

HO CHI MINH UNIVERSITY OF TECHNOLOGY AND EDUCATION
FACULTY OF INTERNATIONAL EDUCATION



IT PROJECT

BUILD A WEBSITE TO MANAGE PROJECT OF
IT FACULTY

Lecturer name: PhD. Tran Nhat Quang

Subject ID: PROJ215879E

Class: PROJ215879E_24_1_10FIE

List of members

Student ID	Student name
19110152	Huỳnh Gia Kiện

Ho Chi Minh City, 11/2024

[illegible]

[illegible]

(Sign, write full name)

Trần Nhật Quang

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my sincerest gratitude to my instructor, PhD Tran Nhat Quang, for his invaluable support and guidance throughout this course. His unwavering dedication to helping me succeed in my final project has been nothing short of inspiring, and I am incredibly grateful for the opportunity to learn from such a knowledgeable and experienced developer.

Furthermore, I would like to extend my heartfelt thanks to my classmates, whose contributions have been instrumental in helping me develop a strong and well-informed thesis. Their insights and feedback have been invaluable, and I am grateful for the knowledge and expertise they have shared with me.

Despite the challenges I faced, I managed to complete the subject and report in a short amount of time, with limited resources and varying levels of expertise in programming implementation. As a result, I acknowledge that there may be flaws in my work, and I welcome any constructive criticism or suggestions for improvement.

In conclusion, I would like to express my appreciation to everyone who has supported me on this journey. This final project would not have been possible without the collective effort of my instructor, classmates, and mentors, and I am truly grateful for the opportunity to learn and grow in this field. Thank you all for your support and encouragement. I welcome any feedback as well as suggestions for improving my project. I deeply appreciate it. Sincerely!

Table of Contents

CHAPTER 1: PROJECT SPECIFICATION.....	1
1.1. PURPOSE.....	1
1.2. INPUT DATA AND INFORMATION	1
1.3. USE CASES	1
1.4. EXPECTED INTERFACE.....	2
CHAPTER 2: WORK DIVISION.....	3
CHAPTER 3: SYSTEM DESIGN.....	4
3.1. USECASE DIAGRAM.....	4
3.2. FILE STRUCTURES	4
3.2.1. File structures	4
3.2.2. Method applied.....	7
3.3. DATABASE DESIGN	10
3.3.1. Database design diagram.....	10
3.3.2. Tables in the database.....	10
3.4. DESCRIBING FIELDS IN TABLES	10
3.4.1. User table.....	10
3.4.2. Topic table	11
3.5. USER INTERFACE DESIGN	12
CHAPTER 4: TESTING.....	16
CHAPTER 5: CONCLUSION	20
5.1. ACHIEVEMENTS.....	20
5.2. STRENGTHS AND DRAWBACKS	20
5.3. REFLECTION	20
5.4. FUTURE DEVELOPMENTS.....	21
CHAPTER 6: REFERENCES	22

Table of figure

Figure 1. Usecase Diagram.....	4
Figure 2. Front-end File Structures	5
Figure 3. Back-end File Structures.....	6
Figure 4. ERD Diagram.....	10

List of tables

Table 1. Usecase Table.....	1
Table 2. Work Division Table	3
Table 3. Method Applied Table.....	7
Table 4. User Database Table.....	11
Table 5. Topic Database Table	11
Table 6. User Interface Design Table.....	12
Table 7. Testing Table	16

Chapter 1: Project Specification

1.1. Purpose

Create a website with all the functions of a project management website that is convenient for users. Website ensures stability and security for users.

The software will facilitate:

- Providing an accessible and secure platform for all users to interact with project topic efficiently.

Usage Context:

- Students: Register project topics, view project instructor and reviewer.
- Lecturers: Register to supervise specific topics
- Deans: Approve project topics, assign reviewers and register to supervise specific topics.
- Admins: Manage users, topics.

1.2. Input data and information

- Login credentials (username, password).
- Students's personal information
- Lecturers's personal information
- Deans's personal information
- Topics's detail

1.3. Use cases

Table 1. Usecase Table

No.	User Role	Purpose	Features
1	Student	Topic registration	<ul style="list-style-type: none">• Sign in/out• Register topic• Cancel topic registration

2	Lecturer	Oversee project topics	<ul style="list-style-type: none">• Sign in/out• View topics• Register to supervise
3	Dean	Oversee project topics and approve topics	<ul style="list-style-type: none">• Sign in/out• Register to supervise• Assign reviewers• Approve topic
4	Admin	Manage system operations	<ul style="list-style-type: none">• Sign in/out• Manage user accounts, project topics

1.4. Expected interface

Login Page: Fields for username and password.

