

Project Report

BU Course Community - A Web Application

Summer 2022
CS411 Software Engineering
Instructor: Prof.Ibrahim

Hao Chen
Lingwen Deng
Chengjie Gu
Qintian Huang
Zhe Tu

Problem Definition

Students are not able to acquire enough information about the course before the course starts. During the course registration, what students get is merely a general short description about the course on studentlink, and it is hard to tell whether the course meets the student's expectation. Students do not have enough ways to discuss and communicate with other students. A student's perspective of a course can be rarely known by other students. Exchanging ideas and comments among students is an effective way to learn useful information about the courses they want to take, but there is such platform to discuss or exchange information.

Project Objective

- Create a forum where students could get information and create posts within a few steps.
- Develop a clean and user-friendly UI
- The Project Development Team could easily manage the database

Stakeholders List

- Internal Stakeholder: Project Development Team
Project development team is in charge of the project development. The team has the responsibility to produce all the deliverables of this project and the further maintenance.
- External Stakeholder: BU Students
Students are the main users of this application. They are interested in both sharing experience of taking a specific class and viewing others' comments on the classes they are considering to take. The impact of this project to BU students is this forum provides a platform for students to communicate and exchange information with each other more easily. BU Course Community could potentially increase students' understanding of the course and have a better performance in class.
