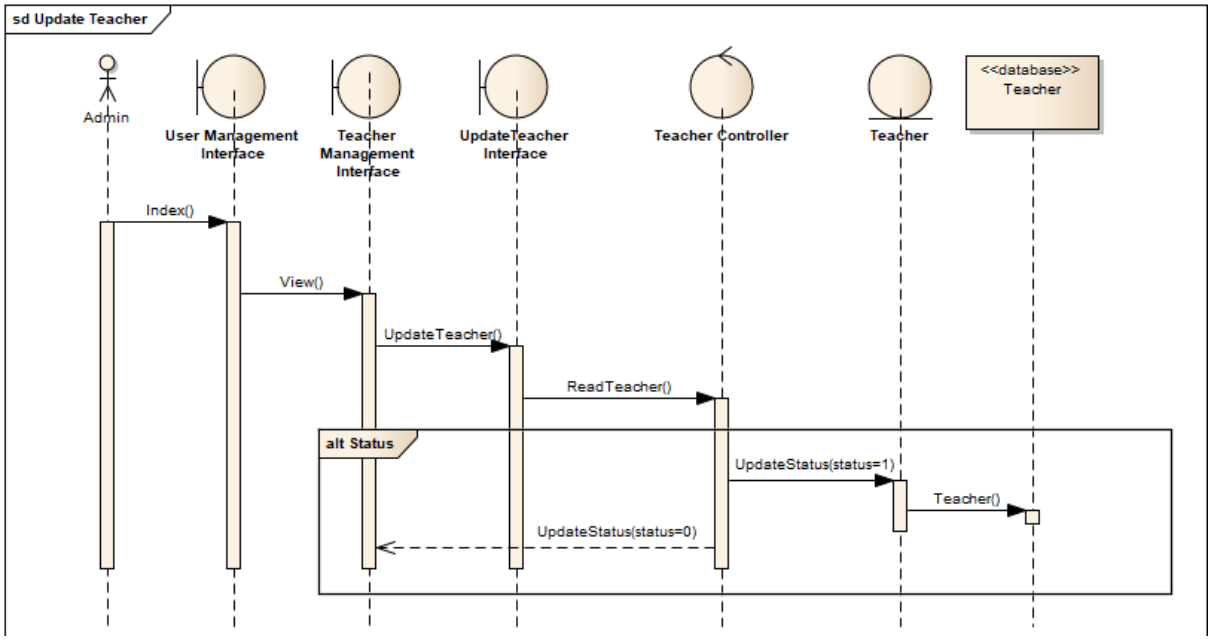

Figure 2.10: Sequence Diagram for Update User (Teacher)

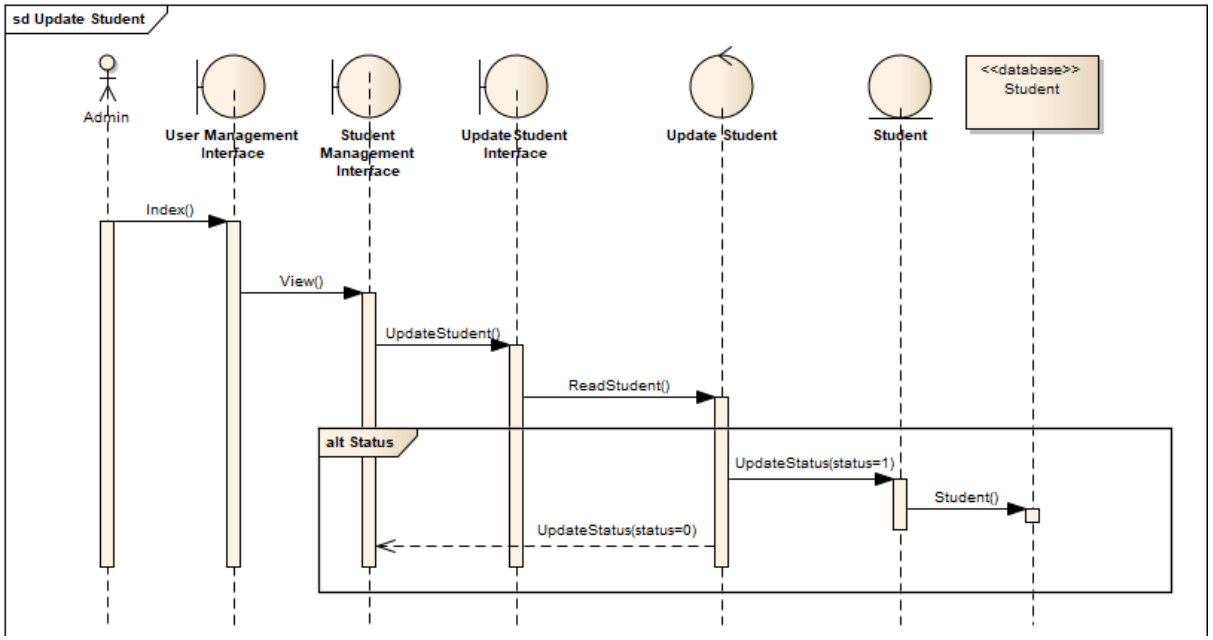

Figure 2.11: Sequence Diagram for Update User (Student)

2.2.3 UC003: Use Case <Delete User>

Use case: Delete Users
ID: UC003
Actors: Admin
Preconditions: 1. A valid admin is logged on to the system.
Flow of events: 1. Admin selects the “Manage User” module from the navbar. It will be dropdowned another 2 options 2. If admin selects “Teacher” a. It will go to the teacher management page. b. Admin select the teachers from the list or search their id c. Admin select the “Delete” option to delete the teachers’ account 3. Else admin selects “Students” a. It will go to the student management page. b. Admin select the students from the list or search their id c. Admin select the “Delete” option to delete the students’ account 4. Admin save the account deletion.
Postconditions: The deleted account info has been updated and saved.
Alternative flow *1*: Admin can leave the user deletion screen as long as not press the save button.
Postconditions: Changes discarded after a few minutes.

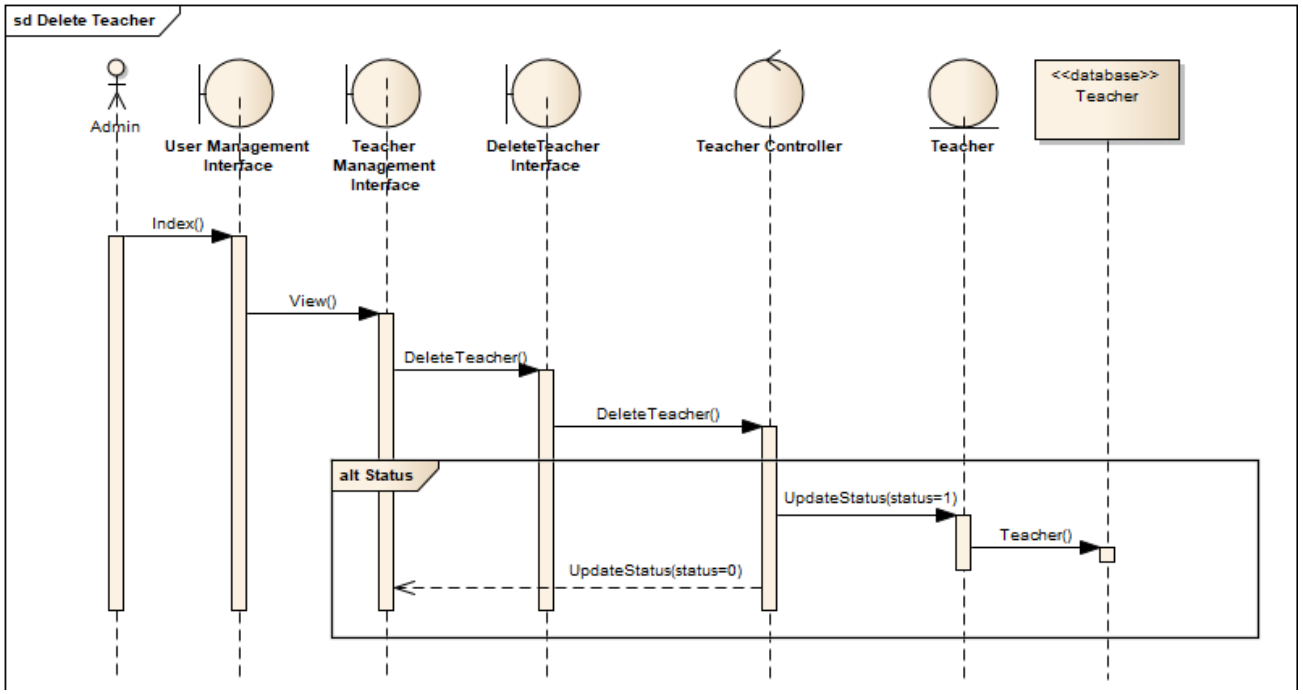

Figure 2.12: Sequence Diagram for Delete User (Teacher)

