Fpt University

TRƯỜNG ĐẠI HỌC FPT

CAPSTONE PROJECT REPORT

Report 4 – Software Design Document

– Hanoi, December 2024 –

Table of Contents

I. Record of Changes

Date	A* M, D	In charge	Change Description
17/09/2024	A	NinhNT	Initiation
17/09/2024	A	NinhNT	Add system architecture
17/09/2024	A	NinhNT	Add front-end package diagram
17/09/2024	A	NinhNT	Add back-end package diagram
18/09/2024	A	NinhNT	Database
19/09/2024	M	HauNX	Edit system architecture
19/09/2024	M	HauNX	Edit front-end package diagram
19/09/2024	M	HauNX	Edit back-end package diagram
22/09/2024	A	HauNX	Add class diagram and sequence diagram of iteration 1
13/10/2024	M	NinhNT	Edit database
17/10/2024	A	HauNX	Add class diagram and sequence diagram of iteration 2
03/11/2024	M	NinhNT	Edit database
07/11/2024	A	HauNX	Add class diagram and sequence diagram of iteration 3
25/11/2024	M	NinhNT	Edit system diagram
29/11/2024	A	HauNX	Add class diagram and sequence diagram of iteration 4, finalize the document

*A - Added M - Modified D - Deleted

II. Software Design Document

1. System Design

1.1 System Architecture

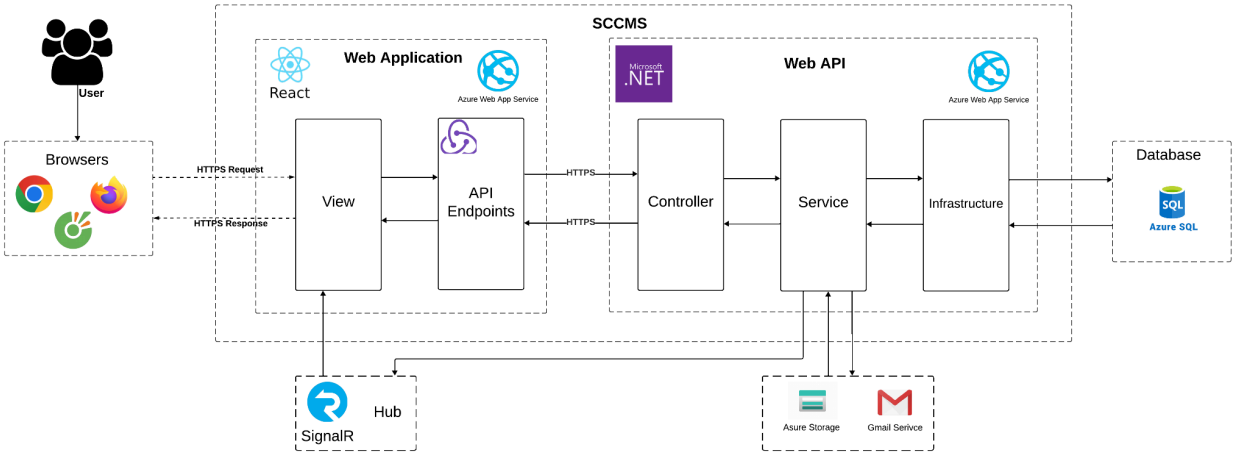

Figure 1: System architecture

The link of image: System architecture

The system architecture depicted in the diagram represents a comprehensive web application structure with secure, high-performance communication through HTTPS. Below is a detailed description of the system:

1. **Browers:** Users can use most browsers to access our website such as: Chrome, Firefox, Microsoft Edge
2. **Web Application (React):** The front-end application is built using React, a popular JavaScript library. React handles the user interface and sends HTTP/HTTPS requests to the Web API to retrieve data and perform various operations.
3. **Redux:** Redux is a state management library used with React to manage the application's state in a single store. It simplifies state sharing across components and enables predictable state updates through actions and reducers. Dispatching an action triggers state changes, making it easier to track and debug complex app behaviors.
4. **Web API (.NET):** The backend is developed using the .NET framework, which provides API services for the web application. All requests from React to the backend occur via

HTTPS to ensure data security and privacy. The .NET API communicates with several services, including Firebase, SignalR, Azure Storage, and SQL Server.

5. **SignalR**: SignalR is integrated into the system to provide real-time bi-directional communication between the client (browser) and the server. It enables features like live updates, notifications, and collaborative tools. SignalR operates over WebSockets (or other fallback protocols like Server-Sent Events or Long Polling), ensuring efficient and low-latency communication. The integration is secured via HTTPS to maintain data confidentiality and integrity.
6. **Gmail API:** The system integrates with the Gmail API to send emails, such as notifications or email confirmations to users. All interactions between the Web API and the Gmail API are secured through HTTPS, protecting user information.
7. **Azure Services (Azure Storage Account):** The Azure Storage Account is used for storing large files or data, such as images or attachments. The Web API securely communicates with this service to store and retrieve data through HTTPS.
8. **Azure SQL:** Azure SQL is a cloud-based relational database service provided by Microsoft. It offers high availability, security, and scalability, making it ideal for applications requiring a secure and reliable SQL database. Azure SQL is fully managed, meaning Microsoft handles maintenance, updates, and scaling automatically.

9. **Azure Web App Service:** Azure Web App Service is a Platform-as-a-Service (PaaS) that allows easy deployment and management of web applications without handling the infrastructure. It supports multiple programming languages, offers automatic scaling, and manages the server-side setup, allowing developers to focus on the code rather than the underlying infrastructure.

1.2 Package Diagram

1.2.1 Front-end package diagram

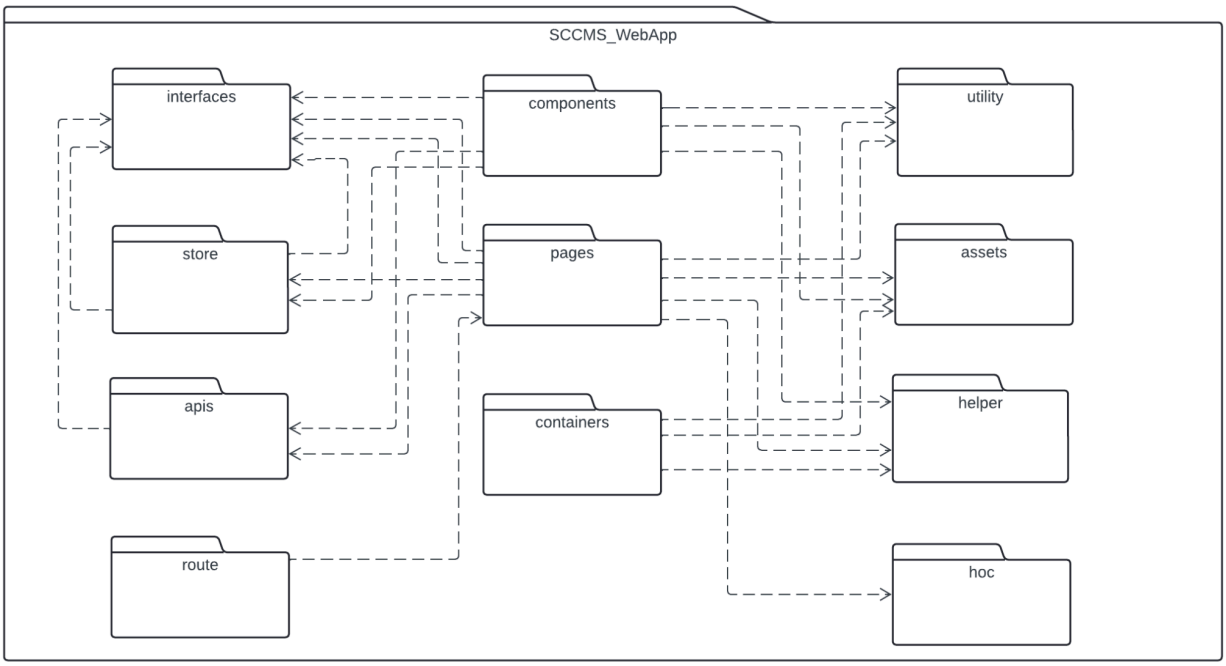

Figure 2: Front-end package diagram

The link of image: Front-end package diagram

Package descriptions

No	Package	Description
1	assets	Contains static resources such as images, fonts, and stylesheets. These assets are used across the application to provide styling and visual content.
2	components	This directory likely holds reusable UI components such as buttons, forms, tables, or any other visual elements used throughout the application. These components are typically independent, and they can be combined to build a larger UI.
3	pages	Pages generally represent full-screen views or routes in the application. Each page could be a collection of multiple components and containers, forming a complete section of the application.

4	containers	Contains the overall layout of the project.
5	hoc (Higher-Order Components)	This folder may contain higher-order components, which are React components that wrap other components to add additional functionality, such as authorization handling, data loading, or conditional rendering.
6	route	This package is responsible for defining the routes of the application. It probably maps different URL paths to their corresponding pages or components.
7	store	This is where state management logic resides for the Redux library. It centralizes the application's state, ensuring consistent data flow and state management.
8	helper	Contains helpers used across different parts of the application. These are not tied to any specific feature but provide essential services like formatting, validation, or shared business logic.
9	utility	Contains utility functions, helpers, or common logic used across different parts of the application. These are not tied to any specific feature but provide essential services like formatting, validation, or shared business logic.
10	apis	The APIs folder most likely handles the interaction with the backend services. This package could contain functions or classes for making HTTP requests, handling responses, and managing data fetching from the server.
11	interfaces	This package likely contains TypeScript or other interface definitions that are shared across various parts of the application. These interfaces help ensure consistent data structures throughout the app. The package also contains constant values, configuration settings, or enums in the application

1.2.2 Back-end package Diagram

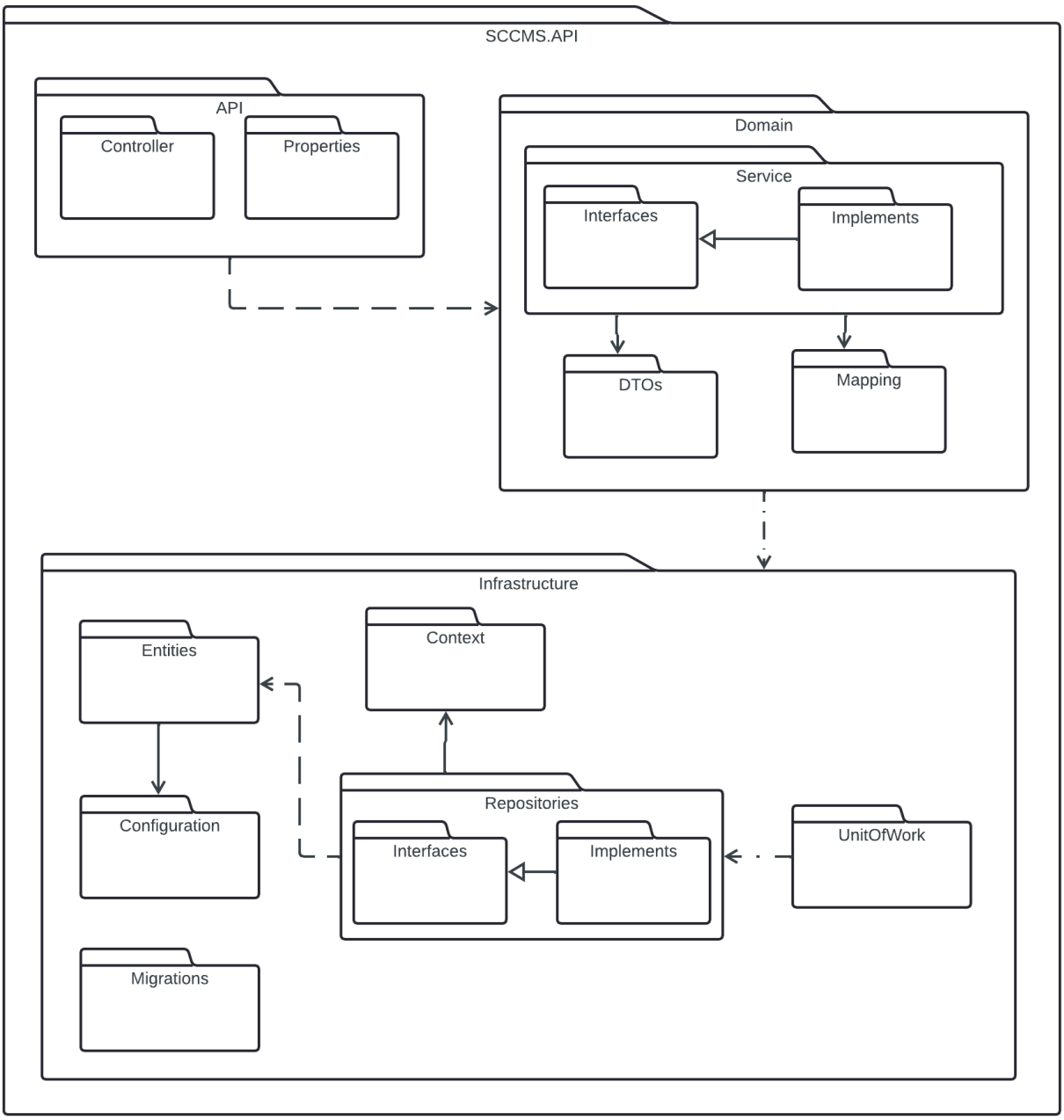

Figure 3: Back-end package diagram
The link of image: Back-end package diagram

Package descriptions

No	Package	Description
1	API	Contains the API layer, with Controller for handling HTTP requests and Properties for application settings or metadata. It serves as the main entry point for external interactions with the system.
2	Controller	Handles incoming HTTP requests, routes them to appropriate services in the Domain layer, and returns HTTP responses.
3	Properties	Contains settings and metadata related to the API layer, such as configurations or constants used in the Controllers.
4	Domain	The core business logic and domain services. It includes Interfaces to define service contracts, Implements for business logic implementations, DTOs for data exchange between layers, and Mapping for converting between entities and DTOs.
5	Interfaces	Defines contracts for services that handle business logic, ensuring a separation between implementation and usage.
6	Implements	Contains concrete implementations of the interfaces where the actual business logic and service methods are defined.
7	DTOs	Data Transfer Objects used to exchange data between layers, particularly between the API and Domain layers, to avoid exposing entities directly.
8	Mapping	Handles mapping logic between models and DTOs, ensuring smooth conversion between data structures.
9	Infrastructure	Manages data access logic and related infrastructure. It includes Entities for database table mapping, Context for interacting with the database, Data for managing data services, Repositories for encapsulating data access operations, UnitOfWork for transaction management, and Migrations for managing database schema changes.
10	Entities	Represents domain models that map directly to database tables, encapsulating the core data of the application.

11	Context	Represents the database context, use Entity Framework to interact with the database.
12	Repositories	Interfaces: Define contracts for data access, including common methods like Get, Add, Update, and Delete. Implements: Concrete implementations of these interfaces, containing the logic for data access and manipulation.
13	UnitOfWork	Manages transactions by coordinating between multiple repositories, ensuring that multiple changes can be committed as a single unit.
14	Configuration	Contains configurations related to infrastructure components, such as database connection settings or service parameters.
15	Migrations	Manages database schema changes through migrations, ensuring the database schema is kept in sync with changes in entity models.

2. Database Design

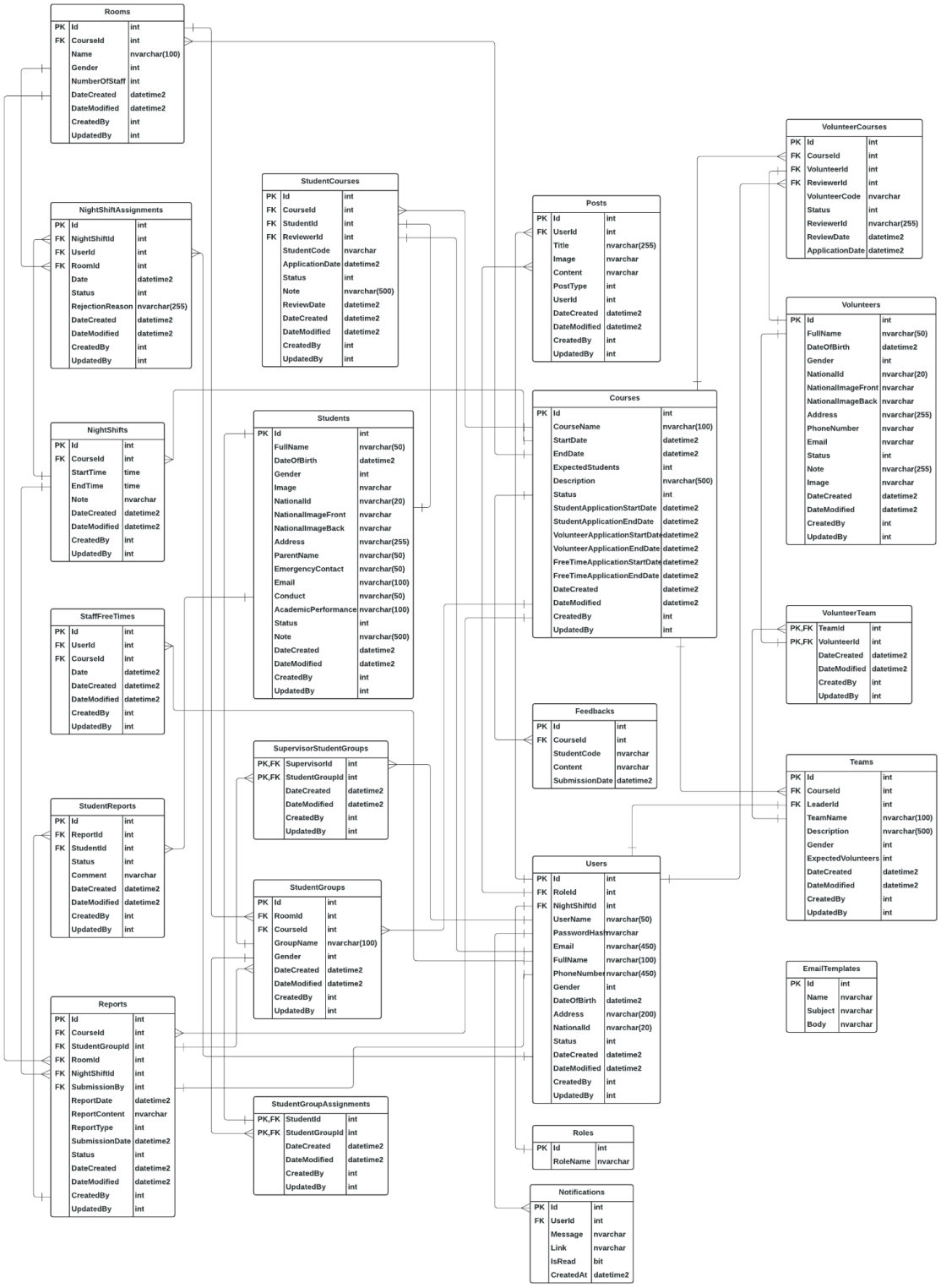

Figure 4: Database design

The link of image: [Database *design*]

2.1 Courses

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	CourseName	nvarchar(100)		x		Max Length: 100
03	StartDate	datetime2		x		
04	EndDate	datetime2		x		
05	ExpectedStudents	int		x		
06	Description	nvarchar(500)				Max Length: 500
07	Status	int		x		Max Length: 20
08	StudentApplicationStartDate	datetime2		x		
09	StudentApplicationEndDate	datetime2		x		
10	VolunteerApplicationStartDate	datetime2		x		
11	VolunteerApplicationEndDate	datetime2		x		
12	FreeTimeApplicationStartDate	datetime2				Nullable
13	FreeTimeApplicationEndDate	datetime2				Nullable
14	DateCreated	datetime2		x		
15	DateModified	datetime2		x		
16	CreatedBy	int		x		
17	UpdatedBy	int		x		

Table 32: Courses Table

2.2 Email Template

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	Name	nvarchar(max)		x		
03	Subject	nvarchar(max)		x		
04	Body	nvarchar(max)		x		

Table 33: Email Templates Table

2.3 Roles

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	RoleName	nvarchar(max)		x		

Table 34: Roles Table

2.4 Students

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	FullName	nvarchar(50)		x		Max Length: 50
03	DateOfBirth	datetime2		x		
04	Gender	int		x		
05	Image	nvarchar(max)		x		
06	NationalId	nvarchar(20)		x		Max Length: 20

07	NationalImageFront	nvarchar(max)		x		
08	NationalImageBack	nvarchar(max)		x		
09	Address	nvarchar(255)		x		Max Length: 255
10	ParentName	nvarchar(50)		x		Max Length: 50
11	EmergencyContact	nvarchar(50)		x		Max Length: 50
12	Email	nvarchar(100)		x		Max Length: 100
13	Conduct	nvarchar(50)				Max Length: 50 (Nullable)
14	AcademicPerformance	nvarchar(100)				Max Length: 100 (Nullable)
15	Status	int		x		
16	Note	nvarchar(500)				Max Length: 500 (Nullable)
17	DateCreated	datetime2		x		
18	DateModified	datetime2		x		
19	CreatedBy	int		x		
20	UpdatedBy	int		x		

Table 35: Students Table

2.5 Volunteers

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	FullName	nvarchar(50)		x		Max Length: 50
03	DateOfBirth	datetime2				Nullable
04	Gender	int		x		

05	NationalId	nvarchar(20)		x		Max Length: 20
06	NationalImageFront	nvarchar(max)		x		
07	NationalImageBack	nvarchar(max)		x		
08	Address	nvarchar(255)		x		Max Length: 255
09	PhoneNumber	nvarchar(max)				Nullable
10	Email	nvarchar(max)				Nullable
11	Status	int		x		
12	Note	nvarchar(255)				Max Length: 255 (Nullable)
13	Image	nvarchar(max)		x		
14	DateCreated	datetime2		x		
15	DateModified	datetime2		x		
16	CreatedBy	int		x		
17	UpdatedBy	int		x		

Table 36: Volunteers Table

2.6 Feedbacks

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	StudentCode	nvarchar(max)		x		
03	CourseId	int		x	FK	FK to Courses(Id)
04	Content	nvarchar(max)		x		
05	SubmissionDate	datetime2		x		

Table 37: Feedbacks Table

2.7 Night Shifts

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	CourseId	int		x	FK	FK to Courses(Id)
03	StartTime	time		x		
04	EndTime	time		x		
05	Note	nvarchar(max)				Nullable
06	DateCreated	datetime2		x		
07	DateModified	datetime2		x		
08	CreatedBy	int		x		
09	UpdatedBy	int		x		

Table 38: Night Shifts Table

2.8 Rooms

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	CourseId	int		x	FK	FK to Courses(Id)
03	Name	nvarchar(100)		x		Max Length: 100
04	Gender	int		x		
05	NumberOfStaff	int		x		
06	DateCreated	datetime2		x		
07	DateModified	datetime2		x		
08	CreatedBy	int		x		
09	UpdatedBy	int		x		

Table 39: Rooms Table

2.9 Users

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	UserName	nvarchar(50)	x	x		Max Length: 50
03	PasswordHash	nvarchar(max)		x		
04	Email	nvarchar(450)	x	x		
05	FullName	nvarchar(100)		x		Max Length: 100
06	PhoneNumber	nvarchar(450)	x			Nullable
07	Gender	int				Nullable
08	DateOfBirth	datetime2				Nullable
09	Address	nvarchar(200)				Max Length: 200, Nullable
10	NationalId	nvarchar(20)	x	x		Max Length: 20
11	Status	int				Nullable
12	RoleId	int		x	FK	FK to Roles(Id)
13	NightShiftId	int			FK	FK to NightShifts(Id), Nullable
14	DateCreated	datetime2		x		
15	DateModified	datetime2		x		
16	CreatedBy	int		x		
17	UpdatedBy	int		x		

Table 40: Users Table

2.10 Student Groups

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	CourseId	int		x	FK	FK to Courses(Id)
03	GroupName	nvarchar(100)		x		Max Length: 100
04	Gender	int		x		
05	RoomId	int			FK	FK to Rooms(Id), Nullable
06	DateCreated	datetime2		x		
07	DateModified	datetime2		x		
08	CreatedBy	int		x		
09	UpdatedBy	int		x		

Table 41: Student Groups Table

2.11 Student Courses

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	CourseId	int		x	FK	FK to Courses(Id)
03	StudentId	int		x	FK	FK to Students(Id)
04	StudentCode	nvarchar(max)				Nullable
05	ApplicationDate	datetime2				Nullable
06	Status	int		x		
07	Note	nvarchar(500)				Max Length: 500, Nullable

08	ReviewerId	int			FK	FK to Users(Id), Nullable
09	ReviewDate	datetime2				Nullable
10	DateCreated	datetime2		x		
11	DateModified	datetime2		x		
12	CreatedBy	int		x		
13	UpdatedBy	int		x		

Table 42: Student Courses Table

2.12 Staff Free Times

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	UserId	int		x	FK	FK to Users(Id)
03	CourseId	int		x	FK	FK to Courses(Id)
04	Date	datetime2		x		
05	DateCreated	datetime2		x		
06	DateModified	datetime2		x		
07	CreatedBy	int		x		
08	UpdatedBy	int		x		

Table 43: Staff Free Times Table

2.13 Reports

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	CourseId	int		x	FK	FK to Courses(Id)

03	UserId	int		x	FK	FK to Users(Id)
04	StudentGroupId	int			FK	FK to StudentGroups(Id), Nullable
05	ReportDate	datetime2		x		
06	ReportContent	nvarchar(max)				Nullable
07	ReportType	int		x		Max Length: 50
08	SubmissionDate	datetime2		x		
09	Status	int		x		
10	DateCreated	datetime2		x		
11	DateModified	datetime2		x		
12	CreatedBy	int		x		
13	UpdatedBy	int		x		