Student Dashboard: View list of available project topics, option to register a topic, section to view registered project topics and option to cancel project registration.

Instructor Dashboard: List of assigned and available project topics, functionality to register as a supervisor.

Dean Dashboard: List of pending project topics for approval, options to approve/reject topics, functionality to register as a supervisor and assign reviewers to specific projects.

Admin Dashboard: Management panels for users and project topics, options for adding, editing, or deleting accounts and topics.

Chapter 2: Work Division

Table 2. Work Division Table

Student name	Tasks	Contribution percentage
Huỳnh Gia Kiện	<ul style="list-style-type: none">• Back-end developer• Front-end developer• Database design• Write Report	100%

Chapter 3: System Design

3.1. Usecase Diagram

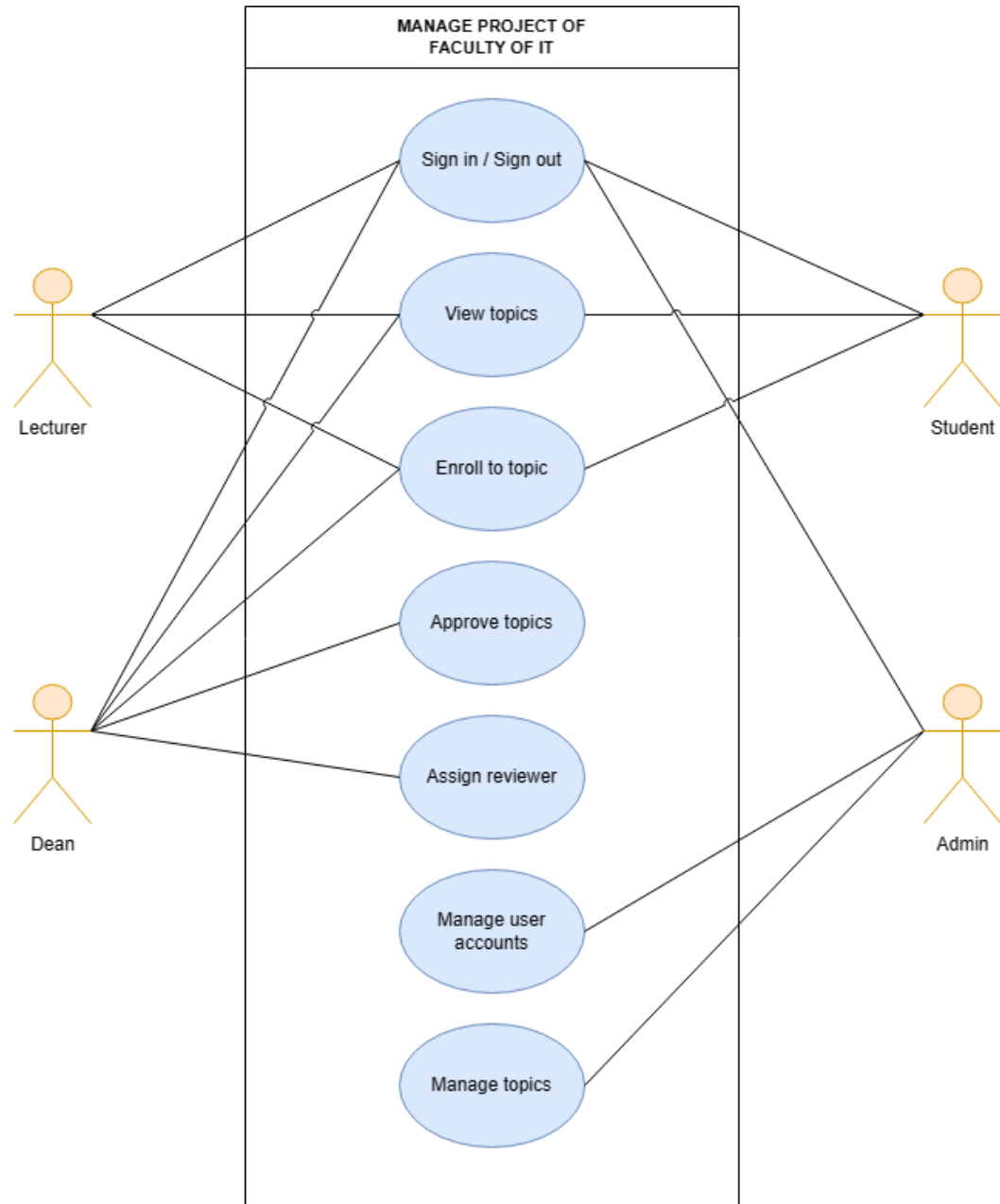


Figure 1. Usecase Diagram

3.2. File Structures

3.2.1. File structures

3.2.1.1. Front-end