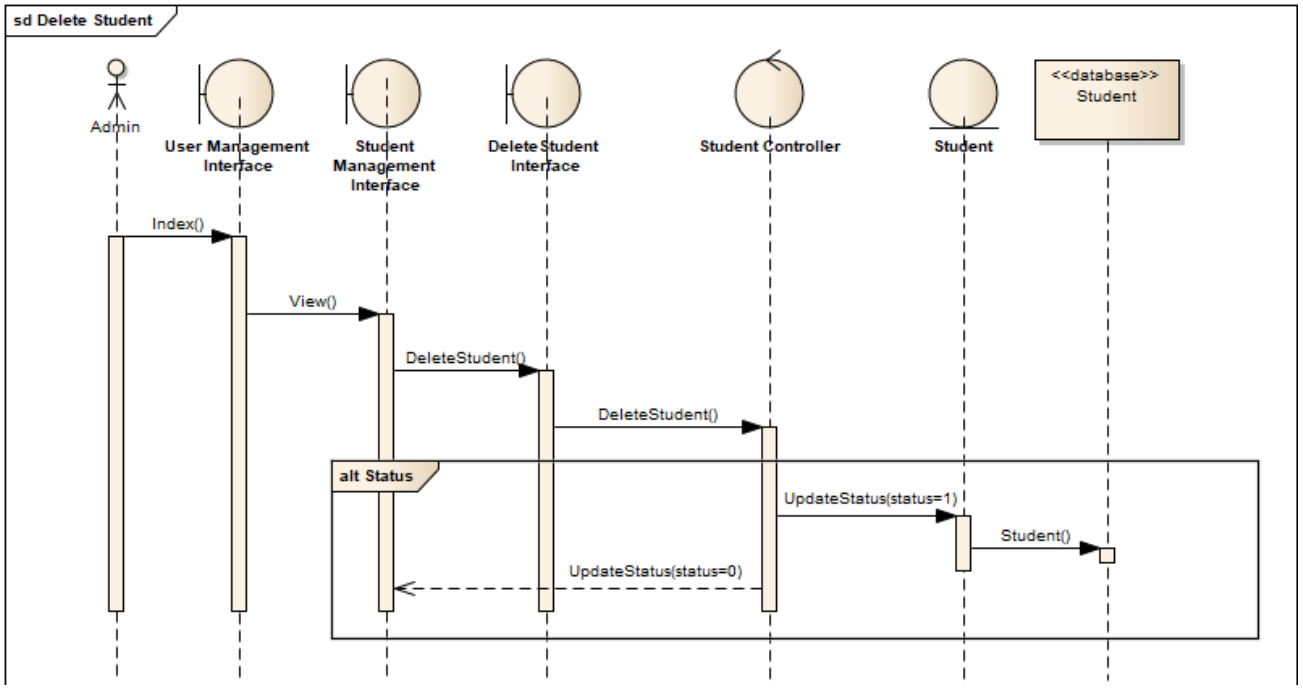

Figure 2.13: Sequence Diagram for Delete User (Student)

2.2.4 UC004: Use Case <Salary Adjustment>

Use case: Salary Adjustment
ID: UC004
Actors: Admin
Preconditions: 1. A valid admin is logged on to the system.
Flow of events: 1. Admin selects the “Salary Adjustment” button in the Salary Record page.. 2. Admin insert new salary rate in the input field. 3. Admin click the “Update” button.
Postconditions: The system updates the salary rate in the database.
Alternative flow *1*: At any point, the admin may leave the salary adjustment page without clicking on the “Update” button.
Postconditions: The system will not update the salary rate in the database.

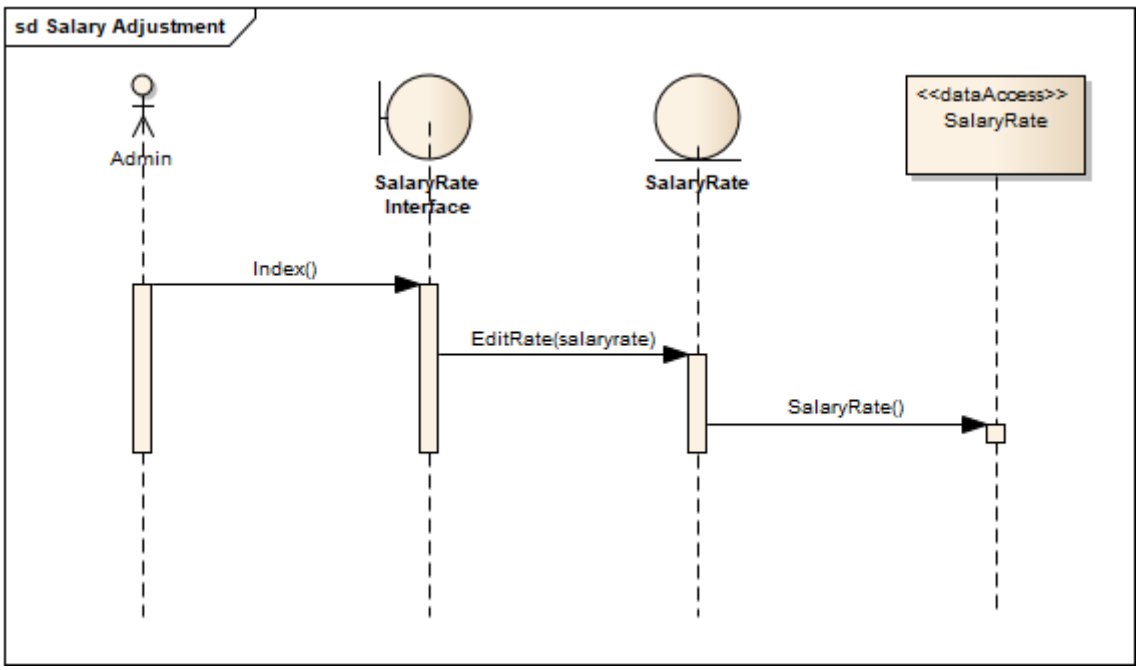

Figure 2.14: Sequence Diagram for Salary Adjustment

2.2.5 UC005: Use Case <Create Salary Record>

Use case: Create Salary Record
ID: UC005
Actors: Admin
Preconditions: 1. A valid admin is logged on to the system. 2. Teacher's teaching hour is verified.
Flow of events: 1. Admin selects the "Salary Payment" page from the navbar. 2. The system displays the list of salary records. 3. Admin selects the row to make payment. 4. Admin copies the teacher's banking number and makes salary payment manually. 5. Admin clicks the "Payment Made" button. 6. System generates an invoice.
Postconditions: Once the admin clicks on the "Payment Made" button, the status of salary payment is changed to "Paid" or "1" and the date is the current date and time.

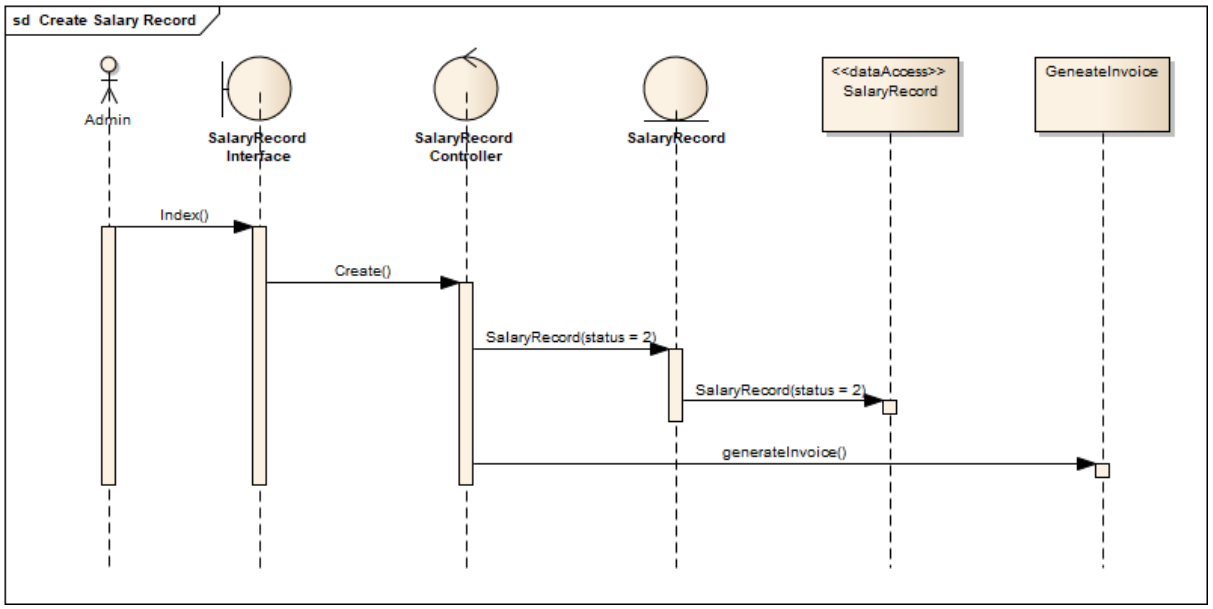

Figure 2.15: Sequence Diagram for Create Salary Record

2.2.6 UC006 Use Case <Read Salary Record>

Use case: Read Salary Record
ID: UC006
Actors: Admin, Teacher
Preconditions: 1. A valid admin is logged on to the system. 2. A valid teacher is logged on to the system. 3. Teacher's teaching hour is verified.
Flow of events: 1. Admin or teacher selects the "Salary Payment" page from the navbar. 2. The system displays the list of salary records. 3. Admin or teacher selects the row to read. 4. The system displays the details of the salary record. 5. Admin or teacher can click on the "Download Invoice" to download the invoice of the salary payment.