Table 44: Reports Table

2.14 Student Group Assignments

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	StudentId	int		x	PK, FK	PK: Composite Key (StudentId, StudentGroupId)
02	StudentGroupId	int		x	FK	FK to StudentGroups(Id)
03	DateCreated	datetime2		x		
04	DateModified	datetime2		x		
05	CreatedBy	int		x		
06	UpdatedBy	int		x		

Table 45: Student Group Assignments Table

2.15 Supervisor Student Groups

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	SupervisorId	int		x	PK, FK	PK: Composite Key (SupervisorId, StudentGroupId)
02	StudentGroupId	int		x	FK	FK to StudentGroups(Id)
03	DateCreated	datetime2		x		
04	DateModified	datetime2		x		
05	CreatedBy	int		x		
06	UpdatedBy	int		x		

Table 46: Supervisor Student Groups Table

2.16 Volunteer Courses

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	CourseId	int		x	FK	FK to Courses(Id)
03	VolunteerId	int		x	FK	FK to Volunteers(Id)
04	VolunteerCode	nvarchar(max)				Nullable
05	Status	int				Max Length: 20, Nullable
06	Note	nvarchar(255)				Max Length: 255, Nullable
07	ReviewerId	int			FK	FK to Users(Id), Nullable
08	ReviewDate	datetime2				Nullable

09	ApplicationDate	datetime2		x		
10	DateCreated	datetime2		x		
11	DateModified	datetime2		x		
12	CreatedBy	int		x		
13	UpdatedBy	int		x		

Table 47: Volunteer Courses Table

2.17 Volunteer Team

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	TeamId	int		x	PK, FK	PK: Composite Key (TeamId, VolunteerId)
02	VolunteerId	int		x	FK	FK to Volunteers(Id)
03	DateCreated	datetime2		x		
04	DateModified	datetime2		x		
05	CreatedBy	int		x		
06	UpdatedBy	int		x		

Table 48: Volunteer Team Table

2.18 Posts

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	Title	nvarchar(255)		x		Max Length: 255
03	Image	nvarchar(max)		x		
04	Content	nvarchar(max)		x		

05	PostType	int		x		
06	Status	int		x		
07	UserId	int			FK	FK to Users(Id), Nullable
08	DateCreated	datetime2		x		
09	DateModified	datetime2		x		
10	CreatedBy	int		x		
11	UpdatedBy	int		x		

Table 49: Post Table

2.19 Student Reports

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	ReportId	int		x	FK	FK to Reports(Id)
03	StudentId	int		x	FK	FK to Students(Id)
04	Status	int		x		
05	Comment	nvarchar(max)				Nullable
06	DateCreated	datetime2		x		
07	DateModified	datetime2		x		
08	CreatedBy	int		x		
09	UpdatedBy	int		x		

Table 50: Student Reports Table

2.20 Night Shift Assignments

No	Field Name	Type	Unique	Not Null	PK/ FK	Note

01	Id	int	x	x	PK	Identity (1,1)
02	NightShiftId	int		x	FK	FK to NightShifts(Id)
03	UserId	int			FK	FK to Users(Id), Nullable
04	Date	datetime2		x		
05	RoomId	int			FK	FK to Rooms(Id), Nullable
06	Status	int		x		Max Length: 50
07	RejectionReason	nvarchar(255)				Max Length: 255, Nullable
08	DateCreated	datetime2		x		
09	DateModified	datetime2		x		
10	CreatedBy	int		x		
11	UpdatedBy	int		x		

Table 51: Night Shift Assignments Table

2.21 Notifications

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	UserId	int		x	FK	FK to Users(Id)
03	Message	nvarchar(max)		x		
04	Link	nvarchar(max)		x		
05	IsRead	bit		x		
06	CreatedAt	datetime2		x		

Table 52: Notifications Table

2.22 Teams

No	Field Name	Type	Unique	Not Null	PK/ FK	Note
01	Id	int	x	x	PK	Identity (1,1)
02	CourseId	int			FK	FK to Courses(Id), Nullable
03	LeaderId	int		x	FK	FK to Users(Id)
04	TeamName	nvarchar(100)		x		Max Length: 100
05	Description	nvarchar(500)				Max Length: 500, Nullable
06	Gender	int				Nullable
07	ExpectedVolunteers	int		x		
08	DateCreated	datetime2		x		
09	DateModified	datetime2		x		
10	CreatedBy	int		x		
11	UpdatedBy	int		x		

Table 53: Teams Table

3. Detailed Design

3.1 Class Diagram

3.1.1 Account Management

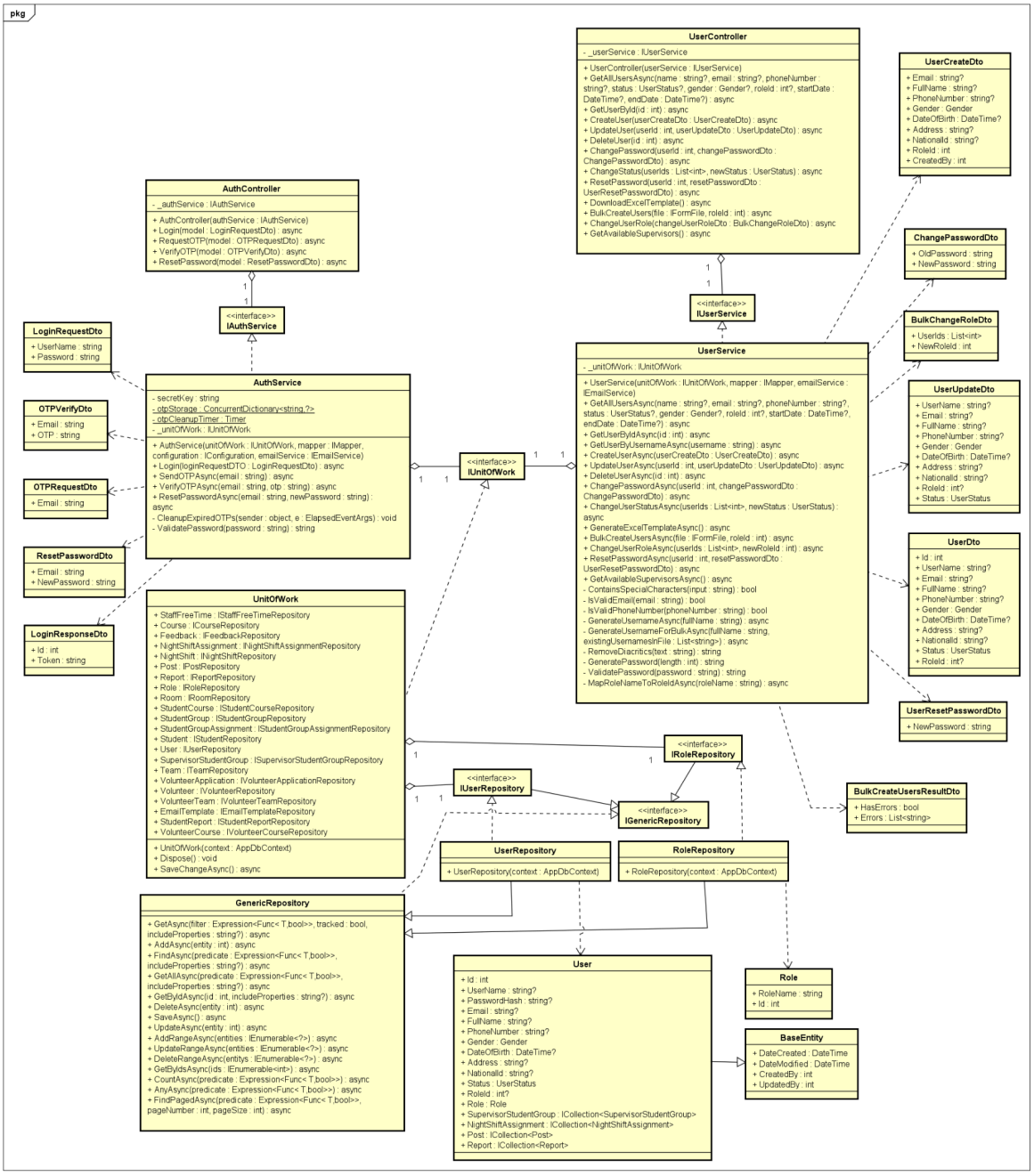

Figure 6: Account management class diagram

The link of image: [Account Management]

3.1.2 Course Management

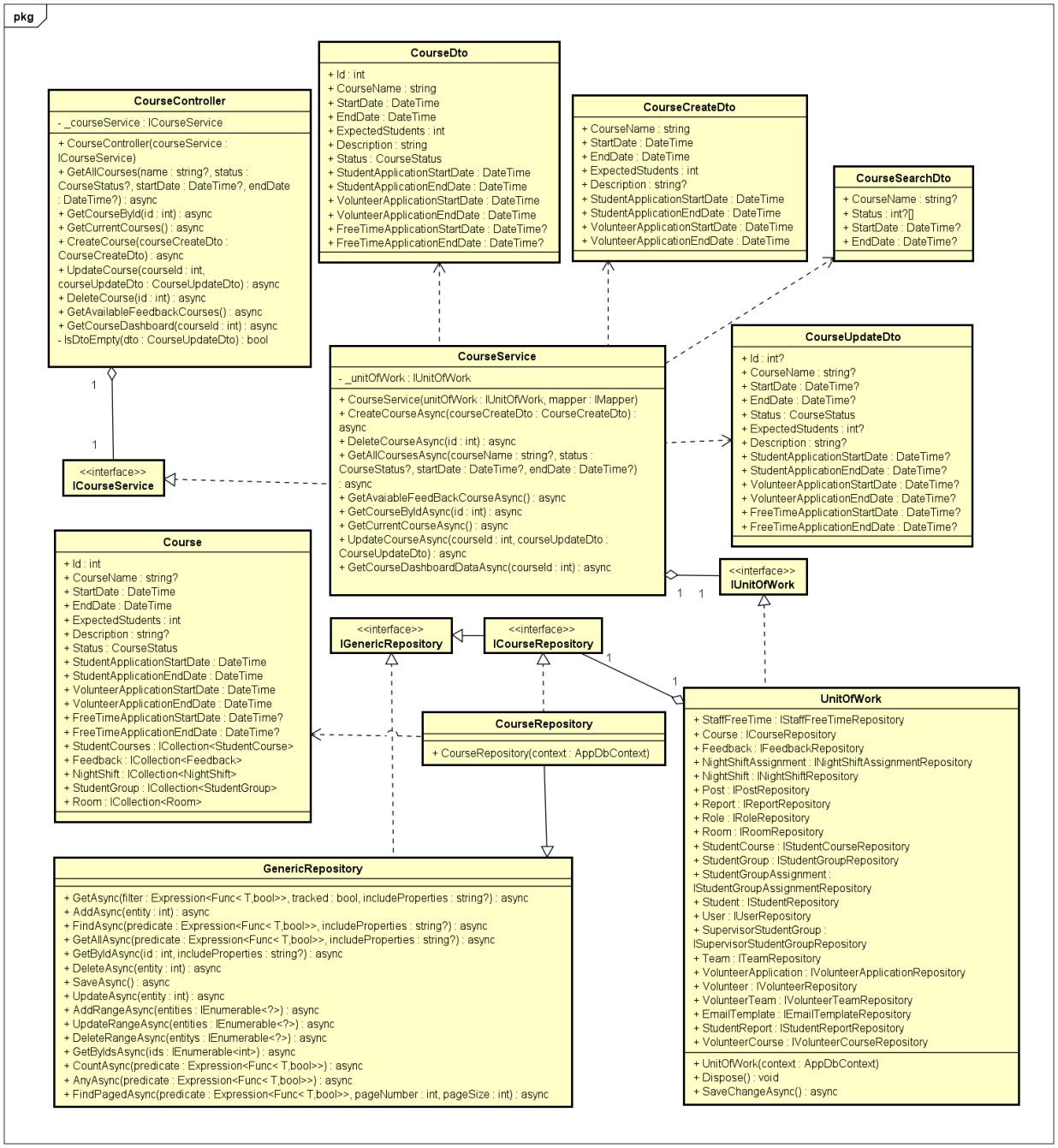

Figure 6: Account management class diagram

The link of image: [Course Management]

3.1.3 Student Management

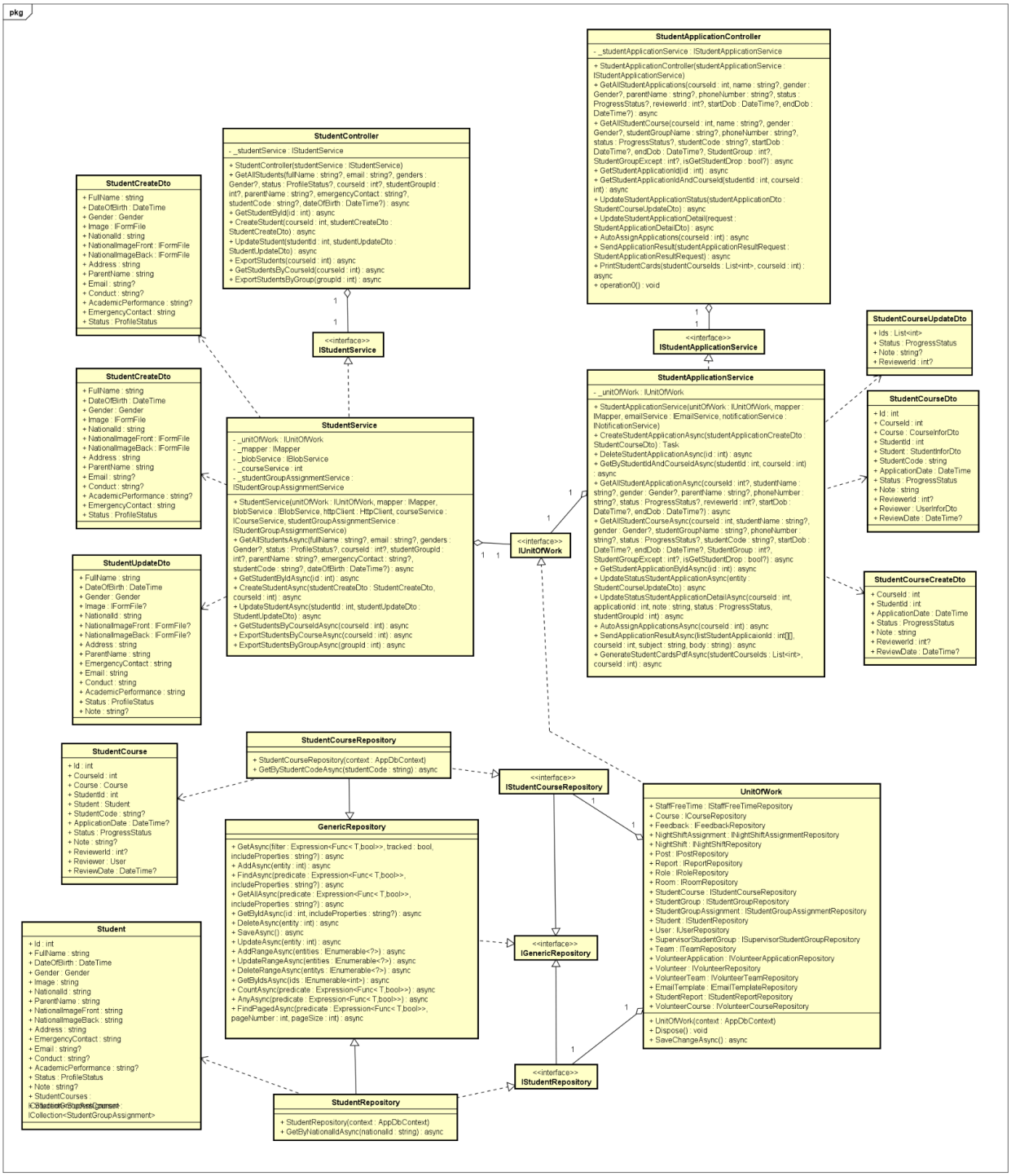

Figure 7: Student management class diagram

The link of image: [Student Management]

3.1.4 Volunteer Management

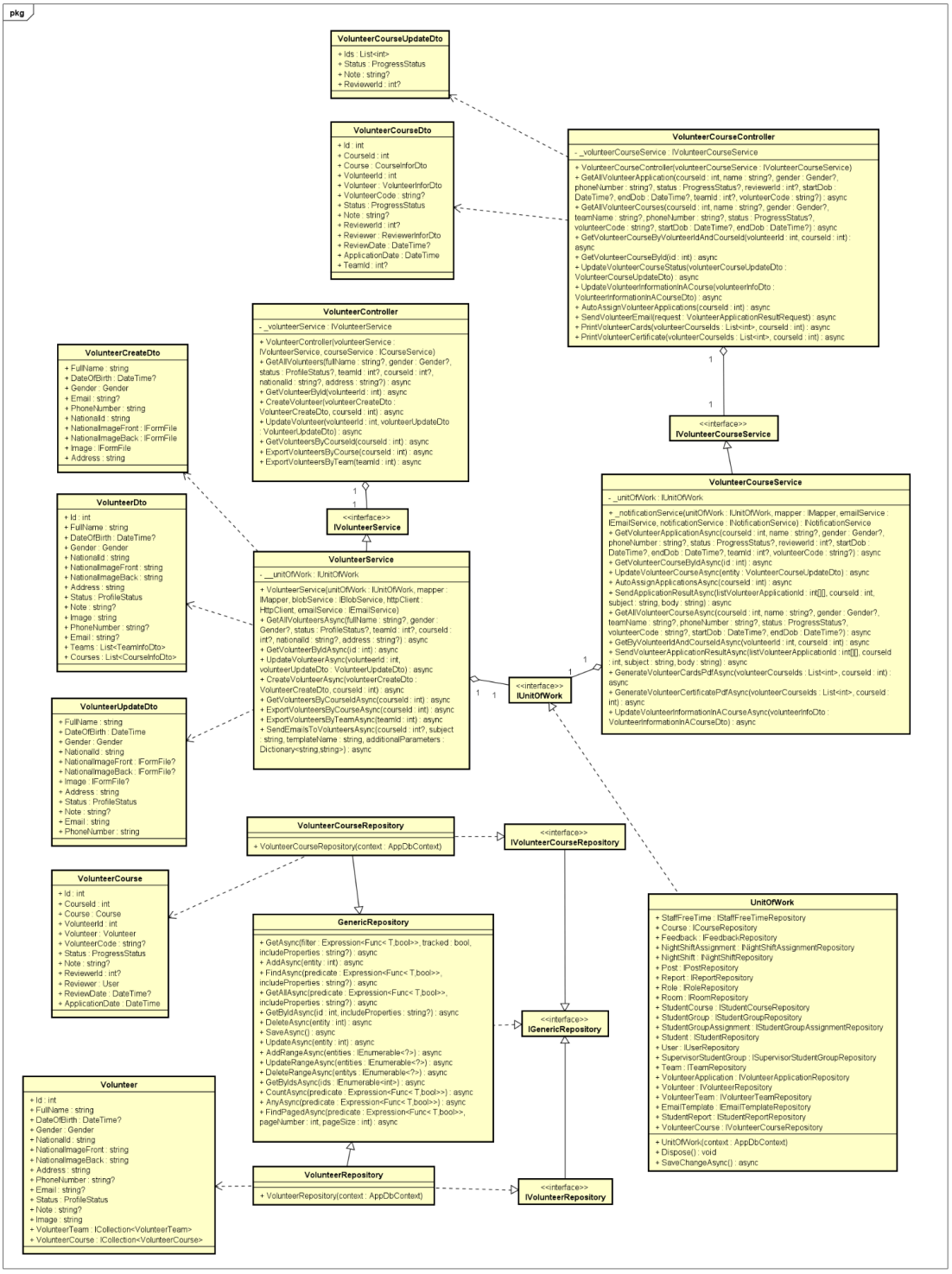

Figure 8: Volunteer management class diagram

The link of image: [Volunteer Management]

3.1.5 Report Management

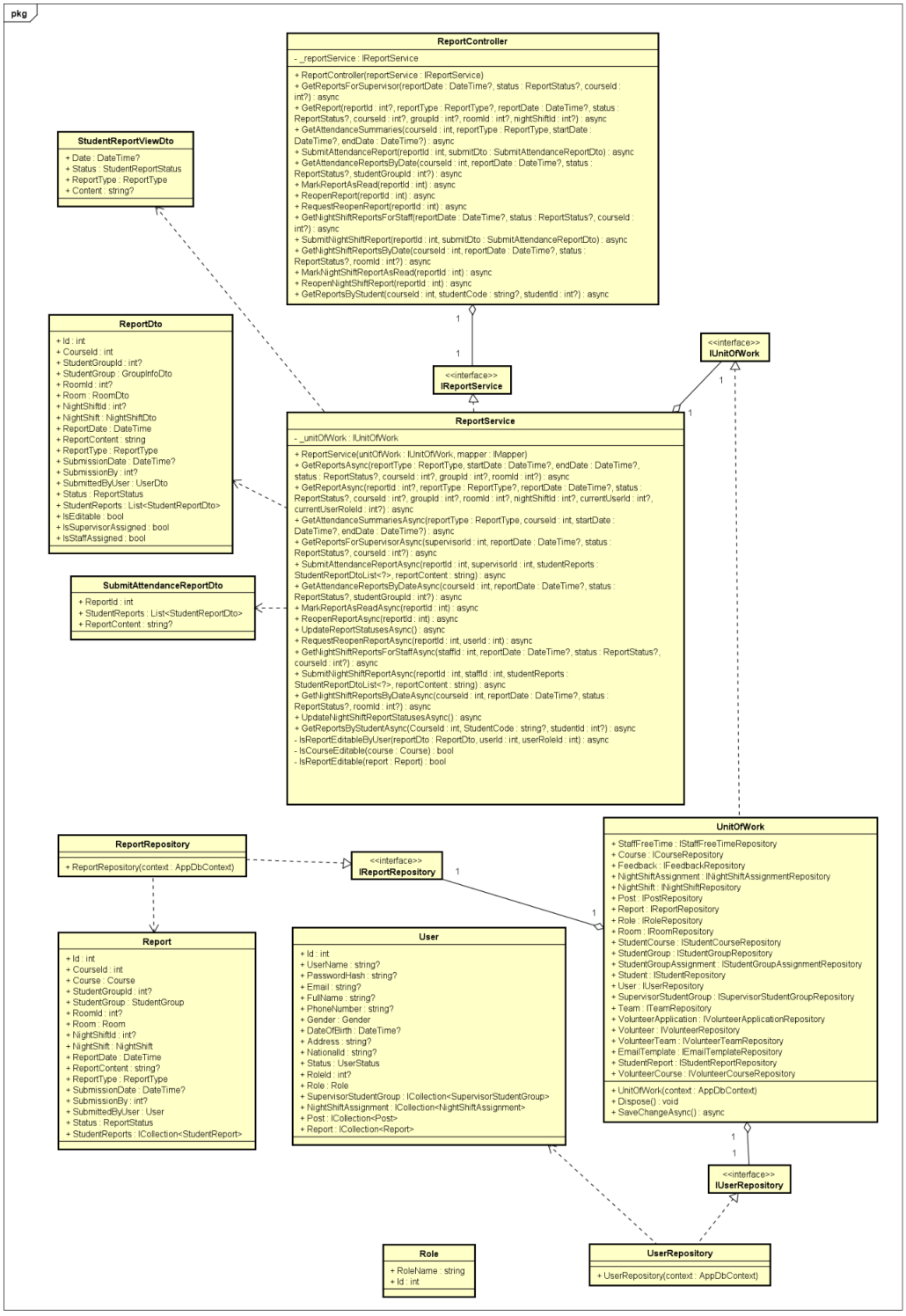

Figure 9: Report management class diagram

The link of image: [Report Management]

