

AN ONLINE ADMISSION APPLICATION AND INTERVIEW IN
UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

A thesis

Present to

Information and Communication Technology Department
University of Science and Technology of Hanoi



by

VU Khanh Huyen

Supervisor : DOAN Nhat Quang

August, 2020

TABLE OF CONTENTS

I. INTRODUCTION	5
1. Context and Motivation	5
2. Thesis Structure	6
II. OBJECTIVES	7
1. Desired Features	7
2. Expected Outcome	8
III. REQUIREMENT ANALYSIS	9
1. Overall System Requirements	9
2. Use Cases	10
2.1. Use cases Diagram	10
2.2. Users Characteristics	11
3. Use Cases and Scenario Description	11
IV. METHODOLOGY	28
1. Tools and Techniques	28
2. System Architecture	31
2.1 Front-end	32
2.2 Bank-end	33
3. Database Design	34
4. Use Cases Implementation	42
V. RESULT AND DISCUSSION	56
VI. CONCLUSION AND FUTURE WORKS	57
1. Conclusion	57
2. Future Work	57
VII. APPENDIX	58

LIST OF TABLES

1.	Register Basic Flow	11
2.	Register Alternatives Flow 1	12
3.	Register Alternatives Flow 2	13
4.	Login Basic Flow	13
5.	Login Alternatives Flow 1	14
6.	Login Alternatives Flow 2	15
7.	Send Form and Upload Files Basic Flow	16
8.	Send Form and Upload Files Alternatives Flow	17
9.	Edit Form Basic Flow	18
10.	Export Form Basic Flow	19
11.	Check Form Status Basic Flow	20
12.	Follow Admission Wave Status Basic Flow	21
13.	View Student Information Basic Flow	22
14.	View Student Information Alternatives Flow 1	23
15.	View Student Information Alternatives Flow 2	23
16.	Evaluate Student Basic Flow	24
17.	Evaluate Student Alternatives Flow	25
18.	Use Question Bank Flow	26
19.	Use Question Bank Alternatives Flow	26
20.	Differences between SQL and NoSQL	28
21.	User Database Design	35
22.	Student Database Design	35
23.	Application Information Database Design	36
24.	Academic History Database Design	37
25.	Address Database Design	38
26.	Employee Database Design	38
27.	English Evaluation Database Design	39
28.	General EvaluationDatabase Design	39
29.	Jury Employee Database Design	40
30.	Jury Database Design	40
31.	Wave Database Design	41
32.	Question Bank Database Design	41

LIST OF FIGURES

1.	Use case diagram	10
2.	System Architecture diagram	31
3.	Frontend Component Design	32
4.	Overall Database Design	34
5.	Login Sequence diagram	42
6.	Register Sequence diagram	44
7.	Send Admission Form Sequence diagram	46
8.	Edit Admission Form Sequence diagram	48
9.	Export Admission Form Sequence diagram	49
10.	Check Admission Form Status Sequence diagram	50
11.	Follow Admission Wave Information Sequence diagram	51
12.	View Student Information Sequence diagram	52
14.	Use Question Bank Sequence diagram	54
15.	Welcome Page user interface	55
16.	Login and Register user interface	58
17.	Admission Form status user interface	58
18.	Admission Wave information user interface	59
19.	Admission Form user interface	59
20.	Upload Files user interface	60
21.	Export Admission Form user interface	61
22.	View Student information user interface	62
23.	Evaluate Student user interface	63
24.	Question Bank user interface	64

I. INTRODUCTION

1. Context and Motivation

Managing admissions can be a huge task for a university. The rising numbers of students applying for admission are causing enormous pressure on the administration department to manage and arrange the admission process manually. It is difficult to manage the admission information accurately and in a timely manner.

In the University of Science and Technology of Hanoi, the entire admission process is handled manually, which is exhausted and time-consuming. Every year, students in hundreds stand in queues for collecting admission forms and then again for submitting the admission forms. After collecting all forms of students, the administration department has to manually import all student applications information into excel files. The evaluation information of students is also imported manually after the admission interview. Manual management also leads to problems in managing the applications, handling queries, distribution of forms and collection of forms. Since the application and evaluation information must be accurate, long-term maintained and easily retrieved while the amount of data becomes larger and larger in quantity, finding a solution for a management system is essential.

In this project, an Online Admission Application and Interview System is proposed to solve the above problem. With this system, the university can easily manage the admission information and the data become more accurate and secure. The project offers many services for USTH officers to improve their work condition: manage the admission information, manage the interview data and manage the admission wave data. Through this application, the students can keep track of their application process and the contact between them and the administration department is direct and improved. Furthermore, the interview evaluation process becomes precise and automatic.

2. Thesis Structure

The thesis will contain all the information necessary to reproduce the result. First, we will introduce the exciting problem in managing enrollment information in USTH and the solution proposed in **section I: Introduction**. In **section II: Objective**, we will define expected requirements of the project, provide a brief overview of the function of the system and the reasons for its development. After that, we will describe the scenarios, use cases, object model, and dynamic models for the system in **section III: Requirement Analysis**. This section contains the complete functional specification, including navigational paths representing the sequence of screens. In **section IV: Methodology**, we will list all the tools and techniques used in the project, the reasons why they are chosen and the detailed use cases implementation. In **section V: Result**, we will list all the functionalities which are implemented in the system. Finally is **Conclusion and Future Work** in **section VI**.

II. OBJECTIVES

In this section, we will define expected requirements of the project, provide a brief overview of the function of the system and the reasons for its development.

1. Desired Features

The main goal of this project is to develop a System for Enrollment and Interview for USTH with basic services:

- **FEATURE 1: Authenticate**
 - **SUB-FEATURE 1.1: Register** allow users to create a new account
 - **SUB-FEATURE 1.2: Login** : allow users to access into the system.
- **FEATURE 2: Maintain Admission Form:**
 - **SUB-FEATURE 2.1: Send Form:** allow Students to add Admission Form
 - **SUB-FEATURE 2.2: Edit Form:** allow Students to update their Admission Form
 - **SUB-FEATURE 2.3: Export Form:** allow Students to export and download their Admission Form as PDF file
 - **SUB-FEATURE 2.4: Check Form Status:** allow Students to keep track with their Admission Form status.
- **FEATURE 3: Upload Files:** allow Students to upload PDF attachments.
- **FEATURE 4: Follow Admission Wave:** allow Students to keep track of Admission Waves information (status, data start, date end, ...)
- **FEATURE 5: Interview:**
 - **SUB-FEATURE 5.1: View Student Information:** allow Lecturer to search Student by name and view Student Information and their attachment files
 - **SUB-FEATURE 5.2: Evaluate Student:**
 - Allow English teacher to add English Evaluation Form
 - Allow Lecturer to add General Evaluation Form
- **FEATURE 6: Use Questions Bank:** allow the lecturer to search questions by Id in Question Banks.

2. Expected Outcome

Our system aims at creating a connection between students and university for a better experiment of enrollment and managing information. The specific goals include:

- Develop a backend engine for managing enrollment and interview data such as Student Admission Form, Evaluation Form, Student attachments (motivation letter, recommendation letter, adward, ...)
- Develop an interface on the web with interaction between Student or Lecturer and the System such as: maintain Admission Form, check Form status, check Wave information (to Student) or search for Student, Interview, search from Questions Bank (to Lecturer)
- The application should have all the features mentioned in Desired Features
- The web page should be able to run in multiple browsers such as Chrome, Safari, etc.

III. REQUIREMENT ANALYSIS

In this section, we will describe a brief overview of the functions in the project, the scenarios and use cases for the system. This part contains the complete functional specification, including navigational paths representing the sequence of screens.

1. Overall System Requirements

In general, this application must satisfy the following main requirements:

- A login system with authentication
- The system limited users with some features:
 - For Student:
 - Create, update and read Admission Form
 - Export Admission Form as PDF file
 - Read Wave information
 - Upload and update Attachment Files
 - For Lecturer:
 - Read Student Admission Form and Attachment Files
 - Create General Evaluation Form (with Lecturer) and English Evaluation Form (with English Teacher)
 - Read Question form Questions Bank

2. Use Cases

2.1. Use cases Diagram

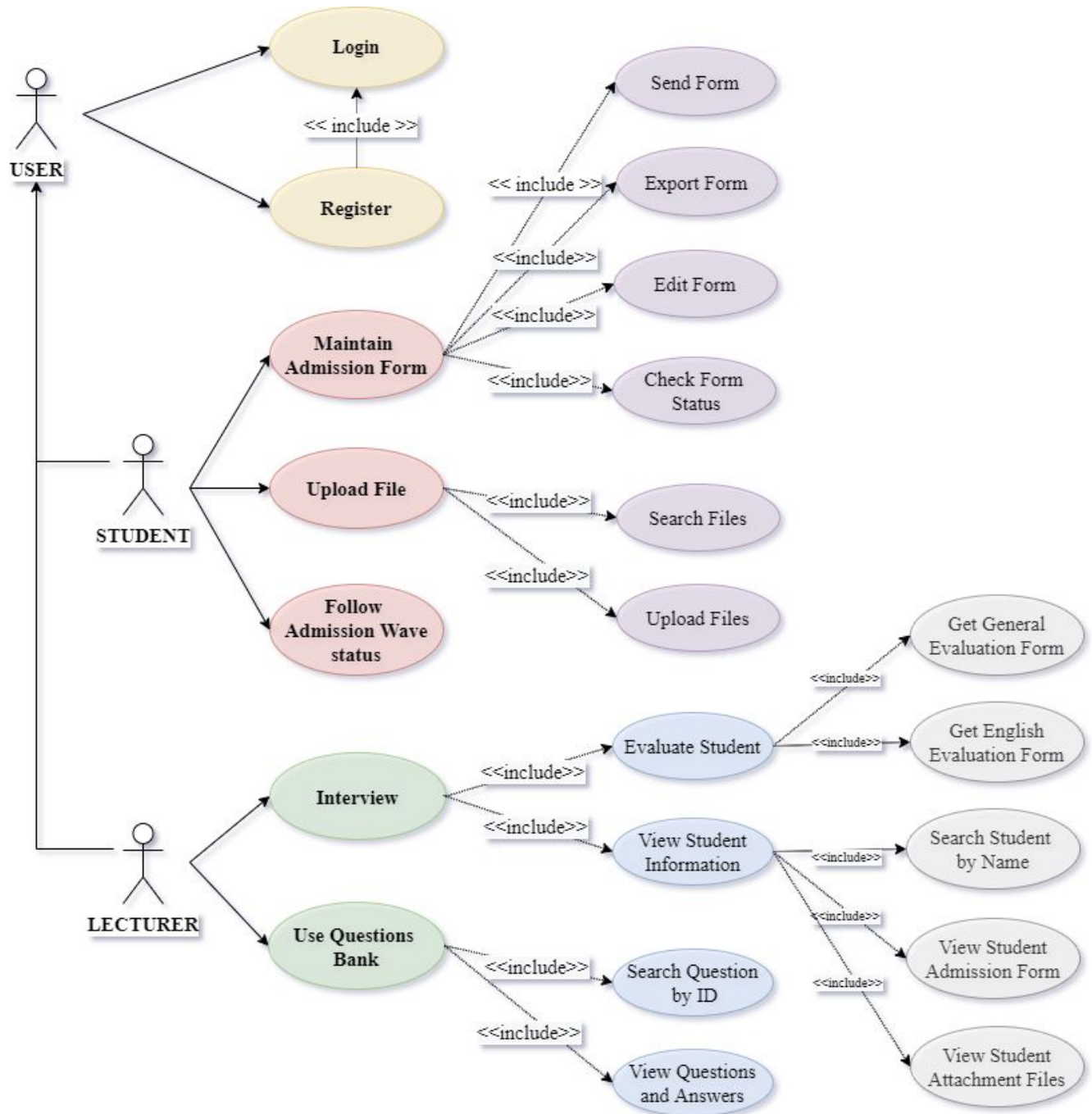


Figure 1. Use case diagram

Figure 1 shows the Use-case diagram that consists of all features/sub-features of the system.

2.2. Users Characteristics

There are 2 types of users that interact with the system: **Student** and **Professor**. Each user type has different uses in the system so each of them has their own requirements:

- **Student:** a user who uses the system to apply for Admission, keep track of the Admission Wave status and check his own Admission form status. After the user application form is approved, he becomes an application.
- **Professor:** a user who uses the system to interview applications: search applications by name, view applications information and evaluate. Professor also uses Questions Bank to search questions and answers for applications.

3. Use Cases and Scenario Description

3.1. Use case: **Register (SUB-FEATURE 1.1)**

3.1.1. Brief description

This use case describes how a student registers to the Online Admission Application and Interview System.

3.1.2. Flow of events

3.1.2.1. Basic Flow

Table 1. Register Basic Flow

Actor Action	System Action	Data
1. The actor wishes to register into the system.	2. The system requests the actor to select user type: Student or Lecture.	

3. The actor select his/ her type	4. The system requests the actor to enter email, username and password.	
5. The actor enters the requested information.	6. The system checks if the entered email is unique from the table “user” in Database. If unique, the system saves the user information and assigns a unique id to the user.	- Email - Username - Password

3.1.2.2. Alternatives Flow

a. Email is already in use

At step 5 in the Basic Flow:

Table 2. Register Alternatives Flow 1

Actor Action	System Action	Data
5. The actor enters the requested information.	6. The system announces that the email is in use and requests the user to re-enter email, username and password.	
7. The actor enters another username, email and password.	8. The system checks if that username is used or not. If not, the system saves the user information. The system assigns a unique id to the user. Else, repeats steps from step 3.	- Email - Username - Password

b. Professor want to Register

At step 3 in the Basic Flow:

Table 3. Register Alternatives Flow 2

Actor Action	System Action	Data
3. The actor select type “Lecturer” and wishes to register into the system	4. The system send an error message “Please contact Staff to register”	

3.1.3. Special requirements

Only Students can Register into the system. Professors have to contact Staff to register.