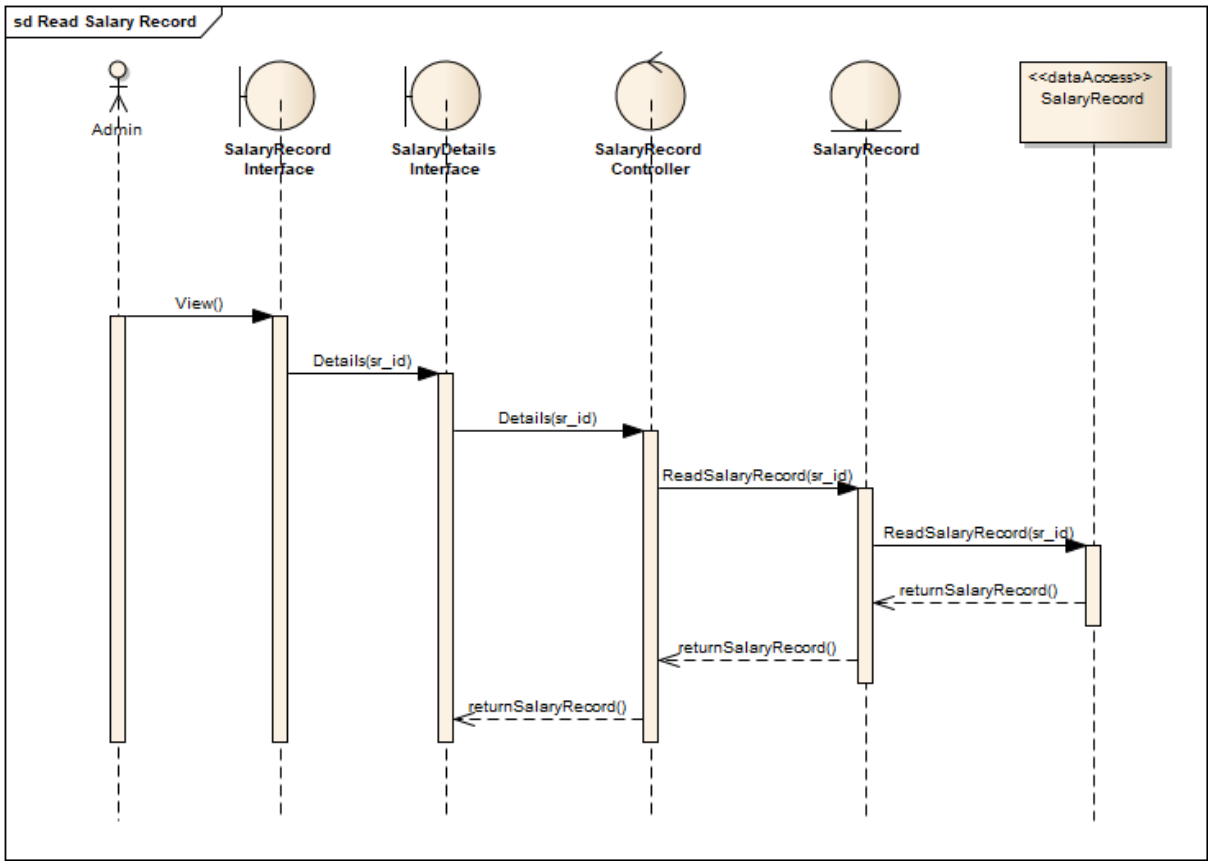

Figure 2.16: Sequence Diagram for Read Salary Record

2.2.7 UC007 Use Case <Update Salary Record>

Use case: Update Salary Record
ID: UC007
Actors: Admin
Preconditions: 1. A valid admin is logged on to the system. 2. Teacher's teaching hour is verified.
Flow of events: 1. Admin selects the "Salary Payment" page from the navbar. 2. The system displays the list of salary records. 3. Admin selects the row to update. 4. The system displays the details of the salary record. 5. Admin selects the "Unpaid/Pending" button to update the status from "Paid" to "Pending".

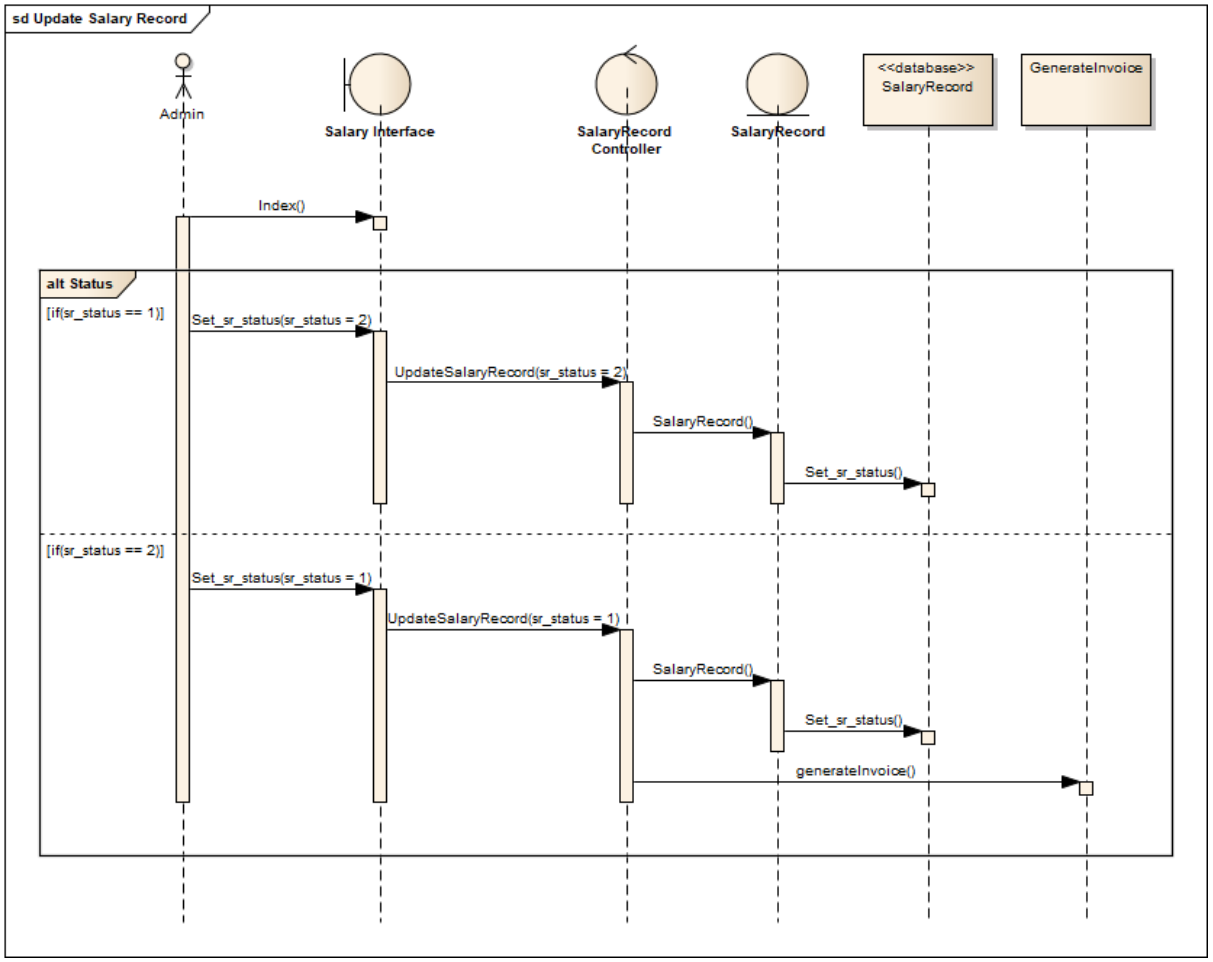

Figure 2.17: Sequence Diagram for Update Salary Record

2.2.8 UC008 Use Case <Create Class Record>

Use case: Create Class Record
ID: UC008
Actors: Teacher
Preconditions: 1. Teacher login the system 2. Teacher finished teaching a class
Flow of events: 1. Teacher select the “Teacher Class Record” module from the navbar 2. Teacher click ‘Create’ button shown at the top of page 3. Teacher key in the details of the class, date, start time and end time 4. Teacher choose the student involved in this class using the drop-down list 5. Teacher click “Save” button to save the record
Postconditions: The system recorded the new class record information
Alternative flow *1*: 1. Teacher Click “Cancel”
Postconditions: System didn’t save the record
Alternative flow *2*: 1. Teacher logout the system or close the browser tab
Postconditions: System didn’t save the record
Exception flow (if any):

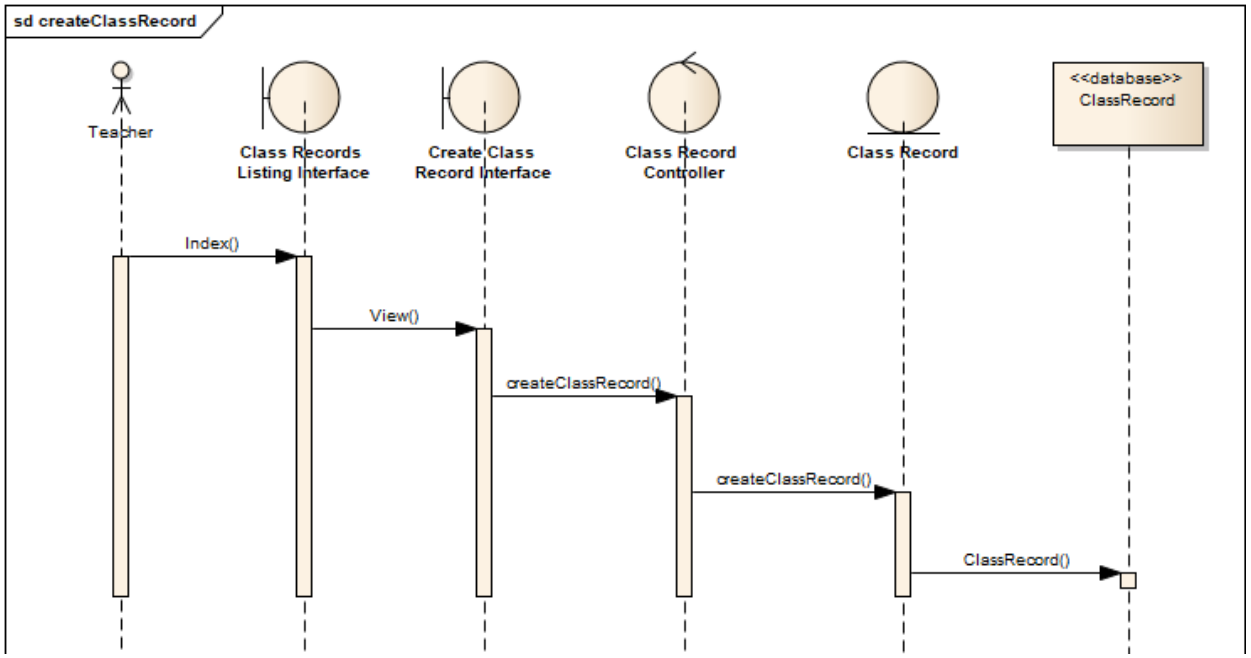

Figure 2.18: Sequence Diagram for Create Class Records (Teacher)