- External Stakeholder: BU Professors
Professors could be interested in this project because it provides a student perspective to look at the course. By using this application, professors could know about how students think about this course, for example, whether the pace is too fast or too slow, what part do students find most difficult to understand. Professors could potentially adjust how to deliver the course according to the comments made by students from this forum instantly. In this way, professors could have better interactions with students and the course could become more understandable.
- External Stakeholder: Boston University
Boston University could be interested in this forum because it helps the university to know how students think about the courses provided by BU. The students only fill in a course evaluation survey at the end of the semester, however, the information from the survey is limited to survey questions, and this forum provides a more diverse information about student's ideas of courses. BU could improve the course quality by looking at the comments, and in this way the whole education level would increase.

Success / Accept Criteria for each Stakeholder

- FOR internal stakeholders:

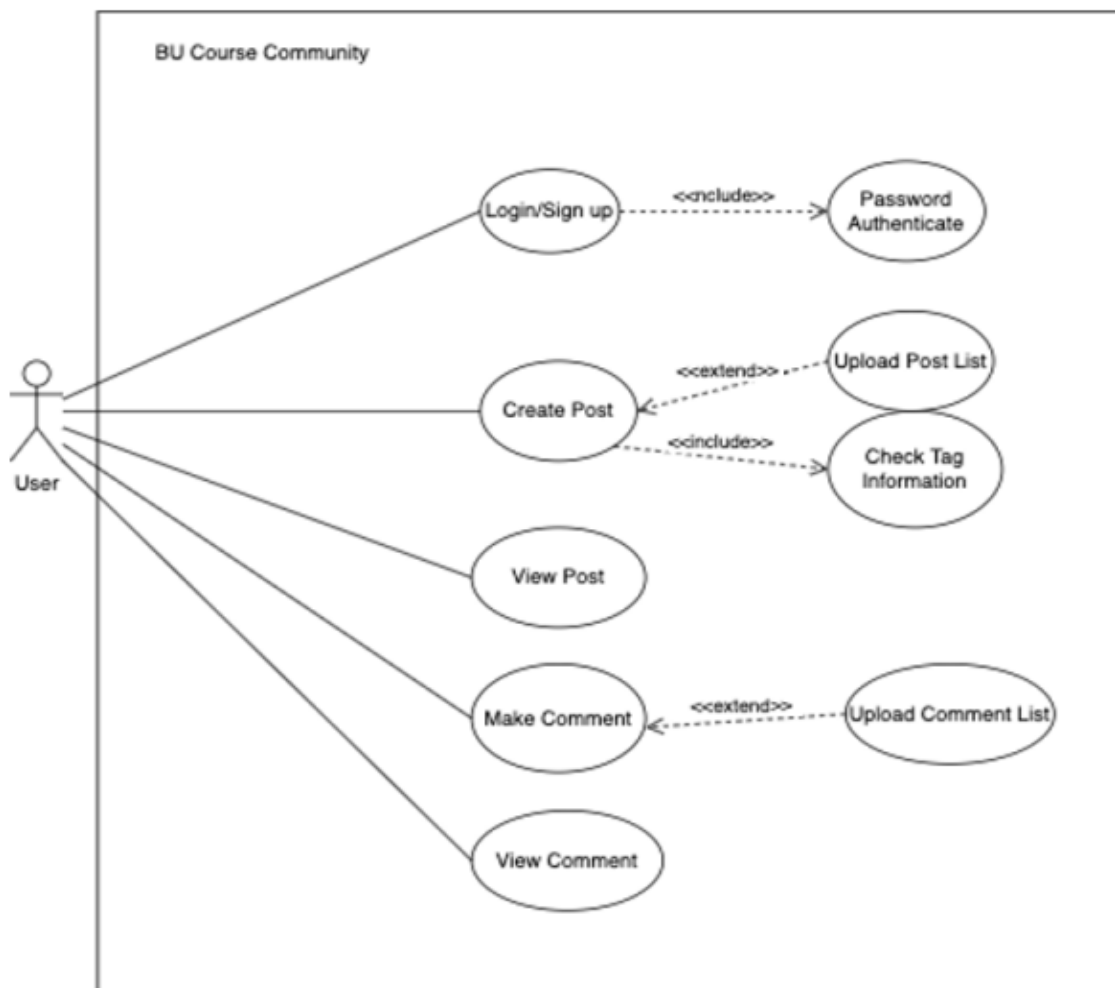
We consider the application to be shippable if the team provides an application fully satisfying the original functionality objectives. Furthermore, the window for further improvement and maintenance would remain open so that the application could be developed with more functions realized to meet further requirements.