3.1.4. Pre-conditions

User is not logged in.

3.1.5. Post-conditions

If the use case is successful, the user registered successfully. If not, the system state is unchanged.

3.2. Use case: **Login (SUB-FEATURE 1.2)**

3.2.1. Brief description

This use case describes how an actor logs into the Online Admission Application and Interview System.

The actor can be a student or a professor.

3.2.2. Flow of events

3.2.2.1. Basic Flow

Table 4. Login Basic Flow

Actor Action	System Action	Data
1. The actor wishes to enter the system.	2. The system requests the actor to select his/ her type (student or lecturer)	

3. The actor selects his/ her type	4. The system requests the actor to enter email and password.	
5. The actor enters the email and password.	6. The system validates the email, password and type entered from the user table in the Database. If valid, the system logs the actor in.	- Username - Password - Type

3.2.2.2. Alternatives Flow

a. Invalid login parameters.

At step 5 in the Basic Flow:

Table 5. Login Alternatives Flow 1

Actor Action	System Action	Data
5. The actor enters the username and password.	6. The system validates the username and password. If the username or password is invalid, the system displays an error messenger “ User not found ” and requests the user to re-enters username and password.	- Username - Password - Type

b. Users select invalid type.

At step 5 in the Basic Flow:

Table 6. Login Alternatives Flow 2

Actor Action	System Action	Data
5. The actor enters the username and password.	6. The system validates the username and password. If the username or password is valid but the type is invalid the system displays an error messenger “ You aren’t Professor” and requests the user to re-enters username and password.	- Username - Password - Type

3.2.3. Special requirements

None.

3.2.4. Pre-conditions

The user account must exist.

3.2.5. Post-conditions

If the use case is successful, the user logged into the system successfully. Otherwise, the system state is unchanged.

3.3. Use case: **Maintain Admission Form: Send Form and Upload Files** (SUB-FEATURE 2.1 and FEATURE 3)

3.3.1. Brief description

This use case describes how a student Creates and Sends Admission Form .

3.3.2. Flow of events

3.3.2.1. Basic Flow

Table 7. Send Form and Upload Files Basic Flow

Actor Action	System Action	Data
1. The actor select type “Student” and logged into the system	2. The system checks if there is any Admission Wave available. If there is, the system checks if there is an Admission form data in the Database with the actor id. If there is not, system displays 1 option in the current Admission Wave: “Admission Form”	- Wave status - Admission Information
3. The actor chooses “Admission Form”	4. The system requests the actor to add his/ her information	
5. The actor enters his/ her information and chooses “Next”	6. The system adds his/her information into the Database and requests the actor to upload his/ her attachment: Motivation letter(required) and Adwards(optional) with right name and format	- Student Information - Application Information
7. The actor chooses files and chooses “Submit”	8. The system validates the format of the file and creates a folder with name as the student_id and uploads the student files in that folder. The system returns a review of the Student Admission Form as PDF.	- Files

3.3.2.2. Alternatives Flow

a. Invalid file format.

At step 7 in the Basic Flow:

Table 8. Send Form and Upload Files Alternatives Flow

Actor Action	System Action	Data
7. The actor chooses files and chooses “Submit”	8. The system validates the format of the file. If the file has invalid format, the system displays an error “This file format is invalid” and requests the actor to upload the right format file	- Files

3.3.3. Special requirements

There is an Active Admission Wave

3.3.4. Pre-conditions

The user must logged into the system

3.3.5. Post-conditions

If the use case is successful, the Student Admission Form is saved in the Database.

3.4. Use case: **Maintain Admission Form: Edit Form (SUB-FEATURE 2.2)**

3.4.1. Brief description

This use case describes how a student Edit and Update Admission Form .

3.4.2. Flow of events

3.4.2.1. Basic Flow

Table 9. Edit Form Basic Flow

Actor Action	System Action	Data
1. The actor select type “Student” and logged into the system	2. The system checks if there is any Admission Wave available. If there is, the system checks if there is an Admission form data in the Database with the actor id. If there is, system display 2 options in current Admission Wave: “Edit Form” and “Export Form”	- Wave status - Admission Information
3. The actor chooses “Edit Form”	4. The system displays the user Admission Form and requests the actor to edit his/ her information	
5. The actor enters his/ her information and chooses “Next”	6. The system updates his/her information into the Database and requests the actor to upload his/ her attachment: Motivation letter(required) and Adwards(optional) with right name and format	- Student Information - Application Information
7. The actor chooses files and chooses “Submit”	8. The system validates the format of the file and updates the student file in the folder with name as the student_id. The system returns a review of the Student Admission Form.	- Files

3.4.3. Special requirements

There is an Active Admission Wave and there is Admission Form data with this student Id in the Database

3.4.4. Pre-conditions

The user must logged into the system

3.4.5. Post-conditions

If the use case is successful, the Student Admission Form is updated in the Database.

3.5. Use case: **Maintain Admission Form: Export Form (SUB-FEATURE 2.3)**

3.5.1. Brief description

This use case describes how a student Export Admission Form as a PDF file.

3.5.2. Flow of events

3.5.2.1. Basic Flow

Table 10. Export Form Basic Flow

Actor Action	System Action	Data
1. The actor select type “Student” and logged into the system	2. The system checks if there is any Admission Wave available. If there is, the system checks if there is an Admission form data in the Database with the actor id. If there is, system display 2 options in current Admission Wave: “Edit Form” and “Export Form”	- Wave status - Admission Information
3. The actor chooses “Export Form”	4. The system returns a review of the Student Admission Form.	- Student Information - Application Information
5. The actor chooses “Export as PDF”	6. The system display a small window with options for the actor	- Student Information - Application Information

	to custom his/ her Admission Form	
7. The actor chooses “Download”	8. The system converts the Admission Form into PDF.	

3.5.3. Special requirements

There is an Active Admission Wave and there is Admission Form data with this student Id in the Database

3.5.4. Pre-conditions

The user must logged into the system

3.5.5. Post-conditions

If the use case is successful, the Admission Form is downloaded as a PDF file.

3.6. Use case: **Maintain Admission Form: Check Form Status (SUB-FEATURE 2.4)**

3.6.1. Brief description

This use case describes how a student can keep track of his/ her Admission Form Status.

3.6.2. Flow of events

3.6.2.1. Basic Flow

Table 11. Check Form Status Basic Flow

Actor Action	System Action	Data
1. The actor select type “Student” and logged into the system	2. The system checks if there is any Admission Wave available. If there is, the system checks if there is an	- Wave status - Admission Information

	Admission form data in the Database with the actor id. If there is, system display the status of actor Admission Form (Submitted/ Pending... or Approved)	
--	--	--

3.6.3. Special requirements

There is an Active Admission Wave and there is Admission Form data with this student Id in the Database

3.6.4. Pre-conditions

The user must logged into the system

3.6.5. Post-conditions

If the use case is successful, the status of the Actor Admission Form will be displayed in the Student Home Page.

3.7. Use case: **Follow Admission Wave Status (FEATURE 4)**

3.7.1. Brief description

This use case describes how a student can keep track of Admission Waves Information.

3.7.2. Flow of events

3.7.2.1. Basic Flow

Table 12. Follow Admission Wave Status Basic Flow

Actor Action	System Action	Data
1. The actor select type “Student” and logged into the system	2. The system list all the Admission Wave of this current year	- Wave status - Admission Information

3.7.3. Special requirements

There is an Active Admission Wave and there is Admission Form data with this student Id in the Database

3.7.4. Pre-conditions

The user must logged into the system

3.7.5. Post-conditions

If the use case is successful, the status of the Actor Admission Form will be displayed in the Student Home Page.

3.8. Use case: **Interview: View Student Information (SUB-FEATURE 5.1)**

3.8.1. Brief description

This use case describes how a lecturer can View a Student Information and Attachment.

3.8.2. Flow of events

3.8.2.1. Basic Flow

Table 13. View Student Information Basic Flow

Actor Action	System Action	Data
1. The actor select type “Lecturer” and logged into the system	2. The system gets the Jury of the actor and checks if this Jury is available. If it is, the system lists all the students in this Jury.	- Jury Status - Employee Information
3. The actor type name of the target Student and chooses “Search”	4. The system displays the Student with entered name	
5. The actor selects the target Student and choses “Detail”	6. The system displays the target Student Admission form and attachment files.	- Admission Information - Files

3.8.2.2. Alternatives Flow

a. There is no Jury active.

At step 1 in the Basic Flow:

Table 14. View Student Information Alternatives Flow 1

Actor Action	System Action	Data
1. The actor select type “Lecturer” and logged into the system	2. The system gets the Jury of the actor and checks if this Jury is available. If this Jury is not available, the system displays an error message “Your Jury is not available.”	- Jury Status - Employee Information

b. Student name not found.

At step 3 in the Basic Flow:

Table 15. View Student Information Alternatives Flow 2

Actor Action	System Action	Data
3. The actor type name of the target Student and chooses “Search”	4. The system displays an empty list.	

3.8.3. Special requirements

The Jury of the Lecturer is available.

3.8.4. Pre-conditions

The user must logged into the system with type “Lecturer”

3.8.5. Post-conditions

If the use case is successful, the information and attachment files of the target Student will be displayed

.

3.9. Use case: **Interview: Evaluate Student (SUB-FEATURE 5.2)**

3.9.1. Brief description

This use case describes how a Lecturer evaluate Students

3.9.2. Flow of events

3.9.2.1. Basic Flow

Table 16. Evaluate Student Basic Flow

Actor Action	System Action	Data
1. The actor select type “Lecturer” and logged into the system	2. The system gets the Jury of the actor and checks if this Jury is available. If it is, the system lists all the students in this Jury.	- Jury Status - Employee Information
3. The actor type name of the target Student and chooses “Search”	4. The system displays the Student with entered name	
5. The actor selects the target Student and chooses “Evaluate”	6. The system checks if this Lecturer is an English teacher. If he/she is not, displays the General Evaluation form	- Employee Information
7. The actor evaluate Student and chooses “Submit”	8. The system add this General Evaluation form with Lecturer Id and Student Id	- General Evaluation Information

3.9.2.2. Alternatives Flow

a. If the Lecturer is an English teacher.

At step 5 in the Basic Flow:

Table 17. Evaluate Student Alternatives Flow

Actor Action	System Action	Data
5. The actor selects the target Student and chooses “Evaluate”	6. The system checks if this Lecturer is an English teacher. If he/she is,, displays the English Evaluation form	- Employee Information
7. The actor evaluate Student and chooses “Submit”	8. The system add this English Evaluation form with Lecturer Id and Student Id	- English Evaluation Information

3.9.3. Special requirements

The Jury of the Lecturer is available.

3.9.4. Pre-conditions

The user must logged into the system with type “Lecturer”

3.9.5. Post-conditions

If the use case is successful, the General/ English Evaluation will be saved in the Database.

3.10. Use case: **Use Questions Bank (FEATURE 6)**

3.10.1. Brief description

This use case describes how a Lecturer searches a question in Question Bank

3.10.2. Flow of events

3.10.2.1. Basic Flow

Table 18. Use Question Bank Flow

Actor Action	System Action	Data
1. The actor select type “Lecturer” and logged into the system	2. The system gets the Jury of the actor and checks if this Jury is available. If it is, the system lists all the students in this Jury.	- Jury Status - Employee Information
3. The actor type question Id of the target Question and chooses “Search”	4. The system gets the question with the entered Id from the Question Bank in the Database displays Question and Answer.	- Question Bank

3.10.2.2. Alternatives Flow

a. There is no question with entered Id

At step 3 in the Basic Flow:

Table 19. Use Question Bank Alternatives Flow

Actor Action	System Action	Data
3. The actor type question Id of the target Question and chooses “Search”	4. The system gets the question with the entered Id from the Question Bank in the Database. If there is no question with that Id, displays an error message “Question not found”	- Question Bank

3.10.3. Special requirements

None.

3.10.4. Pre-conditions

The user must be logged into the system with type “Lecturer”

3.10.5. Post-conditions

If the use case is successful, the question and answer will be displayed.

IV. METHODOLOGY

In this section, we will list all the tools and techniques used in the project, the reasons why they are chosen and the detailed use cases implementation.

1. Tools and Techniques

1.2. SQL

With the amount of data in our system, it is important to choose the proper Database to manage the data. There are two types of Database: using SQL and NoSQL, each database has its own advantages and disadvantages.

Table 20. Differences between SQL and NoSQL

	SQL	NoSQL
<i>Type of database</i>	Relational Database	Non-relational Database
<i>Schema</i>	Pre-defined Schema	Dynamic Schema
<i>Database Categories</i>	Table based Databases	Document-based databases, Key-value stores, graph stores, wide column stores
<i>Complex Queries</i>	Good for complex queries	Not a good fit for complex queries
<i>Hierarchical Data Storage</i>	Not the best fit	Fits better when compared to SQL
<i>Scalability</i>	Vertically Scalable	Horizontally Scalable

SQL is suitable for a project with a fixed schema, high transaction, low maintenance, data security with a limited budget and NoSQL is for unstable schema, high availability, cloud computing, with in-built sharding.

In this project, the data has been fully predefined and it may be expanded in size but contain in its structure and relation. Furthermore, the student application information, lecturer evaluation and

USTH administration department must be strictly connected. Therefore, the best solution of our system is using SQL Database.

1.3. PHP

PHP- Hypertext Preprocessor is used for server-side programming for many reasons. First of all, PHP is compatible with MySQL which is our database so it makes sense for us to use it. This language also has very good online documents with a huge community makes it much more convenient to search and learn. Finally, PHP is open source so the cost of using it is minimal.