3.1.6 Post Management

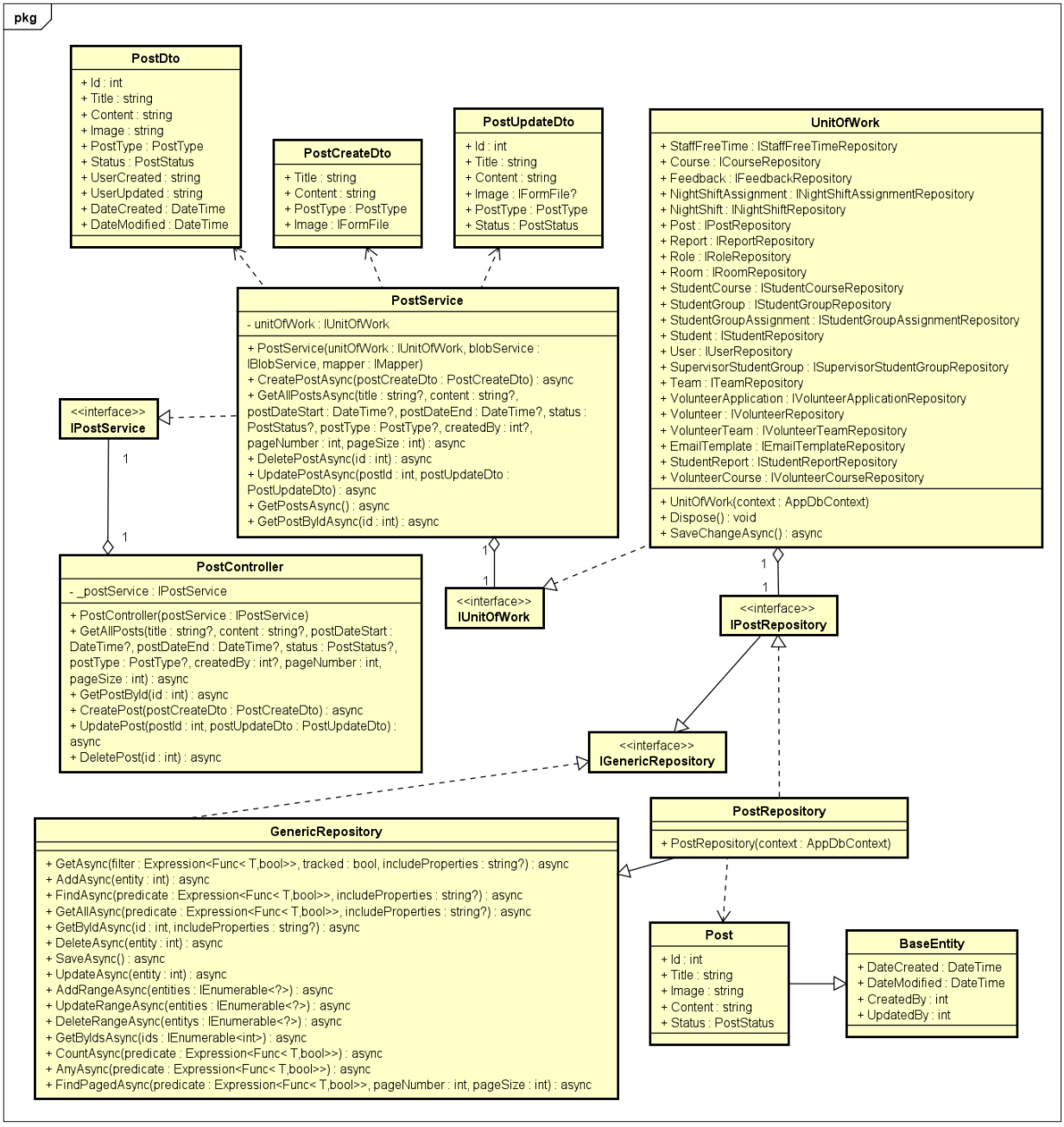

Figure 10: Post management class diagram

The link of image: [Post Management]

3.1.7 Supervisor Management

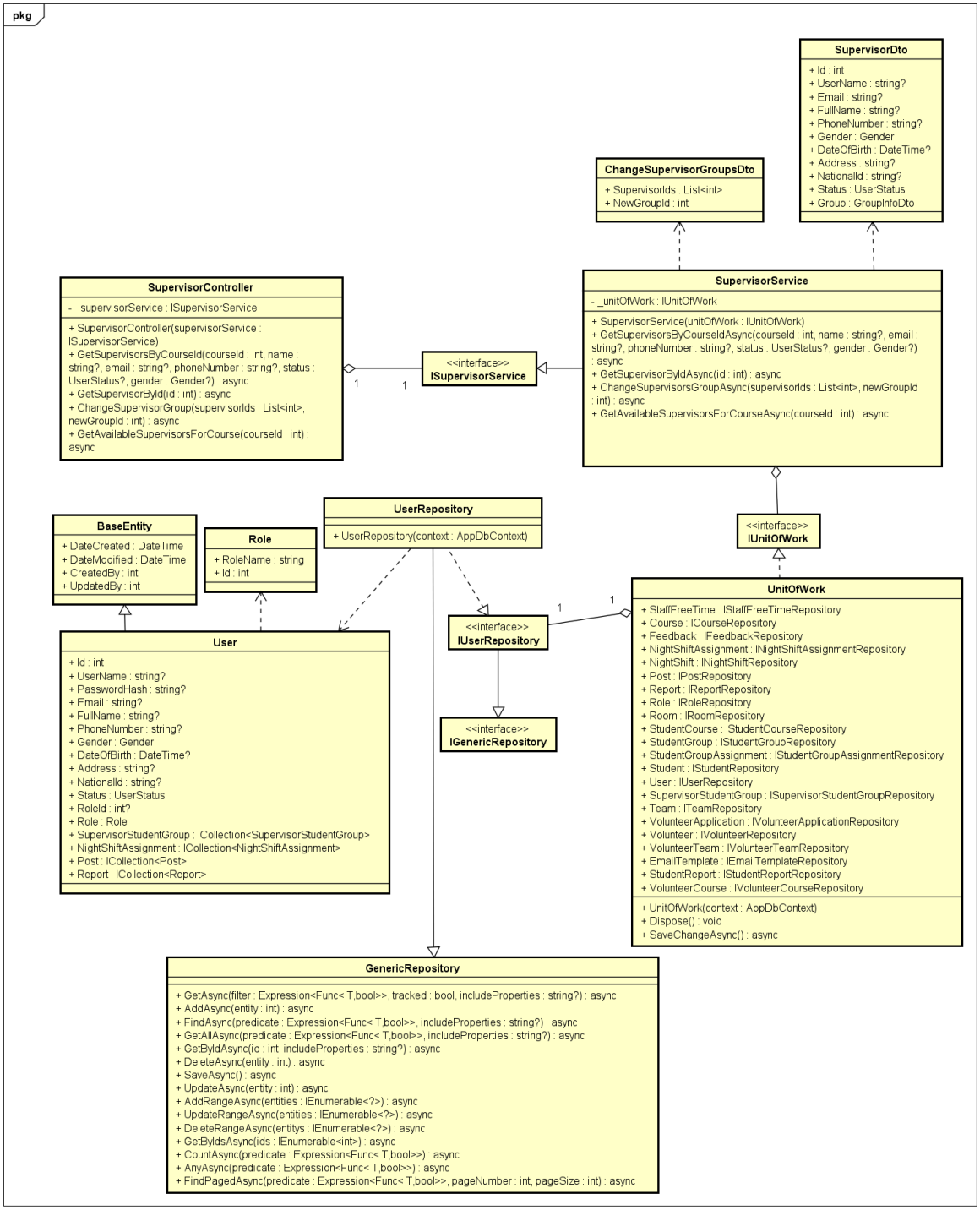

Figure 11: Supervisor management class diagram

The link of image: [Supervisor Management]

3.1.8 Student Group Management

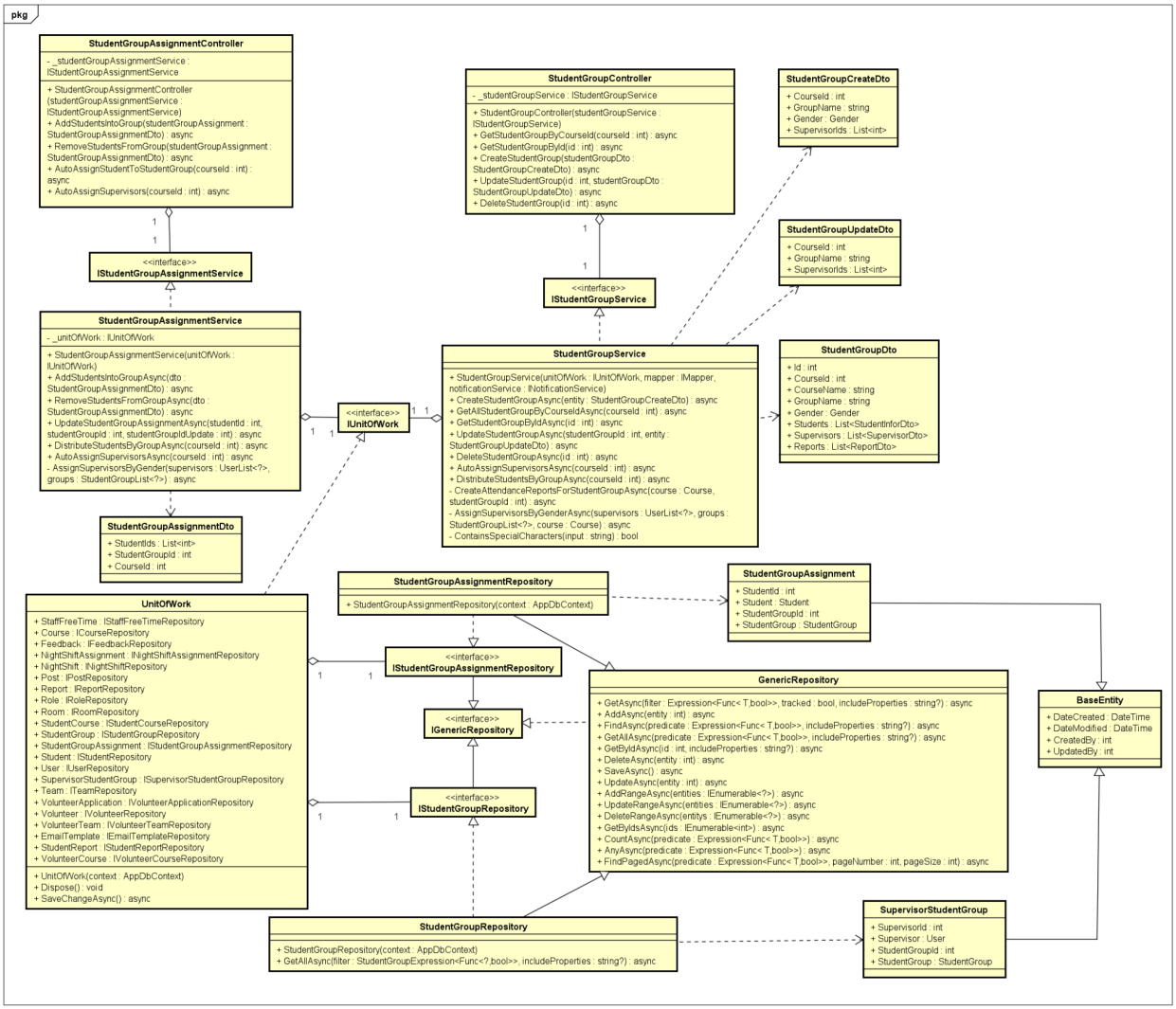

Figure 12: Student group management class diagram

The link of image: [Student Group Management]

3.1.9 Team Management

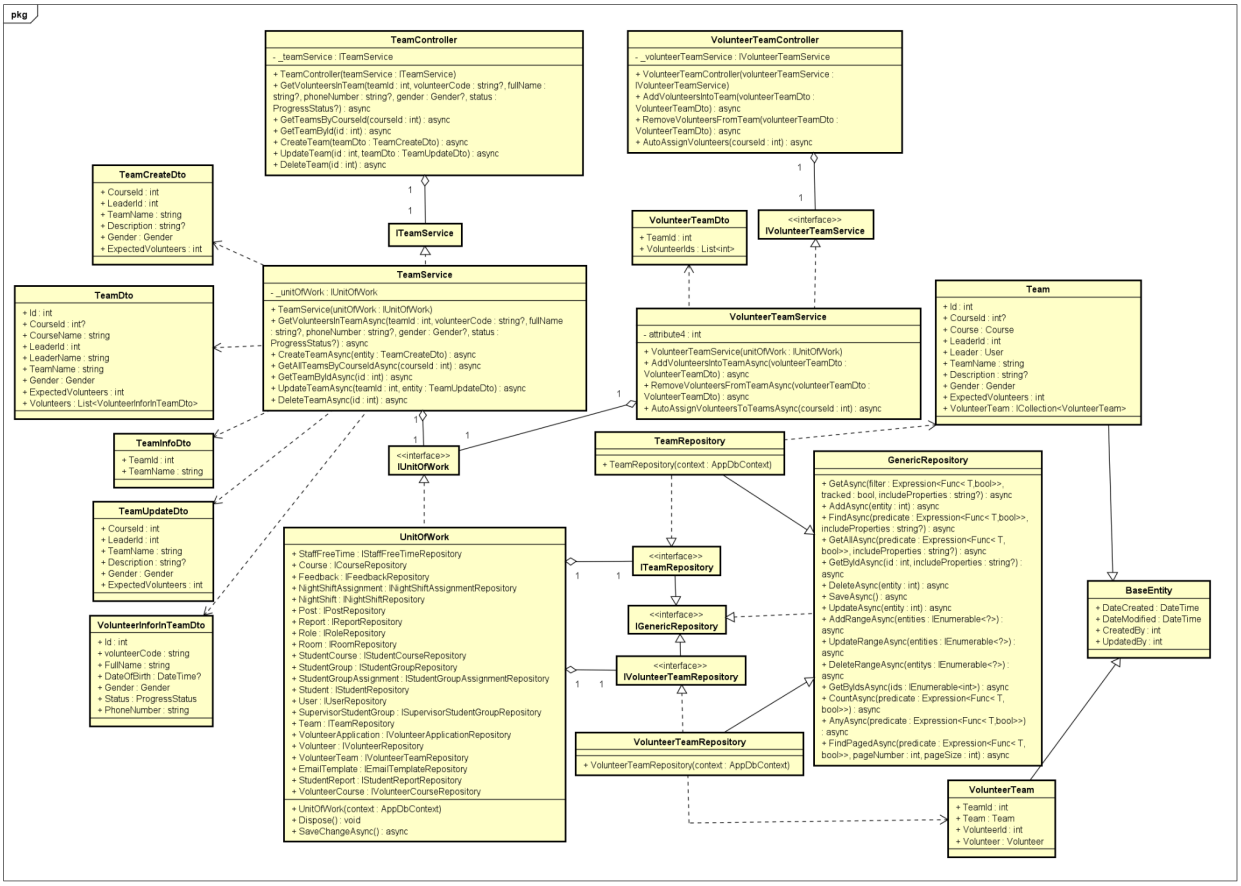

Figure 13: Team management class diagram

The link of image: [Team Management]

3.1.10 Feedback Management

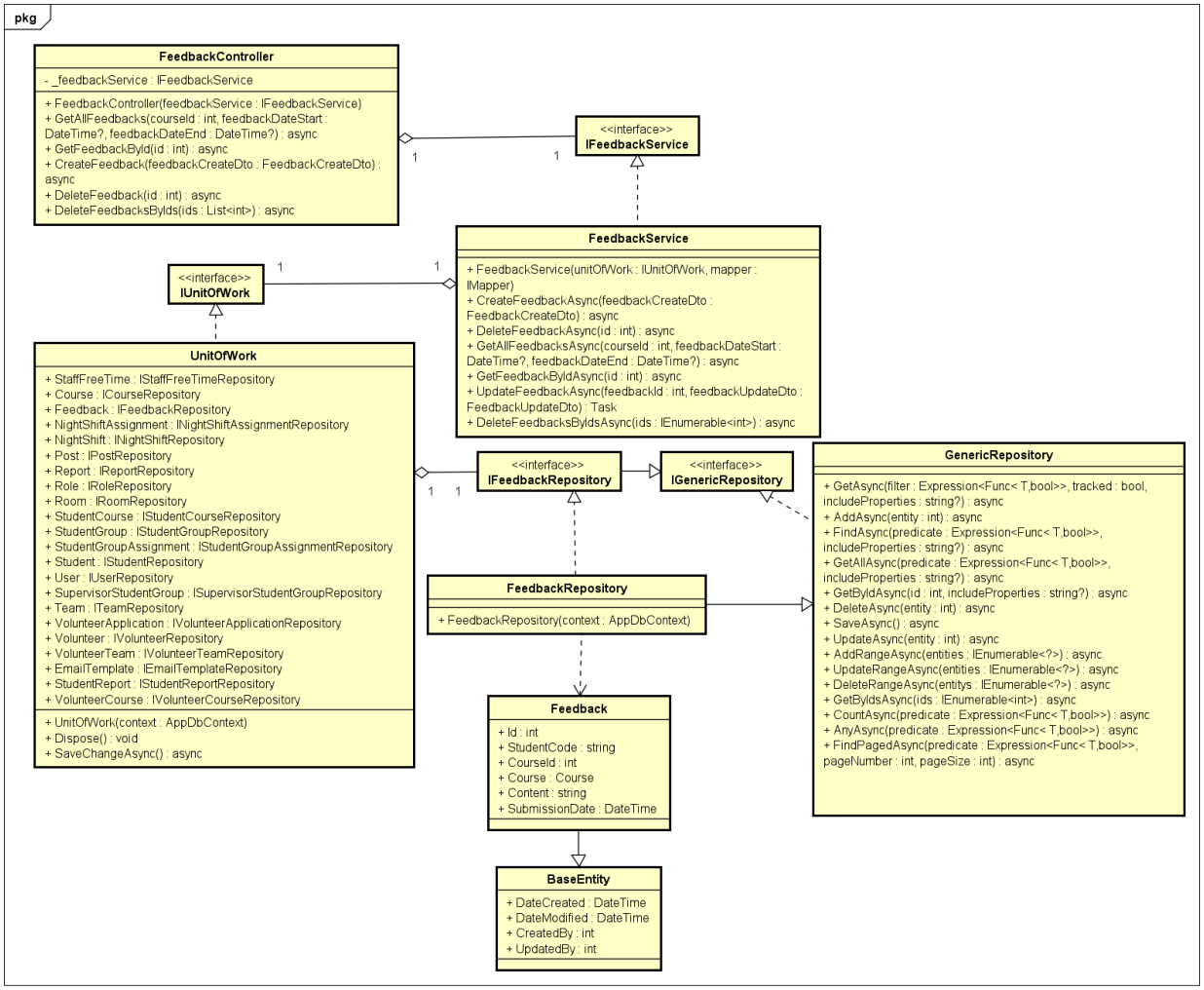

Figure 14: Feedback management class diagram

The link of image: [Feedback Management]

3.1.11 Night Shift Management

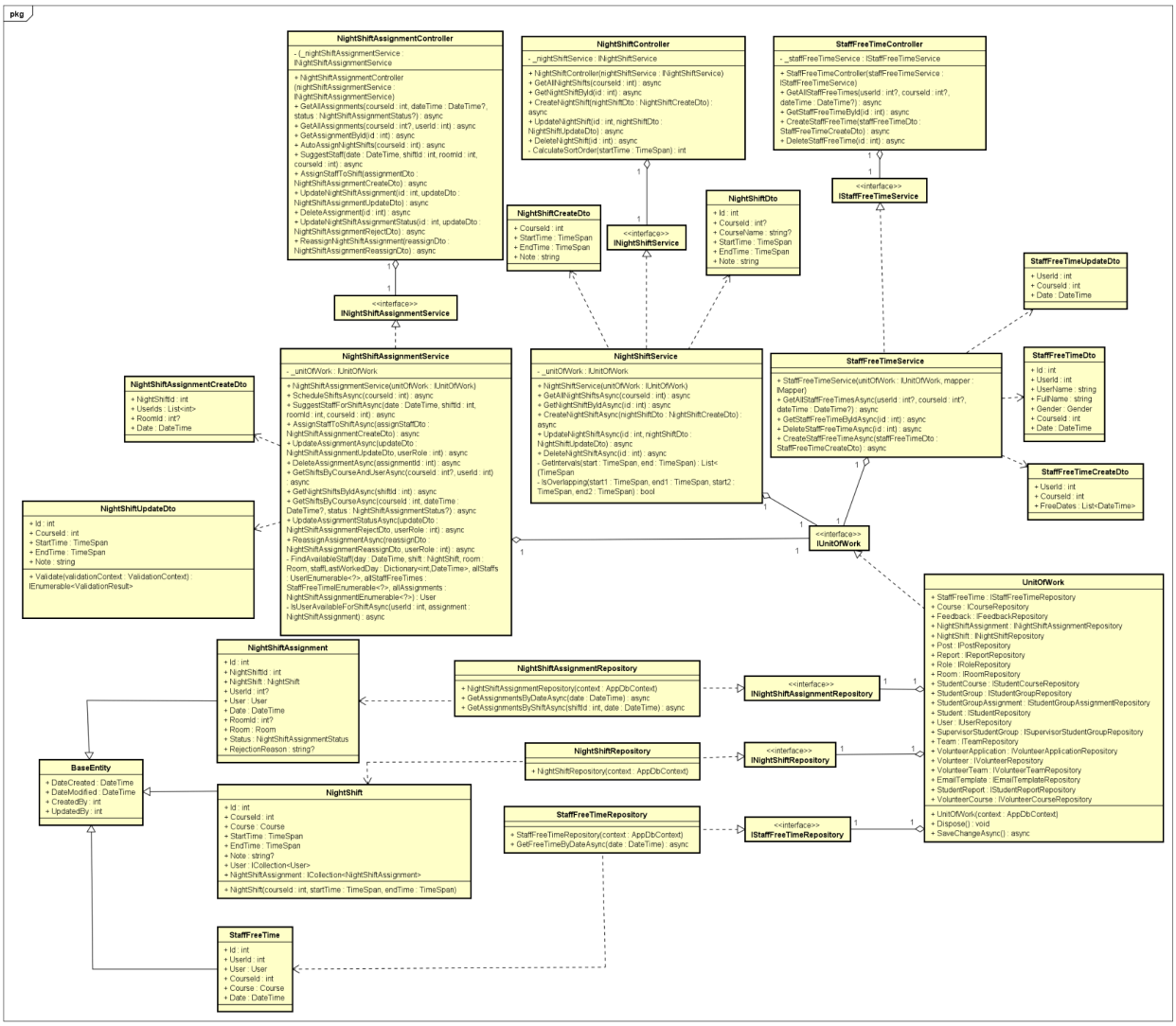

Figure 15: Night shift management class diagram

The link of image: [Night Shift Management]

3.2 Sequence Diagram

3.2.1 Login

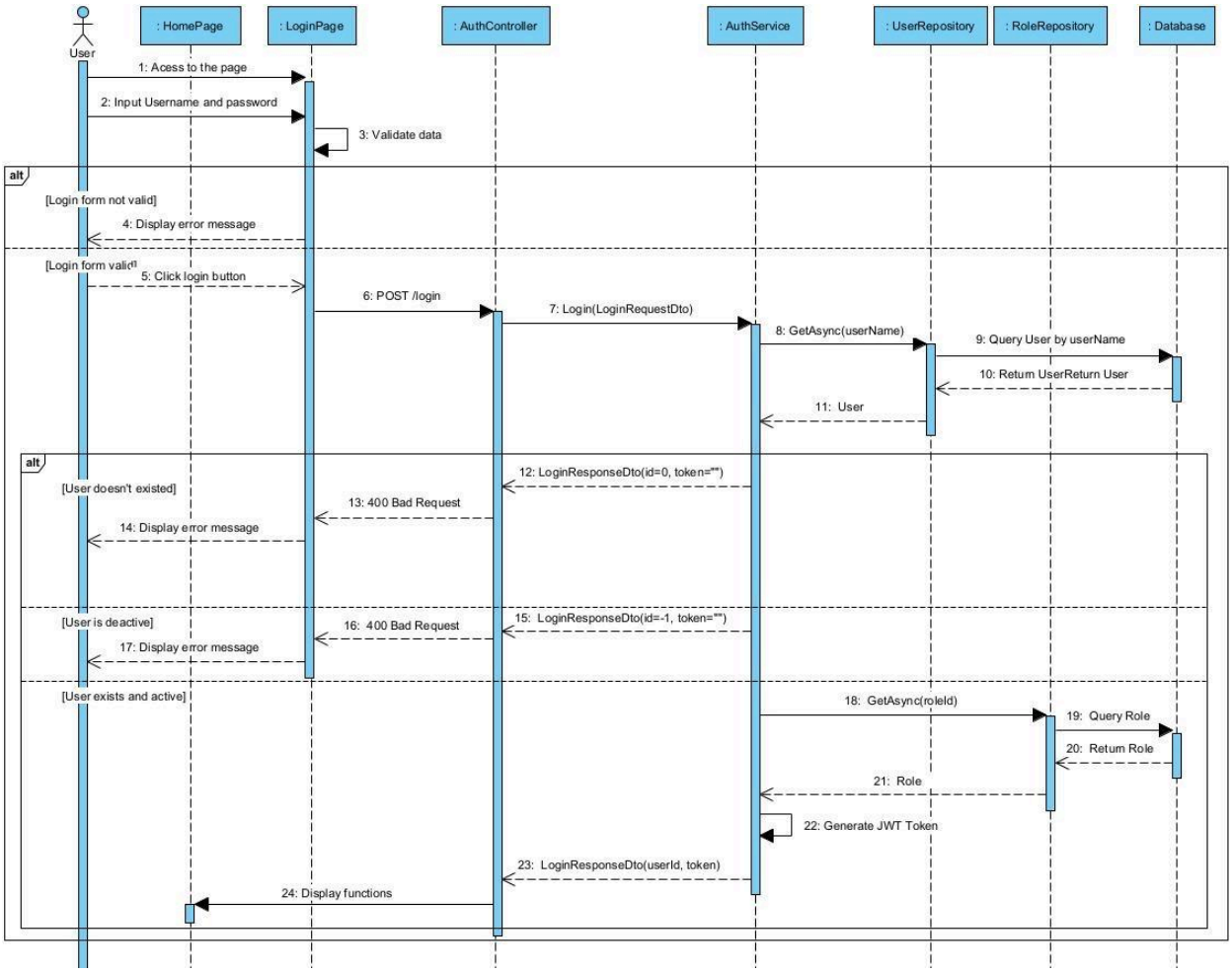

Figure 16: Login sequence diagram

The link of image: Login

3.2.2 Create course

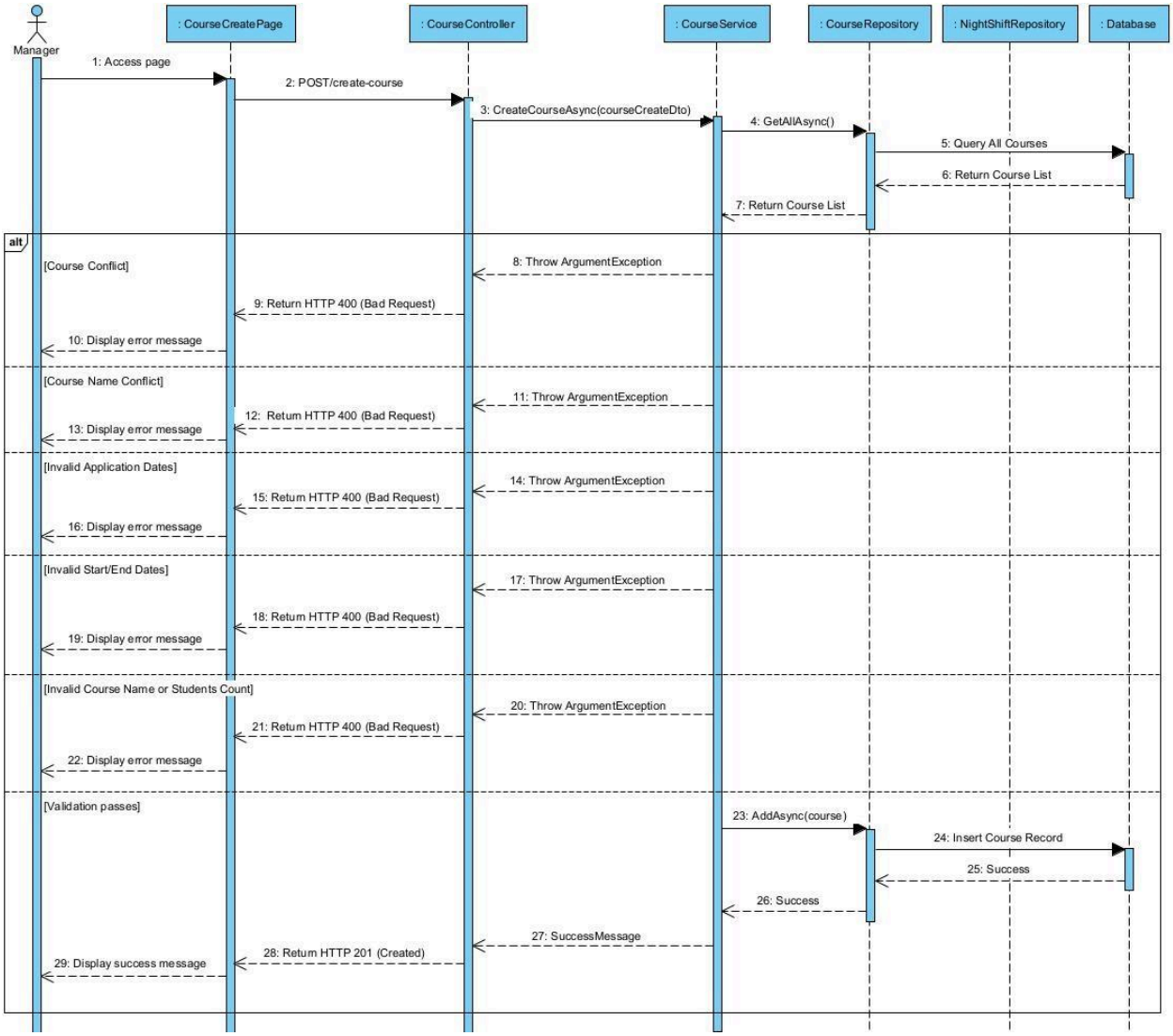

Figure 17: Create course sequence diagram

The link of image: [Create course]