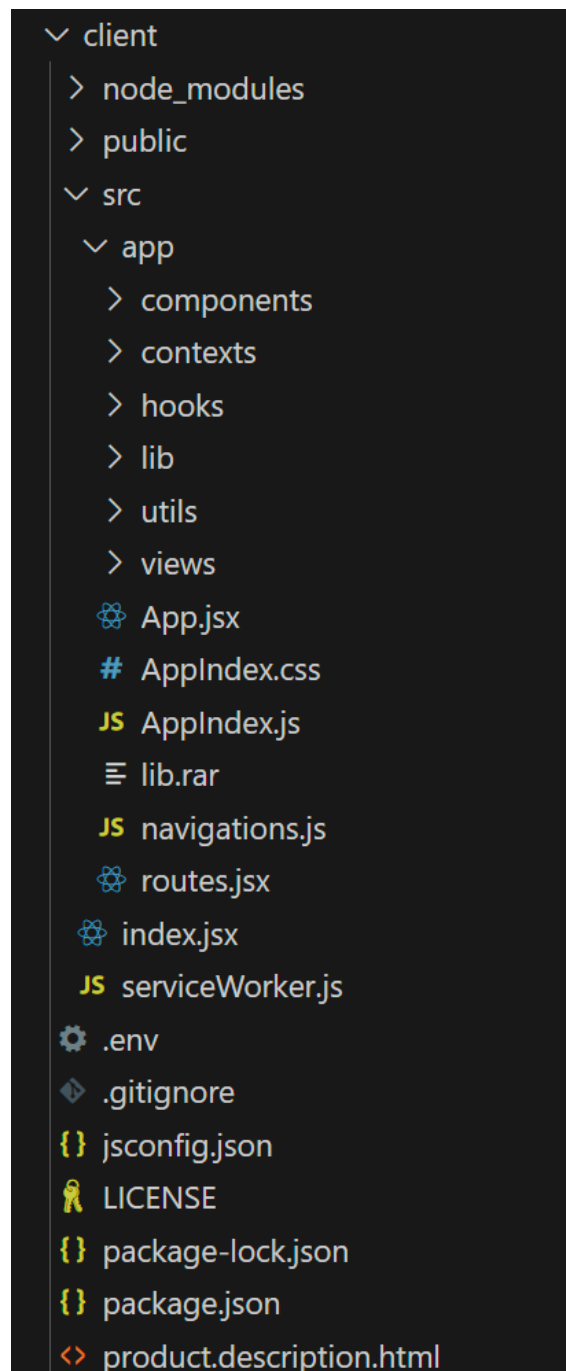


Figure 2. Front-end File Structures

- **public:** This folder holds static assets like images, CSS files, and other resources that are directly served to the browser.
- **src/app:** This directory houses the core source code of my frontend application. It is further divided into subdirectories for better organization:

- **components:** This folder contains reusable UI components that make up your application's interface.
- **contexts:** This folder contain context setting file
- **hooks:** This folder contain use setting file
- **lib:** This folder contains utility functions that are used throughout my application, such as helper functions for formatting data, making API requests, or handling errors.
- **utils:** This folder contains functions that are used to check device and time
- **views:** This folder contains files that represent individual pages or screens of my application. Each page typically consists of a combination of components.