2.2.9 UC009 Use Case <Read Class Record>

Use case: Read Class Record
ID: UC009
Actors: Teacher, Admin
Preconditions: Teacher or Admin login the system
Flow of events: 1. Teacher or Admin selects the “Class Record” module from the navbar. 2. If the user is a teacher a. The system displays the class ID, class date and time, students involved of every class records 3. If the user is an Admin a. The system will have 2 buttons. One write “Class Records”, and another one write “Teacher’s Performance” b. Admin select the “Class Records” c. The system displays the class ID, class date and time, students involved of every class records
Postconditions:
Exception flow (if any):

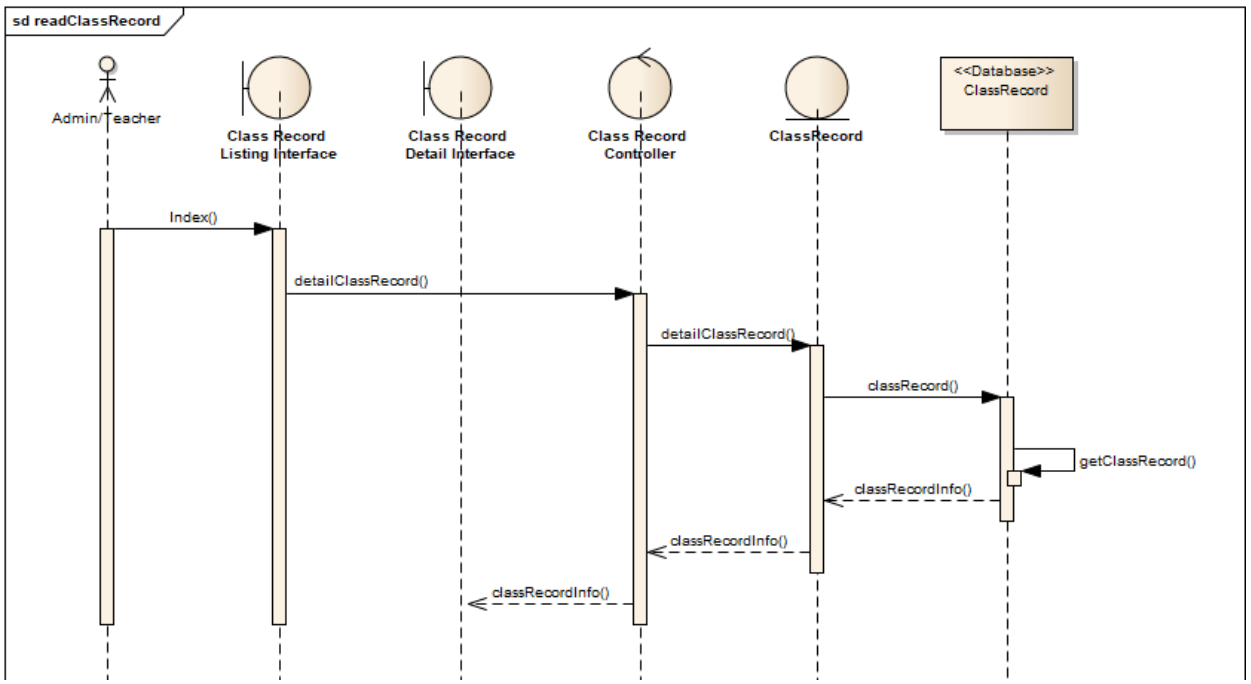

Figure 2.19: Sequence Diagram for Read Class Records (Teacher)

2.2.10 UC010 Use Case <Update Class Record>

Use case: Update Class Record
ID: UC010
Actors: Teacher
Preconditions: 1. Teacher is logged on to the system.
Flow of events: 1. Teacher selects the “Manage Class Record” module from the navbar. 2. System displays a list of existing class records. Evry class record has an “Edit” button at its end 3. Teacher choose the class record he wants to edit by clicking the “Edit” button 4. Teacher edit the detail he/she wants to update 5. Teacher click “Save” button to save the changes
Postconditions: The system save the changes made to the existing class records
Alternative flow *1*: 1. Teacher Click “Cancel”
Postconditions: System didn’t save the record
Alternative flow *2*: 1. Teacher logout the system or close the browser tab
Postconditions: System didn’t save the record
Exception flow (if any):

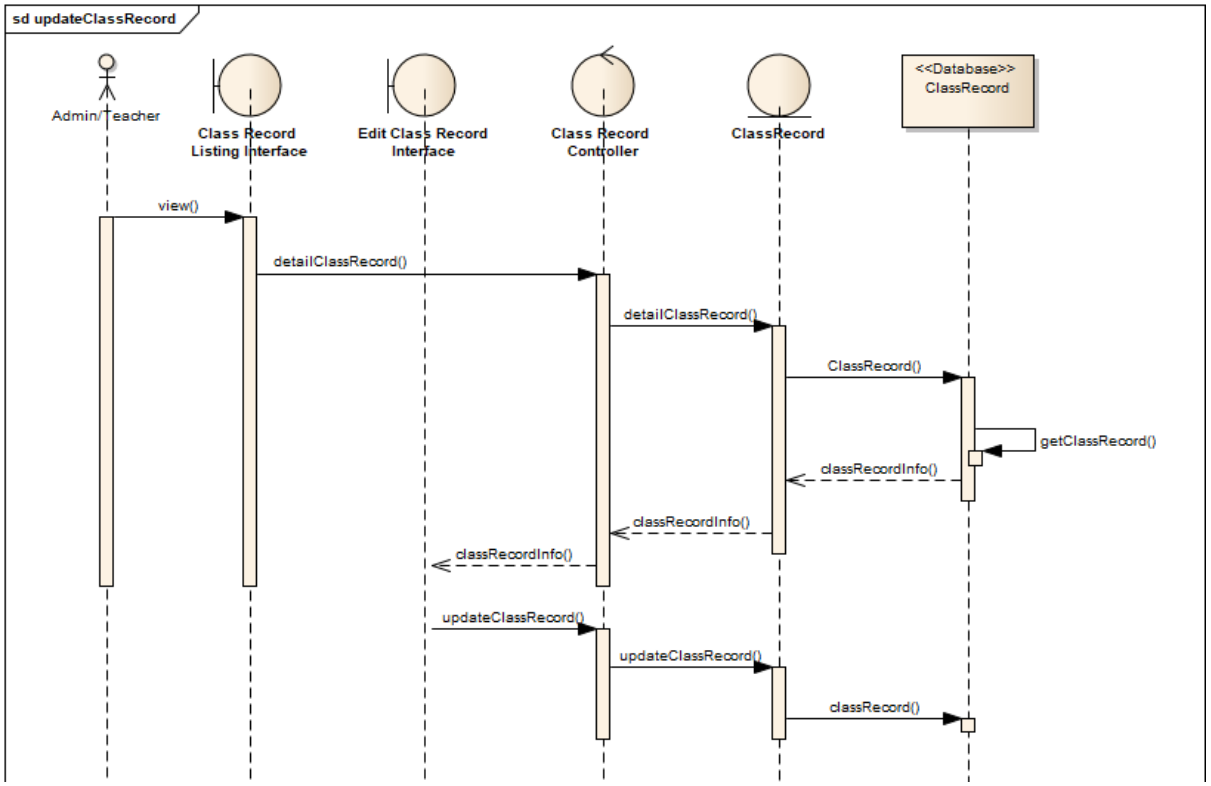

Figure 2.20: Sequence Diagram for Update Class Records (Teacher)

2.2.11 UC011 Use Case <Delete Class Record>

Use case: Delete Class Record
ID: UC011
Actors: Teacher
Preconditions: 1. A teacher is logged on to the system.
Flow of events: 1. Teacher selects the “Manage Class Records” module from the navbar 2. System displays a list of existing class records. Evry class record has an “Delete” button at its end 3. Teacher choose the class record he wants to edit by clicking the “Delete” button 4. System will display a prompt message confirming teacher’s action 5. Teacher choose “Yes” to execute the deletion a. System execute the record deletion 6. Else a. System won’t execute the record deletion
Postconditions:
Alternative flow *1*: 1. Teacher logout the system or close the browser tab
Postconditions: System didn’t save the record
Exception flow (if any):