- FOR external stakeholders:

We consider the application to meet the expectations of the university students if the application gives users an open and informative platform with easy access to and provides a unanimous comment function.

For other external stakeholders, we expect the application to meet their interest as long as it provides useful information for university staff to make adjustments and objective judgment to the course curriculum.

Use Case Diagram



Graph #1: Use Case Diagram

Selected use case description

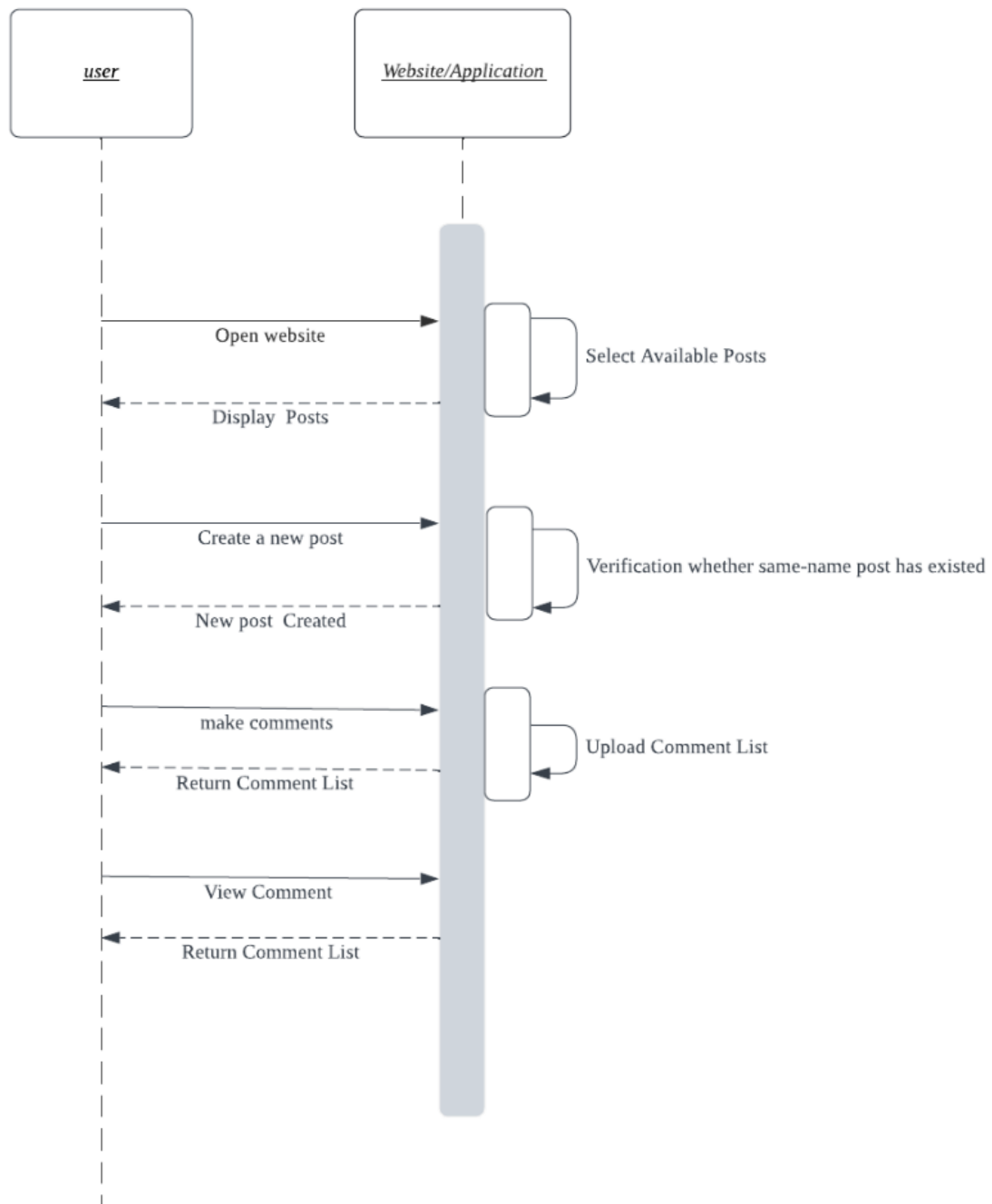
For the **Login/Sign up** case, the user will be able to access the system with its unique id and password. The system will authenticate the id and password so that the user could successfully login.

For the **Create Post** case, the system will first check the tag of the post user created. If the tag has already been created in the Database, the system will update the post into the corresponding list. If the tag does not exist in the Database, the system will create a new post list to store the creating post.

Use case	Login
Pre-condition	The user wants to log into his account
Post-condition	User successfully log in with the right password
Basic Path	<ol style="list-style-type: none">1. This use case starts with the user inputs an effective email as its username and a password for security.2. The application acknowledges the users' request, and verifies the user's identity.3. The application approves the user's request for login.
Alternative path	At step 2, the application recognizes the password as an incorrect one, and requests the user to give another password.

Use case	Create Post
Pre-condition	The user wants to create a post
Post-condition	The user successfully creates the post under the wanted tag
Basic Path	<ol style="list-style-type: none">1. The user logs into his account.2. The user opens the website for creating posts3. The user inputs tag under which he wants his post to be shown4. The user edit the text in his post, and gives his post a headline5. The application approves the user's request.6. The application displays the updated list of posts under the specific tag
Alternative path	At step 5, if the user's input tag is not existing yet, the application would create a new tag, and put the user's post into the newly-created tag.