1.4. Postman

Postman is a collaboration platform for API development, it simplifies each step of building an API and streamline collaboration. In this project, Postman is used to create collections of integration tests to ensure the API is working as expected. This helps to save hours of manually testing.

1.5. VueJS

In the front-end development environment, the most common JavaScript in the instance of time are AngularJS and VueJS, the most popular JavaScript library is ReactJS.

VueJS and ReactJS share many similarities. They both: utilize a virtual DOM, provide reactive and composable view components and maintain focus in the core library, with concerns such as routing and global state management handled by companion libraries. But in ReactJS, all components express their UI using JSX, a declarative JavaScript XML-like syntax while VueJS uses HTML templates to create views. Since HTML-based templates are much more familiar VueJS provides an easy learning curve.

AngularJS uses TypeScript, a subset of JavaScript, it offers a variety of structure types like Injectables, Components, Modules, Pipes and more so its learning curve is steep. The complexity of Angular is largely due to its design goal of targeting only large, complex applications. Since our project is aimed at a small to medium scale so using AngularJS is not suitable.

In general, VueJS is a progressive framework for building user interfaces, it is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is

easy to pick up and integrate with other libraries or existing projects. For this project, according to the condition and requirement of the system environment, VueJS should be a great choice compared to others.

1.6. Axios

Axios is a Javascript library used to perform HTTP requests for retrieving, posting, deleting, and modifying data from APIs.

1.7. Element

Element UI is a Desktop UI toolkit for Vue.js, containing many pre-built components that match perfectly with Vue Data Binding properties. The project documentation is very well written and organized so it is easy for us to follow.

2. System Architecture

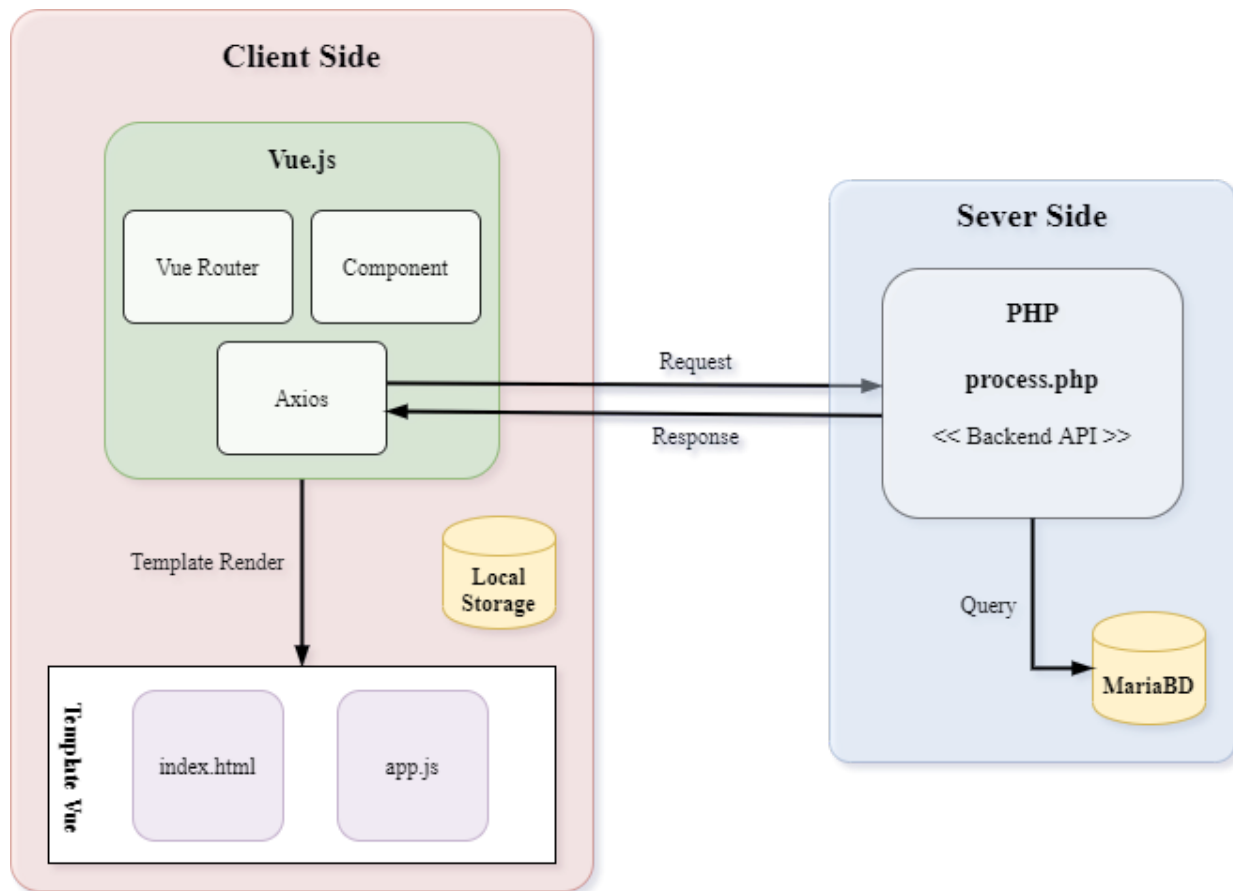


Figure 2. System Architecture diagram

This Online Admission Application and Interview System is a full side project with a completely developed Client side(Front-end) and Server side(Back-end):

- The Server side receives the request from the Client-side, processes and gets data from the Database then returns a response.
- The Client side interacts with the user, retrieves the user request, sends it to Sever side, receives the response from the Server then renders the response data for display.

2.1 Client Side

Client side of the application includes three main parts: Vue component, Vue Router and Axios:

- Vue Router is the official router for Vue.js. It deeply integrates with Vue.js core to make building Single Page Applications with Vue.js much more easier. This allows us to navigate between pages in our application.
- Axios is imported into components in order to send requests and handle responses from HTTP requests between client and server.
- Components are reusable Vue instances with a name. Many components are included in the web front-end. These components are arranged as the below figure. Sharing data between components is divided into two types: pass data from parent component to child component (using prop) and from child component to parent component (by emitting event).

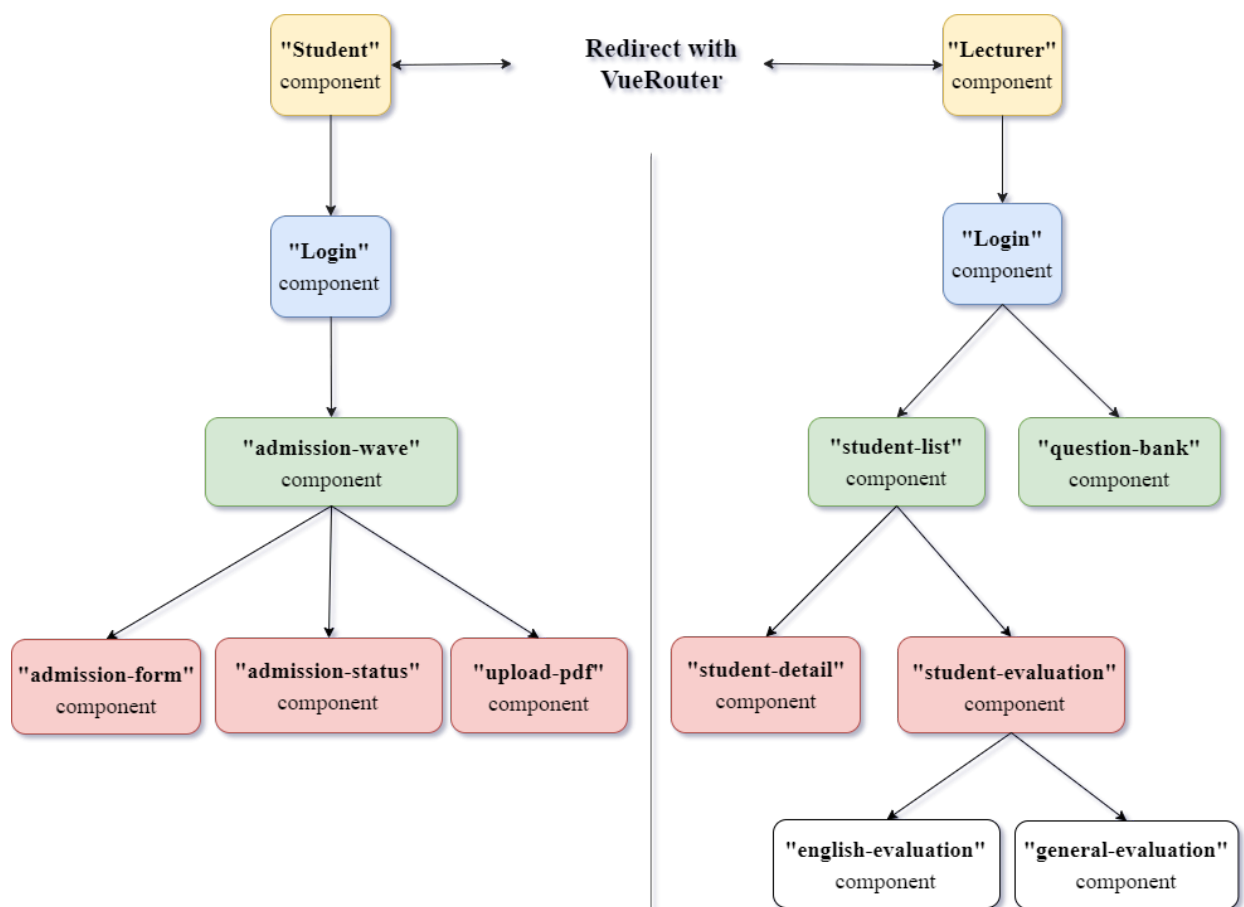


Figure 3. Frontend Component Design

2.2 Server Side

This is a Client side rendering web application, the Back-end takes the responsibility of creating a secure connection between the web browser and system's database. This side include two main parts:

- MySQL: A database management system to manage all the system data.
- PHP for API processing (including receive requests, authenticating requests, communicating with database, return response in JSON form). Structure of an API contains:
 - Define action.
 - Request verification with token.
 - Query from Database.
 - Response:
 - Return a response data in JSON format if success.
 - Return an error message if there is an error.

3. Database Design

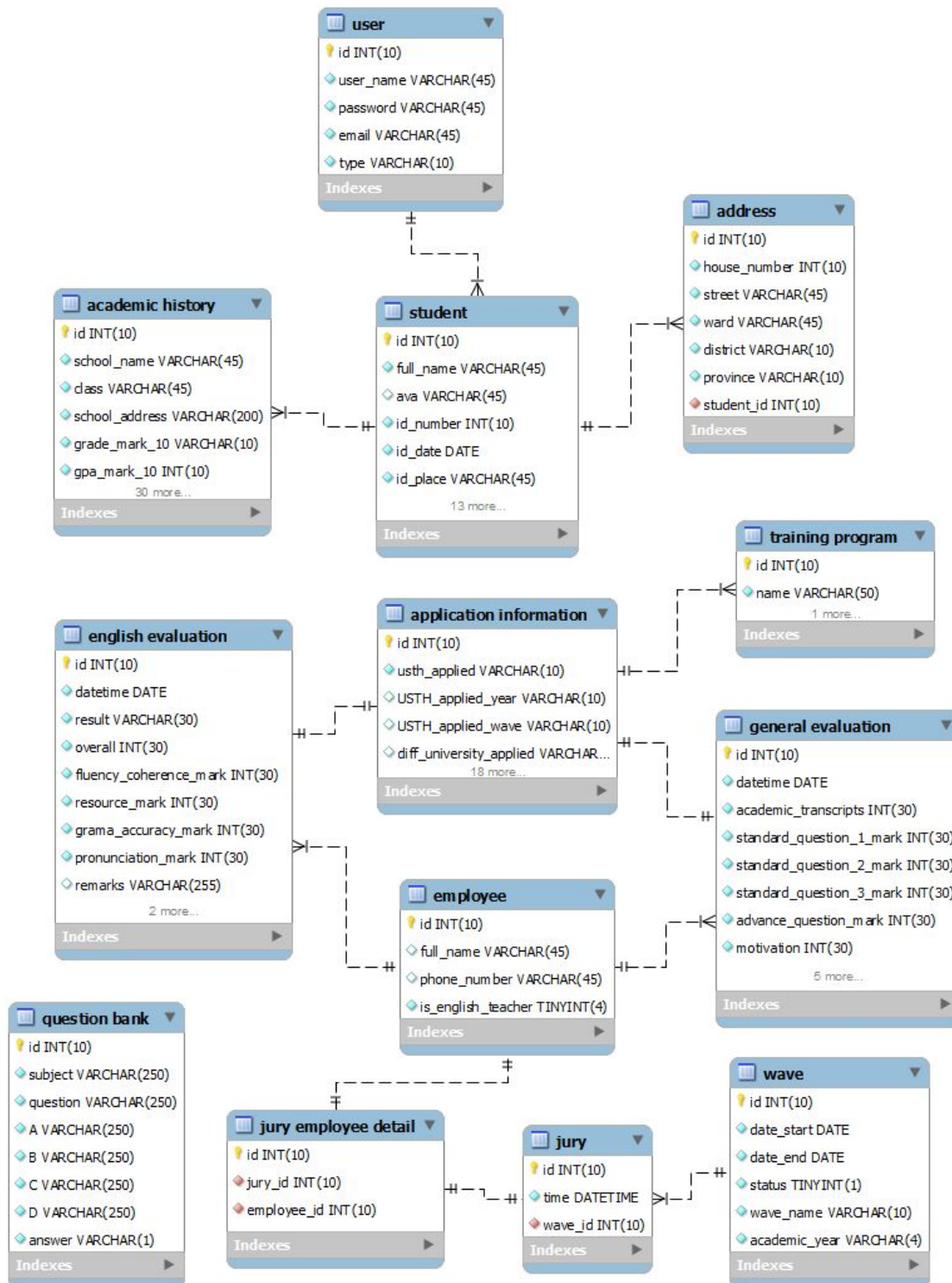
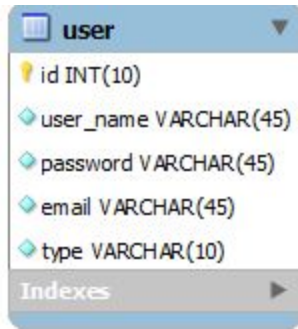


Figure 4. Overall Database Design

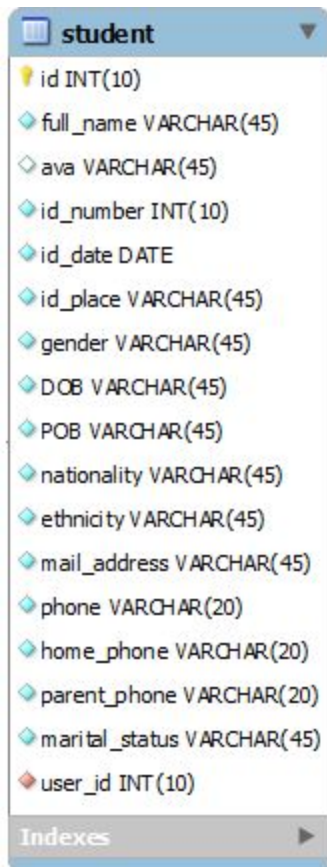
Table 21. User Database Design



user	
id	INT(10)
user_name	VARCHAR(45)
password	VARCHAR(45)
email	VARCHAR(45)
type	VARCHAR(10)
Indexes	

- “user” table is the table that contains the information of accounts. This table has sensitive data of users such as: “password”
- The ‘password’ field in this table is saved in encrypted form so that it can not be easily used by attackers.
- After the user register, the system saves the user information and assigns a unique “id” to the user and this “id” is the Primary key of the table.