3.2.1.2. Back-end

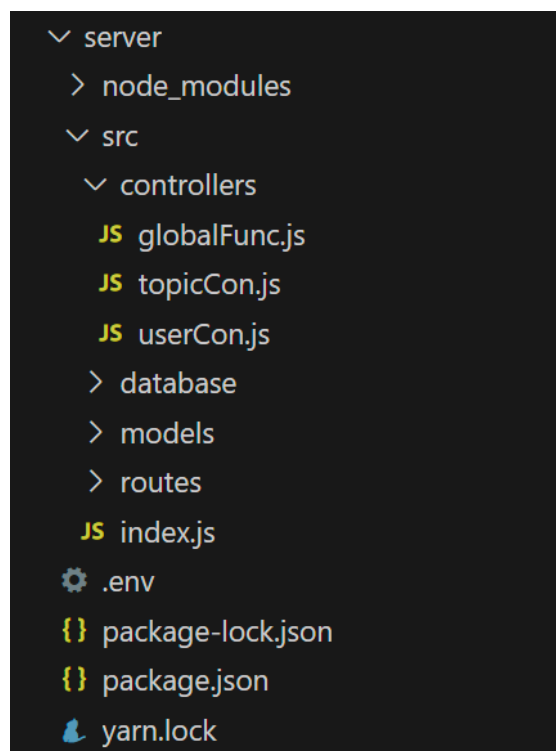


Figure 3. Back-end File Structures

- **src:** This directory contains the core source code of your application:

- **controllers:** This folder holds logic functions that handle specific tasks, such as processing incoming requests, manipulating data, and generating responses.
- **database:** This folder contains configuration files for my database connection, including database credentials, connection strings, and other settings.
- **models:** This folder defines the structure of my data using database schemas. These schemas represent the data entities that my application will store and manage.
- **routes:** This folder maps API endpoints to specific controller functions. It defines the URLs that my application can handle and the corresponding actions to be executed.
- **index.js:** setup server setting

3.2.2. Method applied

Student in charge: Huỳnh Gia Kiện

Table 3. Method Applied Table

No.	Method	Purpose	File name, line number
1	EmptyCheck(request) Input: request Output: isValid (boolean)	To validate that no keys in the request.body object have empty string values	GlobalFunc.js (2)
2	register(req, res) Input: req Output: response (success or fail) and a content message	To register a new user	userCon.js (6)
3	login(req, res) Input: req	To authenticate and log in a user	userCon.js (43)

	Output: response (success or fail) and a content message		
4	getAll(req, res) Input: req Output: response (success) and user list	To retrieve all users or filter by role	userCon.js (76)
5	logout(req, res) Input: req Output: response (success or fail) and a content message	To log out a user	userCon.js (90)
6	update(req, res) Input: req Output: response (success or fail) and a content message	To update user details	userCon.js (108)
7	checkAuthen(req, res) Input: req Output: response (success or fail) and user information	To check if a user is authenticated	userCon.js (135)
8	getDetail(req, res) Input: req Output: response (success or fail) and user details	To get detailed user information	userCon.js (146)
9	delete(req, res) Input: req Output: response (success or fail) and a content message	To delete a user account	userCon.js (159)
10	create (req, res) Input: req	Retrieve all topics	topicCon.js (5)

	Output: response with all topics or error		
11	getAll (res) Input: none Output: response (success or fail) and user information	To check if a user is authenticated	topicCon.js (38)
12	update (req, res) Input: req Output: response (success or fail)	Update details of a topic	topicCon.js (56)
13	enroll (req, res) Input: req Output: response (success or fail)	Enroll a student into a topic	topicCon.js (83)
14	disEnroll (req, res) Input: req Output: response (success or fail)	Remove a student from a topic	topicCon.js (124)
15	getDetail (req, res) Input: req Output: response response with topic details or error	Retrieve details of a specific topic	topicCon.js (149)
16	delete (req, res) Input: req Output: response (success or fail)	Delete a specific topic	topicCon.js (162)
17	approve (req, res) Input: req Output: response (success or fail)	Approve a topic	topicCon.js (173)

18	checkAuthen(req, res) Input: req Output: response (success or fail)	Assign an instructor or reviewer to a topic	topicCon.js (184)
----	---	---	----------------------