Detailed Sequence Diagram



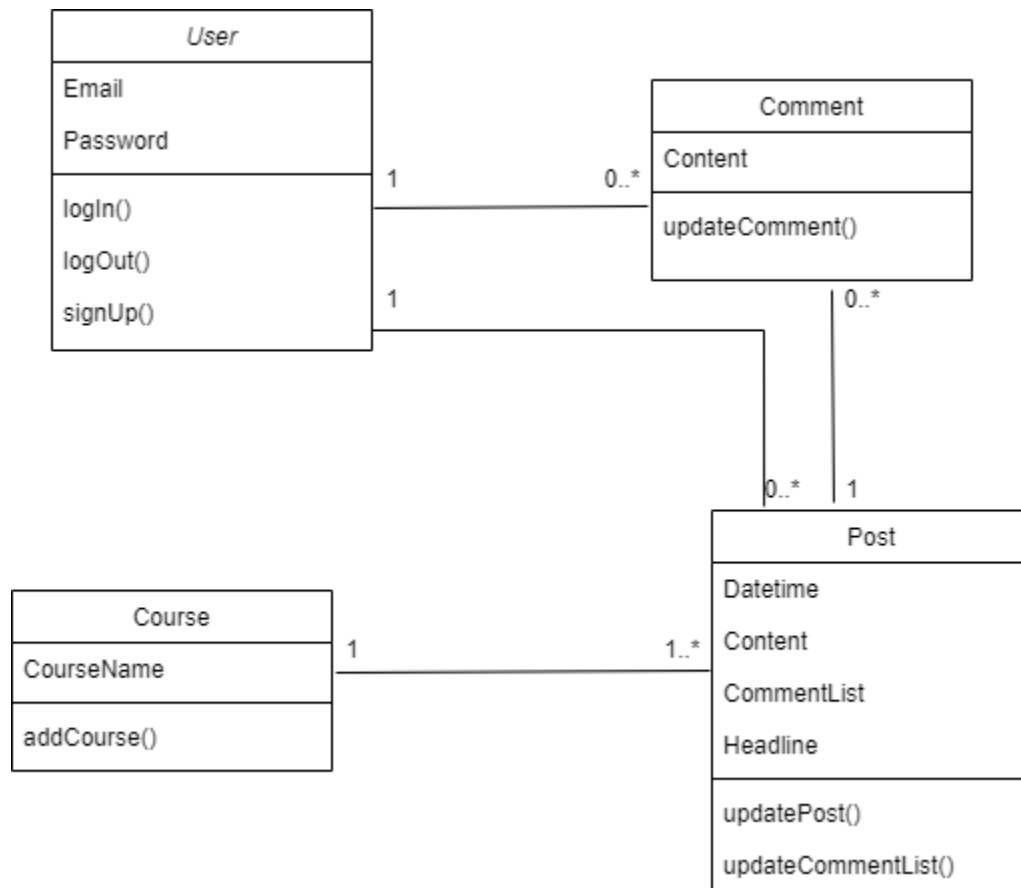
Graph #2: Detailed Sequence Diagram

System Architecture

The architecture pattern of this application is **client-server** architecture pattern. This architecture pattern is used because there are many clients that need to access this application at any time at any location. The architecture of having one server and many clients is suitable for this application. The server keeps listening to make sure the clients, which are users, can access the service provided at any time. The service provided by the server is the service of making posts, making the comments, view posts, etc.

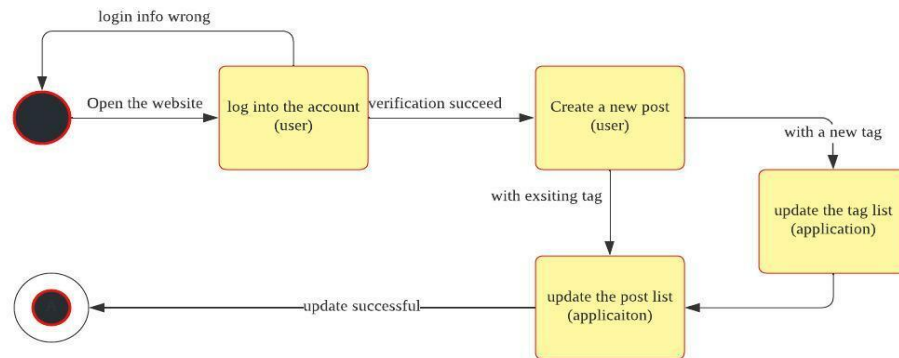
The architecture pattern of the server is the **model-view-controller** architecture pattern. The model part is in charge of the operation of the database. The view is in charge of how to show the webpage to the users. The controller is in charge of the logic behind, and uses the model and view when needed. Choosing this architecture pattern is because it is relatively loosely coupled between components, and this feature makes it easy to make modifications.