Table 22. Student Database Design



student	
id	INT(10)
full_name	VARCHAR(45)
ava	VARCHAR(45)
id_number	INT(10)
id_date	DATE
id_place	VARCHAR(45)
gender	VARCHAR(45)
DOB	VARCHAR(45)
POB	VARCHAR(45)
nationality	VARCHAR(45)
ethnicity	VARCHAR(45)
mail_address	VARCHAR(45)
phone	VARCHAR(20)
home_phone	VARCHAR(20)
parent_phone	VARCHAR(20)
marital_status	VARCHAR(45)
user_id	INT(10)
Indexes	

- “student” table is a table containing the basic personal information of a student.
- This table has “id” as Primary Key
- This table connects with the “user” table with the Foreign Key: ‘user_id’. This is “one to one” relationship: Each “User” corresponding to a “Student”

Table 23. Application Information
Database Design

application information	
id	INT(10)
usth_applied	VARCHAR(10)
USTH_applied_year	VARCHAR(10)
USTH_applied_wave	VARCHAR(10)
diff_university_applied	VARCHAR(10)
diff_university_applied_year	INT(10)
diff_university_applied_mark	VARCHAR(45)
diff_university_applied_group	VARCHAR(45)
diff_university_applied_name	VARCHAR(45)
national_exam_student_number	INT(10)
national_exam_math_result	INT(10)
national_exam_physics_result	INT(10)
national_exam_chemistry_result	INT(10)
national_exam_biology_result	INT(10)
national_exam_english_result	INT(10)
motivation_letter	VARCHAR(255)
first_training_program_id	INT(10)
second_training_program_id	INT(10)
third_training_program_id	INT(10)
fourth_training_program_id	INT(10)
status	TINYINT(4)
student_id	INT(10)
jury_id	INT(10)
Indexes	

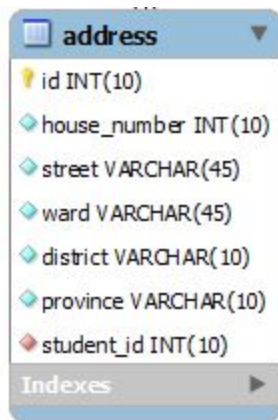
- “application information” table contains information that applicants used to apply for a specific enrollment registration period.
- This table has “id” as Primary Key
- This table connects with the “jury” table with the Foreign Key: “jury_id” . This is “one to one” relationship: Each “application form” has a “jury”
- This table connects with the “student” table with the Foreign Key: “student_id” . This is “one to many” relationship: Each “student” can have many “application information” (which means 1 student can apply multiple times into the university)

Table 24. Academic History
Database Design

academic history	
id	INT(10)
school_name	VARCHAR(45)
class	VARCHAR(45)
school_address	VARCHAR(200)
grade_mark_10	VARCHAR(10)
gpa_mark_10	INT(10)
maths_mark_10	INT(10)
physics_mark_10	INT(10)
chemistry_mark_10	INT(10)
biology_mark_10	INT(10)
it_mark_10	INT(10)
english_mark_10	INT(10)
grade_mark_11	VARCHAR(10)
gpa_mark_11	INT(10)
maths_mark_11	INT(10)
physics_mark_11	INT(10)
chemistry_mark_11	INT(10)
biology_mark_11	INT(10)
it_mark_11	INT(10)
english_mark_11	INT(10)
grade_mark_12	VARCHAR(10)
gpa_mark_12	INT(10)
maths_mark_12	INT(10)
physics_mark_12	INT(10)
chemistry_mark_12	INT(10)
biology_mark_12	INT(10)
it_mark_12	INT(10)
english_mark_12	INT(10)
english_certi	VARCHAR(45)
english_score	VARCHAR(45)
french_certi	VARCHAR(45)
french_score	VARCHAR(45)
viet_certi	VARCHAR(45)
viet_score	VARCHAR(45)
adward	VARCHAR(255)
student_id	INT(10)

- “academic history” table contains academic result in highschool and national exam of an application
- This table has “id” as Primary Key
- This table connects with the “student” table with the Foreign Key: “student_id” . This is “one to many” relationship: Each “student” can have many “academic history”

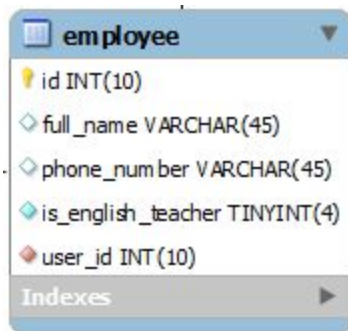
Table 25. Address Database Design



address	
id	INT(10)
house_number	INT(10)
street	VARCHAR(45)
ward	VARCHAR(45)
district	VARCHAR(10)
province	VARCHAR(10)
student_id	INT(10)
Indexes	

- “address” table contains address of an application
- This table has “id” as Primary Key
- This table connects with the “student” table with the Foreign Key: “student_id” . This is “one to many” relationship: Each “student” can have many “academic history”

Table 26. Employee Database Design



employee	
id	INT(10)
full_name	VARCHAR(45)
phone_number	VARCHAR(45)
is_english_teacher	TINYINT(4)
user_id	INT(10)
Indexes	

- “employee” table contains basic information about the lecturers and staff.
- This table has “id” as Primary Key
- This table connects with the “user” table with the Foreign Key: ‘user_id’. This is “one to one” relationship: Each “User” corresponding to a “Employee”

Table 27. English Evaluation
Database Design

Field	Type	Key
id	INT(10)	Primary Key
datetime	DATE	
result	VARCHAR(30)	
overall	INT(30)	
fluency_coherence_mark	INT(30)	
resource_mark	INT(30)	
grama_accuracy_mark	INT(30)	
pronunciation_mark	INT(30)	
remarks	VARCHAR(255)	
employee_id	INT(10)	
application_information_id	INT(10)	

- “english evaluation” table contains english assessment criteria for the interview.
- This table has “id” as Primary Key
- This table connects with the “employee” table with the Foreign Key: ‘employee_id’. This is “one to one” relationship: Each “English evaluation” corresponding to a “Employee”
- This table connects with the “application information” table with the Foreign Key: ‘application_information_id’. This is “one to one” relationship: Each “English evaluation” corresponding to a “Application Information”

Table 28. General Evaluation
Database Design

Field	Type	Key
id	INT(10)	Primary Key
datetime	DATE	
academic_transcripts	INT(30)	
standard_question_1_mark	INT(30)	
standard_question_2_mark	INT(30)	
standard_question_3_mark	INT(30)	
advance_question_mark	INT(30)	
motivation	INT(30)	
other_achievements	INT(30)	
remarks	VARCHAR(500)	
employee_id	INT(10)	
application_information_id	INT(10)	

- “general evaluation” table contains general assessment criteria for the interview.
- This table has “id” as Primary Key
- This table connects with the “employee” table with the Foreign Key: ‘employee_id’. This is “one to one” relationship: Each “General evaluation” corresponding to a “Employee”
- This table connects with the “application information” table with the Foreign Key: ‘application_information_id’. This is “one to one” relationship: Each “General evaluation” corresponding to a “Application Information”

Table 29. Jury Employee Database Design

jury employee detail	
id	INT(10)
employee_id	INT(10)
jury_id	INT(10)
Indexes	

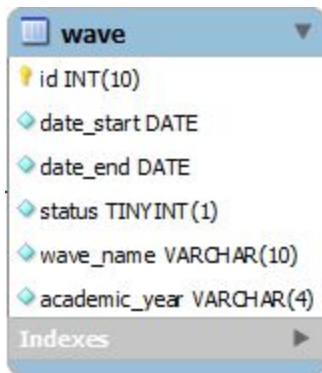
- “jury employee detail” table represent for the distribution of employee into different jury (In a wave, each “Employee” belong to a “Jury”; Each “Jury” has many “Employees”)
- This table has “id” as Primary Key
- This table connects with the “employee” table with the Foreign Key: ‘employee_id’. This is an “one to one” relationship.
- This table connects with the “jury” table with the Foreign Key: ‘jury_id’. This is an “one to one” relationship.

Table 30. Jury Database Design

jury	
id	INT(10)
time	DATETIME
wave_id	INT(10)
Indexes	

- “jury” table contains basic information about the jury: time, wave,...
- This table has “id” as Primary Key
- This table connects with the “jury” table with the Foreign Key: ‘jury_id’. This is “one to many” relationship: Each “Wave” has many “Juries”

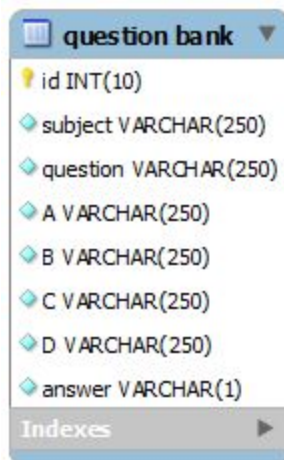
Table 31. Wave Database Design



wave	
id	INT(10)
date_start	DATE
date_end	DATE
status	TINYINT(1)
wave_name	VARCHAR(10)
academic_year	VARCHAR(4)
Indexes	

- “wave” table contains basic information about the jury: date_start, date_end, status, academic year...
- This table has “id” as Primary Key

Table 32. Question Bank
Database Design



question bank	
id	INT(10)
subject	VARCHAR(250)
question	VARCHAR(250)
A	VARCHAR(250)
B	VARCHAR(250)
C	VARCHAR(250)
D	VARCHAR(250)
answer	VARCHAR(1)
Indexes	

- “question bank” table contains a list of subjects, questions and answers for application in general evaluation.
- This table has “id” as Primary Key

4. Use Cases Implementation

4.1. Login

This use case describes how a user logs into the system.

The user can be a student or a lecturer.

When a user wants to login, the system requests the actor to select his/ her type (student or lecturer) (figure 15 in section VII: Appendix). After the type is selected, the system displays a login screen and requests the user to enter email and password (figure 16 in section VII: Appendix). Once the user enters the email and password, the system encrypts the password and validates the email, password and user type entered from the user table in the Database. If valid, the system logs the actor in. Otherwise, the system displays an error message and asks the user to re-enter the information.