3.2.3 Add student group

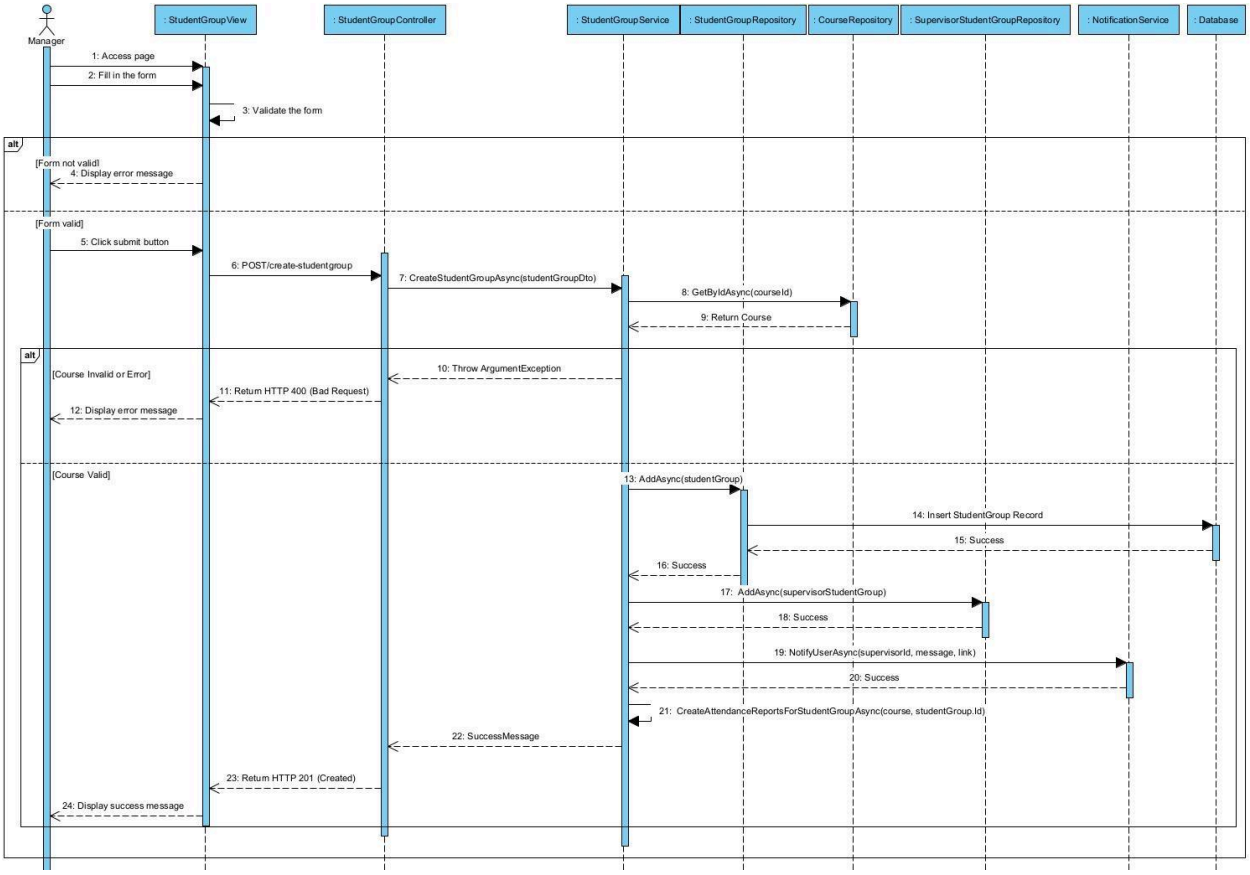

Figure 18: Add student group sequence diagram

The link of image: [Add student group]

3.2.4 Add team

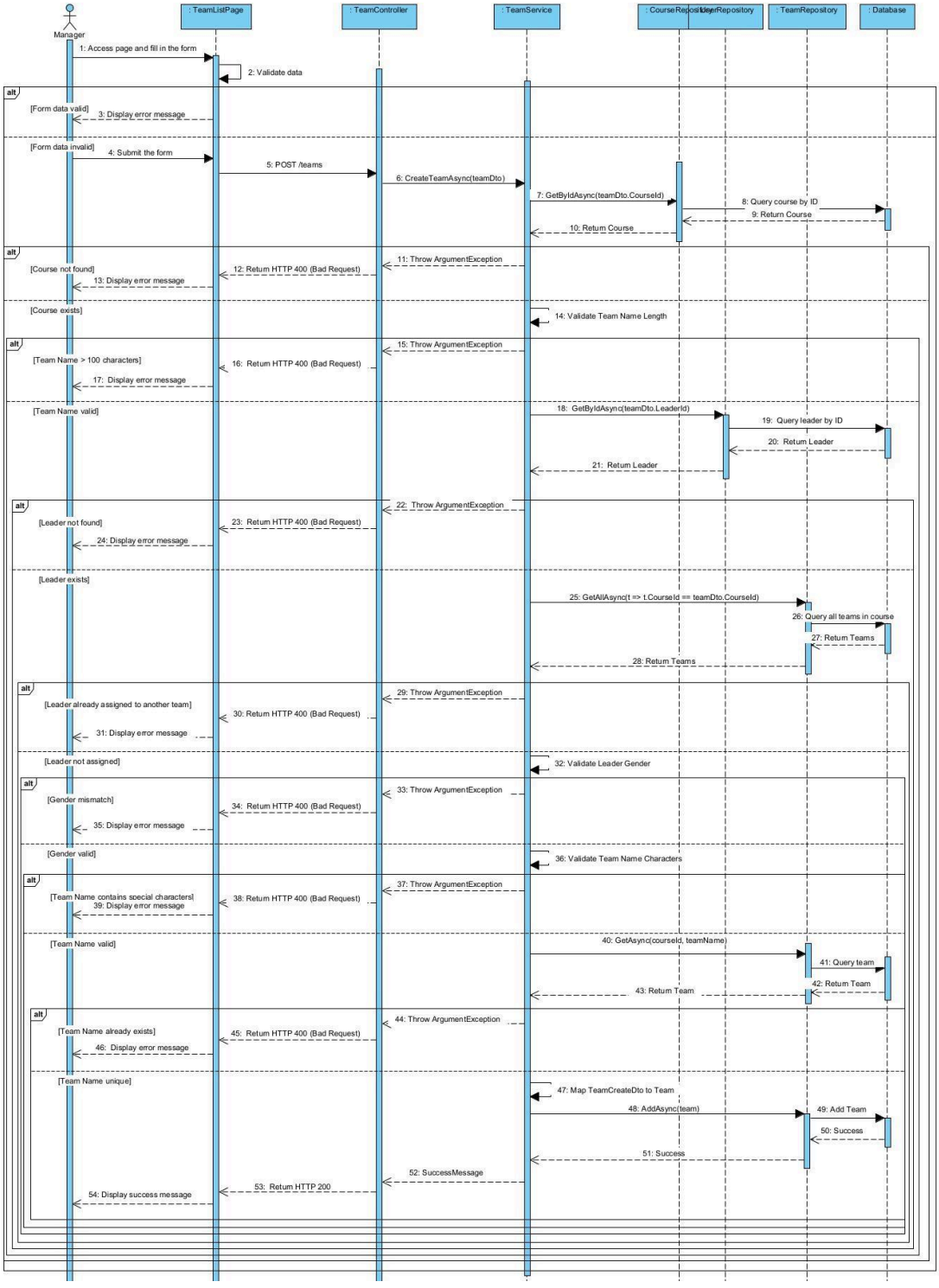

Figure 19: Add team sequence diagram

The link of image: [Add team]

3.2.5 Add supervisors

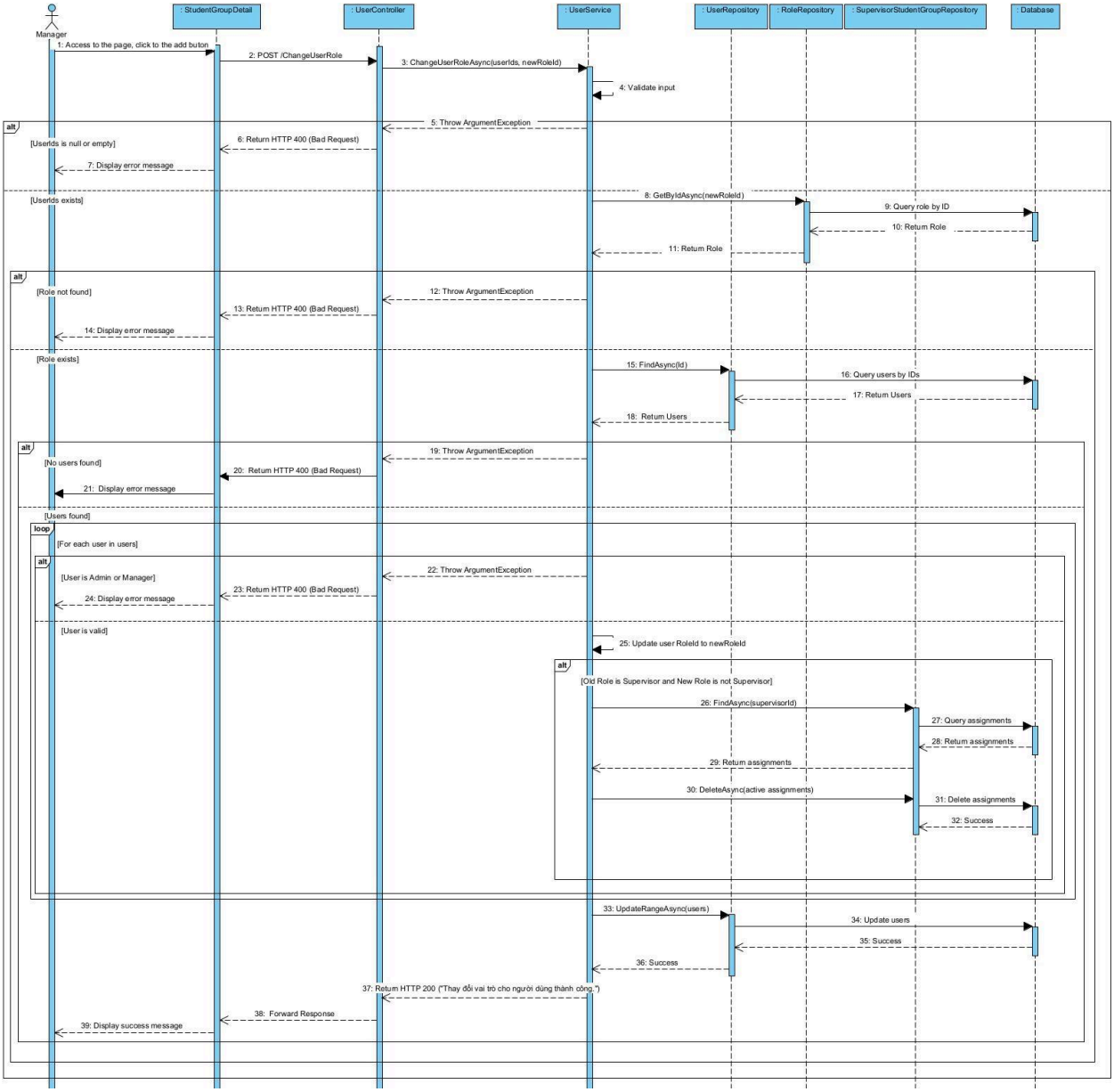

Figure 20: Add supervisors sequence diagram

The link of image: [Add supervisors]

3.2.6 Manually assign supervisors to student group

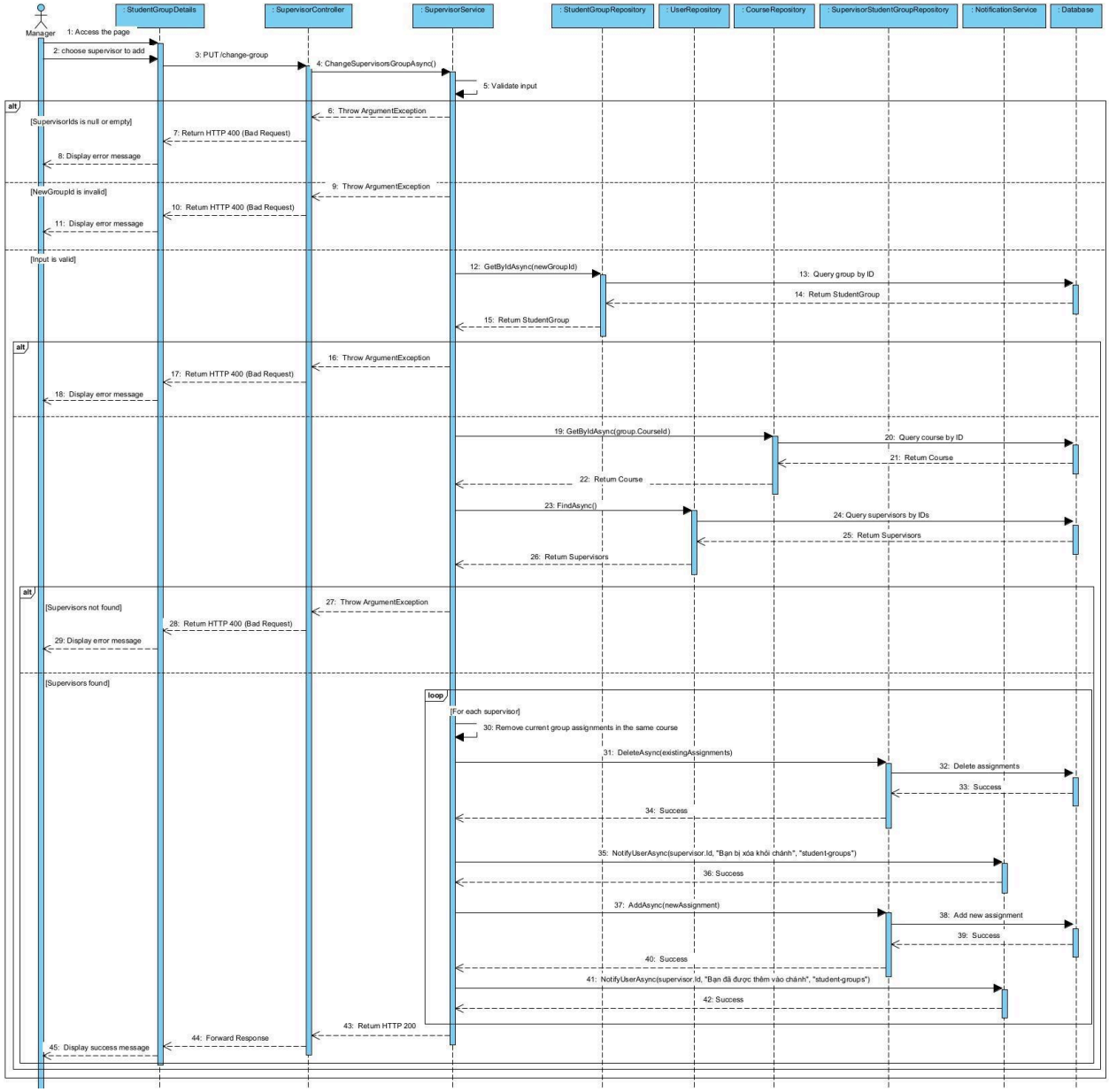

Figure 21: Manually assign supervisors to student group sequence diagram

The link of image: [Manually assign supervisors]

3.2.7 Automatically assign supervisors to student group

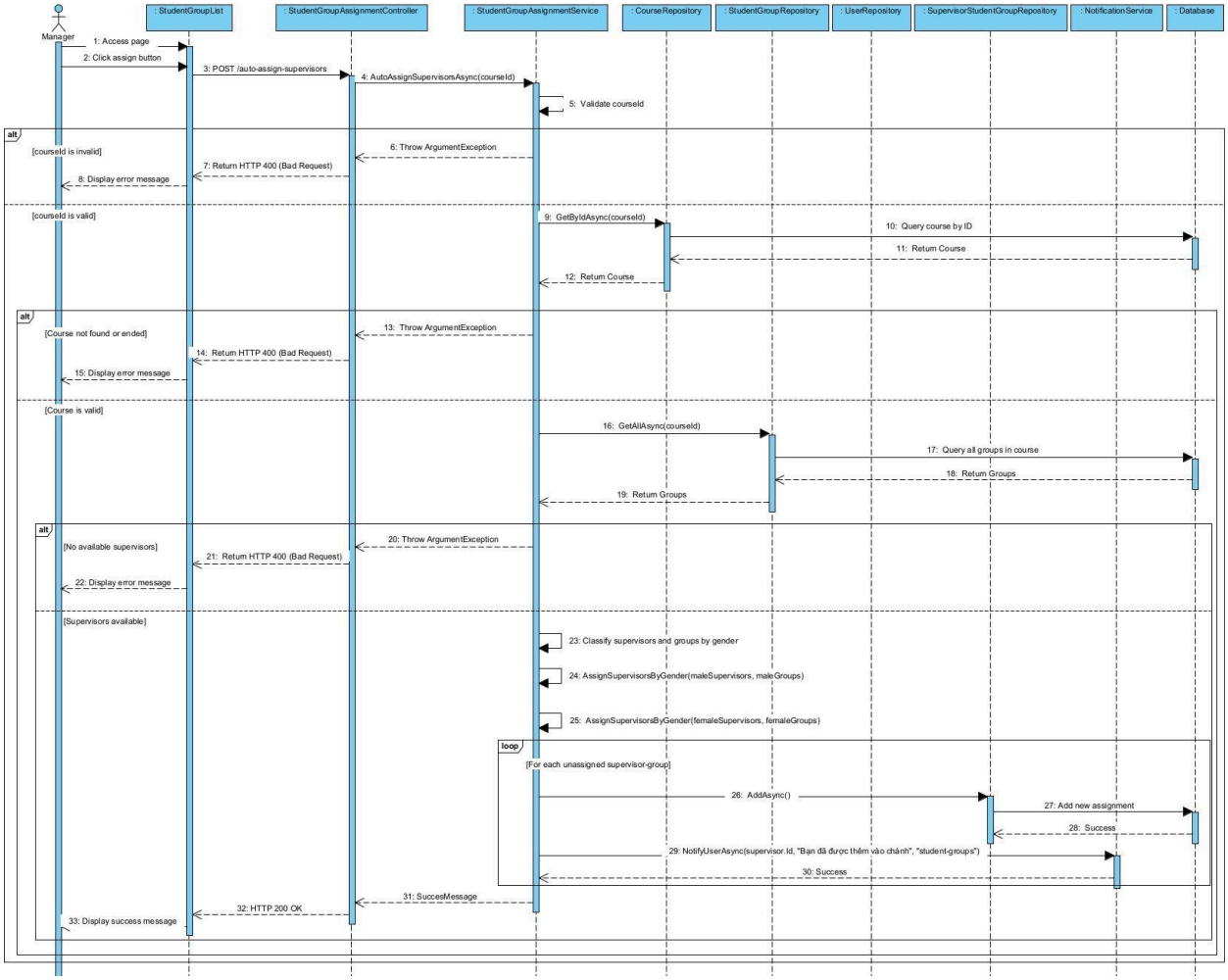

Figure 22: Automatically assign supervisors to student group sequence diagram

The link of image: [Automatically assign supervisors]

3.2.8 Submit student registration

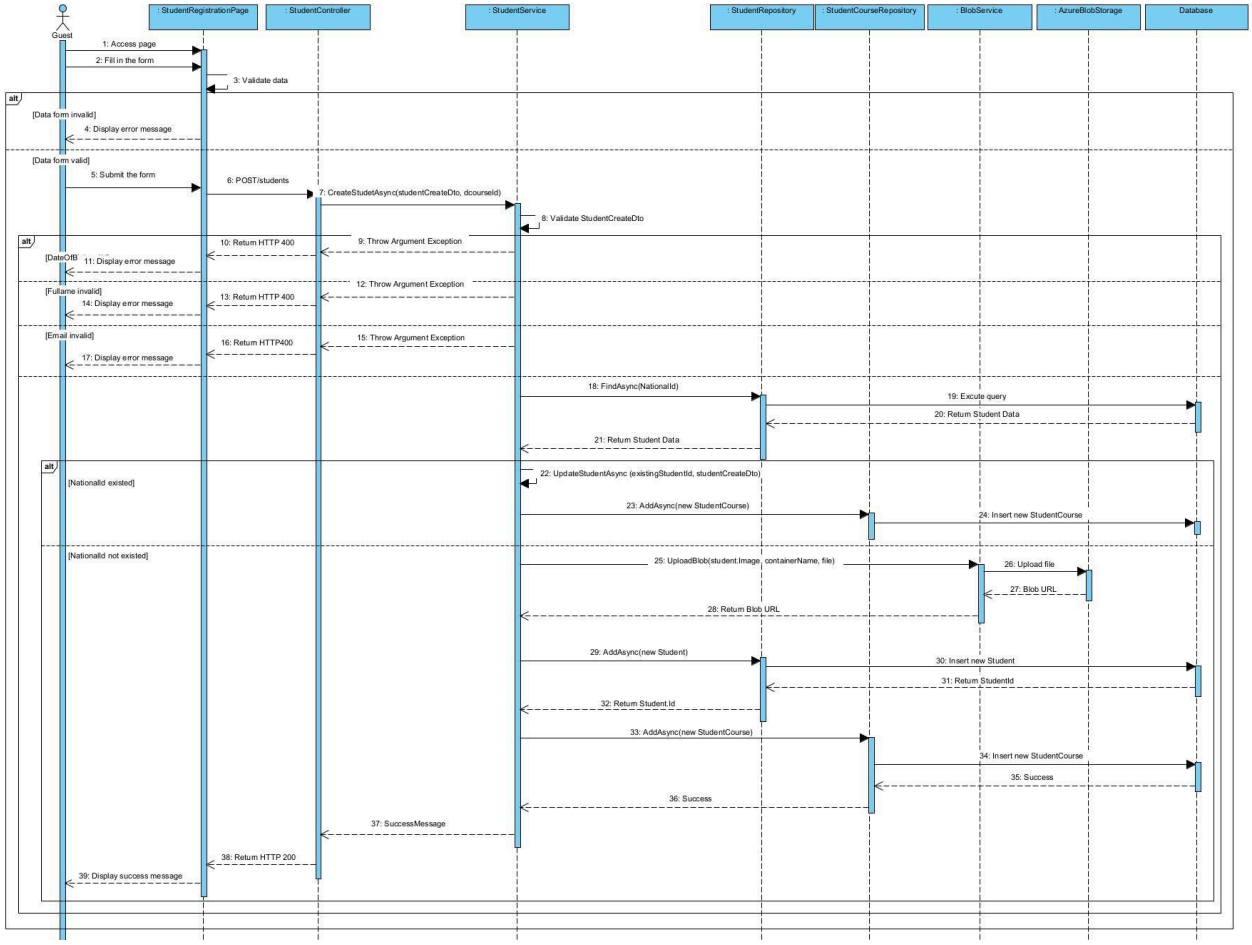

Figure 23: Submit student registration sequence diagram

The link of image: [Submit student registration]