3.3. Database Design

3.3.1. Database design diagram

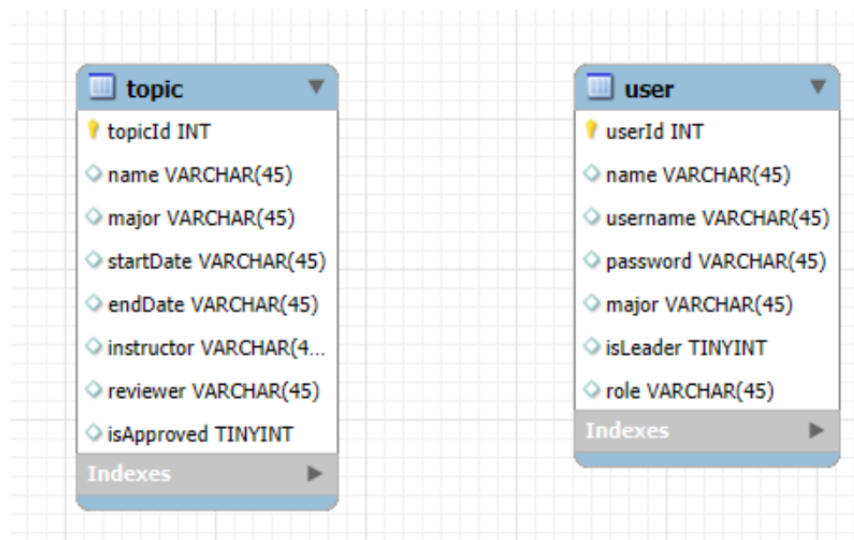


Figure 4. ERD Diagram

3.3.2. Tables in the database

Table 4. Tables In The Database Table

No.	Table Name	Purpose
1	User	Store user account information
2	Topic	Store topic information

3.4. Describing fields in tables

3.4.1. User table

Table 5. User Database Table

No.	Field Name	Data Type	Purpose
1	name	String	Store user's name
2	username	String	Store username
3	password	String	Store password
4	major	String	Store user's major
5	isLeader	Boolean	Clarify user is faculty Dean or Lecturer
6	role	String	Store user's role


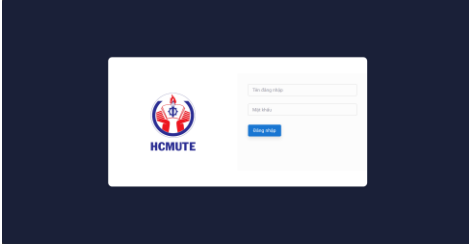
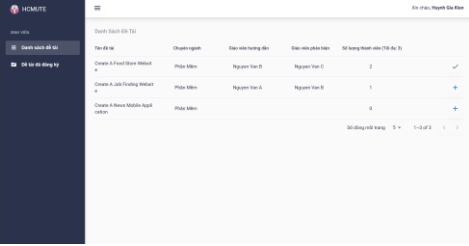
3.4.2. Topic table

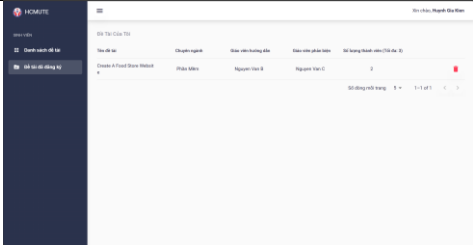
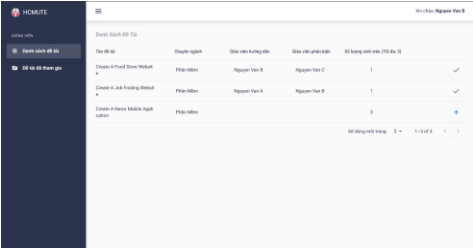
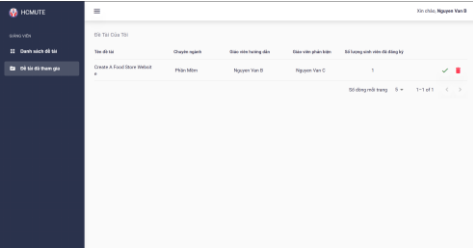
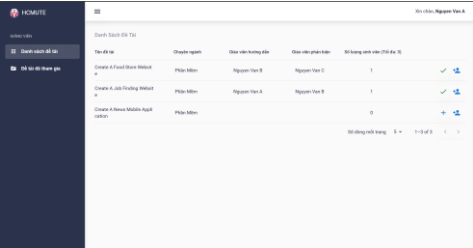
Table 6. Topic Database Table