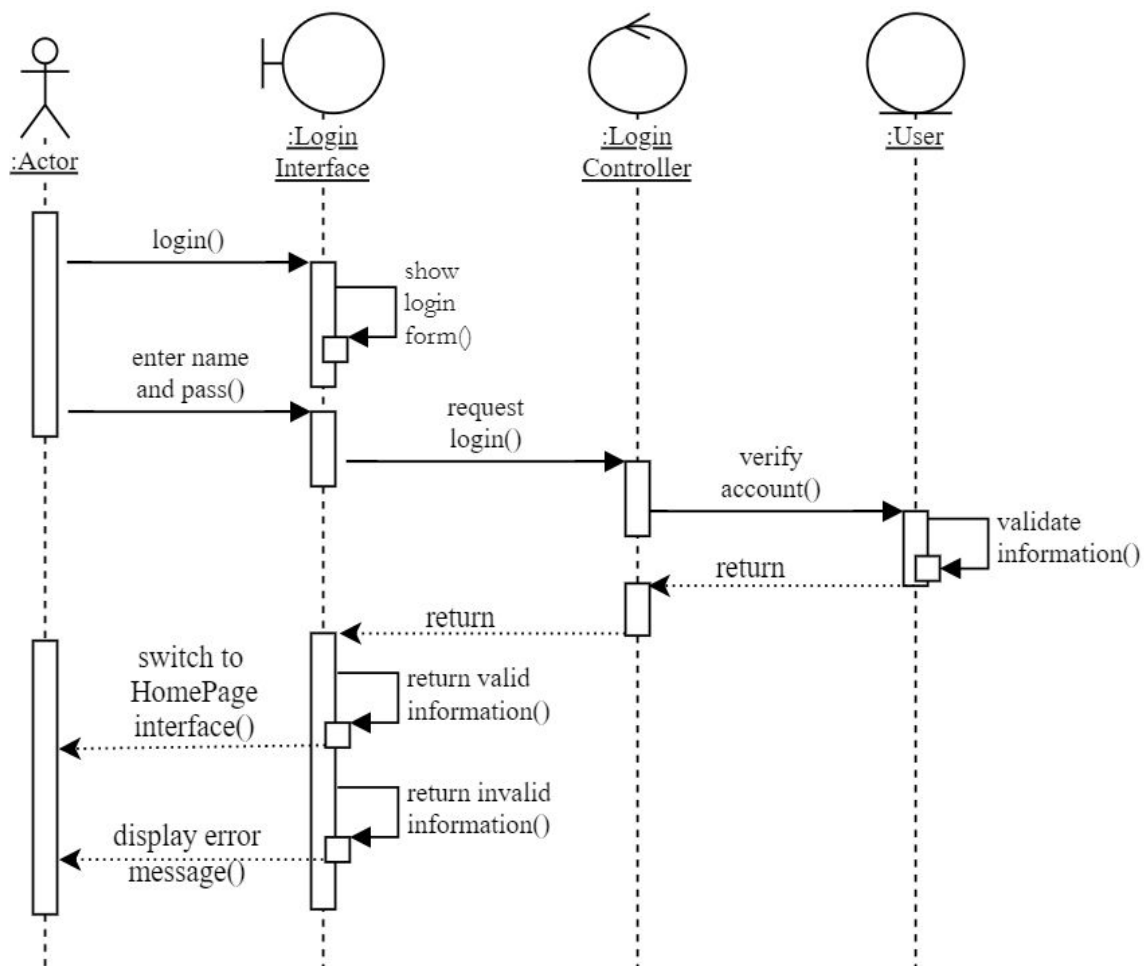


Figure 5. Login Sequence diagram

4.2. Register

This use case describes how a user registers to the Online Admission Application and Interview System.

This user is a student.

When a user wants to register into the system (by clicking on “Create new account”), the system displays a register interface and requests the user to enter email, username and password (figure 16 in section VII: Appendix). After the user enters the required information, the system checks if the entered email is unique from the table “user” in Database. If unique, the system encrypts the password and saves the user information, assigns a unique id to the user and displays a success message. Otherwise, the system displays an error message and asks the user to reenter the information.

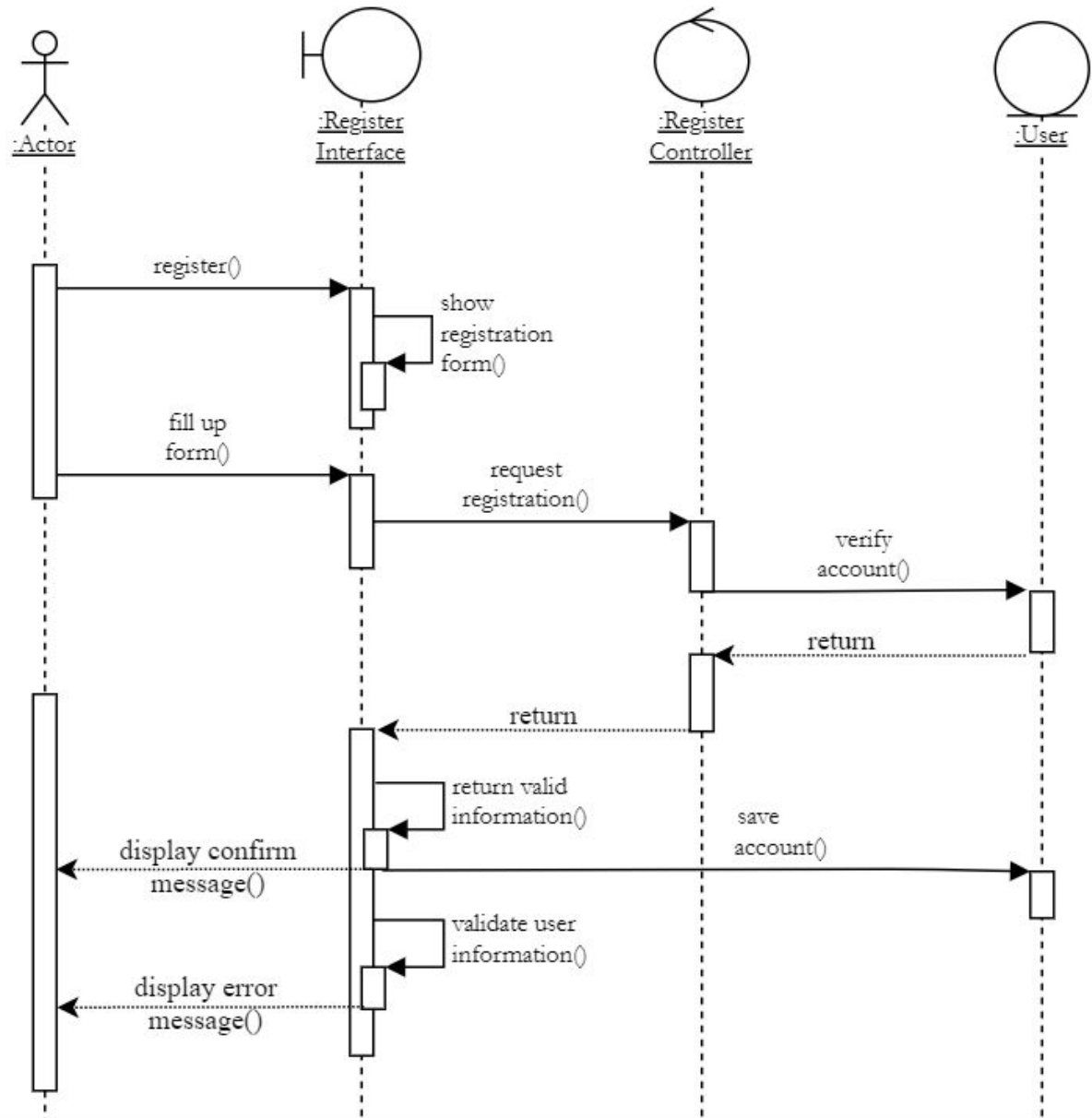


Figure 6. Register Sequence diagram

4.3. Maintain Admission Form: Send Form

This use case describes how a user Creates and Sends Admission Form.

This user is a student.

When the user logged into the system, the system checks if there is any Admission Wave available, the system will check for any existing Admission form data in the Database corresponding to the user id. If there is not, the system displays 1 option in the current Admission Wave: “Admission Form” (figure 18 in section VII: Appendix). After the user selects “Admission Form”, the system displays the Admission Form and requests the actor to add his/ her information (figure 19 in section VII: Appendix). Once the user enters his/ her information and chooses “Next”, the system adds his/her information into the Database and requests the actor to upload his/ her attachment: Motivation letter(required) and Adwards(optional) with right name and format (figure 20 in section VII: Appendix). When the user chooses files and the button “Submit” is clicked, the system validates the format of the file and creates a folder with name as the student_id and uploads the student files in that folder. Finally, the system returns a review of the Student Admission Form.

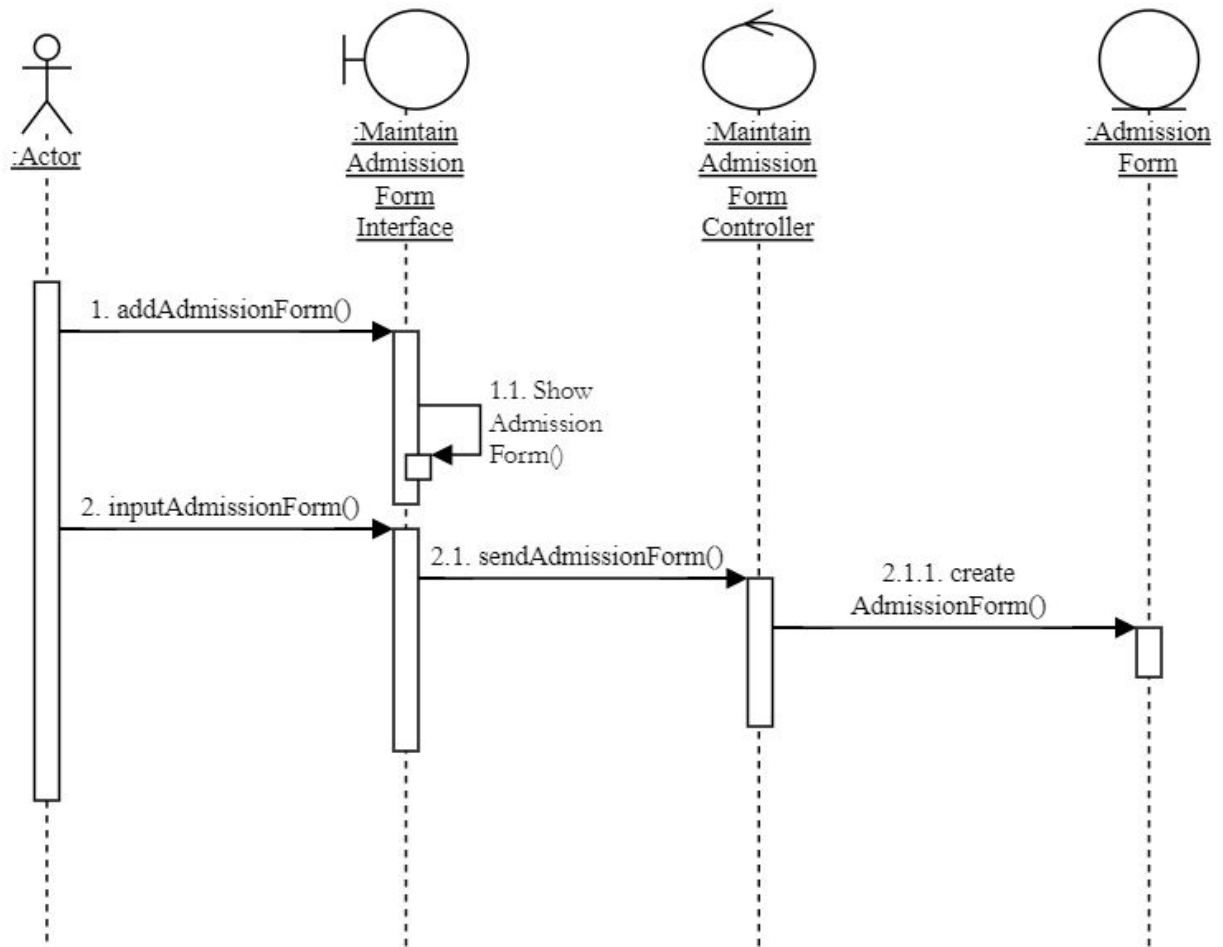


Figure 7. Send Admission Form Sequence diagram

4.4. Maintain Admission Form: Edit Form

This use case describes how a student Edit and Update Admission Form .

This user is a student.

When the user logged into the system, the system checks if there is any Admission Wave available, the system will check for any existing Admission form data in the Database corresponding to the user id. If all conditions are met, the system displays 2 options in the current Admission Wave: “Edit Form” and “Export Form”(figure 17 in section VII: Appendix). After the student chooses “Edit Form”, the system displays the user Admission Form and requests the actor to edit his/ her information(figure 19 in section VII: Appendix). Once the user edits his/ her information and chooses “Next”, the system updates his/her information into the Database and requests the actor to upload his/ her attachment: Motivation letter(required) and Adwards(optional) with right name and format (figure 20 in section VII: Appendix). When the uploads files, the system validates the format of the file and updates the student file in the folder with name as the student_id. The system returns a review of the Student Admission Form.

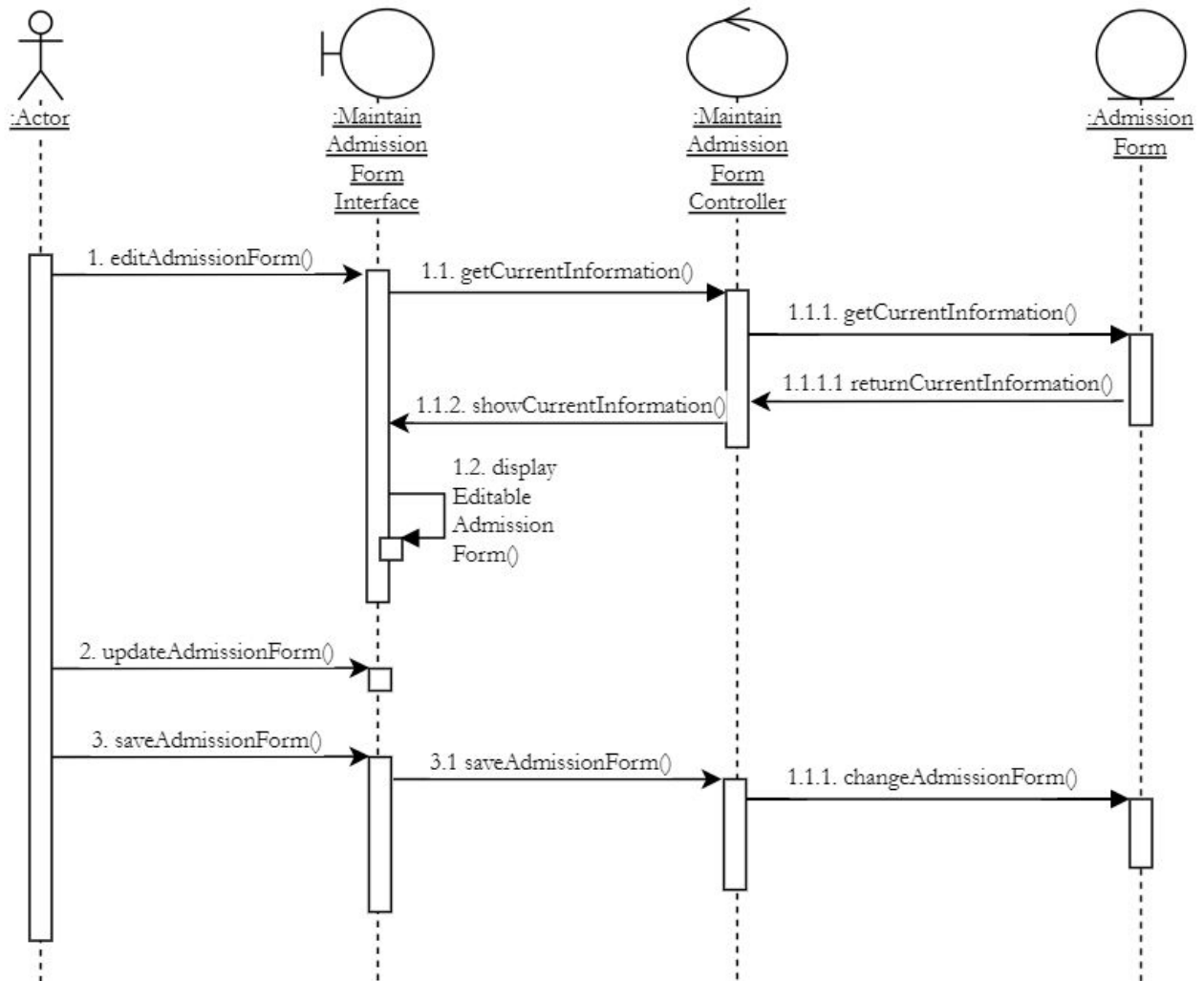


Figure 8. Edit Admission Form Sequence diagram

4.5. Maintain Admission Form: Export Form

This use case describes how a student Export Admission Form as a PDF file.