3.2.9 Manually assign student registration forms to secretary

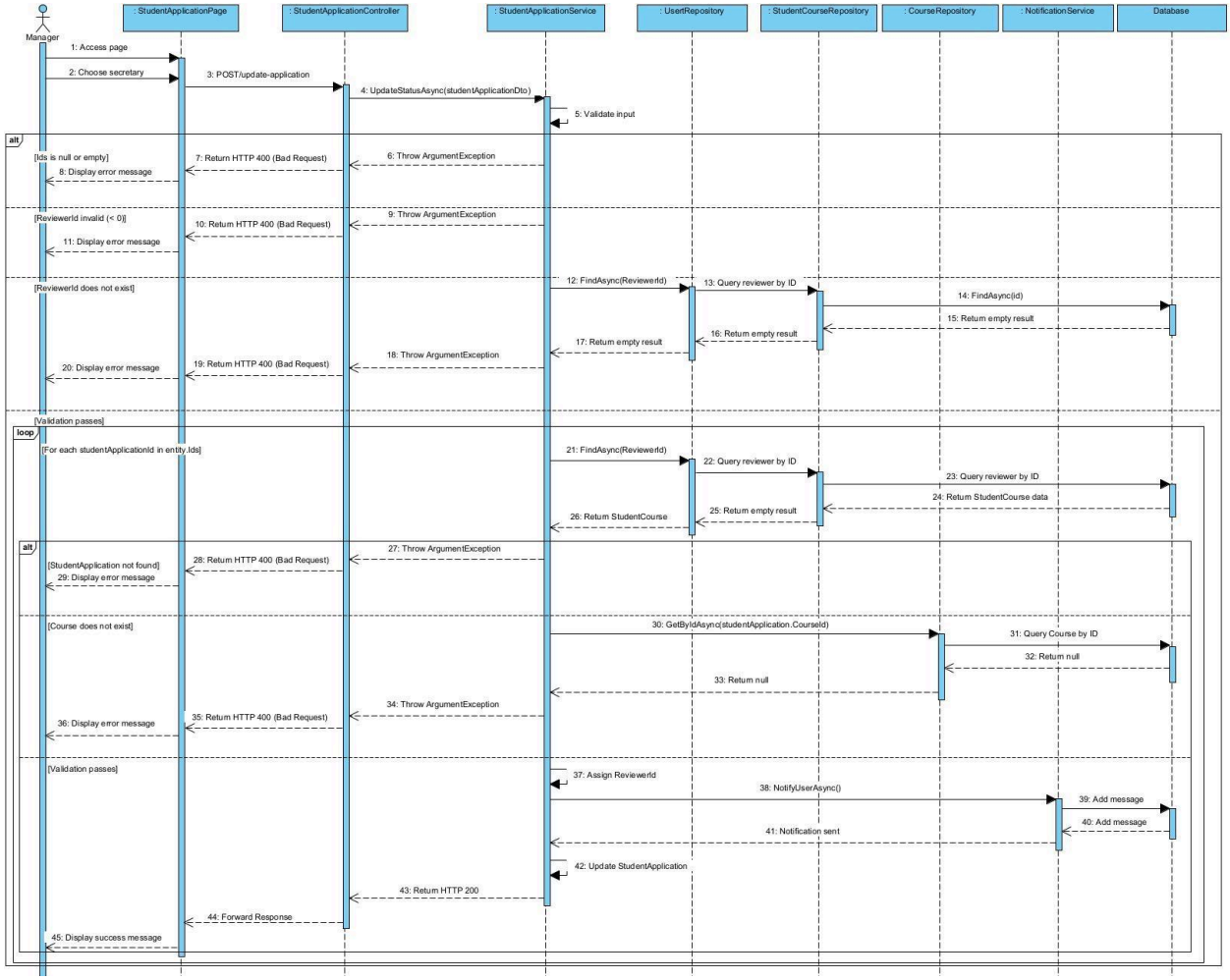

Figure 24: Manually assign student registration forms to secretary sequence diagram

The link of image: [Manually assign student registration]

3.2.10 Automatically student registration forms to secretaries

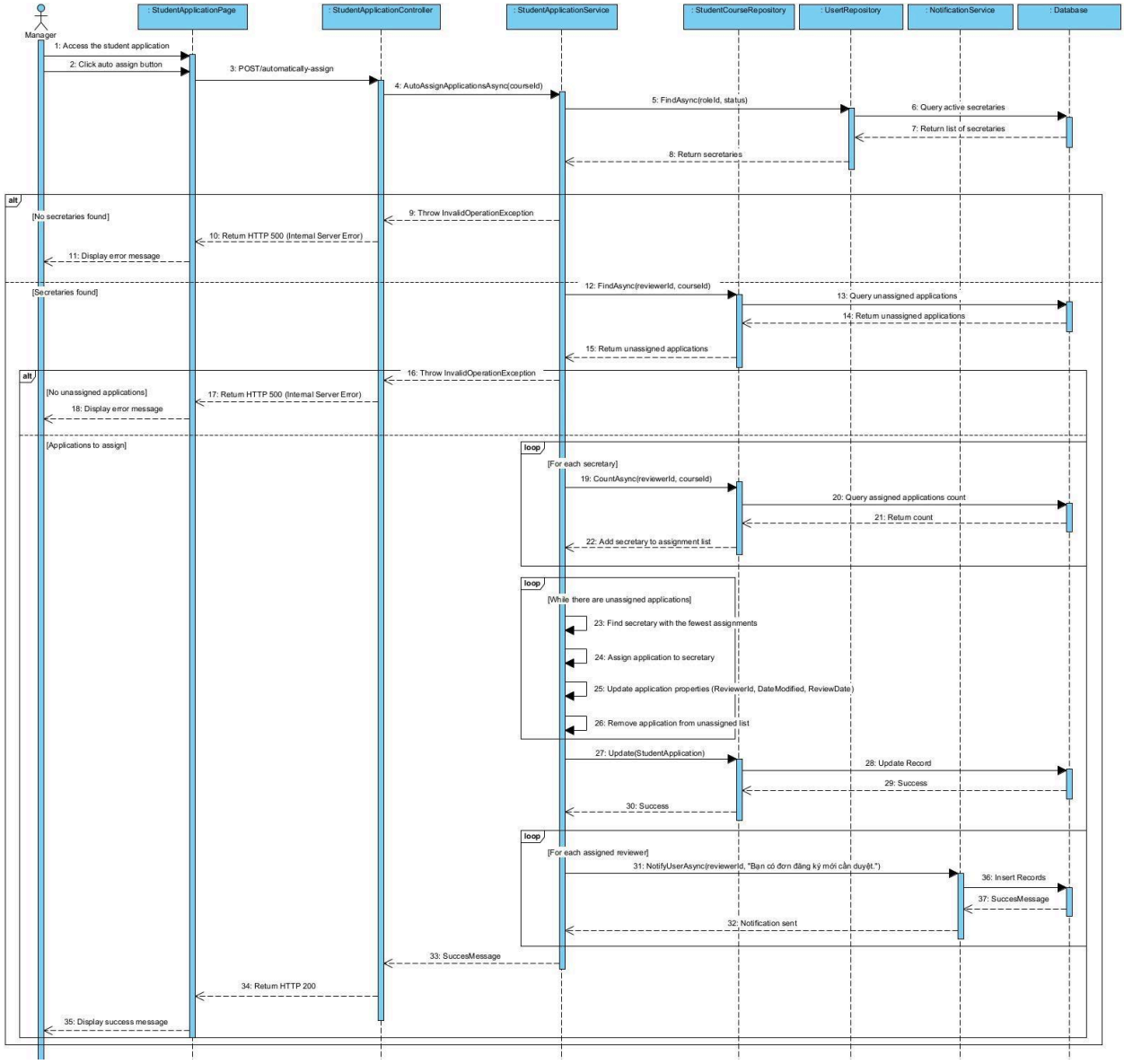

Figure 25: Automatically assign student registration forms to secretaries sequence diagram

The link of image: [Automatically assign student registration]

3.2.11 Accept/Reject student registration

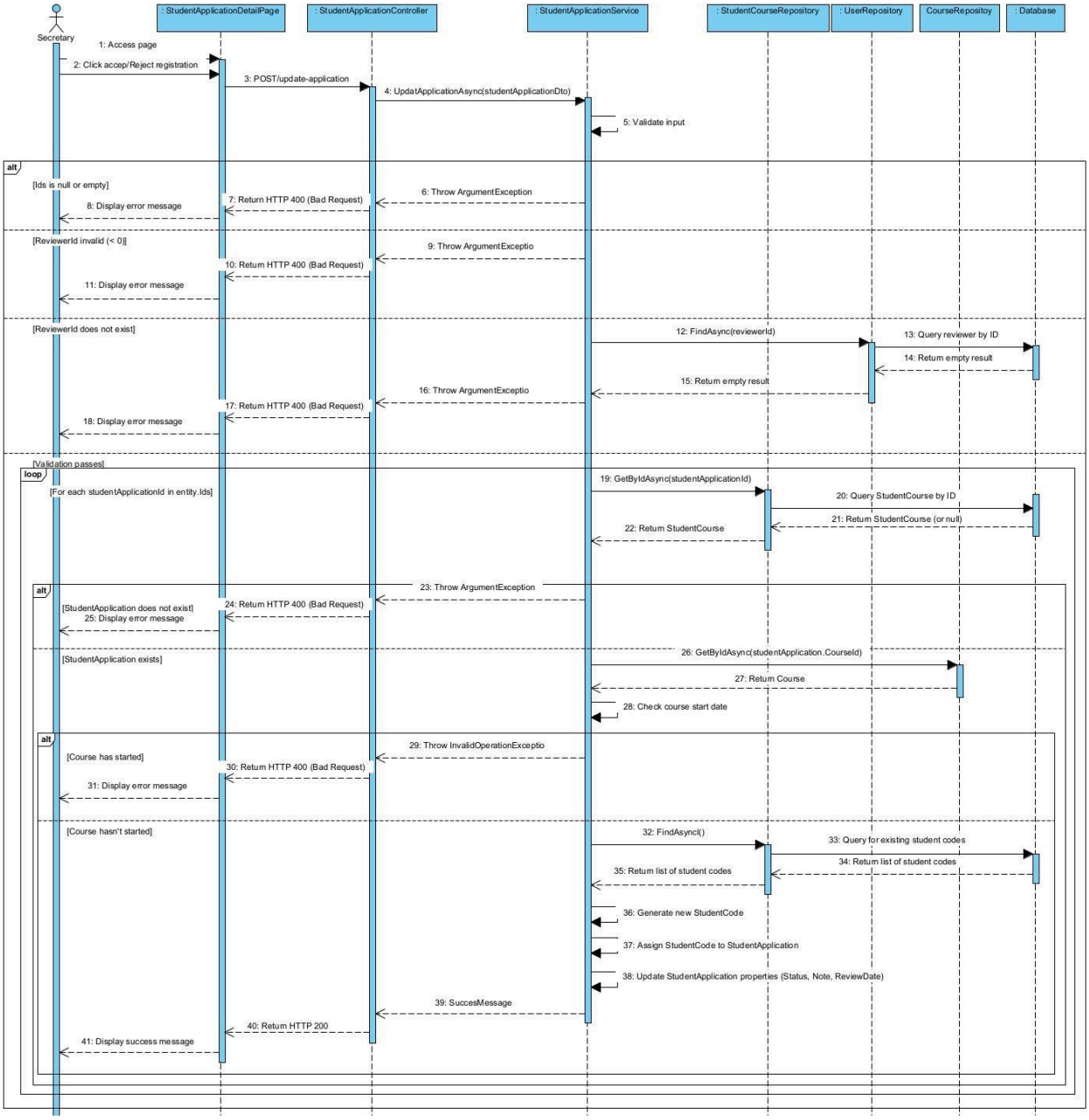

Figure 26: Accept/Reject student registration sequence diagram

The link of image: [Accept/Reject student registration]

3.2.12 Manually assign student to student group

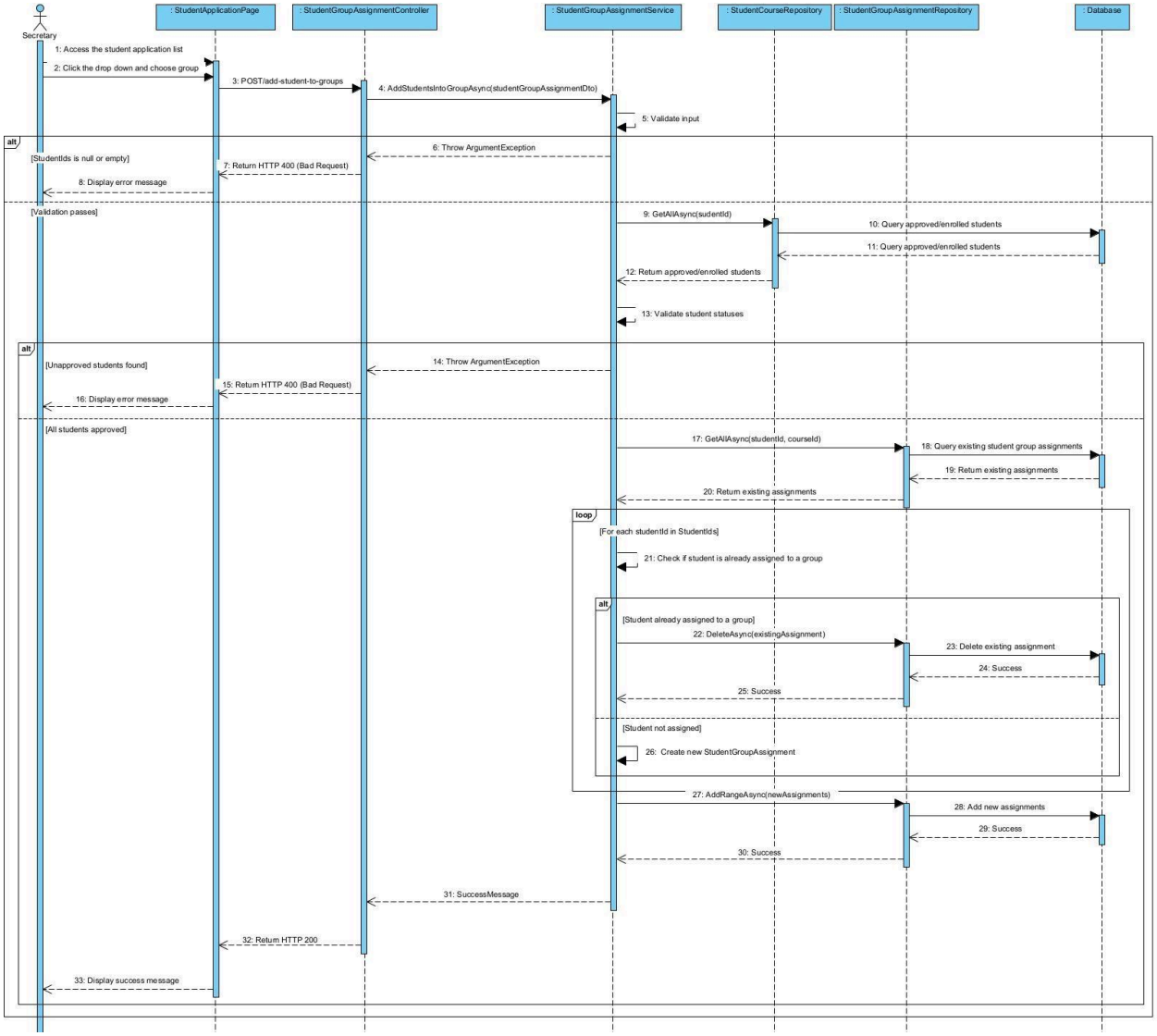

Figure 27: Manually assign student to student group sequence diagram

The link of image: [Manually assign student]

3.2.13 Automatically assign students to student groups

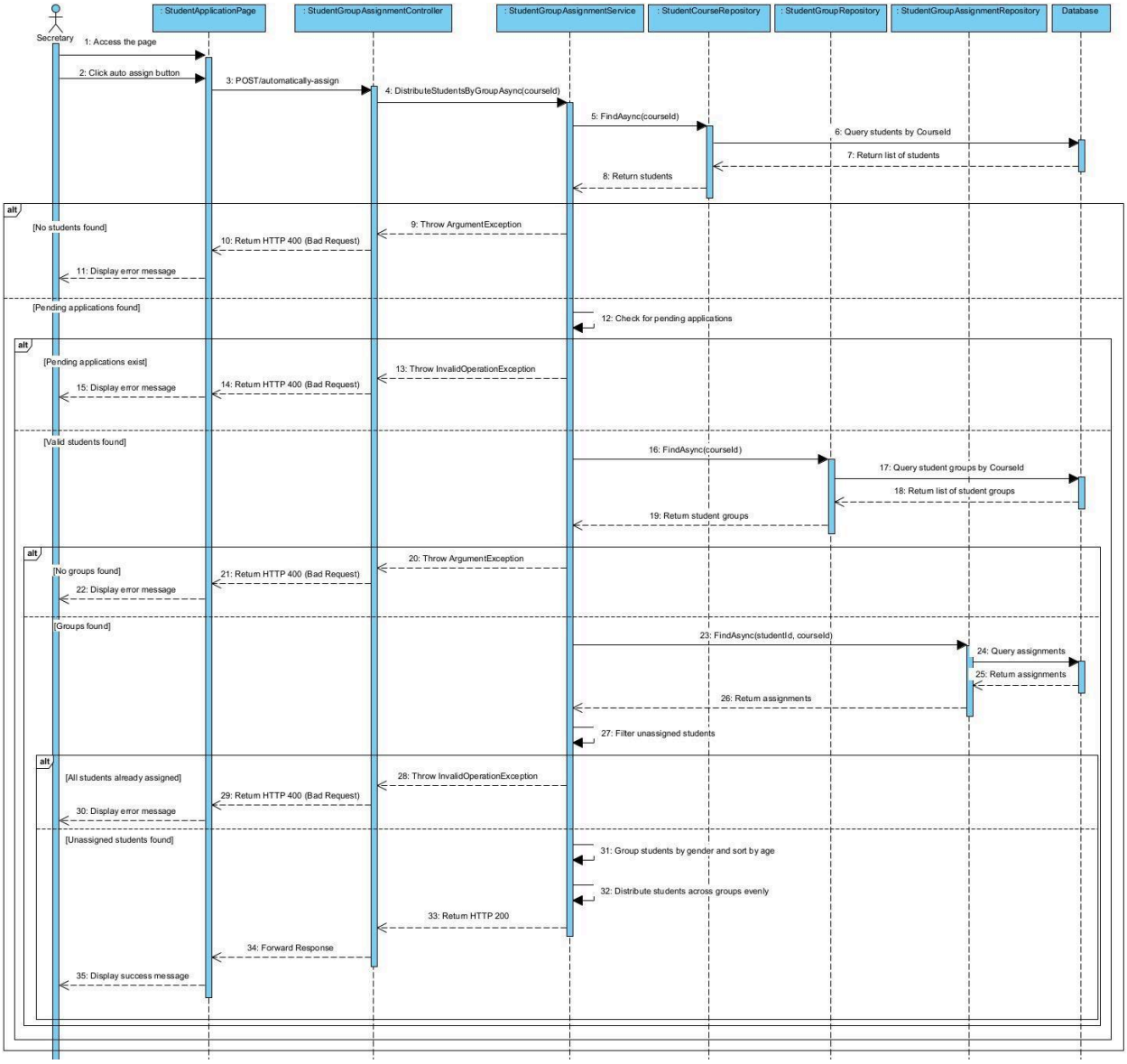

Figure 28: Automatically assign students to student groups sequence diagram

The link of image: [Automatically assign students]

3.2.14 Send email

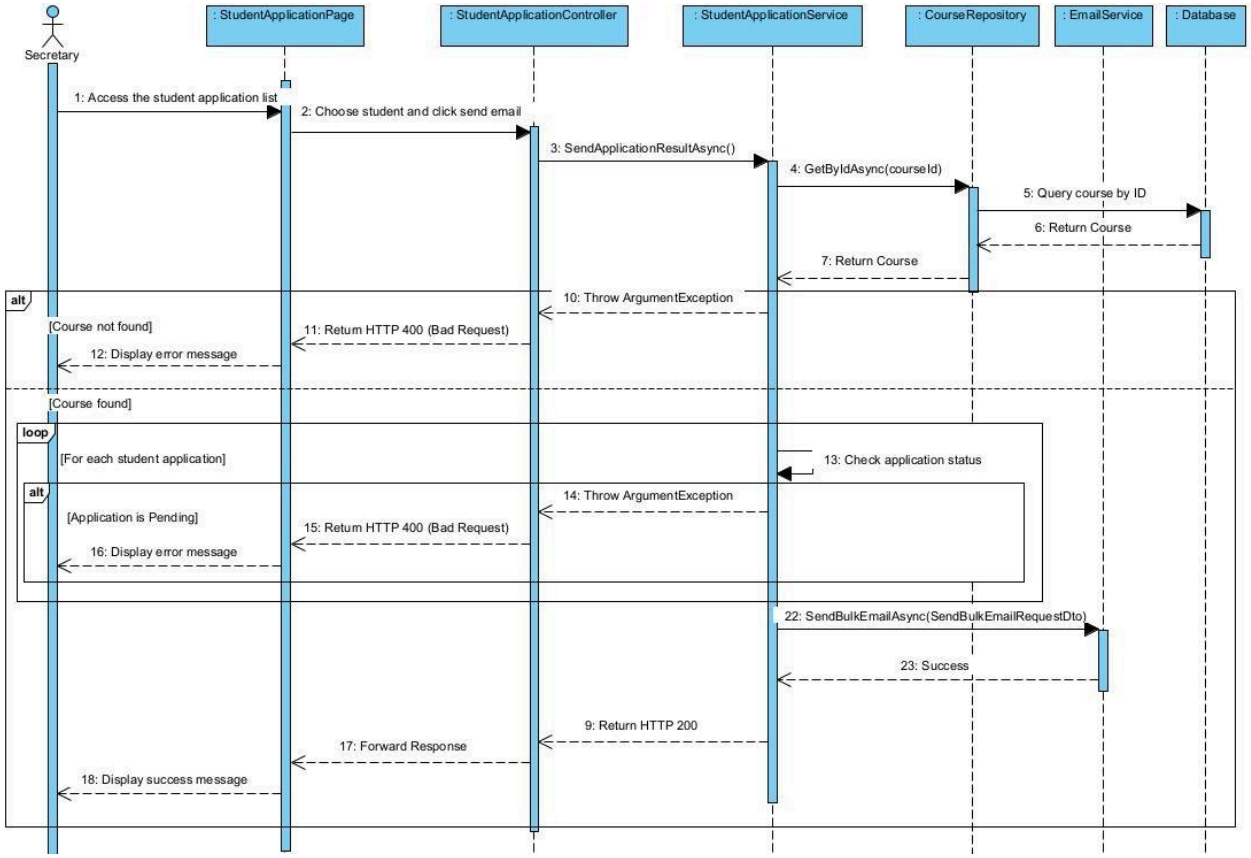

Figure 29: Send email sequence diagram

The link of image: [Send email]

3.2.15 Generate student card

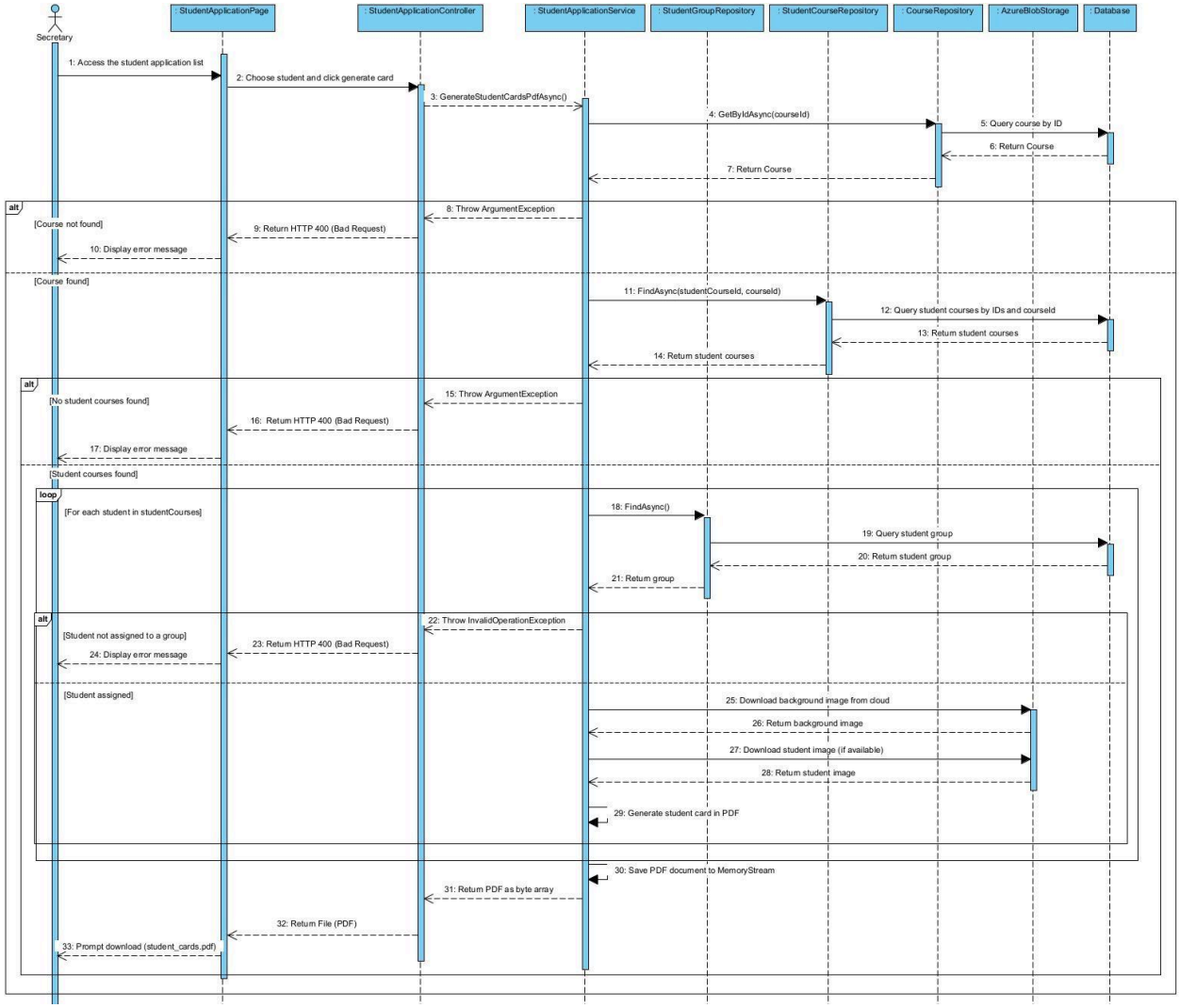

Figure 30: Generate student card sequence diagram

The link of image: [Generate student card]