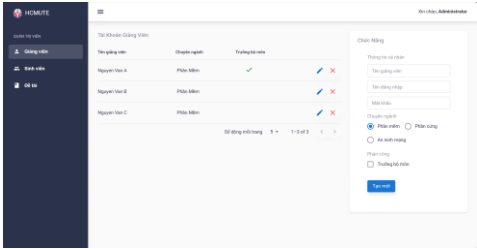
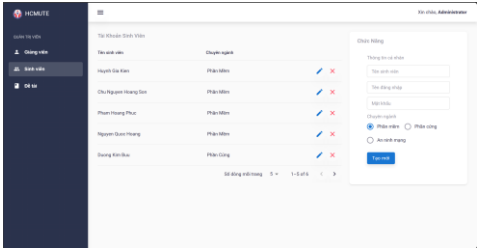
No.	Field Name	Data Type	Purpose
1	name	String	Store user's name
2	major	String	Store topic's major
3	startDate	String	Store topic's start date
4	endDate	String	Store topic's end date
5	instructor	String	Store topic's instructor
6	reviewer	String	Store topic's reviewer
7	isApproved	Boolean	Clarify topic is approved or not
8	students	Object	Store student list of the topic

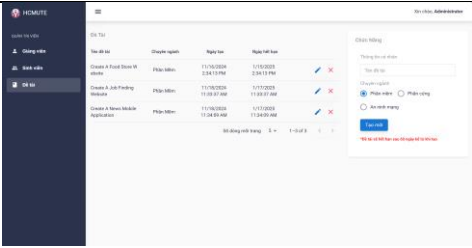
3.5. User Interface design

Table 7. User Interface Design Table

No.	UI name	Purpose	Explain
1	<p>Homepage</p> 	Show information about the website	For user to have an overview about the website
2	<p>Login Page</p> 	Authenticate users and grant them access to personalized or restricted content.	Protect sensitive information, maintain user privacy, and allow access to specific features based on user roles
3	<p>Student's Topic List Page</p> 	Show topic lists based on student's major, show topic's informations and button to register to the topic	Make sure student can view topic's informations and register to the topic
4	<p>Student's Registered Topic Page</p>	Show student's registered topic	Make sure student can view registered topic's informations

		and button to cancel registration	and cancel registration
5	<p>Lecturer's Topic List Page</p> 	Show topic lists based on lecturer's major, show topic's informations and button to register to supervise	Make sure the lecturer can view topic's informations and register to supervise
6	<p>Lecturer's Registered Topic Page</p> 	Show lecturer's registered topics and button to cancel registration	Make sure lecturer can view registered topic's informations and cancel registration
7	<p>Dean's Topic List Page</p> 	Show topic lists based on dean's major Show topic's informations and button to register to supervise, button to assign instructor to the topic	Make sure the dean can view topic's informations, register to supervise and assign instructor

8	<p>Admin Manage Lecturer Page</p> 	<p>Show lecturer's list and their information</p> <p>Show button to edit lecturer information or delete lecturer account</p> <p>Show function to create new lecturer account and assign faculty dean</p>	<p>Make sure admin can manage lecturer's user accounts</p>
9	<p>Admin Manage Student Page</p> 	<p>Show student's list and their information</p> <p>Show button to edit student information or delete student account</p> <p>Show function to create new student account</p>	<p>Make sure admin can manage student's user accounts</p>
10	<p>Admin Manage Topic Page</p>	<p>Show topic's list and their information</p>	<p>Make sure admin can manage topics</p>

	<p>Show button to edit topic</p> <p>information or delete topic</p> <p>Show function to create new topic</p>	
---	--	--

Chapter 4: Testing

Table 8. Testing Table

No.	Test case	Purpose	Explain
1	Login with a valid account Input: account (kien1234, 123456) Expected result: Login successfully Actual result: Login successfully Status: pass	Check that user can login with a valid account	Use an existed account to check
2	Login with an invalid account Input: account (kien12345, 123456) Expected result: Login failed Actual result: Login failed Status: pass	Check that user can not login with an invalid account	Use an non-existed account to check
3	Register to a valid project topic Input: none Expected result: Register successfully Actual result: Register successfully Status: pass	Check that user can register to a valid project topic	A valid project topic is a topic that not full (3/3)