The user is a student.

When the user logged into the system, the system checks if there is any Admission Wave available, the system will check for any existing Admission form data in the Database corresponding to the user id. If all conditions are met, the system displays 2 options in current Admission Wave: “Edit Form” and “Export Form”. After the user chooses “Export Form”, the system returns a review of the Student Admission Form and allows the user to export form as PDF and download (figure 21 in section VII: Appendix).

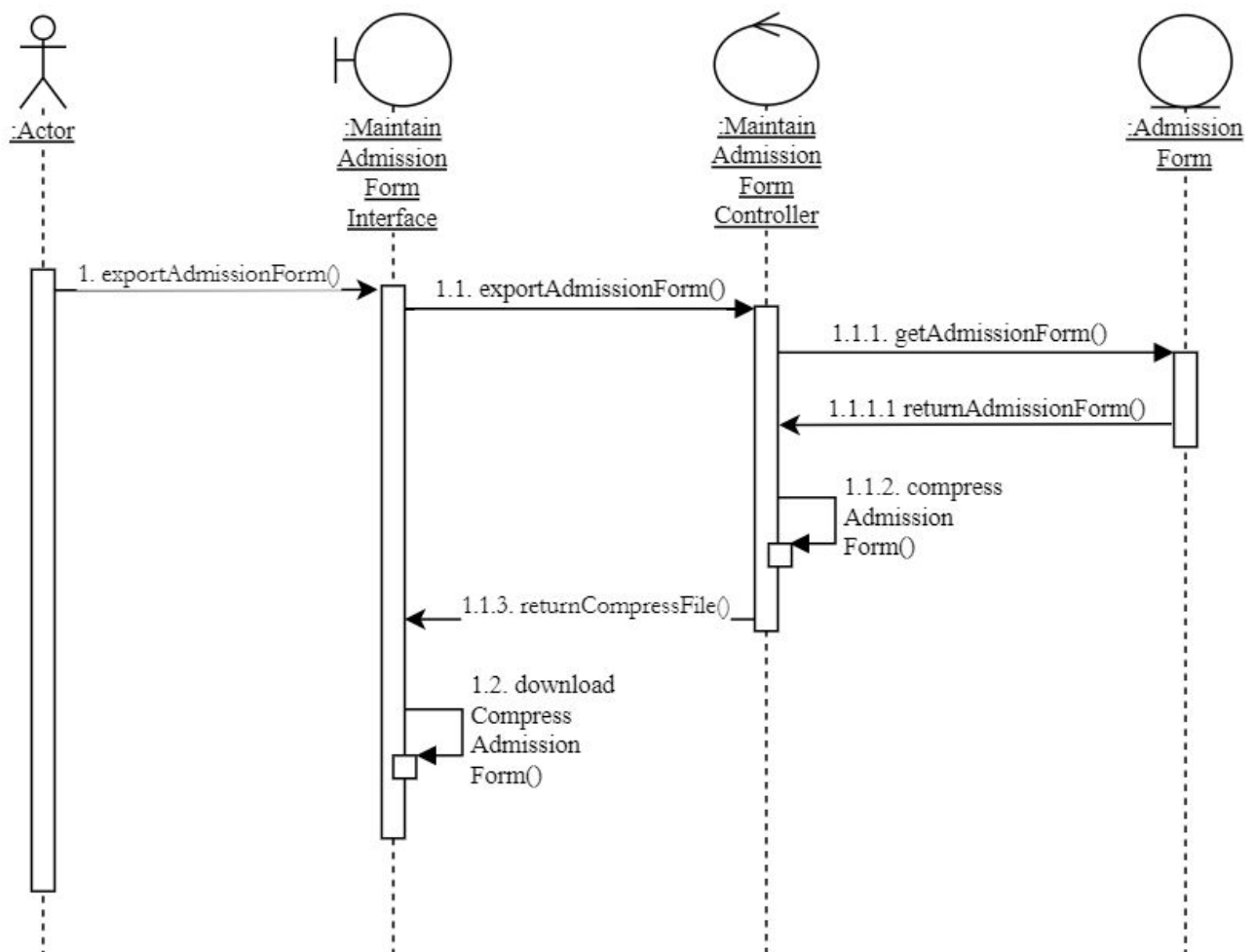


Figure 9. Export Admission Form Sequence diagram

4.6. Maintain Admission Form: Check Admission Form Status

This use case describes how a student can keep track of his/ her Admission Form Status.

The user is a student.

When the user logged into the system, the system checks if there is any Admission Wave available. If there is, the system checks if there is an Admission form data in the Database with the actor id. If there is, system display the status of actor Admission Form (Submitted/ Pending... or Approved) (figure 17 in section VII: Appendix)

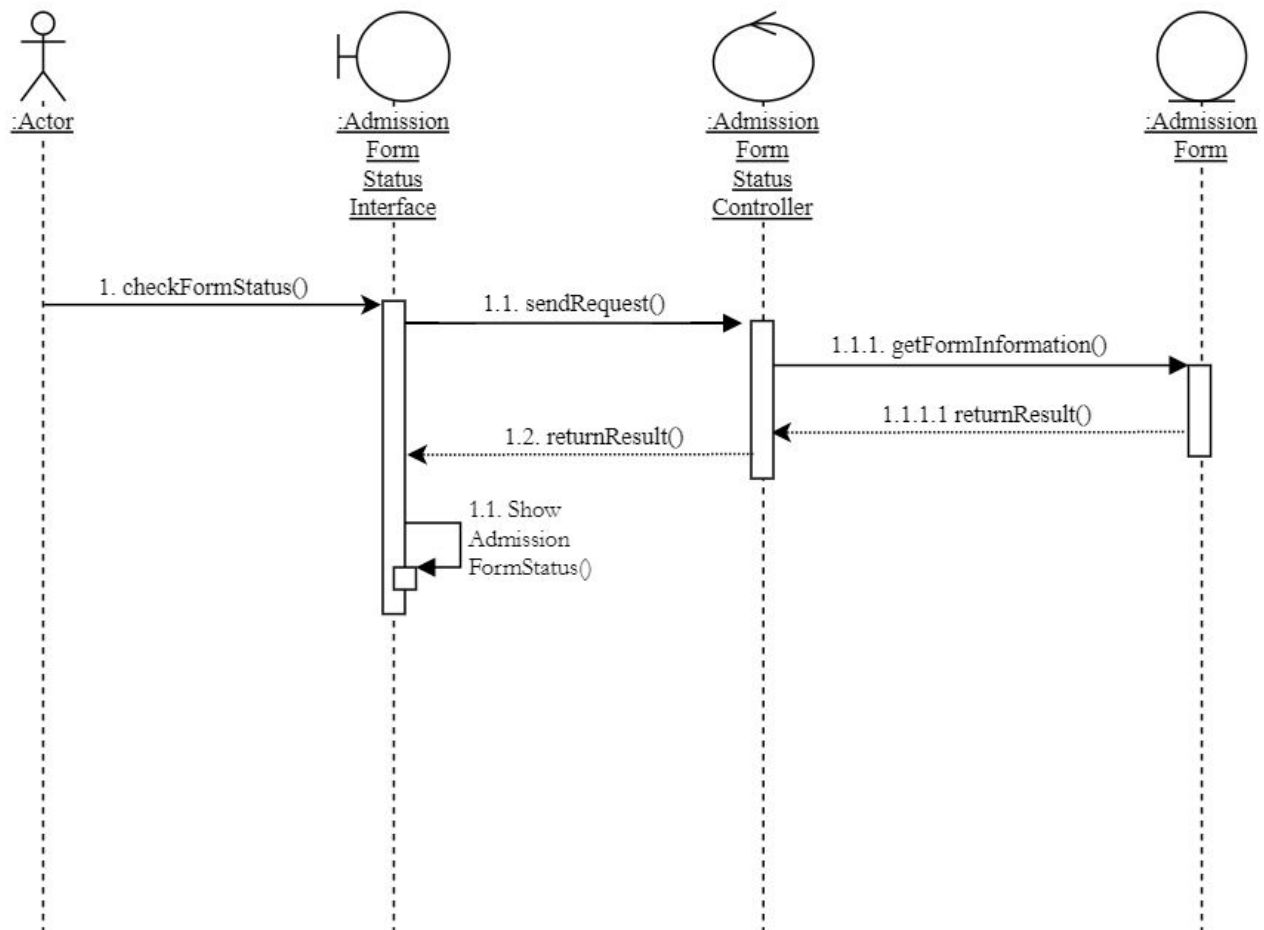


Figure 10. Check Admission Form Status Sequence diagram

4.7. Follow Admission Wave Status

This use case describes how a student can keep track of Admission Waves Information.

The user is a student.

When the user logged into the system, the system lists all the Admission Wave of this current year including start date, end date and the status of this wave. (figure 18 in section VII: Appendix)

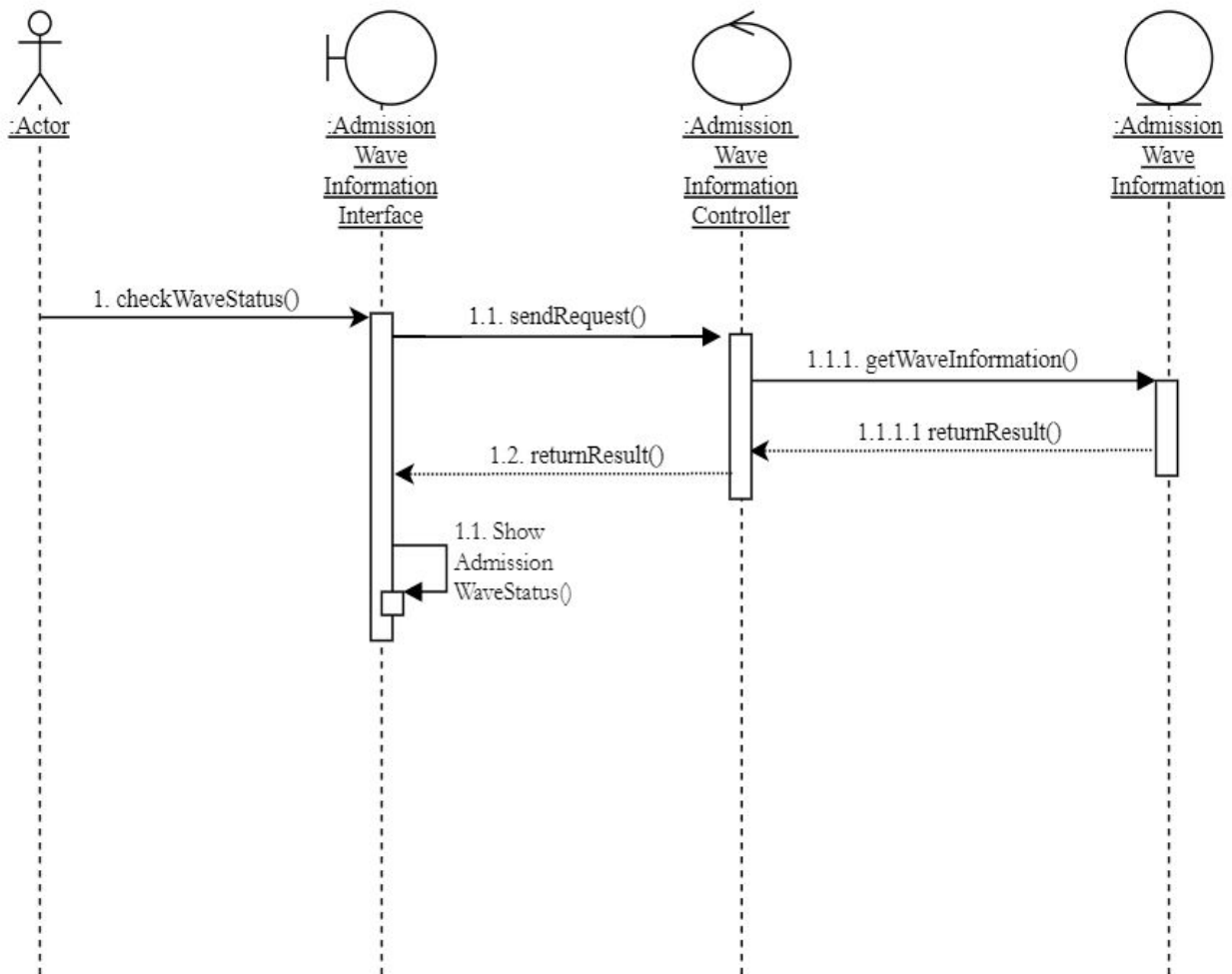


Figure 11. Follow Admission Wave Information Sequence diagram

4.8. Interview: View Student Information

This use case describes how a lecturer can View a Student Information and Attachment.