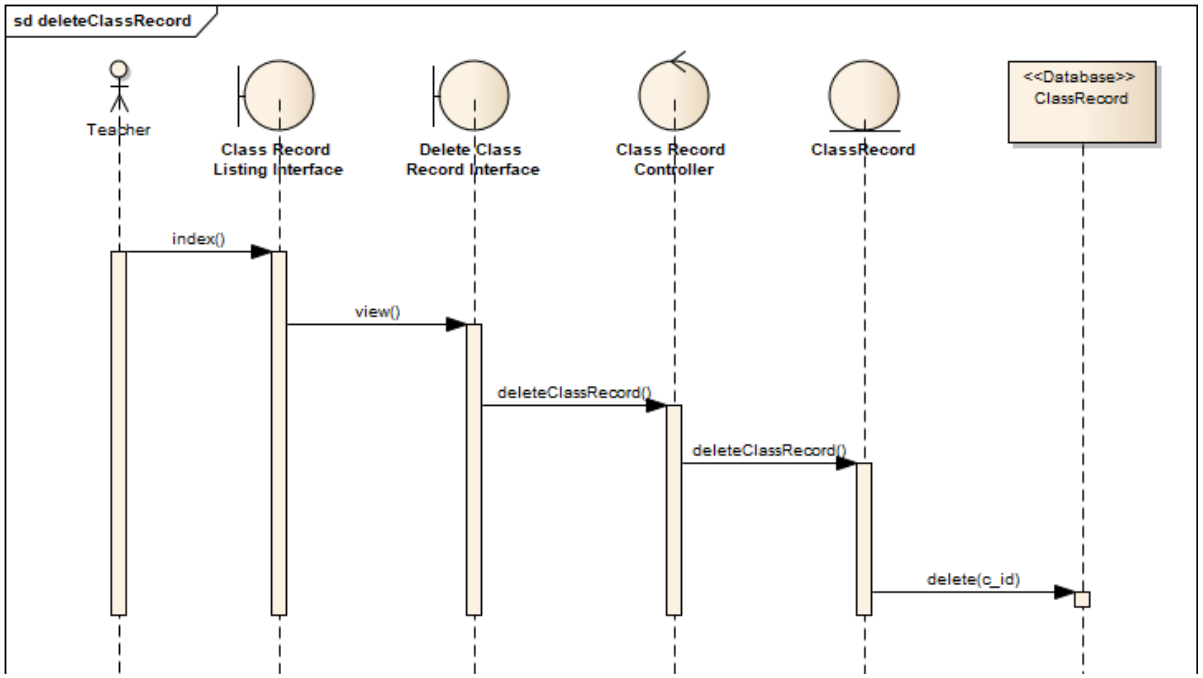

Figure 2.21: Sequence Diagram for Delete Class Records (Teacher)

2.2.12 UC012 Use Case <Create Student Feedback>

Use case: Create Student Feedback
ID: UC012
Actors: Teacher
Preconditions: 1. A valid teacher is logged on to the system. 2. The class must be taken.
Flow of events: 1. The teacher chooses the “Student Feedback” module from the navbar. 2. The teacher selects the “Date” of the class. 3. The system displays the list of the students attended under the tutor. 4. The teacher clicks on the “Create” for a particular student and creates the student feedback. 5. Teacher selects the “Post” button to post the feedback given.
Postconditions: The new student feedback is created.
Alternative flow 1 : If the Teacher changes their mind not to proceed with the feedback,he/she can click on the “Cancel” button.
Postconditions: The user will be redirected to the list of students.

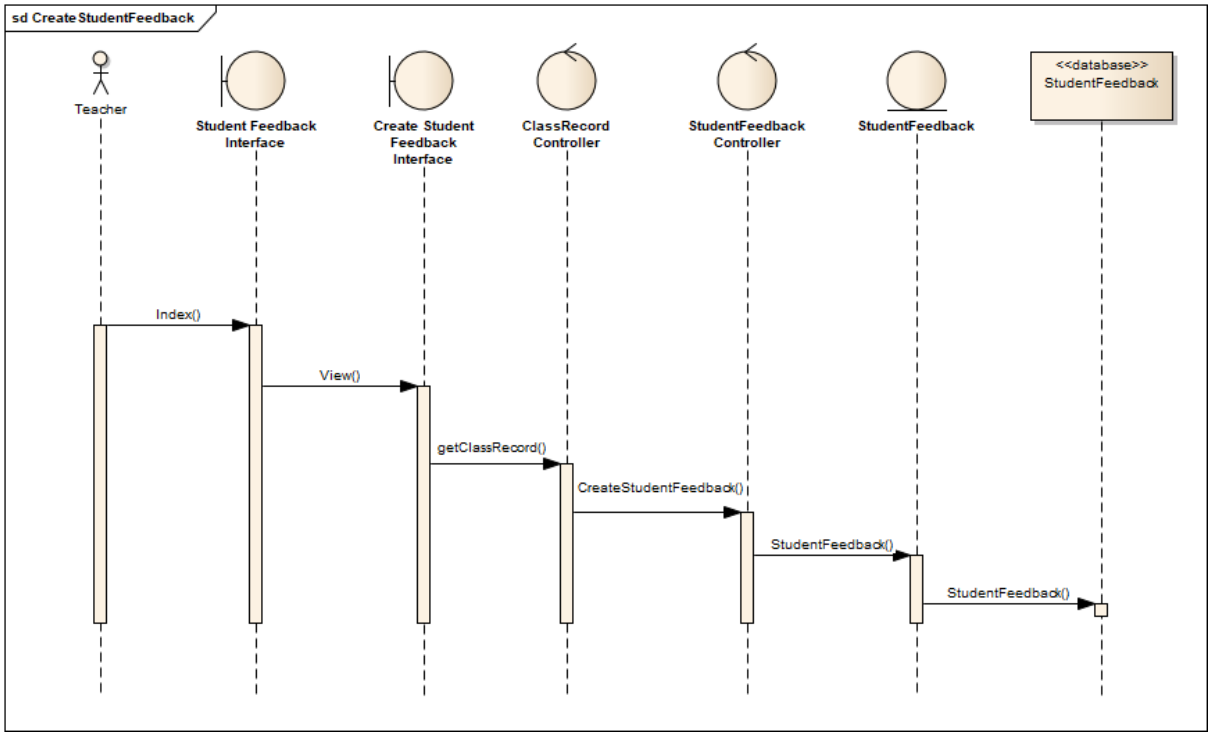

Figure 2.22: Sequence Diagram for Create Student Feedback

2.2.13 UC013 Use Case <Update Student Feedback>

Use case: Update Student Feedback
ID: UC013
Actors: Teacher
Preconditions: 1. A valid teacher is logged on to the system. 2. The class must be taken.
Flow of events: 1. The teacher chooses the “Student Feedback” module from the navbar. 2. The teacher selects the “Date” of the class. 3. The system displays the list of the students attended under the tutor. 4. The teacher clicks on the “Update” for the selected student. 5. Teacher selects the “Save” button to proceed with the update.
Postconditions: The student feedback is updated.
Alternative flow 1 : If the Teacher changes their mind not to proceed with the update,he/she can click on the “Cancel” button.
Postconditions: The user will be redirected to the list of students.

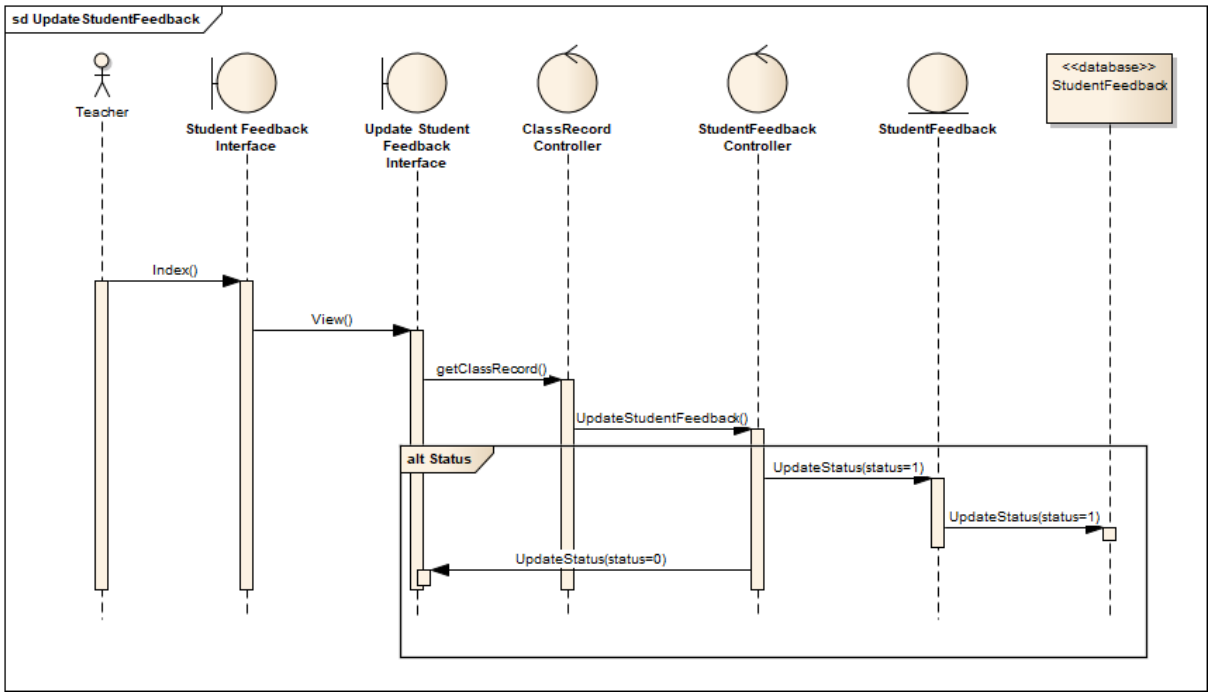

Figure 2.23: Sequence Diagram for Update Student Feedback

2.2.14 UC014 Use Case <Read Student Feedback>

Use case: Read Student Feedback
ID: UC014
Actors: Teacher
Preconditions: 1. A valid teacher is logged on to the system. 2. The class must be taken.
Flow of events: 1. The teacher chooses the “Student Feedback” module from the navbar. 2. The system displays the student feedback given for every student.