4	<p>Register to an invalid project topic</p> <p>Input: none</p> <p>Expected result: Register failed</p> <p>Actual result: Register failed</p> <p>Status: pass</p>	<p>Check that students can not register to an invalid project topic</p>	<p>An invalid project topic is a topic that full (3/3)</p>
5	<p>Register to a project topic when student already registered to another topic</p> <p>Input: none</p> <p>Expected result: Register failed</p> <p>Actual result: Register failed</p> <p>Status: pass</p>	<p>Check that students can not register to a project topic when they already registered to another topic</p>	<p>Make sure that student can only register to 1 topic at a time</p>
6	<p>Cancel registration of a project topic</p> <p>Input: none</p> <p>Expected result: Cancel successfully</p> <p>Actual result: Cancel successfully</p> <p>Status: pass</p>	<p>Check that users (student, lecturer, dean) can cancel registration of a topic</p>	<p>Make sure users (student, lecturer, dean) can cancel registration</p>
7	<p>Register to supervise a valid project topic</p> <p>Input: none</p>	<p>Check that users (lecturer, dean) can register to</p>	<p>A valid topic is a topic that no one is supervising</p>

	Expected result: Register successfully Actual result: Register successfully Status: pass	supervise a valid topic	
8	Register to supervise an invalid project topic Input: none Expected result: Register failed Actual result: Register failed Status: pass	Check that users (lecturer, dean) can not register to supervise an invalid topic	An invalid topic is a topic that another user (lecturer, dean) is supervising
9	Assign reviewer to a project topic Input: none Expected result: Assign successfully Actual result: Assign successfully Status: pass	Check that deans can assign reviewer to a project topic	Make sure that dean can assign reviewer to a project topic
10	Approve/Reject a project topic Input: none Expected result: Approve/Reject successfully Actual result: Approve/Reject successfully Status: pass	Check that deans can approve/reject a project topic	Make sure that dean can approve/reject a project topic

11	<p>Manage user accounts</p> <p>Input: none</p> <p>Expected result: Add/Edit/Delete successfully</p> <p>Actual result: Add/Edit/Delete successfully</p> <p>Status: pass</p>	<p>Check that admin can manage (add, edit, delete) user accounts</p>	<p>Make sure that admin can manage user accounts</p>
12	<p>Manage topics</p> <p>Input: none</p> <p>Expected result: Add/Edit/Delete successfully</p> <p>Actual result: Add/Edit/Delete successfully</p> <p>Status: pass</p>	<p>Check that admin can manage (add, edit, delete) project topics</p>	<p>Make sure that admin can manage project topics</p>

Chapter 5: Conclusion

After spending a lot of hard work with enthusiasm, I have completed this project quite well. I could make all basic operations of a project management website work great in spite of some logical conflicts or struggling in handling complex operations. In addition I have also improved coding skills which can help me to work more smoothly in the future projects.

Through this project, I had lots of chances to learn new knowledge about MERN stack.

Besides the results that have been achieved, this website still has many things to overcome and improve with many development directions.

5.1. Achievements

- Know the theory of MongoDB, NodeJS, ReactJS, ExpressJS. Learn how to build front-end website and build databases and connect them.
- Project: Project management website with all functions that work well. Including four user types (administrator, student, lecturer, dean). Good interface and easy to use, simple and user-friendly.

5.2. Strengths and drawbacks

The website is designed with an easy-to-see and user-friendly interface. In particular, simple operations and functions make users feel comfortable while using the website.

On the other hand, there are still many shortcomings in the design which makes the website not look optimized in components arrangement. Moreover, the website is not so optimized and well-prepared. In terms of functions, due to the lack of expertise and experience, the handling logic and performance are not optimal.

5.3. Reflection

After completing this project, I feel how small I am in this huge technology world, there are enormous things to learn, to fulfill and to archive in the future. I could also understand more how hard it is to build just a basic website, which I thought would be quite easy in the past. Overall, this project has enlightened me about how big this IT world is and will be in the future.

5.4. Future developments

- Responsive website.
- Improve website performance.
- I will work to improve the website and add more features in the future.
- Enhance the user interface and experience.

Chapter 6: References

Nabed Khan. (2024). Uilibrary/matx-react: Matx -free and open-source react material UI admin dashboard template. GitHub. <https://github.com/uilibrary/matx-react>

Meta Platforms. (2024). React. React. <https://react.dev/learn>

OpenJS Foundation. (2024). Node.js v21.4.0 documentation. Node.js. <https://nodejs.org/docs/latest/api/>

Material UI SAS. (2024). Overview - Material UI. MUI: The React component library you always wanted. <https://mui.com/material-ui/getting-started/>