3.2.16 Submit volunteer registration

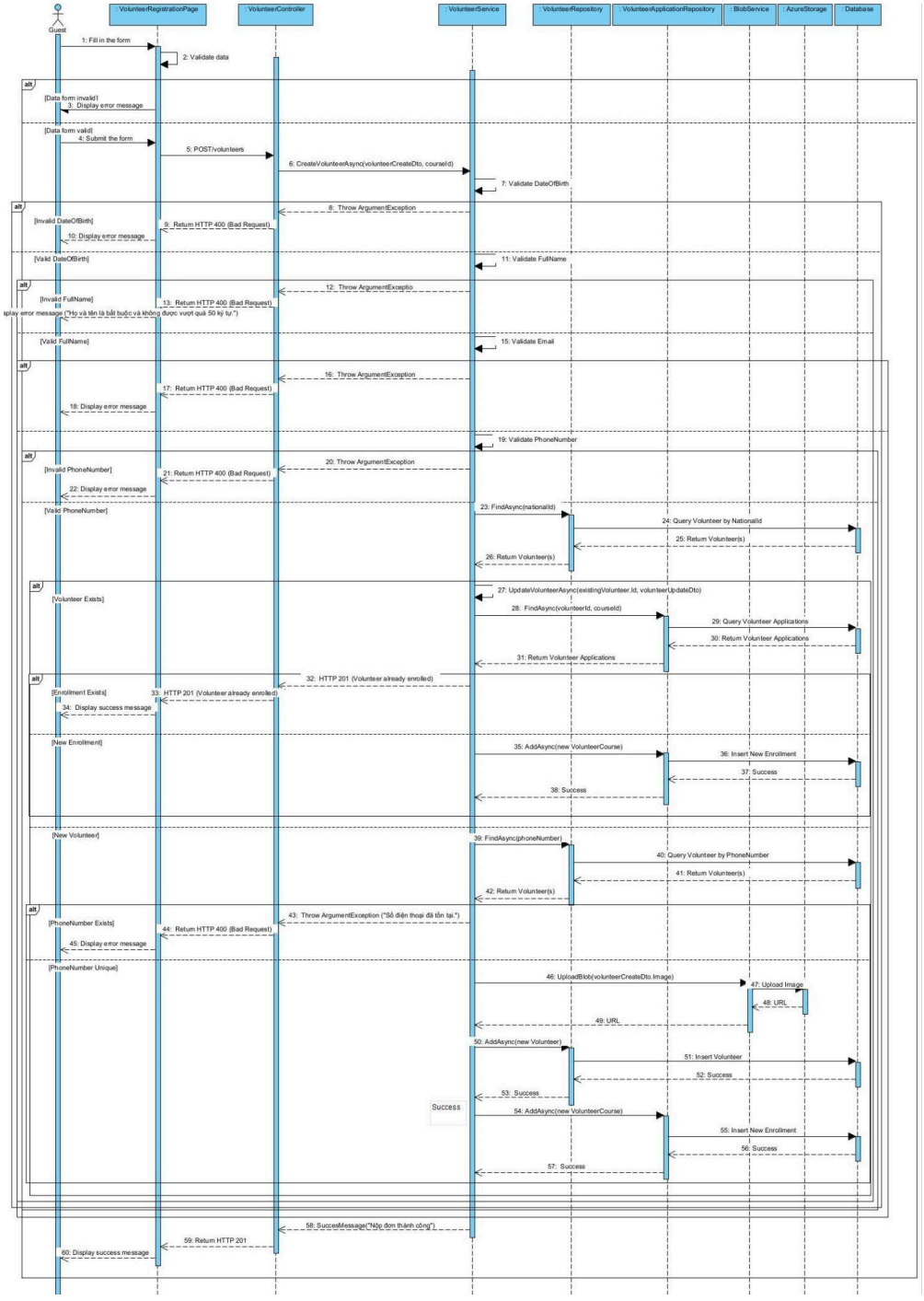

Figure 31: Submit volunteer registration sequence diagram

The link of image: [Submit volunteer registration]

3.2.17 Manually assign volunteer registration forms to secretary

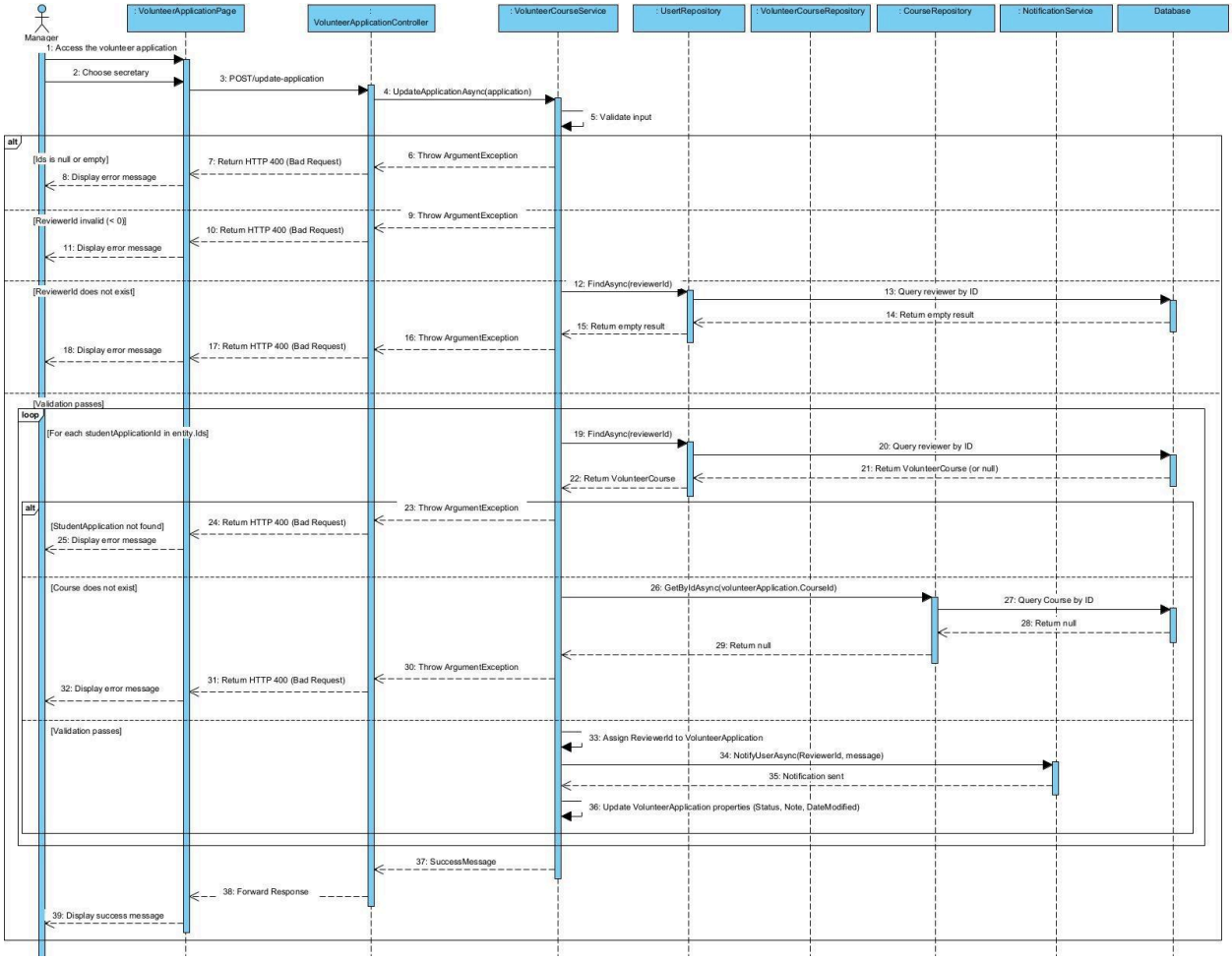

Figure 32: Manually assign volunteer registration forms to secretary sequence diagram

The link of image: [Manually assign volunteer registration]

3.2.18 Automatically assign volunteer registration forms to secretaries

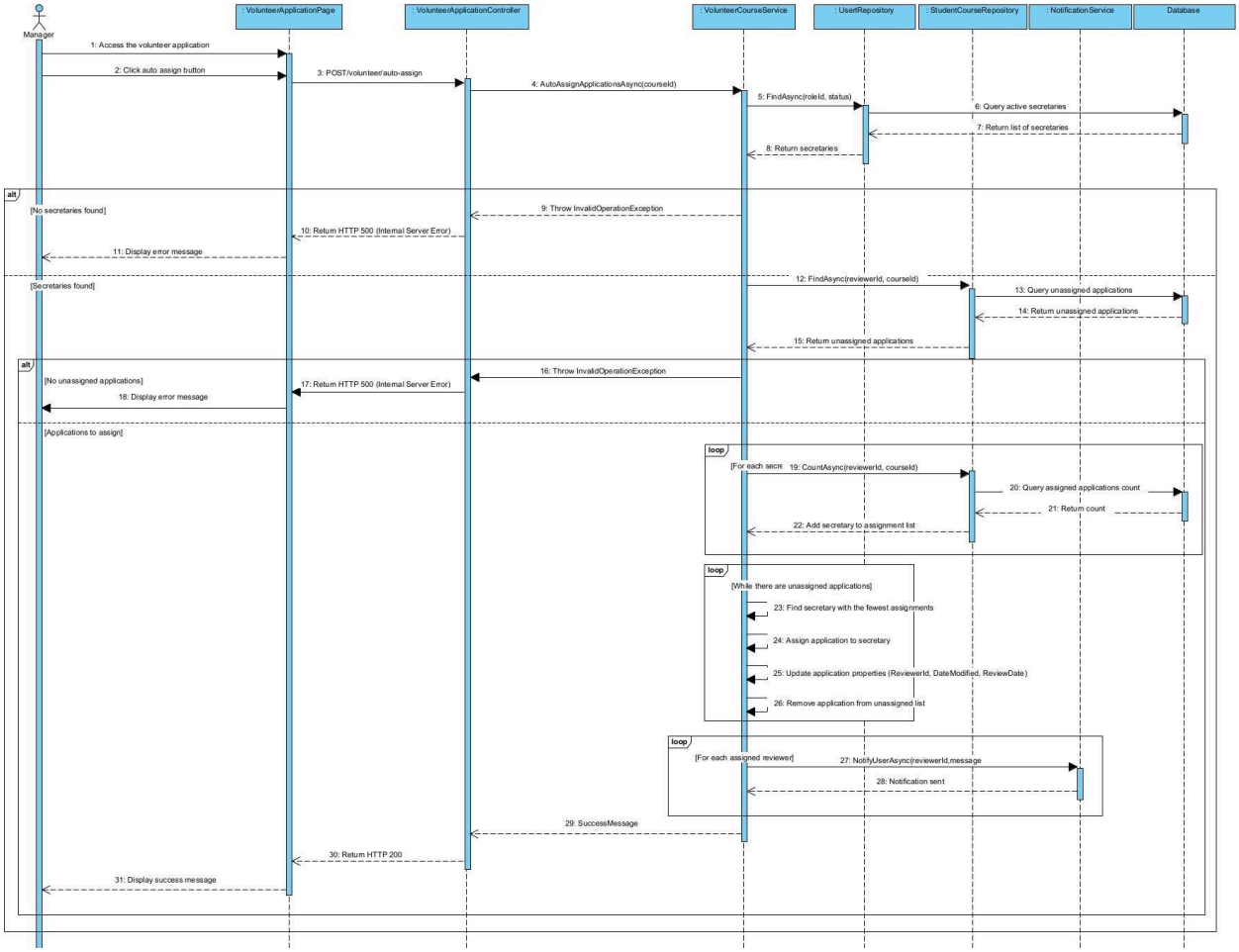

Figure 33: Automatically assign volunteer registration forms to secretaries sequence diagram

The link of image: [Automatically assign volunteer registration]

3.2.19 Accept/Reject volunteer registration

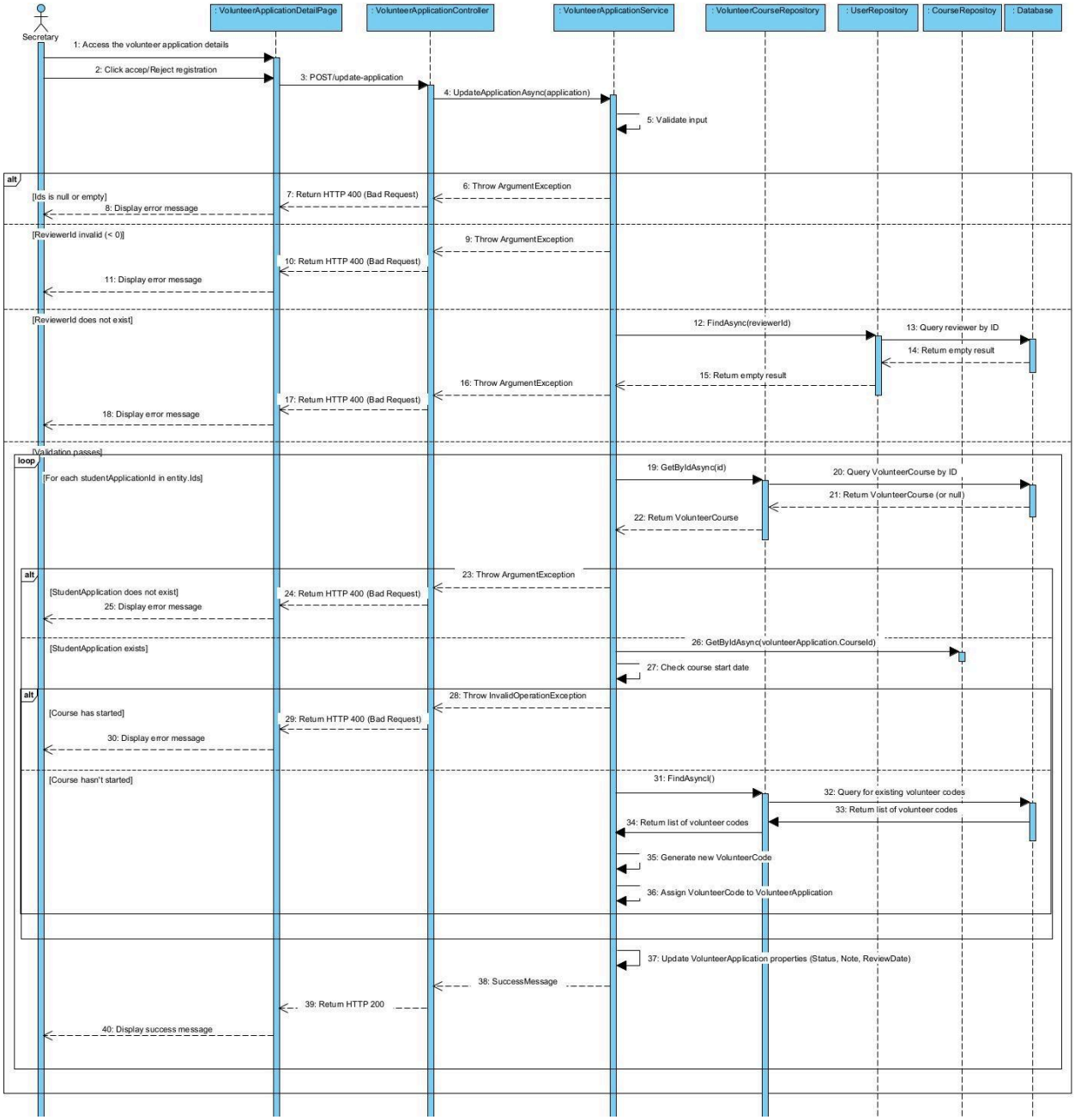

Figure 34: Accept/Reject volunteer registration sequence diagram

The link of image: [Accept/Reject volunteer registration]

3.2.20 Manually assign volunteer to team

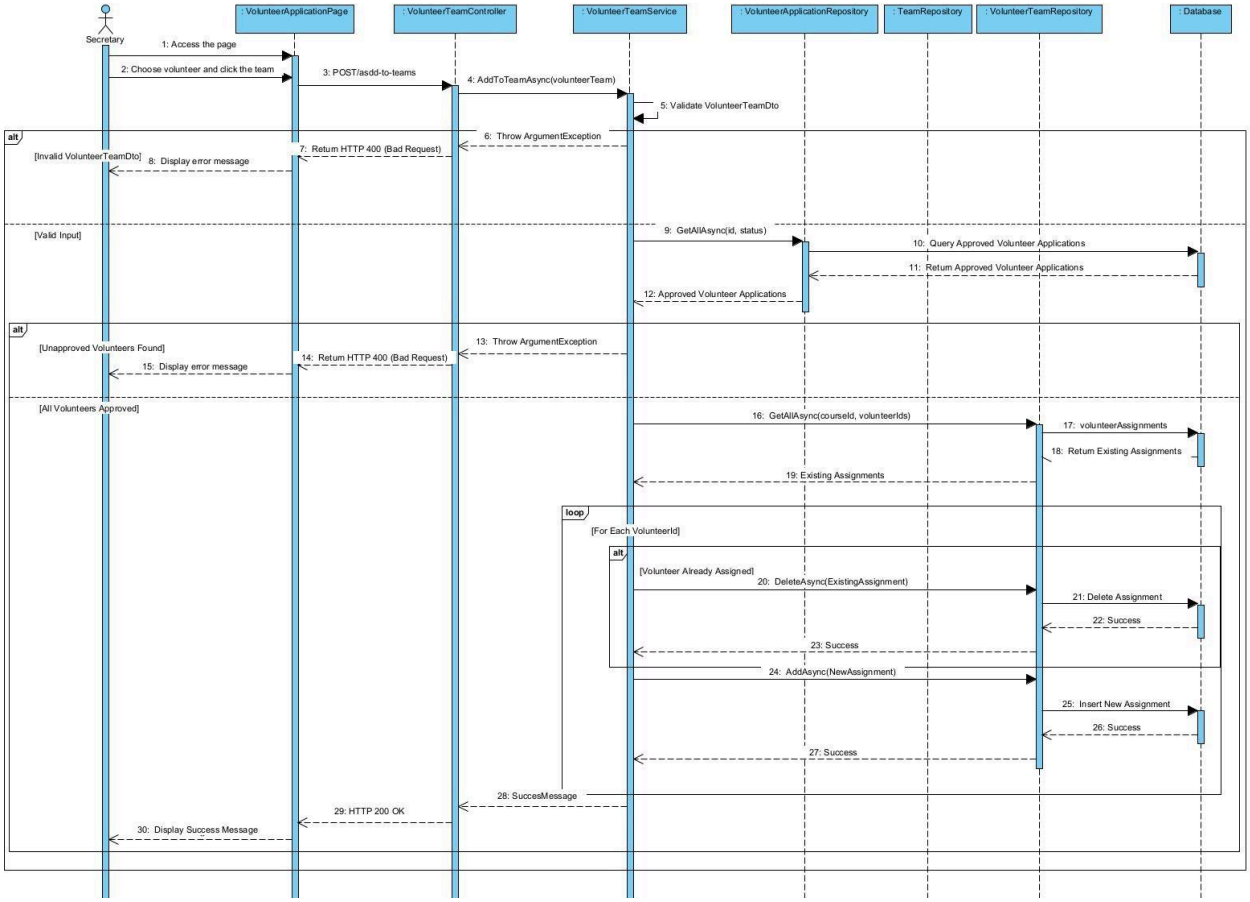

Figure 35: Manually assign volunteer to team sequence diagram

The link of image: [Manually assign volunteer]

3.2.21 Manually assign volunteers to teams

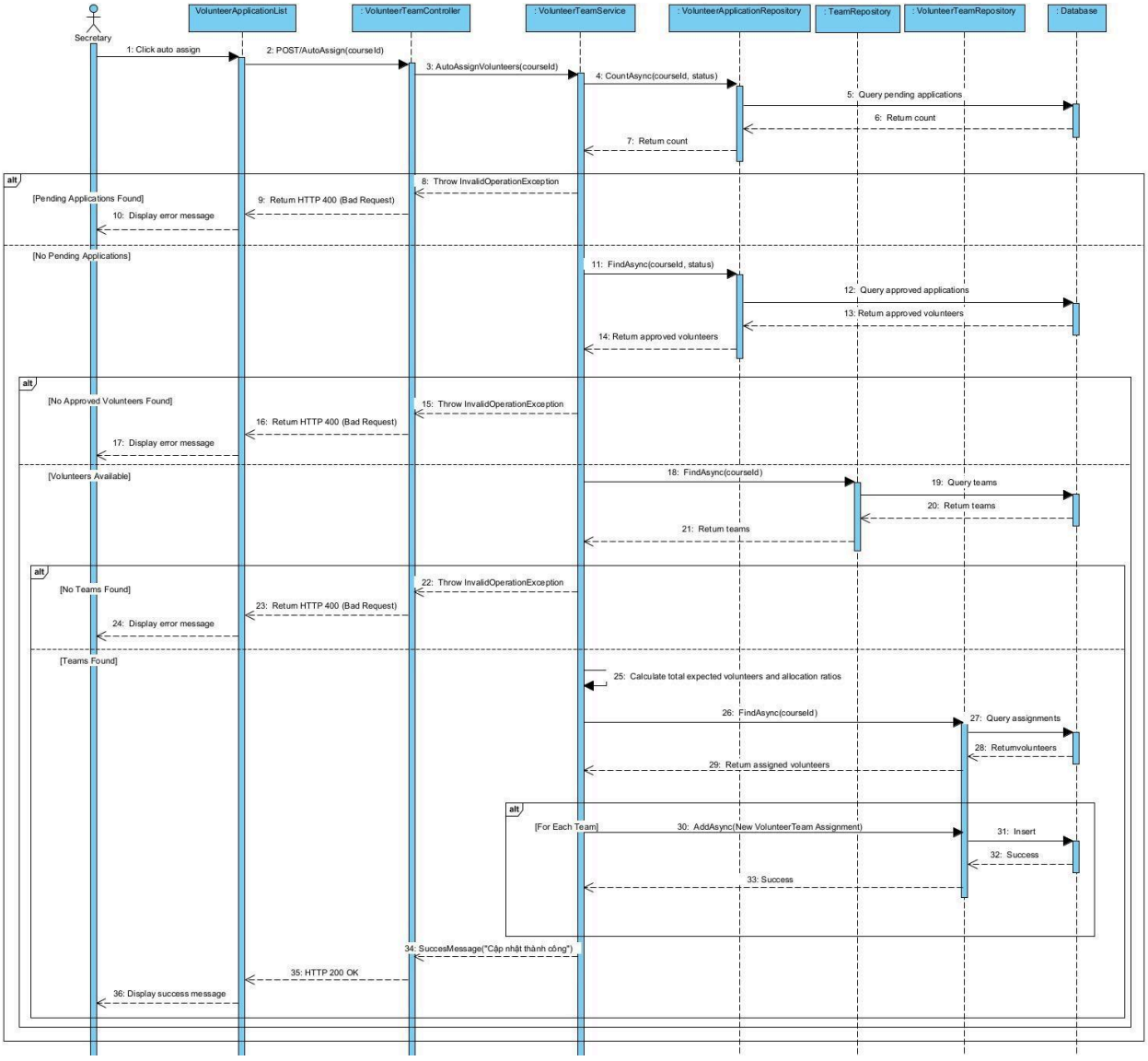

Figure 36: Manually assign volunteers to teams sequence diagram

The link of image: [Manually assign volunteers]