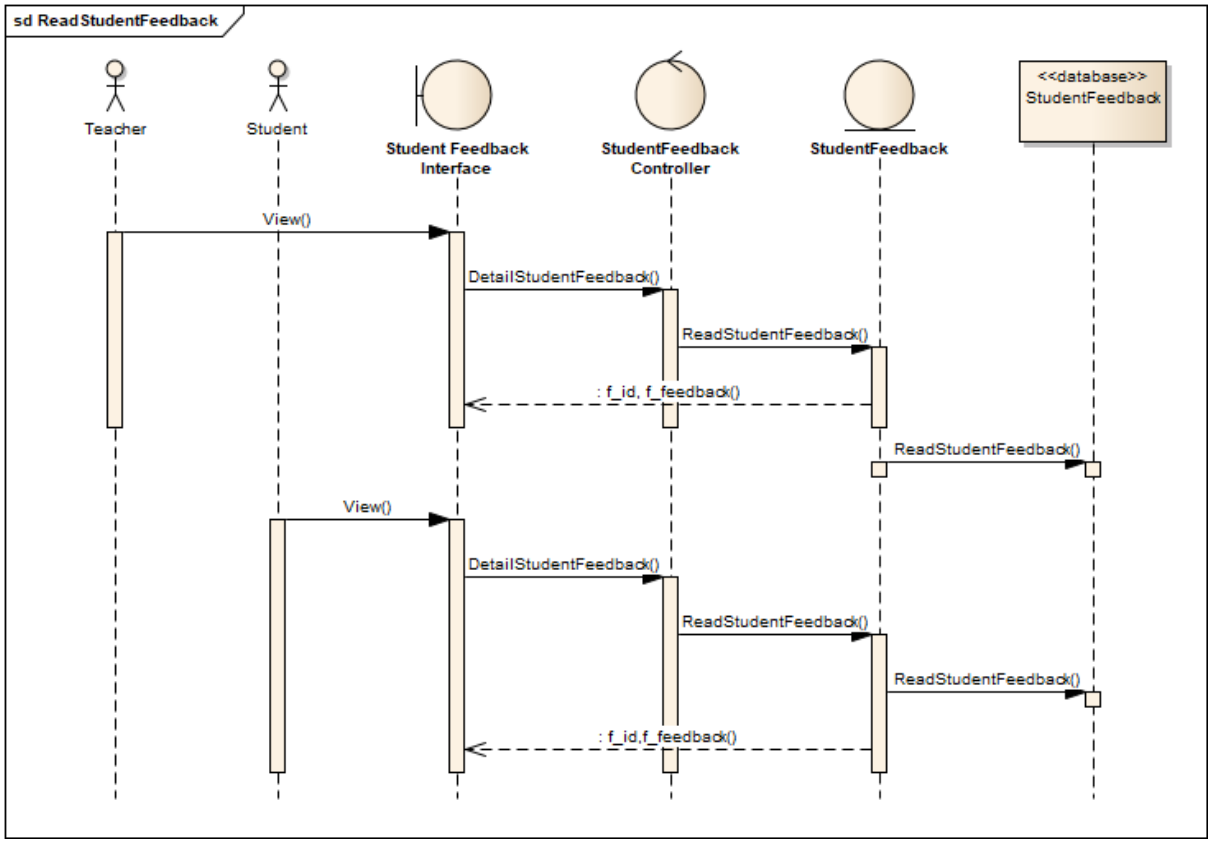

Figure 2.24: Sequence Diagram for Read Student Feedback

2.2.15 UC015 Use Case <Delete Student Feedback>

Use case: Delete Student Feedback
ID: UC015
Actors: Teacher
Preconditions: 1. A valid teacher is logged on to the system. 2. The class must be taken.
Flow of events: 1. The teacher chooses the “Student Feedback” module from the navbar. 2. The teacher selects the “Date” of the class. 3. The system displays the list of the students attended under the tutor. 4. The teacher clicks on the “Delete” for the selected student. 5. A pop up of reconfirmation for the deletion will show up. 6. Teacher selects the “Sure” button to proceed with the deletion.
Postconditions: The student feedback is deleted.
Alternative flow 1 : If the Teacher changes their mind not to proceed with the deletion,he/she can click on the “Cancel” button.
Postconditions: The user will be redirected to the list of students.

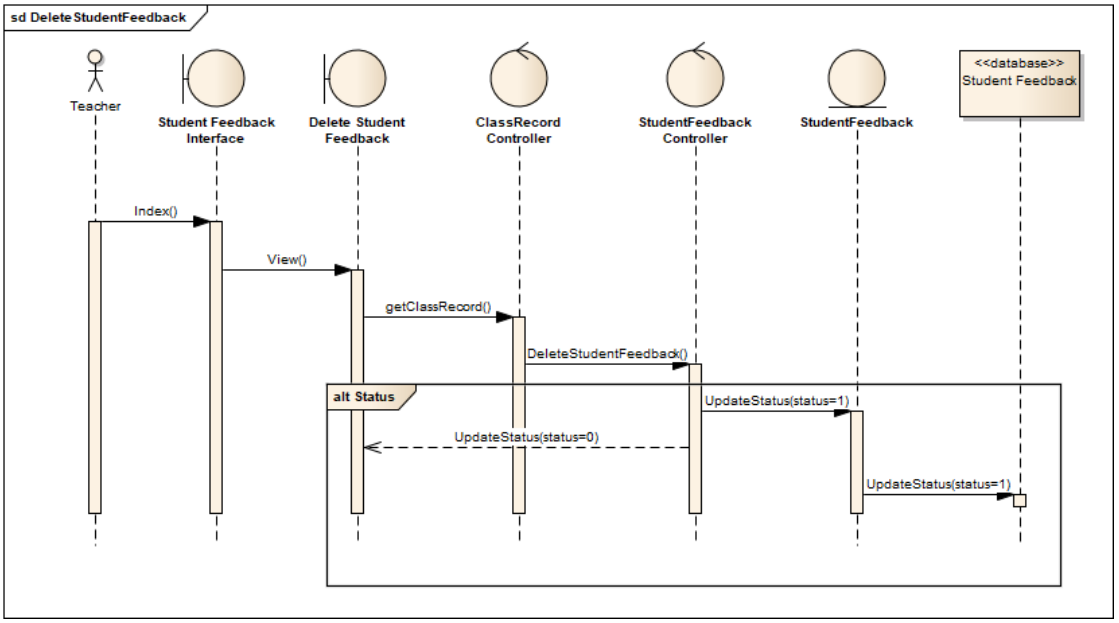

Figure 2.25: Sequence Diagram for Delete Student Feedback

2.2.16 UC016 Use Case <Generate Report>

Use case: Generate Report
ID: UC016
Actors: Teacher
Preconditions: 1. A valid teacher is logged on to the system. 2. The class must be taken.
Flow of events: 1. The teacher chooses the “Student Feedback” module from the navbar. 2. The teacher selects “Report” to generate a monthly student feedback report for the students. 3. Teacher selects the “Generate” button to create the report based on the student feedback given.
Postconditions: The student feedback report is generated.