The user is a lecturer.

When the user logged into the system, the system gets the Jury of the actor and checks if this Jury is available. If it is, the system lists all the students in this Jury. The user can search students by entering the name of the target student and clicking the “Search” system displays the Student with entered name. After the user selects a student and clicks “Detail”, the system displays the target Student Admission form and attachment files (figure 22 in section VII: Appendix).

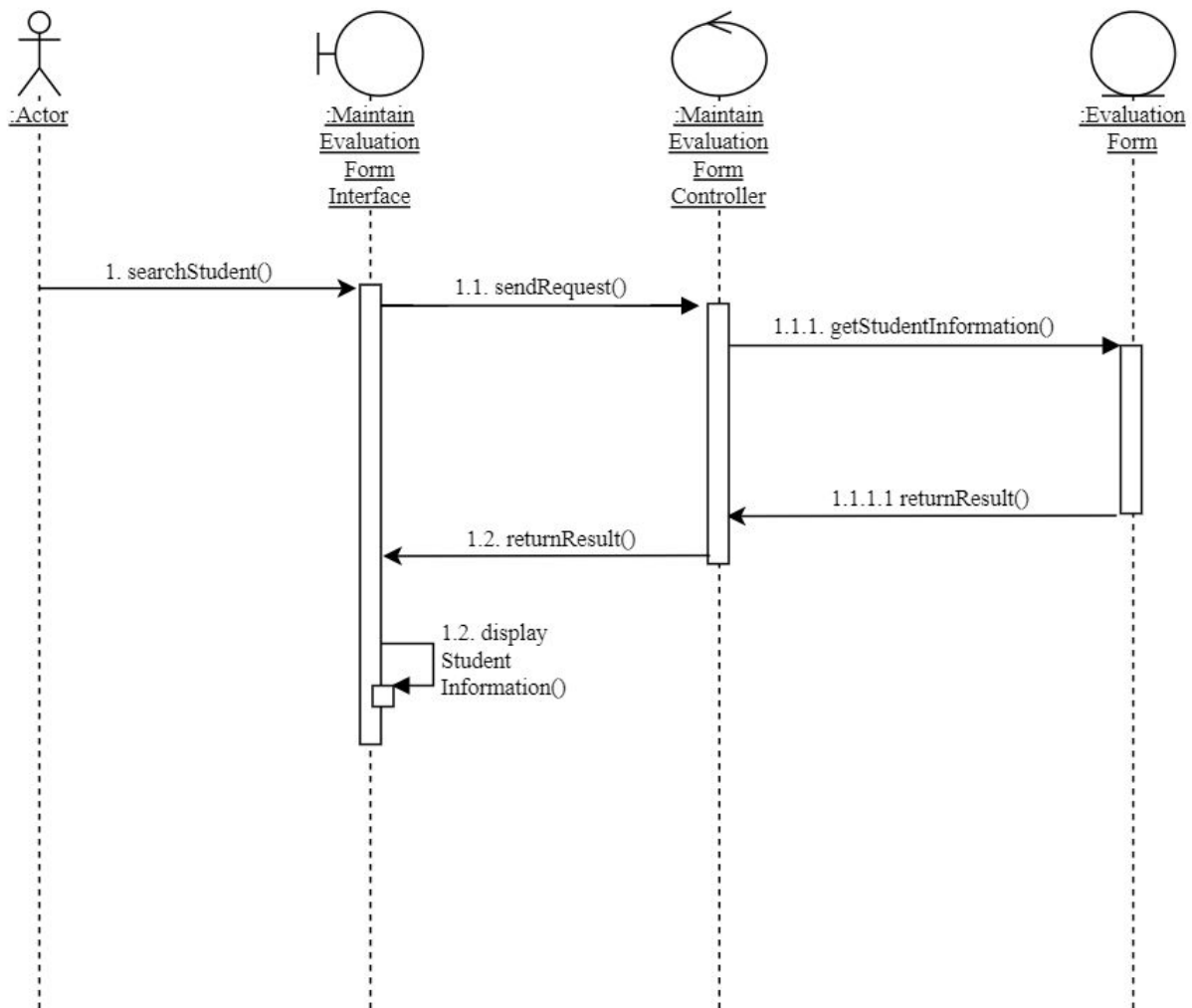


Figure 12. View Student Information Sequence diagram

4.9. Interview: Evaluate Student

This use case describes how a Lecturer evaluate Students

The user is a lecturer.

When the user logged into the system, the system gets the Jury of the actor and checks if this Jury is available. If it is, the system lists all the students in this Jury. The user can search students by entering the name of the target student and clicking the “Search” system displays the Student with entered name. After the user selects a student and clicks “Evaluate”, the system checks if this Lecturer is an English teacher. If he/she is not, displays the General Evaluation form(figure 23 in section VII: Appendix). Once the user evaluates the student and clicks “Submit”, system add this General Evaluation form with Lecturer Id and Student Id.

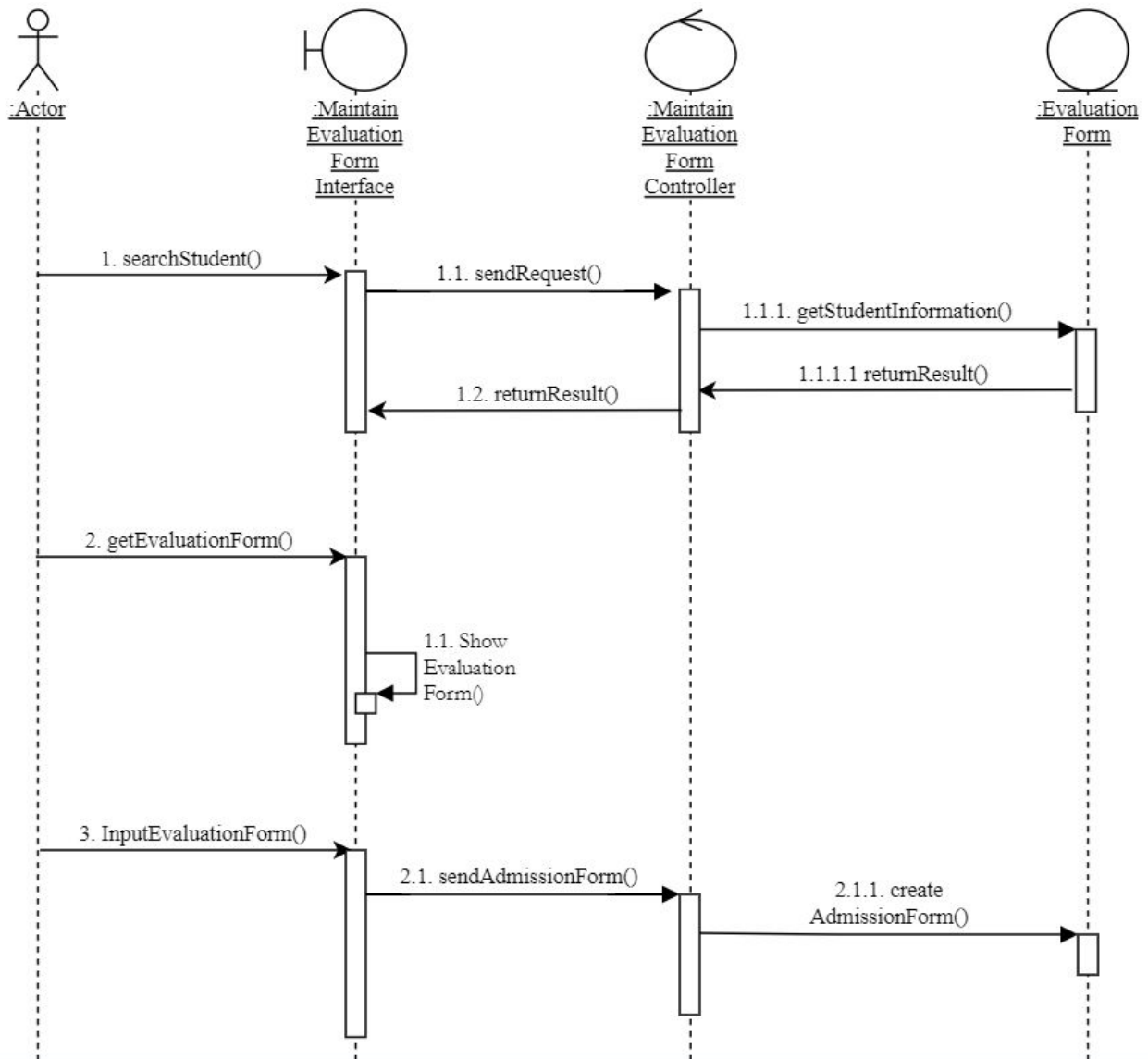


Figure 13. Evaluate Student Sequence diagram

4.10 Use Questions Bank

This use case describes how a Lecturer searches a question in Question Bank.

The user is a lecturer.

When the user logged into the system, the system gets the Jury of the actor and checks if this Jury is available. If it is, the system lists all the students in this Jury. After the user type question Id of the target Question and chooses “Search”, the system gets the question with the entered Id from the Question Bank in the Database displays Question and Answer (figure 24 in section VII: Appendix).

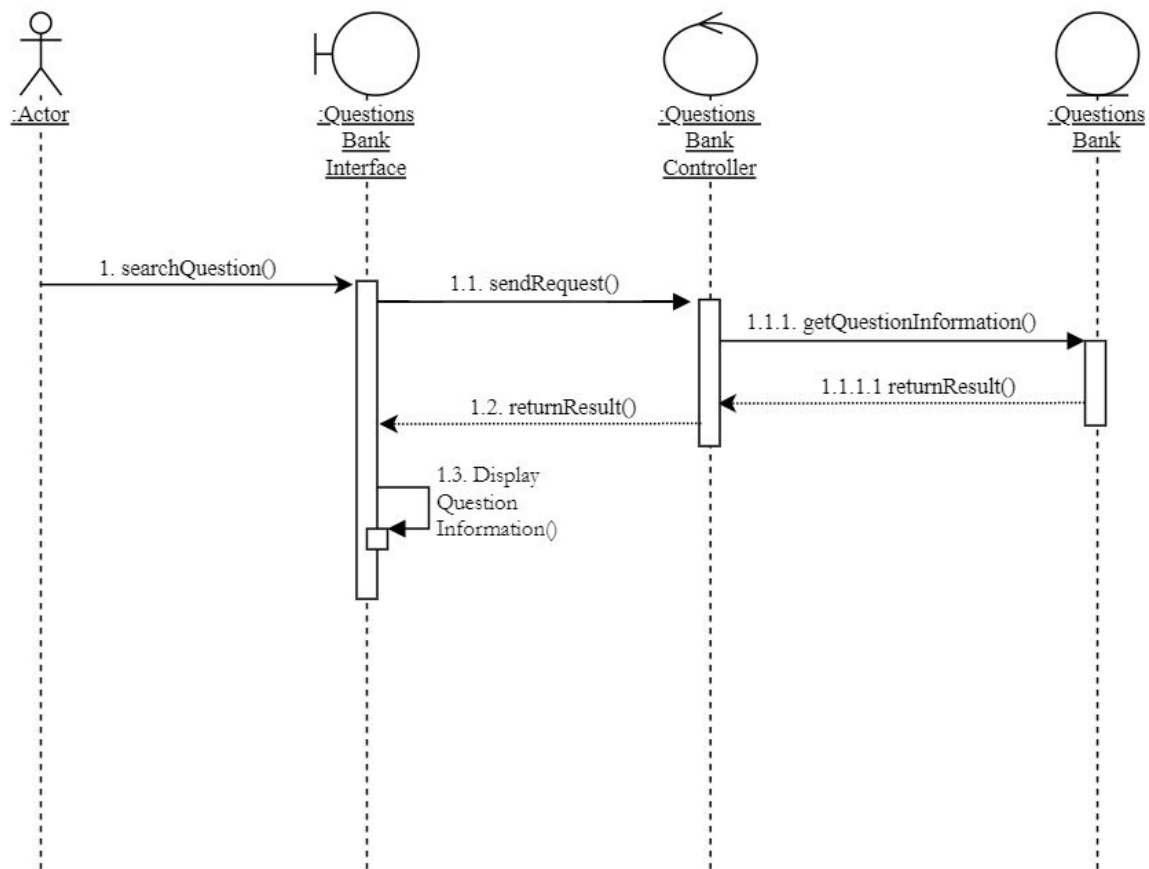


Figure 14. Use Question Bank Sequence diagram

V. RESULT AND DISCUSSION

1. Result

In this system, these main functions have implemented completely:

- **Register and Login** : users can Create a new account and Access into the system.
- **Maintain Admission Form:**
 - **Send Form, Edit Form:** Students can add and update their Admission Form (only allow update when their Admission Form is not yet approved)
 - **Export Form:** Students can export and download their Admission Form as PDF file
 - **Check Form Status:** Students can keep track with their Admission Form status (pending, rejected or approved).
 - **Upload Files:** Students can upload PDF attachments (such as Motivation Letter, Adwads, ...)
- **Follow Admission Wave:** Students can follow Admission Waves information (status, data start, date end, ...)
- **Interview:**
 - **View Student Information:** Lecturer can search Student by name and view Student Information and their attachment files
 - **Evaluate Student:**
 - English teacher can get and submit English Evaluation Form
 - Lecturer can get and submit General Evaluation Form
- **Use Questions Bank:** Lecturer can search questions by Id in Question Banks.