3.2.22 Automatically assign secretaries to volunteer registration forms

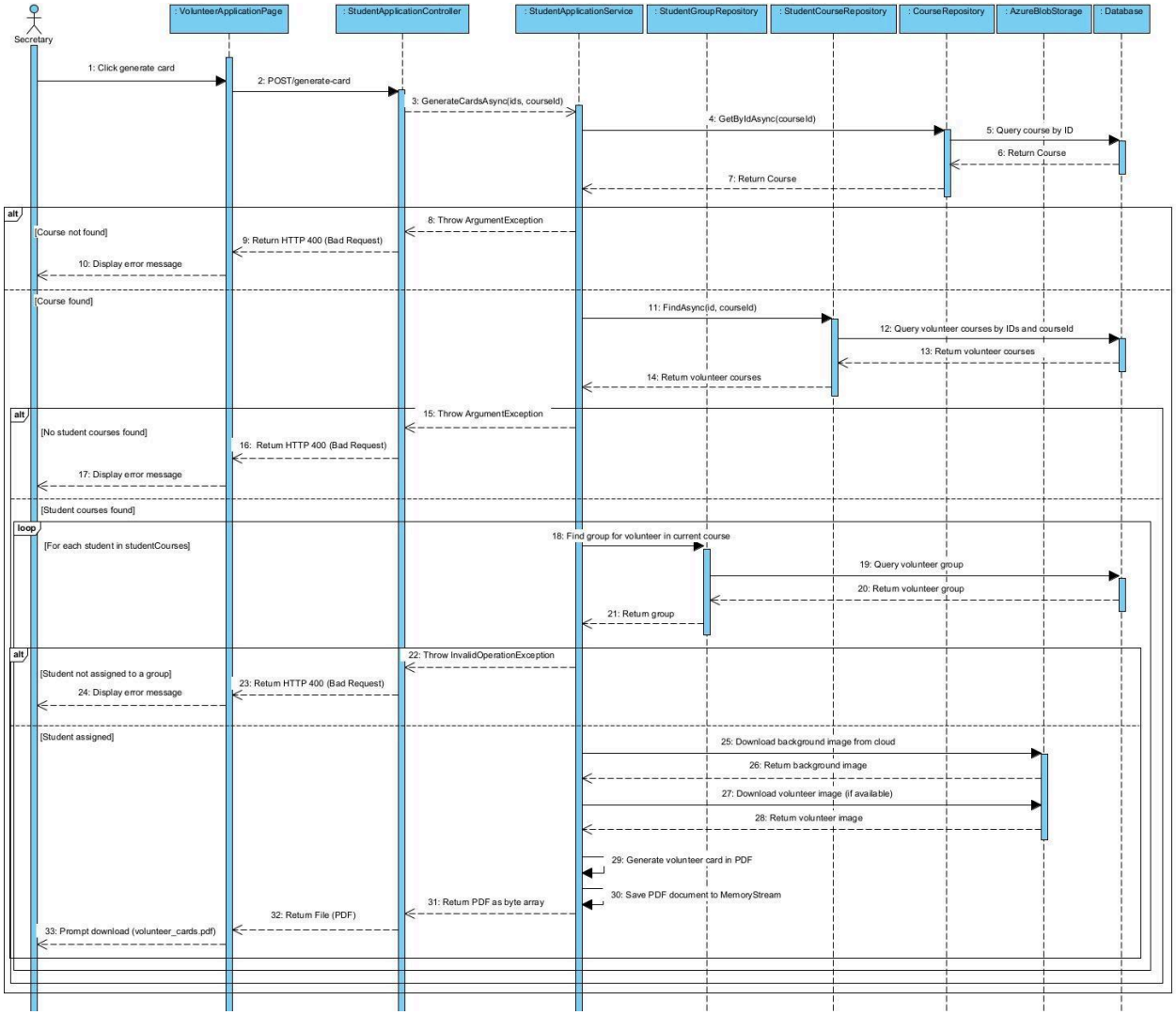

Figure 37: Automatically assign secretaries to volunteer registration forms sequence diagram

The link of image: [Automatically assign secretaries]

3.2.23 Edit volunteer in a course

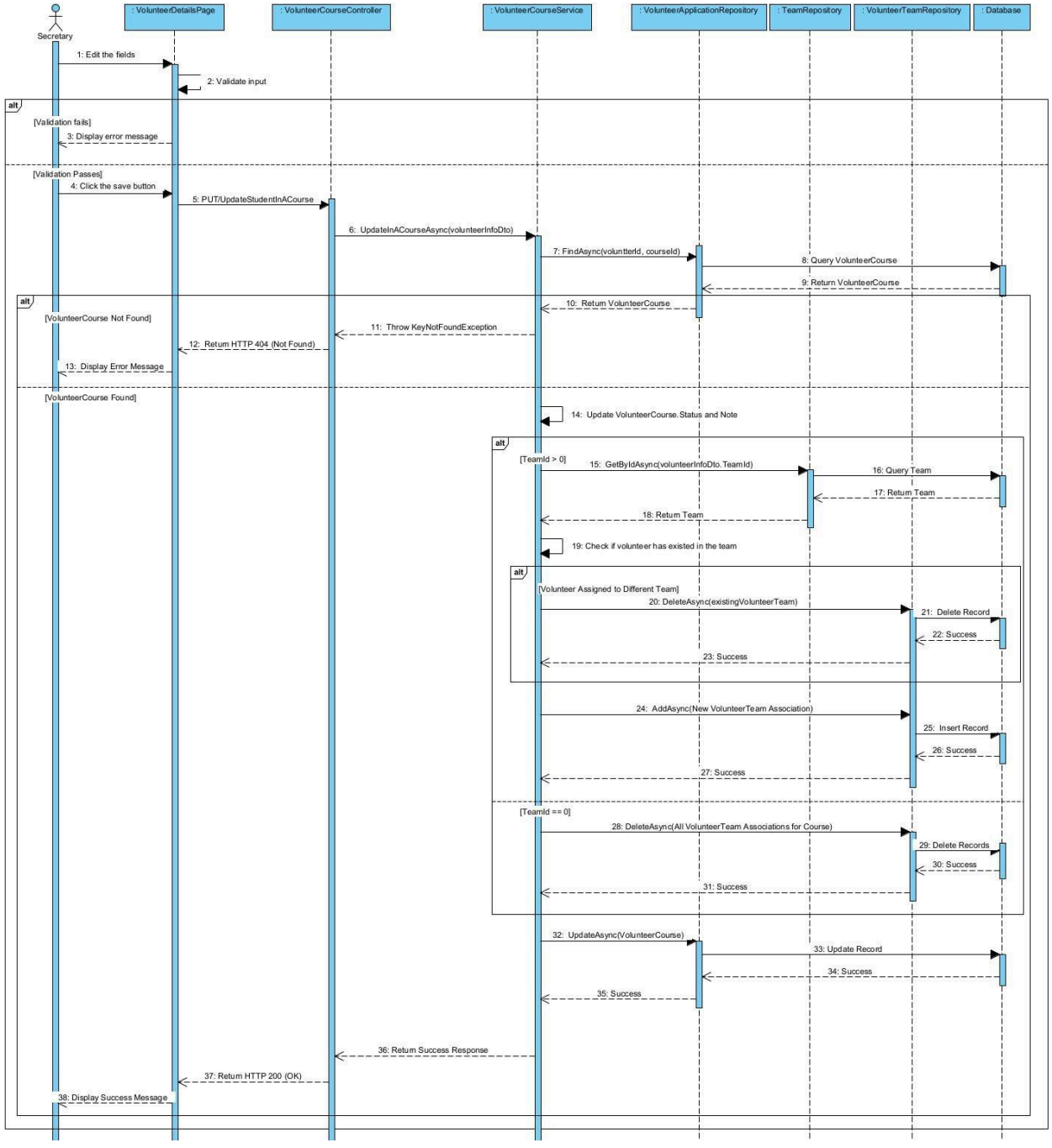

Figure 38: Edit volunteer in a course sequence diagram

The link of image: [Edit volunteer in a course]

3.2.24 Add room

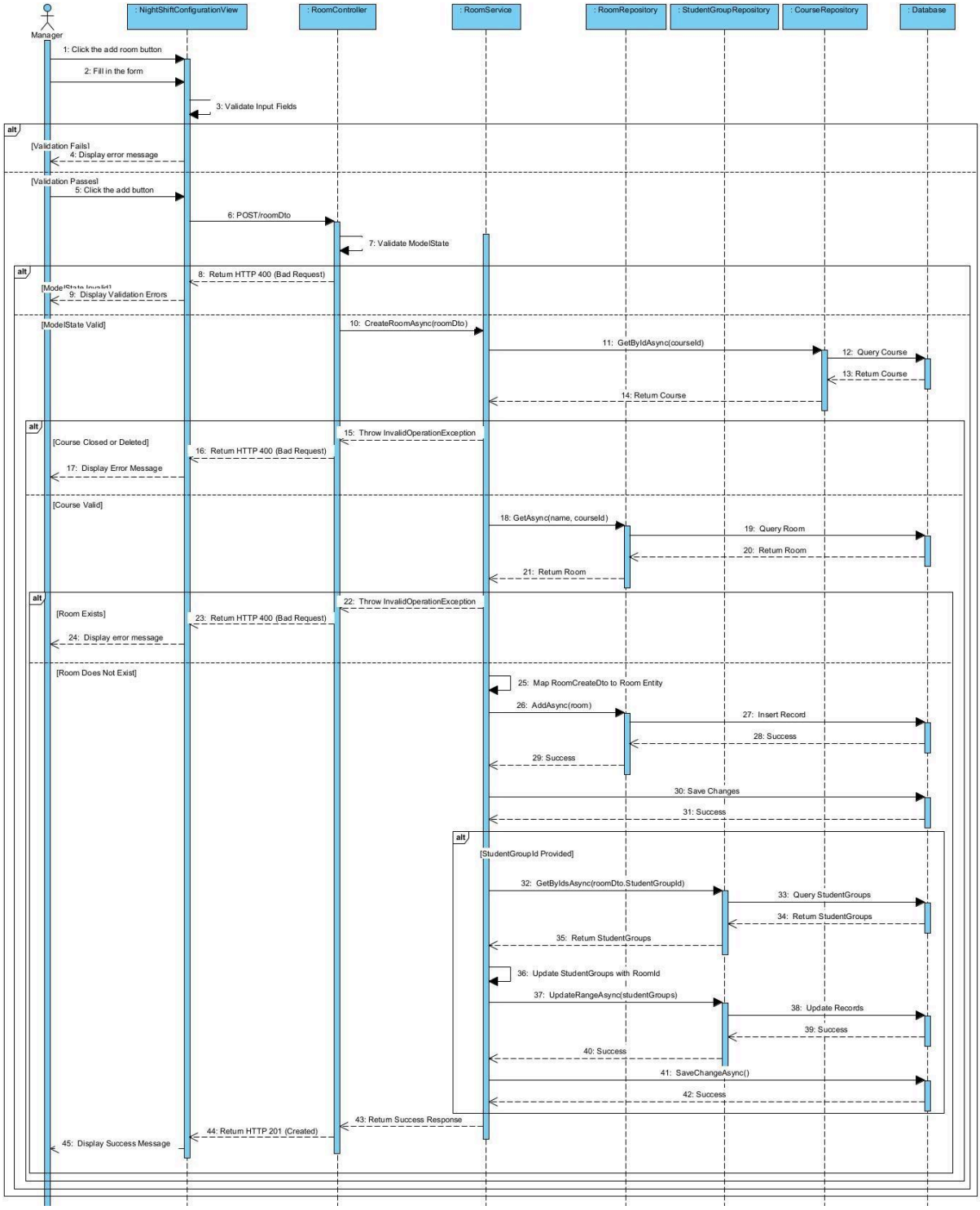

Figure 39: Add room sequence diagram

The link of image: [Add room]

3.2.25 Add Night Shift

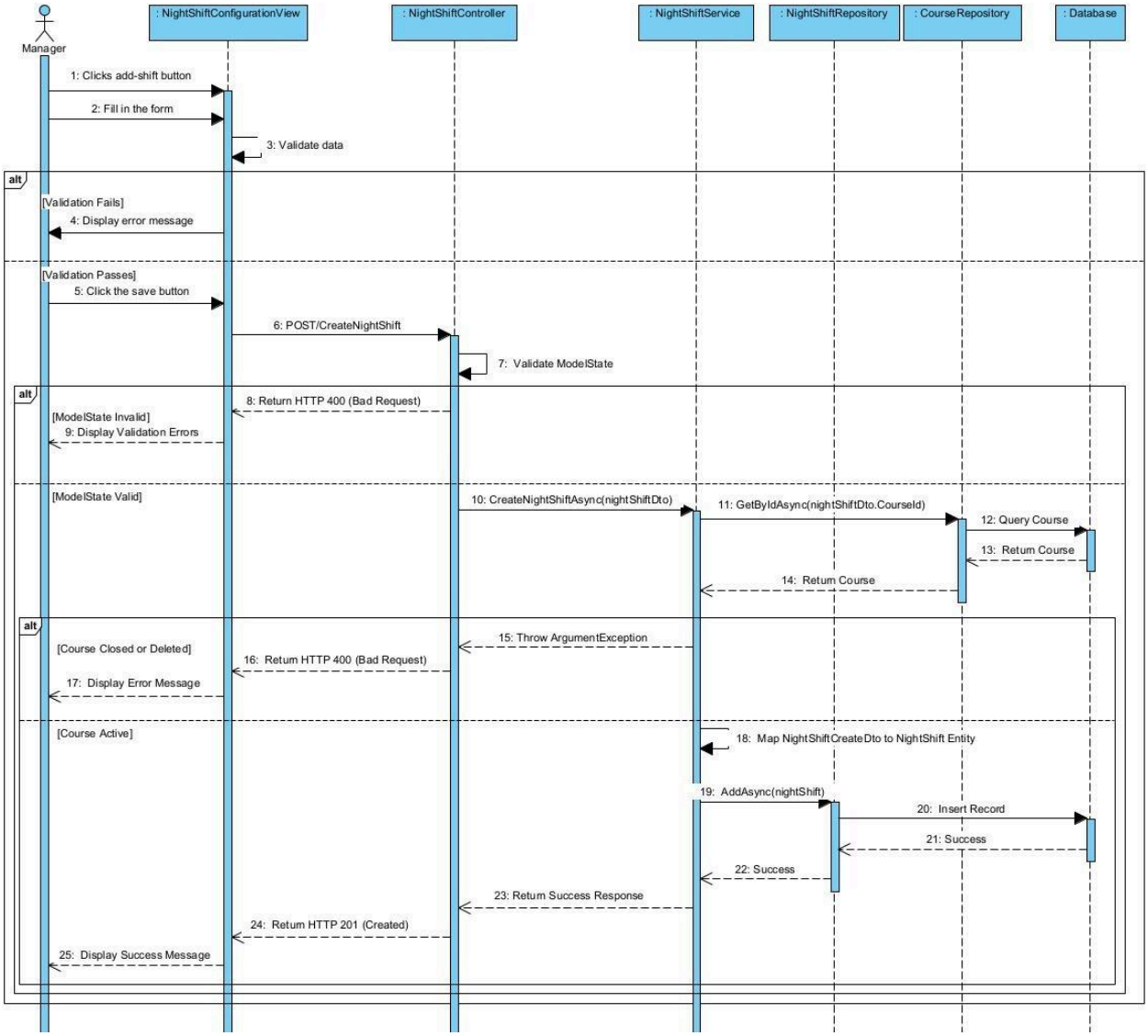

Figure 40: Add Night Shift sequence diagram

The link of image: [Add Night Shift]

3.2.26 Register for free time

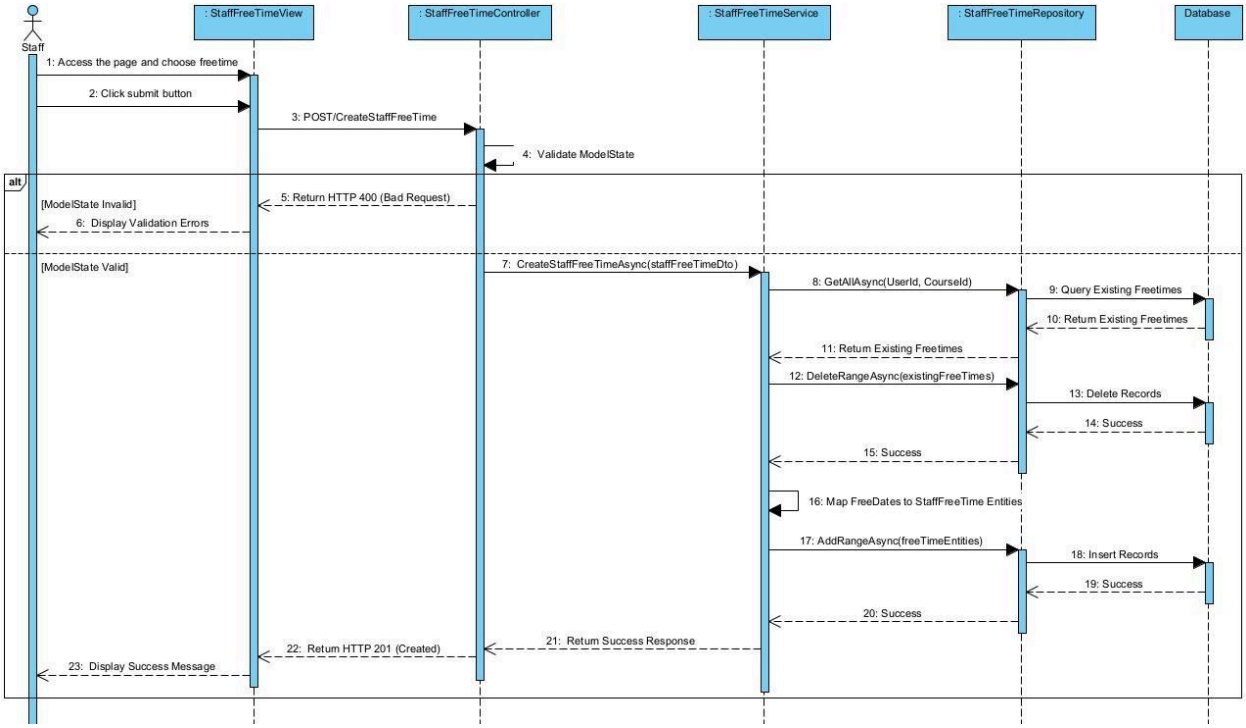

Figure 41: Register for free time sequence diagram

The link of image: [Register for free time]

3.2.27 Automatically assign nightshift to staff

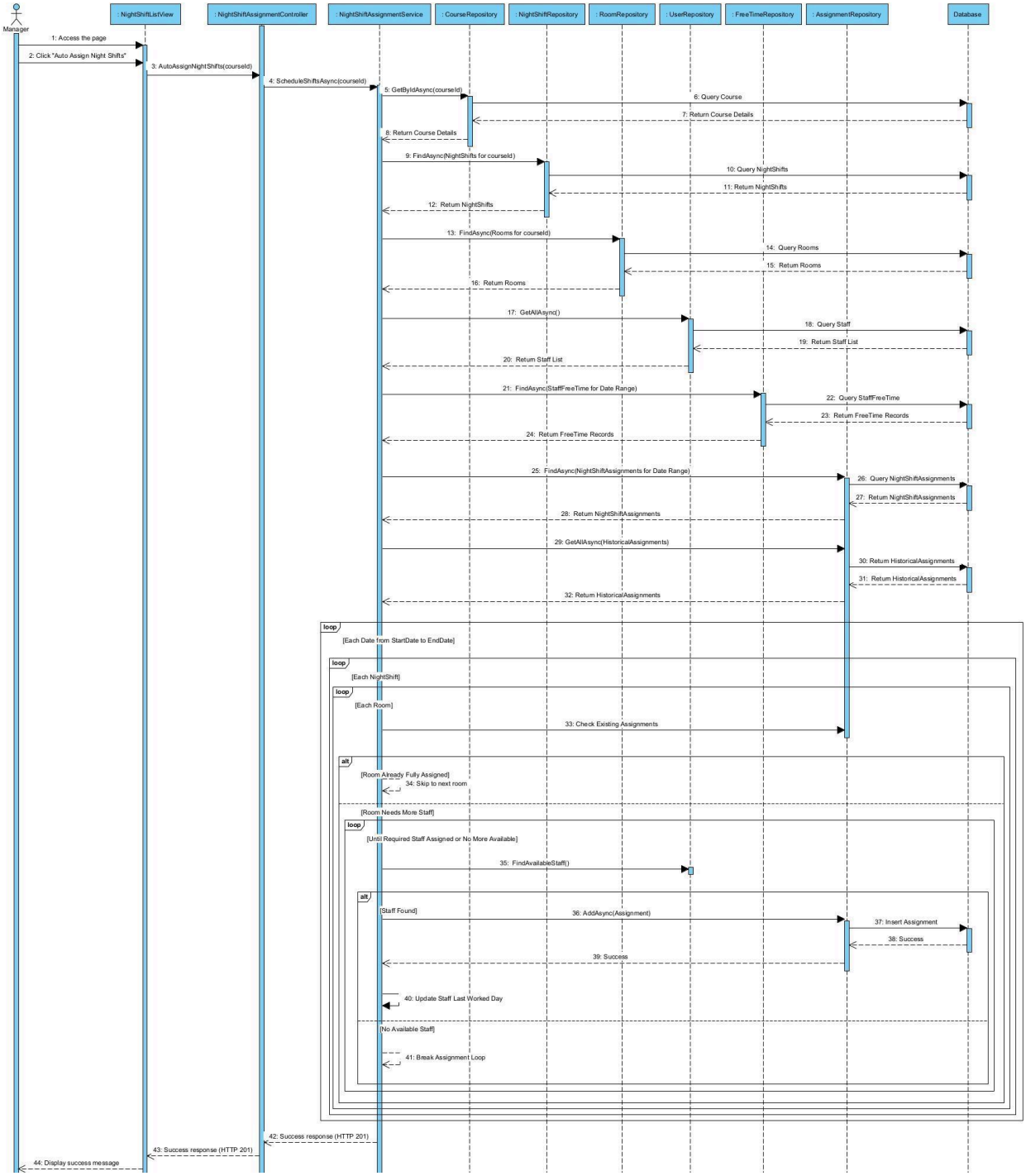

Figure 42: Automatically assign nightshift to staff sequence diagram

The link of image: [Automatically assign nightshift]

3.2.28 Manually assign nightshift to staff

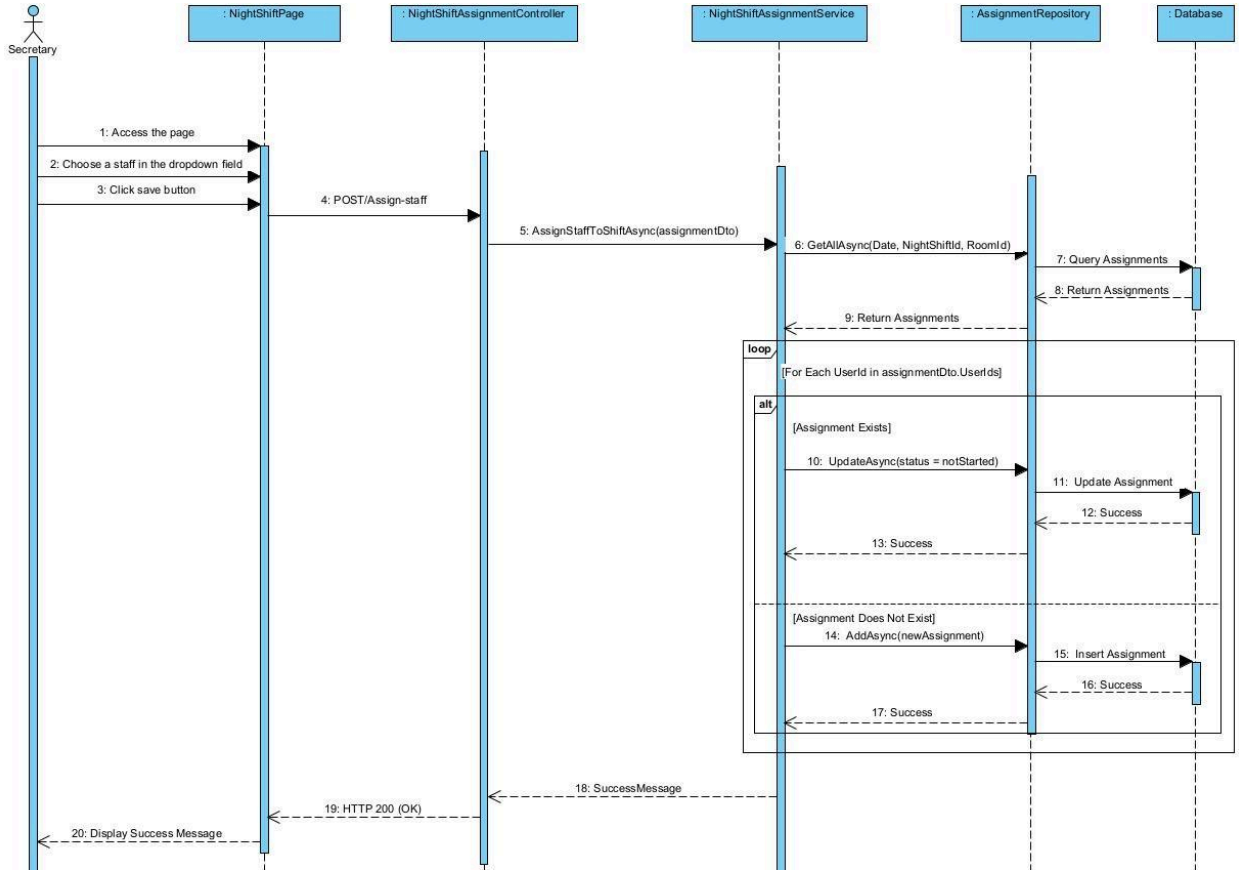

Figure 43: Manually assign nightshift to staff sequence diagram

The link of image: [Manually assign nightshift]