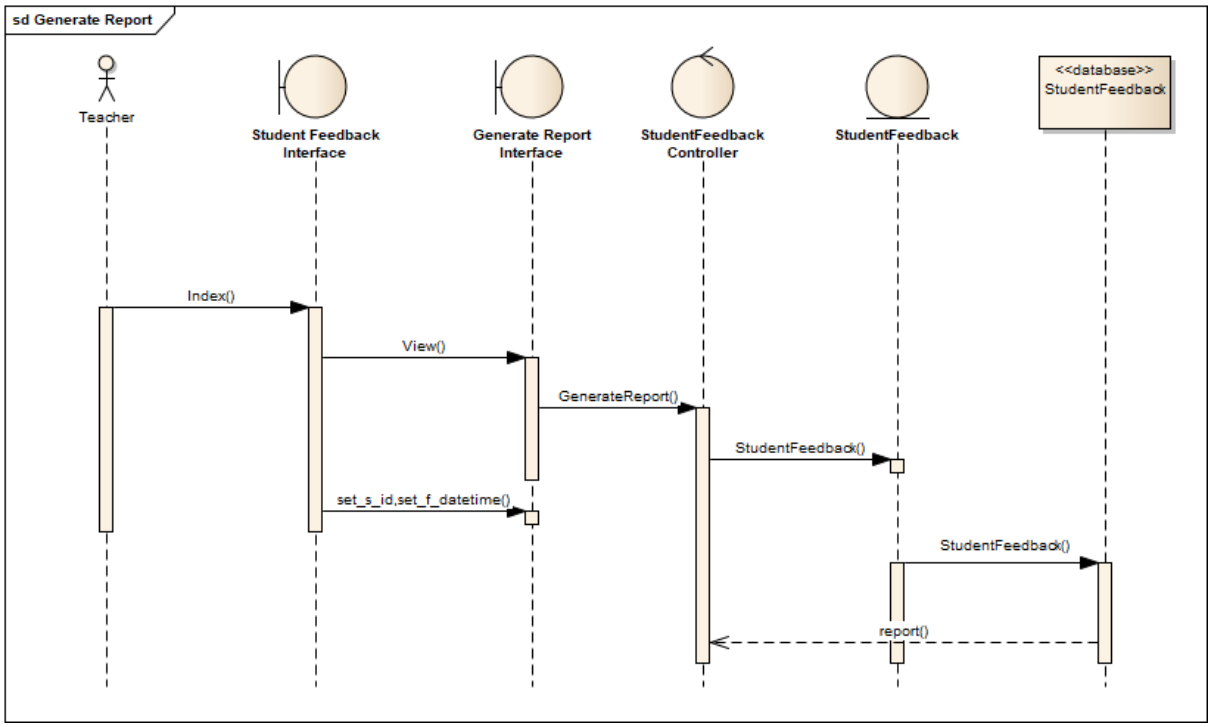

Figure 2.26: Sequence Diagram for Generate Report

2.2.17 UC017 Use Case <Verify Report>

Use case: Verify Report
ID: UC017
Actors: Student
Preconditions: 1. A valid student is logged on to the system. 2. The class must be taken.
Flow of events: 1. The student chooses the “Student Feedback” module from the navbar. 2. The student selects the “Month”. 3. The system displays the feedback report for the selected month.
Postconditions: The student is able to review the feedback report made by their teachers and verify it.
Alternative flow 1 : If the student wishes to ask something, they can click on the “Comment” to post their opinion.

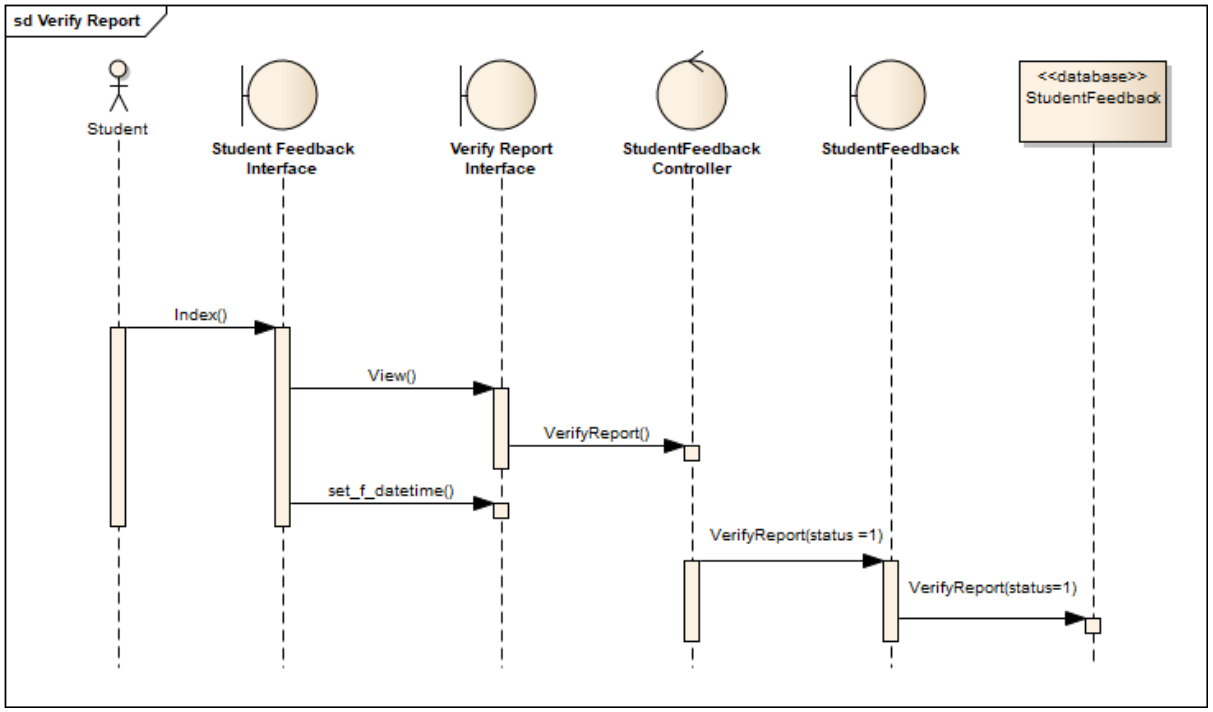

Figure 2.27: Sequence Diagram forVerify Report

2.2.18 UC018 Use Case <Create Teacher-student Pairing>

Use case: Create Teacher-student Pairing
ID: UC018
Actors: Admin
Preconditions: 1. Admin login the system
Flow of events: 1. Admin select the “Teacher-student pairing” module from the navbar 2. Admin click ‘Create’ button shown at the top of page 3. System displays 2 dropdown lists 4. For the first one, the admin need to choose a teacher 5. For the second one, the admin need to choose student that will be under that teacher
Postconditions: The system recorded the new teacher-student pairing record
Alternative flow *1*: 1. Admin Click “Cancel”
Postconditions: System didn’t save the record
Alternative flow *2*: 1. Admin logout the system or close the browser tab
Postconditions: System didn’t save the record
Exception flow (if any):

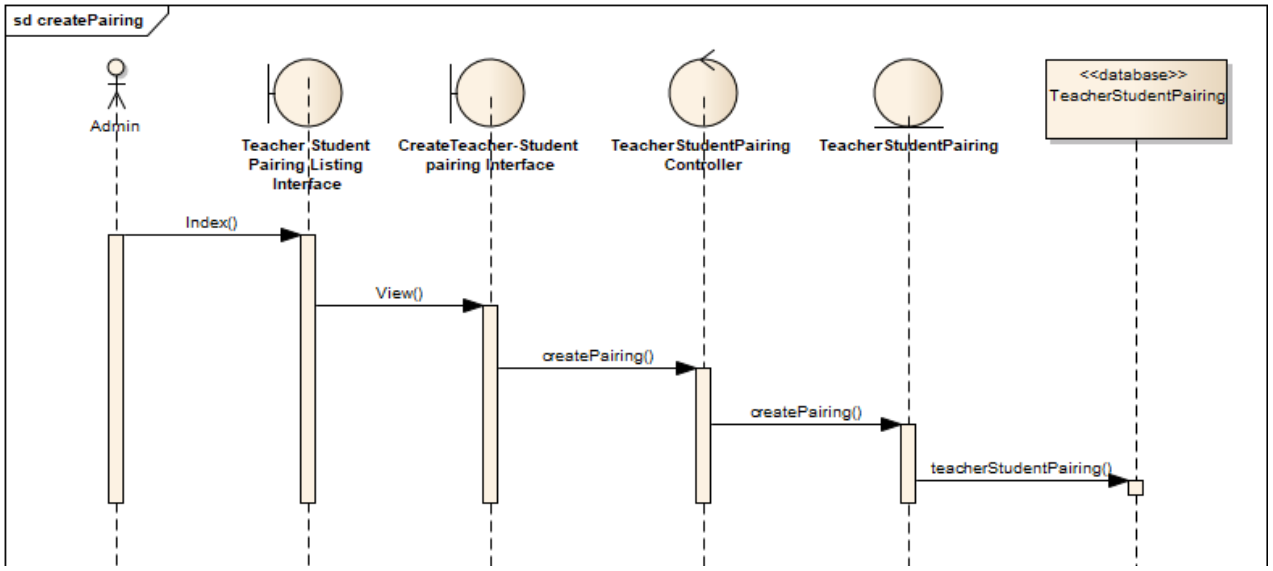

Figure 2.28: Sequence Diagram for Create Teacher-student Pairing

2.2.19 UC019 Use Case <Delete Teacher-student Pairing>

Use case: Delete Teacher-student Pairing
ID: UC019
Actors: Admin
Preconditions: 1. Admin login the system
Flow of events: 1. Admin selects the “Teacher-student Pairing” module from the navbar 2. System displays a list of existing pairing records. Evry record has an “Delete” button at its end 3. Admin choose the class record he wants to edit by clicking the “Delete” button 4. System will display a prompt message confirming Admin’s action 5. Adminr choose “Yes” to execute the deletion a. System execute the record deletion 6. Else a. System won’t execute the record deletion
Postconditions:
Alternative flow *1*: 1. Teacher logout the system or close the browser tab
Postconditions: System didn’t delete the record
Exception flow (if any):