2. Discussion

Despite being implemented completely, the main functions in this project have some existing problems:

- In the function Managing Admission Form, each student can only submit one admission form in a wave. Up to now, we distinguish among students by their user ID. But students can register another account to submit the form.
- The attachments of students are displayed by iframe which brings security risks.
- The Admission Wave has not automatically generated an unique code for each candidate yet.

VI. CONCLUSION AND FUTURE WORKS

1. Conclusion

In conclusion, the purposes and objectives of the Online Admission Application and Interview System is achieved. By providing a user-friendly interface on the web, the interaction between student, lecturer and administration department becomes more convenient. Managing admission data becomes simpler and more effective.

The main functions of the Online Admission Application and Interview System have already been implemented:

- User can register and log into the system
- Students can submit and update Admission Form, keep track of their application process and follow Admission Wave information.
- Lecturer can search and view students application information, submit Evaluation form and get questions from Question Bank

2. Future Work

To further improve this web application in the future, the following tasks need to be done:

- Improve website security.
- Create an online support system allowing students to connect with staff.
- Create a Guideline for using the system.
- Distinguish among students by their ID number from the Identification card instead of their user ID.
- Improve the performance of the application to be more responsive and user-friendly.

VII. APPENDIX

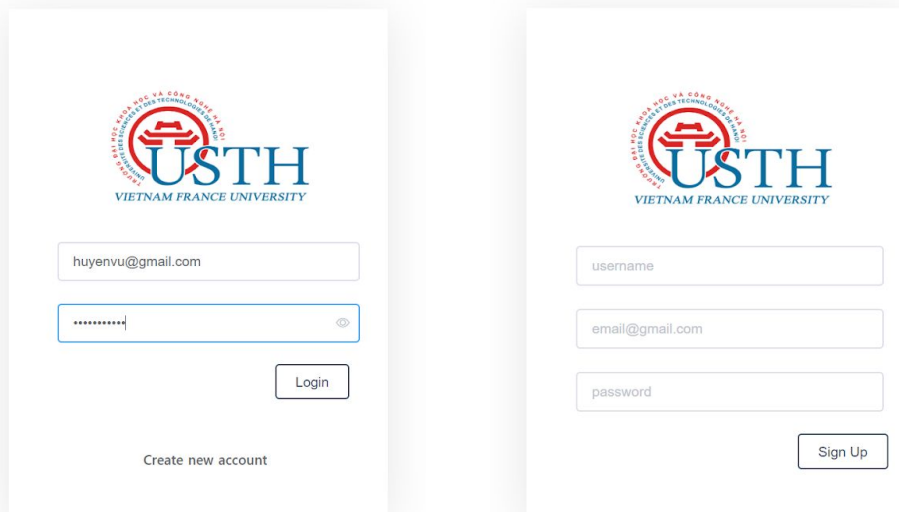
University of Science and Technology of Hanoi

Are you a Student or a Lecturer?

Student

Lecturer

Figure 15. Welcome Page user interface



The figure displays two side-by-side user interface forms for the University of Science and Technology of Hanoi (USTH). Both forms feature the USTH logo at the top, which includes a red circular emblem with a stylized 'U' and 'S' and the text 'USTH VIETNAM FRANCE UNIVERSITY'.

The left form is the Login interface. It contains two input fields: the first is labeled with the email 'huyenvu@gmail.com' and the second is a password field with a masked input (dots) and a toggle icon. Below these fields is a 'Login' button. At the bottom, there is a link that says 'Create new account'.

The right form is the Register interface. It contains three input fields: the first is labeled 'username', the second is labeled 'email@gmail.com', and the third is labeled 'password'. Below these fields is a 'Sign Up' button.

Figure 16. Login and Register user interface

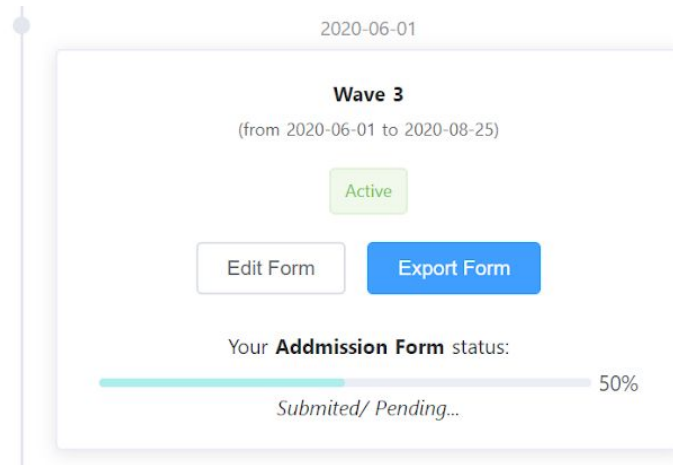


Figure 17. Admission Form status user interface

Admission Wave 2020

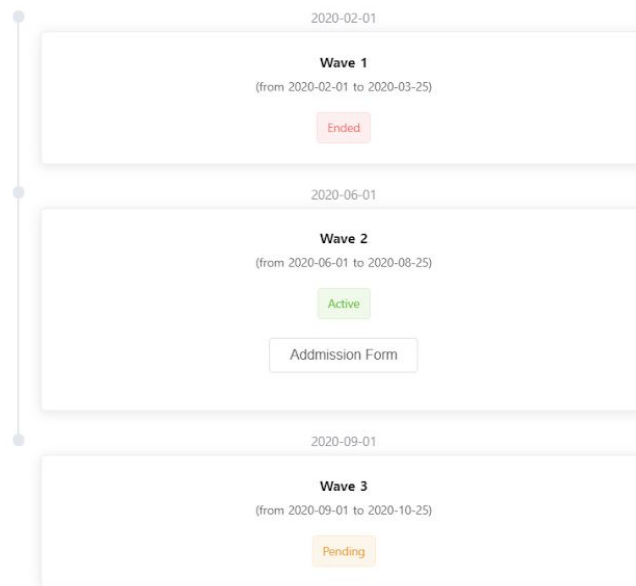


Figure 18. Admission Wave information user interface



ĐƠN ĐĂNG KÝ DỰ TUYỂN HỆ CỬ NHÂN NĂM 2020 APPLICATION FORM FOR BACHELOR COURSES INTAKE 2020

1. Họ và tên/ Full name:

Doan Lien Huong

2. Số chứng minh thư nhân dân/ ID number:

4

Ngày cấp:

2010-06-01

Nơi cấp:

Hanoi

3. Ngày sinh/ Date of birth:

1999-11-12

4. Giới tính/ Gender

☐ Nam/ Male ☒ Nữ/ Female

5. Nơi sinh/ Place of birth:

Hanoi

6. Quốc tịch / Nationality:

Vietnamese

Lưu ý/ Note:

(*) Đối với các ngành không đủ số lượng sinh viên theo điều kiện mở ngành thì sinh viên sẽ được chuyển nguyện vọng sang các ngành theo thứ tự đã đăng ký (trừ ngành Kỹ thuật Hàng không). For program which does not recruit enough students as the regulation, student will be transferred to other program in the order of registration (except for Aeronautical maintenance and Engineering program).

(**) Do đặc thù của chương trình đào tạo nên thí sinh có nguyện vọng học ngành Kỹ thuật Hàng không thì thứ tự ưu tiên số 1 bắt buộc phải là ngành Kỹ thuật hàng không. Students who apply for Aeronautical maintenance and Engineering program must select Aeronautical Engineering program as the 1 st priority.

Next

Figure 19. Admission Form user interface

26. Các thành tích nổi bật (học bổng, giải thưởng ... không bắt buộc)/ Awards and Distinctions(optional):

Choose File No file chosen

File có tên dạng: adward.pdf/ File with format: adward.pdf

27. Thư trình bày mục đích, nguyện vọng/ Motivation letter:

(Trình bày bằng tiếng Anh hoặc Tiếng Việt (khoảng 300 từ) nêu rõ mục đích và lý do đăng ký dự tuyển vào Trường ĐHKHCNHN, mục tiêu nghề nghiệp và lý do chọn ngành, v.v... /Please explain within 300 words in English or Vietnamese the purpose of your application and reasons for applying to USTH, detailing your career objectives and the reason for which you choose the above-mentioned specialty, etc.)

Choose File No file chosen

File có tên dạng: motivation.pdf/ File with format: motivation.pdf

Submit

Figure 20. Upload Files user interface

← Admission Form

THÔNG TIN CÁ NHÂN / PERSONAL INFORMATION

1. Họ và tên/ Full name: Đoàn Lâm Hoàng

2. Số chứng minh thư nhân dân/ ID number: 4
Ngày cấp: 2010-06-01
Nơi cấp: Hanoi

3. Ngày sinh/ Date of birth: 1999-11-12

4. Giới tính/ Gender: female

5. Nơi sinh/ Place of birth: Hanoi

6. Quốc tịch / Nationality: Vietnamese

7. Dân tộc/ Ethnicity: Kinh

8. Địa chỉ gửi thư/ Current mailing address for correspondence: Hanoi

9. Di động/Mobile phone: 0903849347

10. E-mail:

11. Điện thoại/ Home phone: 0437584788

12. Số di động của bố, mẹ/ Parent's mobile No: 0934753847

13. Tình trạng hôn nhân/ Marital status: single

14. Hộ khẩu thường trú/ Permanent Resident Address:

Số nhà/ House No: 21

Đường/ Street: Ngạc Lam

Phường/xã/ Ward, commune: Ngạc Lam

Quận, huyện/ District: Long Biên

Tỉnh, thành phố/ Province: Hà Nội

QUÁ TRÌNH HỌC TẬP BẮC TRUNG HỌC PHỔ THÔNG / ACADEMIC HISTORY

15. Tên trường/ School name: Viet Duc

Trưởng Đại học Khoa học và Công nghệ Hà Nội
University of Science and Technology of Hanoi

ĐƠN ĐĂNG KÝ DỰ TUYỂN HỆ CỬ NHÂN NĂM 2020
APPLICATION FORM FOR BACHELOR COURSES INTAKE 2020

Print

DestinationSave as PDF

PagesAll

LayoutPortrait

More settings

SaveCancel

Figure 21. Export Admission Form user interface

Trường Đại học Khoa học và Công nghệ Hà Nội

University of Science and Technology of Hanoi

Jury 5 Wave 3 Year 2020

ID	Full Name	High School	Training Program	Date of birth	Gender	<input type="text" value="huyen"/>
2	Vu Khanh Huyen	Nguyen Gia Thieu	ICT	1999-08-09	female	Detail Evaluate



Trường Đại học Khoa học và Công nghệ Hà
Nội
University of Science and Technology of
Hanoi

ĐƠN ĐĂNG KÝ DỰ TUYỂN HỆ CỬ NHÂN NĂM 2020

APPLICATION FORM FOR BACHELOR COURSES INTAKE

2020

THÔNG TIN CÁ NHÂN/ PERSONAL INFORMATION

- Họ và tên/ Full name: Vu Khanh Huyen
- Số chứng minh thư nhân dân/ ID number: 2
Ngày cấp: 2010-06-01
Nơi cấp: Hanoi
- Ngày sinh/ Date of birth: 1999-08-09
- Giới tính/ Gender: female
- Nơi sinh/ Place of birth: Hanoi

Motivation Letter

MOTIVATION 2

"What exactly does 'tamed' mean?"

"Well, it's something too often forgotten," said the fox. "I suppose it means: to make some kind of relationship."

"Relationship?"

"Yes," said the fox. "I'll explain. To me, you are just a little boy like any other, like a hundred thousand other little boys. I have no need of you and you have no need of me. To you I am a fox like any other, like a hundred thousand other foxes. But if you tame me, you and I, we will have created a relationship, and so we will need one another. You will be unique in the world for me... If you were to tame me, my whole life would be so much more fun. I would come to know the sound of your footsteps, and it would be different from all the others. At the sound of any other footsteps I would be down in my hole in the earth as quick as you like. But your footsteps would be like music to my ears, and I would come running up out of my hole, quick as you like."

Figure 22. View Student information user interface

Trường Đại học Khoa học và Công nghệ Hà Nội

University of Science and Technology of Hanoi

Jury 5 Wave 3 Year 2020

ID	Full Name	High School	Training Program	Date of birth	Gender	
2	Vu Khanh Huyen	Nguyen Gia Thieu	ICT	1999-08-09	female	<input type="text" value="huyen"/>

[Detail](#)

[Evaluate](#)



Trường Đại học Khoa học và Công nghệ Hà
Nội
University of Science and Technology of
Hanoi

General Evaluation

Bachelor Admission Interview-2020

CRITERIA	Max score	Final score	SCORE SCALE
A. Academic transcripts	30	<input type="text"/>	(find detailed assessment in the attachment)
B. Direct academic assessment			
	No.1 12	<input type="text"/>	
1. Standard Question	No.2 12	<input type="text"/>	Right andswer: 12 points Wrong andswer but Right method: 09 points Wrong andswer: 0 points

[Submit](#)

GPA	Equal Score
6.5-6.9	15 points
7.0-7.9	20 points
8.0-8.9	25 points
>9.0	30 points

Date: 2020-6-30
Student ID: 2
Lecturer ID: 1

Figure 23. Evaluate Student user interface

Please enter the question ID

Search

Questions Bank

Question: Why does the little prince leave his planet?

A. He wants to visit Earth

B. He begins to doubt the rose sincerity

C. A sheep eats his rose

D. Baobabs eats his planet

answer: B

Figure 24. Questions Bank user interface