3.2.29 Reject a night shift

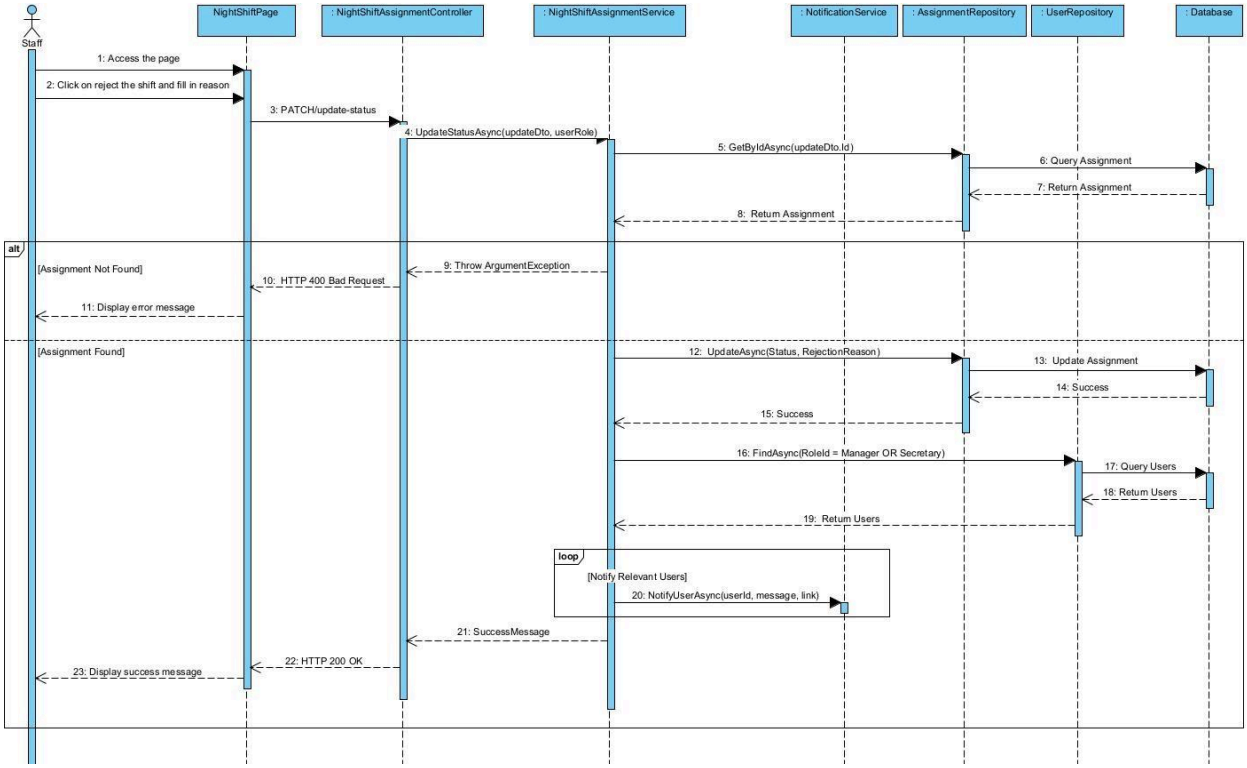

Figure 44: Reject a night shift sequence diagram

The link of image: [Reject a night shift]

3.2.30 Accept night shift rejection

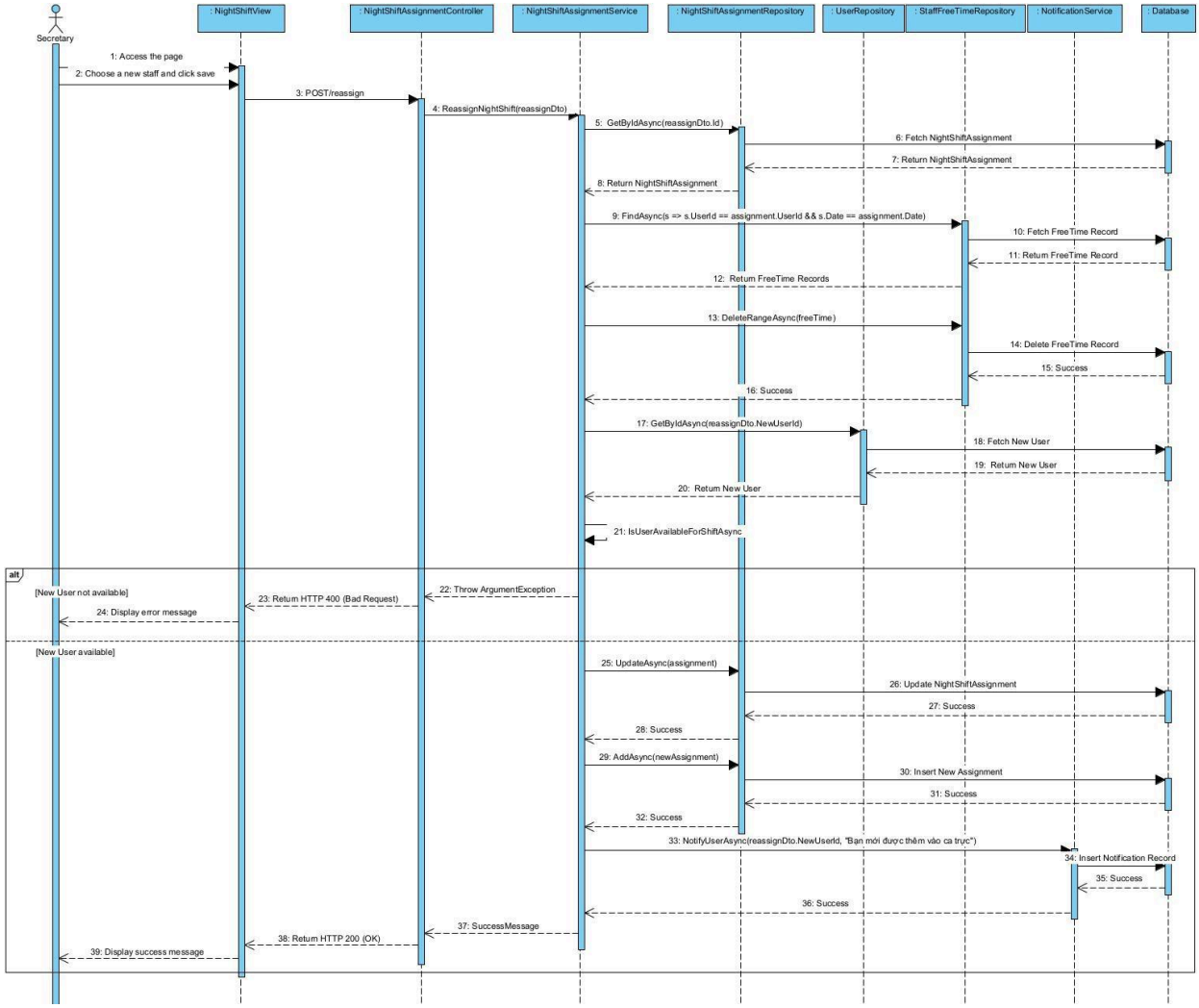

Figure 45: Accept night shift rejection sequence diagram

The link of image: [Accept night shift rejection]

3.2.31 Submit daily report

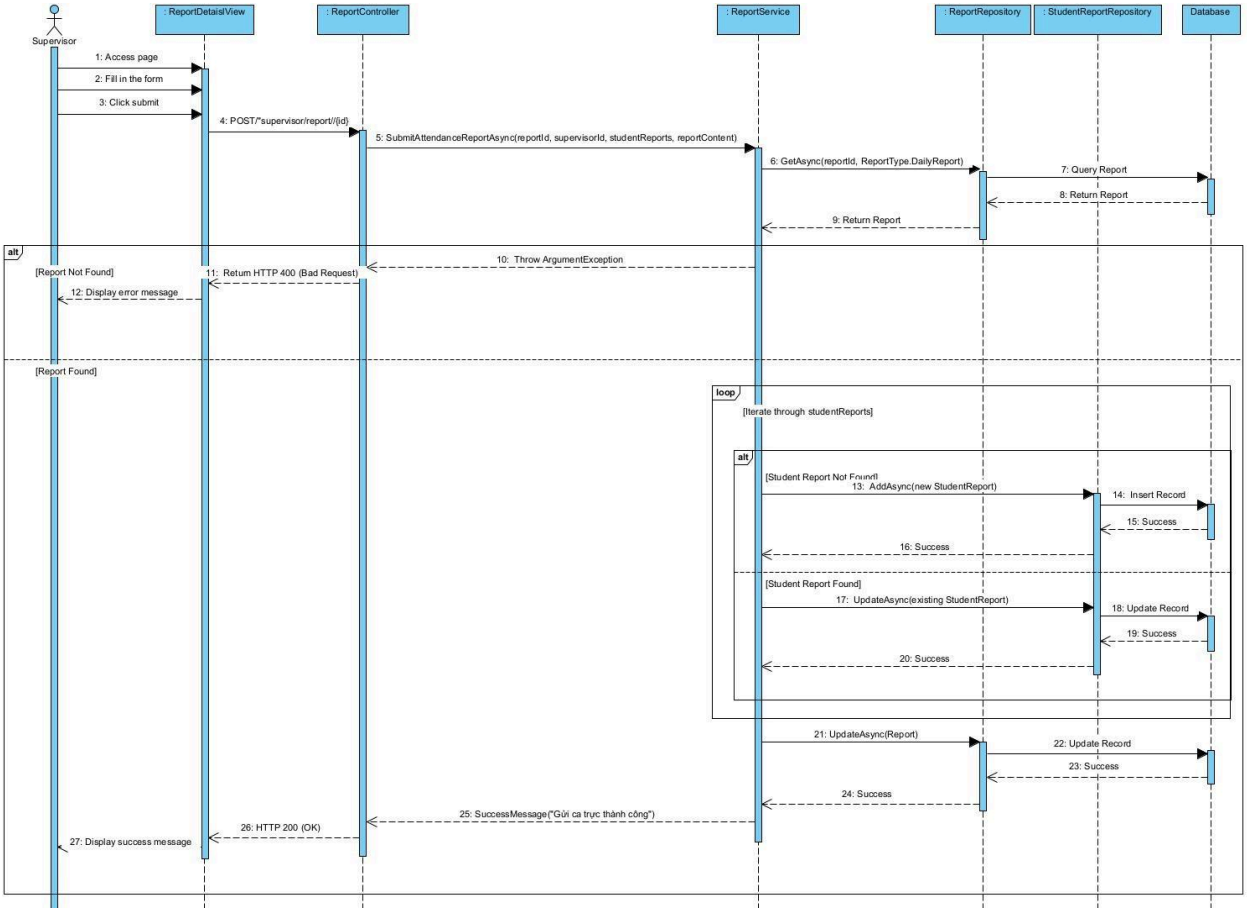

Figure 46: Submit daily report sequence diagram

The link of image: [Submit daily report]

3.2.32 Reopen a report

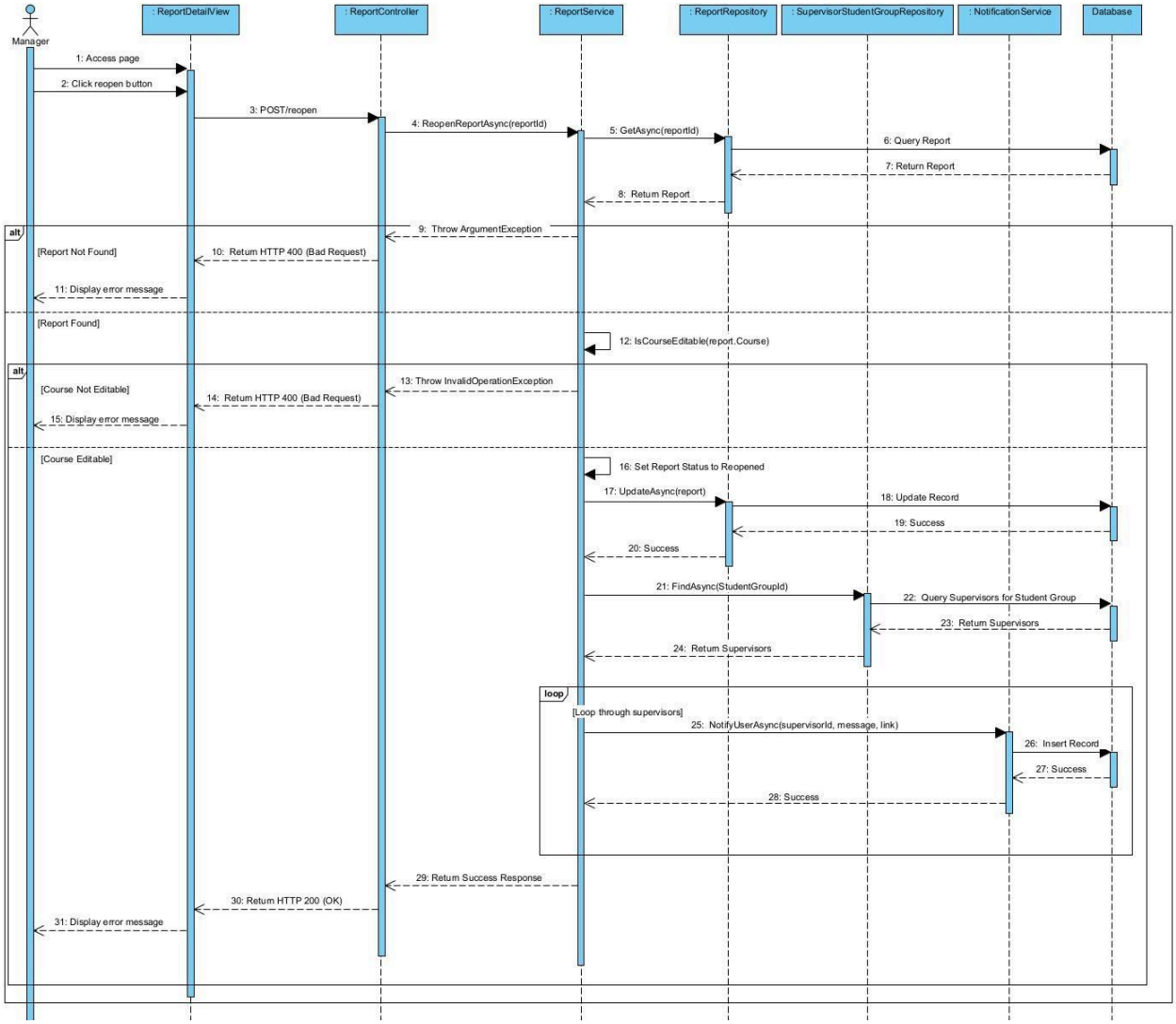

Figure 47: Reopen a report sequence diagram

The link of image: [Reopen a report]

3.2.33 Request to open a report

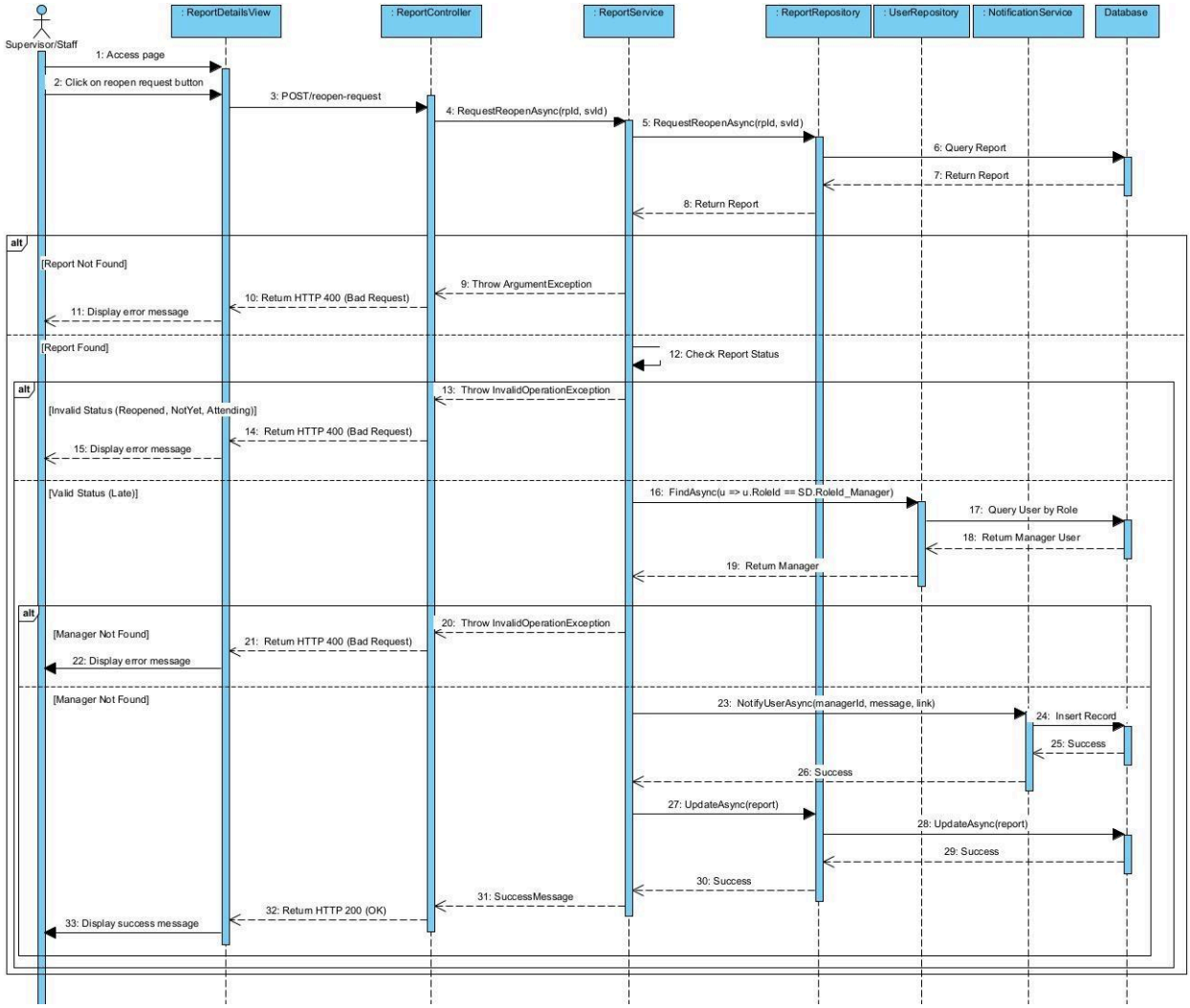

Figure 48: Request to open a report sequence diagram

The link of image: [Request to open a report]

3.2.34 Mark report as read

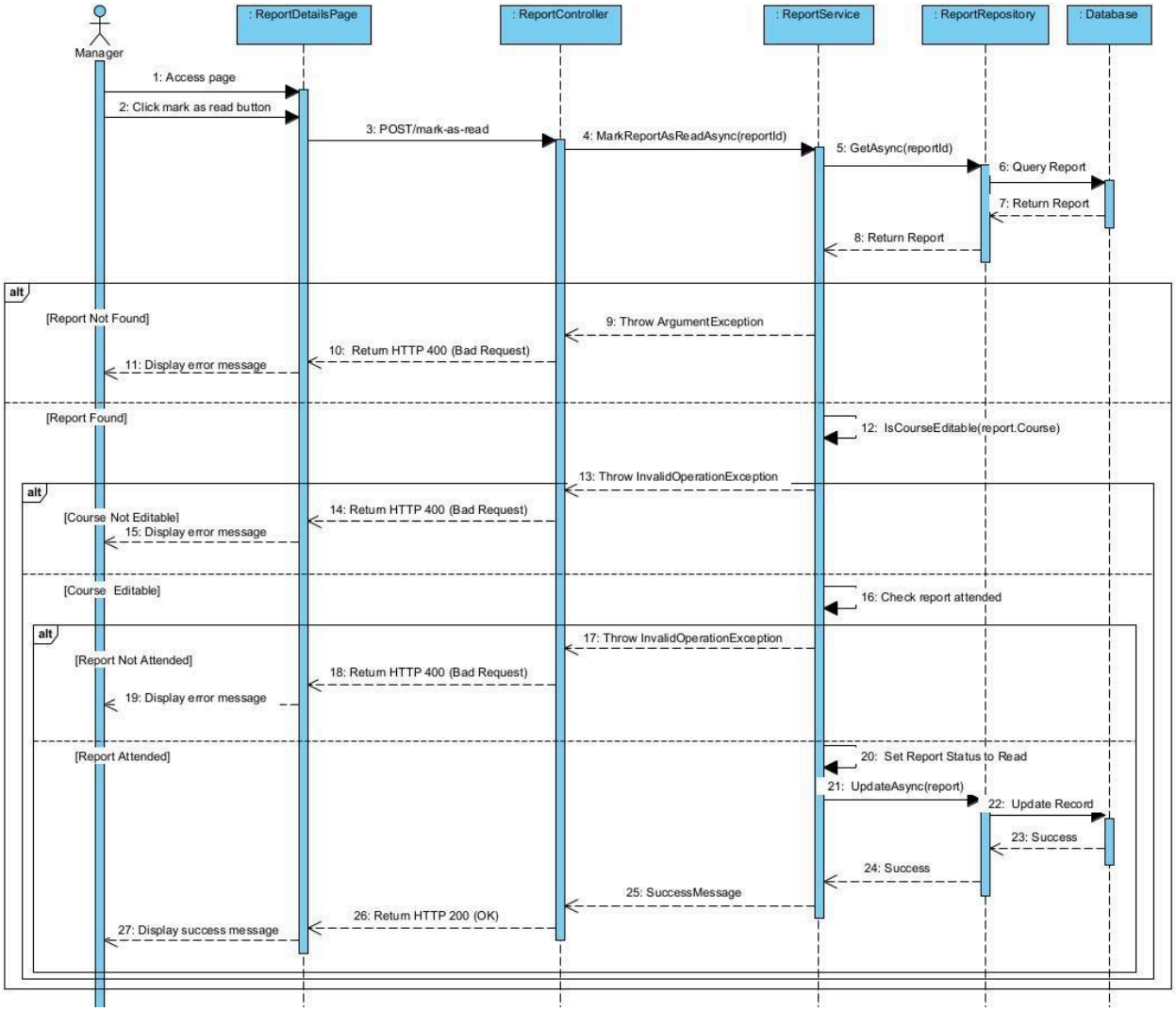

Figure 49: Mark report as read sequence diagram

The link of image: [Mark report as read]

3.2.35 Create Post

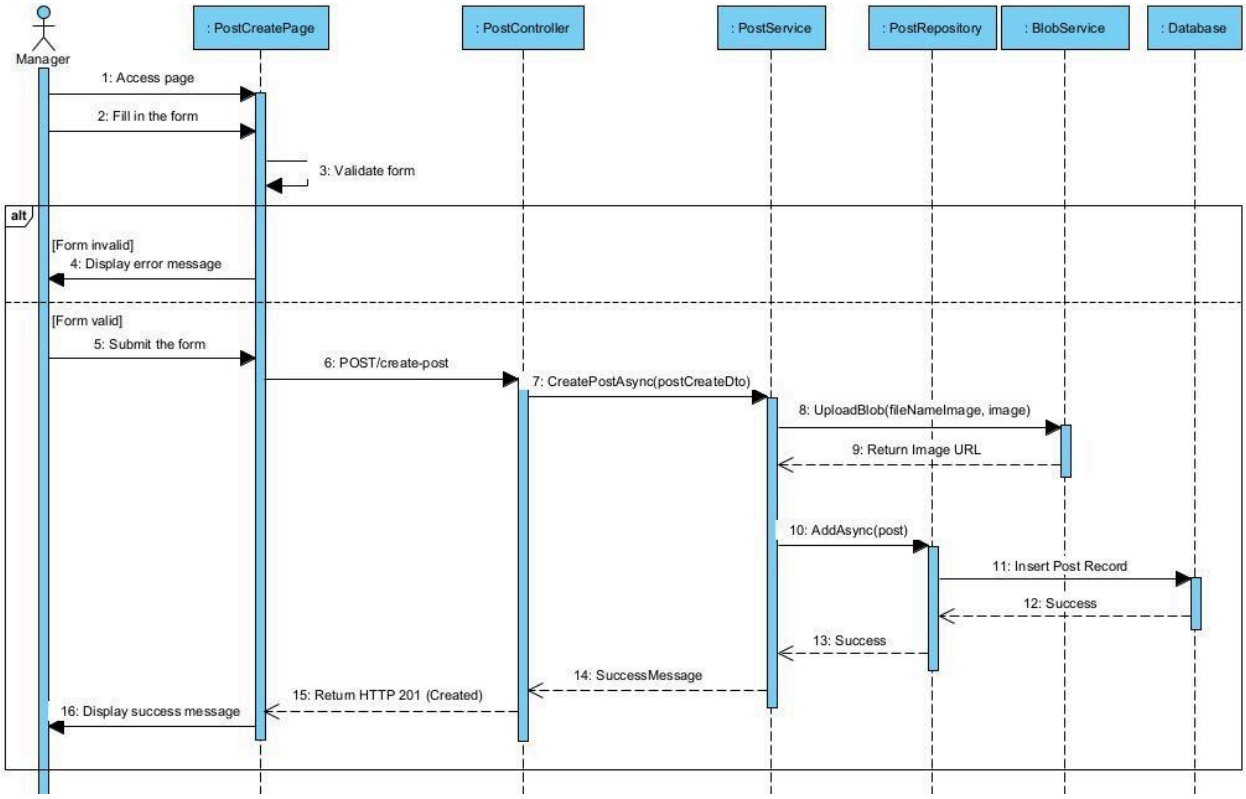

Figure 50: Create Post sequence diagram

The link of image: [Create Post]