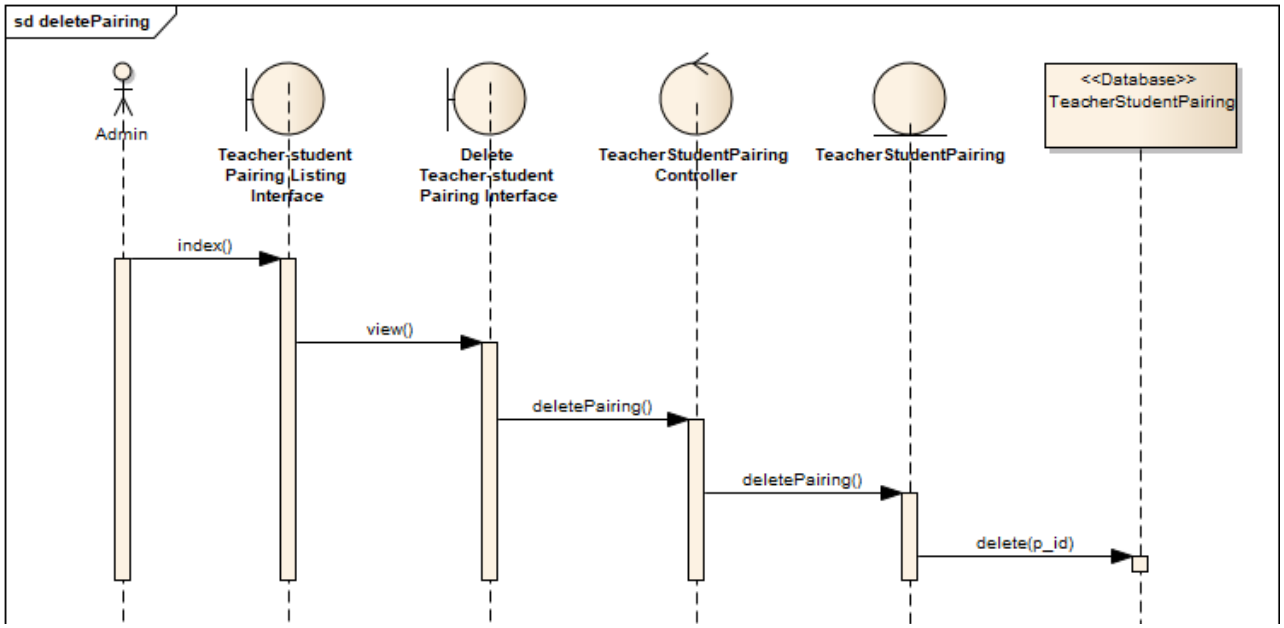

Figure 2.29: Sequence Diagram for Delete Teacher-student Pairing

2.2.20 UC020 Use Case <Read Teacher-student Pairing>

Use case: Update Teacher-Student Pairing
ID: UC020
Actors: Admin
Preconditions: 1. Admin login the system
Flow of events: 1. Admin selects the “Teacher-student Pairing” module from the navbar. 2. System displays a list of pairings. Evry record has an “Edit” button at its end 3. Admin choose the record he wants to edit by clicking the “Edit” button 4. Admin edit the record by changing teacher or students 5. Admin click “Save” button to save the changes
Postconditions: The system save the changes made to the existing pairing records
Alternative flow *1*: 1. Admin Click “Cancel”
Postconditions: System didn’t save the record
Alternative flow *2*: 1. Admin logout the system or close the browser tab
Postconditions: System didn’t save the record
Exception flow (if any):

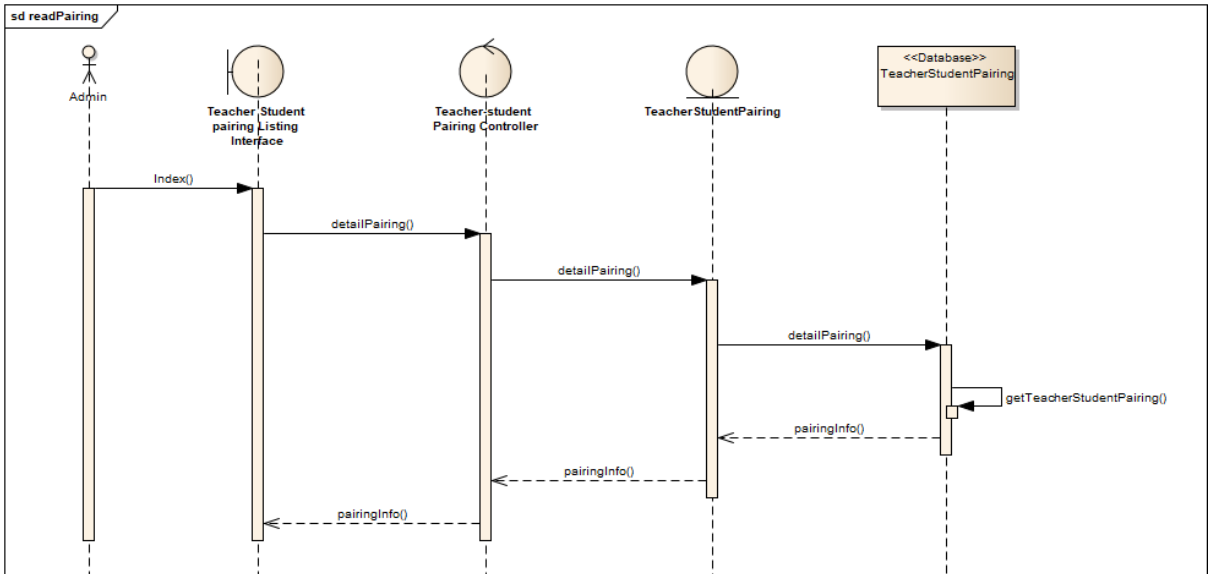

Figure 2.30: Sequence Diagram for Read Teacher-student Pairing

2.3 Performance and Other Requirements

Functional Requirements

Manage Users

Function1 <<Create new teachers' account>>

The admin shall be able to create a new account for teachers

Function2 <<Delete teachers' account>>

The admin shall be able to delete the account for teachers

Function3 <<Update teachers' account>>

The admin shall be able to update the teachers' profile (contact number, email, password, location, gender)

Function4 <<View teachers' account>>

The admin shall be able to view all of the teachers' profiles

Function5 <<Create new students' account>>

The admin shall be able to create a new account for students

Function6 <<Delete students' account>>

The admin shall be able to delete the account for students

Function7 <<Update students' account>>

The admin shall be able to update the students' profile (contact number, email, password, location, gender)

Function8 <<View students' account>>

The admin shall be able to view all of the students' profiles

Function9 <<Login>>

The users shall be able to log in the system using their user ID and password

Function10 <<Manage Profile & Password>>

The users shall be able to update their profile details or password when they feel the need.

Manage Class Record

Function11 <<Create Class Record>>

The teacher shall be able to create a class record after every class.

Function12 <<Read Class Record>>

The teacher shall be able to access the details of every class record made by him/her.

Function13 <<Update Class Record>>

The teacher shall be able to modify the details of every class record made by him/her.

Function14 <<Delete Class Record>>

The teacher shall be able to delete class records when he/she feels the need.

Function15 <<View Teaching Hour>>

The teacher shall be able to view the total teaching hour he/she has achieved in a month

Function16 <<Verify Teaching Hour>>

The admin shall be able to verify the total teaching hour of a teacher when the admin wants to make a salary payment to the teacher

Manage Student Feedback

Function17 <<Create Student Feedbacks>>

The teacher shall be able to create feedback for their students.

Function18 <<Read Student Feedbacks>>

The teacher shall be able to access the details of every student feedback made by him/her.

Function19 <<Update Student Feedbacks>>

The teacher shall be able to modify the details of the feedback made by him/her.

Function20 <<Delete Student Feedbacks >>

The teacher shall be able to delete feedback when he/she feels the need.

Function21 <<Generate Report>>

The system should generate the report and send it to the teacher's student at the end of the month.

Function22 <<Verify Report>>

The student shall be able to review the feedback report made by their teachers and verify it.

Manage Salary

Function23 <<Create teacher salary>>

The admin shall be able to create a teacher salary after the admin verified the teaching hour of the teacher and number of reports submitted.

Function24 <<Generate invoice>>

The system shall be able to generate an invoice once the admin has made payment for the teacher.

Function25 <<Update or edit teacher salary>>

The admin shall be able to modify the status of salary payment.

Function26 <<Read or view teacher salary>>

The admin shall be able to view the teacher salary information.

Function27 <<Update or edit salary rate>>

The admin shall be able to adjust the salary rate.

Manage Teacher-student Pairing

Function28 <<Create Teacher-student Pairing>>

The admin shall be able to assign a teacher to a student

Function29 <<Read Teacher-student Pairing>>

The admin shall be able to access a list of teacher-student pairing

Function30 <<Update Teacher-student Pairing>>

The admin shall be able to assign the teacher to a different student

Function31 <<Delete Teacher-student Pairing>>

The admin shall be able to unassign a teacher from a student.

Non-Functional Requirements

NFR1 <<Operational>>

The user shall authenticate themselves using their user id and password.

NFR2 <<Operational>>

The system shall send email notification to notify teachers when the admin has made salary payment.

NFR3 <<Efficiency>>

The system shall be able to generate an invoice within 5 seconds once the admin changes the status of salary payment to “Approved”.

NFR4 <<Efficiency>>

The system shall be able to calculate the total salary within 2 seconds and display it on the screen.

NFR3 <<Security/Safety>>

The system shall be able to implement user’s privacy provisions to prevent the data to be used by third parties.

NFR5 <<Security/Safety>>

The system shall not allow non-registered users to access the system

NFR6 <<Usability>>

The students shall be able to trace back their feedback from the teacher.

NFR7 <<Dependability>>

The system shall not have downtime longer than 1 hour.

2.4 Design Constraints

Some of the design constraints imposed by the organisation where the MengajiOnetoOne Salary Management System will be used are as below:

- .NET MVC 5.0 is used for the back-end development
- Bootstrap 5.0 is used for the front-end development
- Session is used to check for the user type in order to display the appropriate pages or interface and certain functionalities for each user type
- The users have to authenticate themselves by logging into the system to use it
- Non-registered users are not allowed to access the system

2.5 Software System Attributes

- Only admin can register users where teachers and students are not able to register themselves
- The total salary is calculated based on the salary rate and total hour of teaching
- Admin can verify teaching hours and total students feedback reports submitted before making salary payment to the teachers
- Invoice can be generated once the admin makes the salary payment
- Admin can assign the student to the teacher
- Teachers can submit students' feedback to the system and students are able to review the feedback reports