Detailed Class diagram



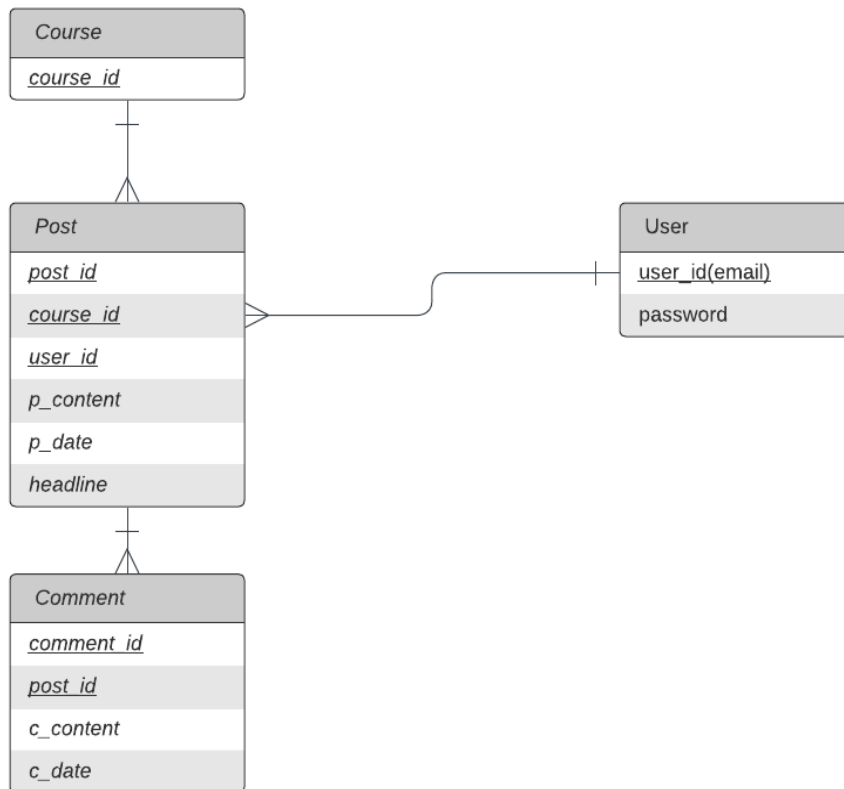
Graph #3: Detailed Class Diagram - Four classes: User, Comment, Course, Post

State-machine diagram



Graph #4: State-Machine Diagram

ER – Diagram (Data modelling)



Graph #5: ER Diagram - Four entities: User, Comment, Course, Post

GitHub link to your project source code

https://github.com/qthuang/cs411_sum2022

Conclusion (lesson learned)

The group members learn from the project about timeline management, collaboration in development and applying the content we learnt from CS411 class.

For time management, since there are only 6 weeks for us to complete the project, we tried to start planning as early as possible. For the first two weeks, we got familiar with each other and explored possible problems that can be solved by developing an application. For the next 2 weeks, we assigned tasks to each member and started the development process. We worked on writing the report and drawing the diagrams together in the last week. Though the time for the project is short, we found that it is enough if we manage to fairly plan the time usage and start everything early.

We also learnt a lot about collaboration in web application development. At first, we were not sure how to assign tasks for each member. We figured this out by introducing ourselves about what we were familiar with, and then quickly divided the group into the front-end part and the back-end part, and then decided to work on the concatenation together. In the development process, we searched on the internet and communicated, and therefore learnt many new techniques on database, css and HTML. What we learnt was that everyone should work on the things that he/she is good at, and keep a smooth communication about any inconsistencies of the development.

When writing the report, we did a good practice on identifying problem definition, stakeholders, as well as creating diagrams including use case diagram, sequence diagram, detailed class diagram, state machine diagram and ER diagram. Applying this software engineering knowledge to the application which is developed by ourselves gives us a better understanding of this professional knowledge, and will help us in the future with actual software development tasks.

Appendix

Reference

W3School HTML&CSS Tutorial

https://www.w3schools.com/html/html_basic.asp

<http://www.w3.org/2000/svg>

BootStrap HTML Template & CSS

<https://getbootstrap.com/docs/5.2/examples/album/>

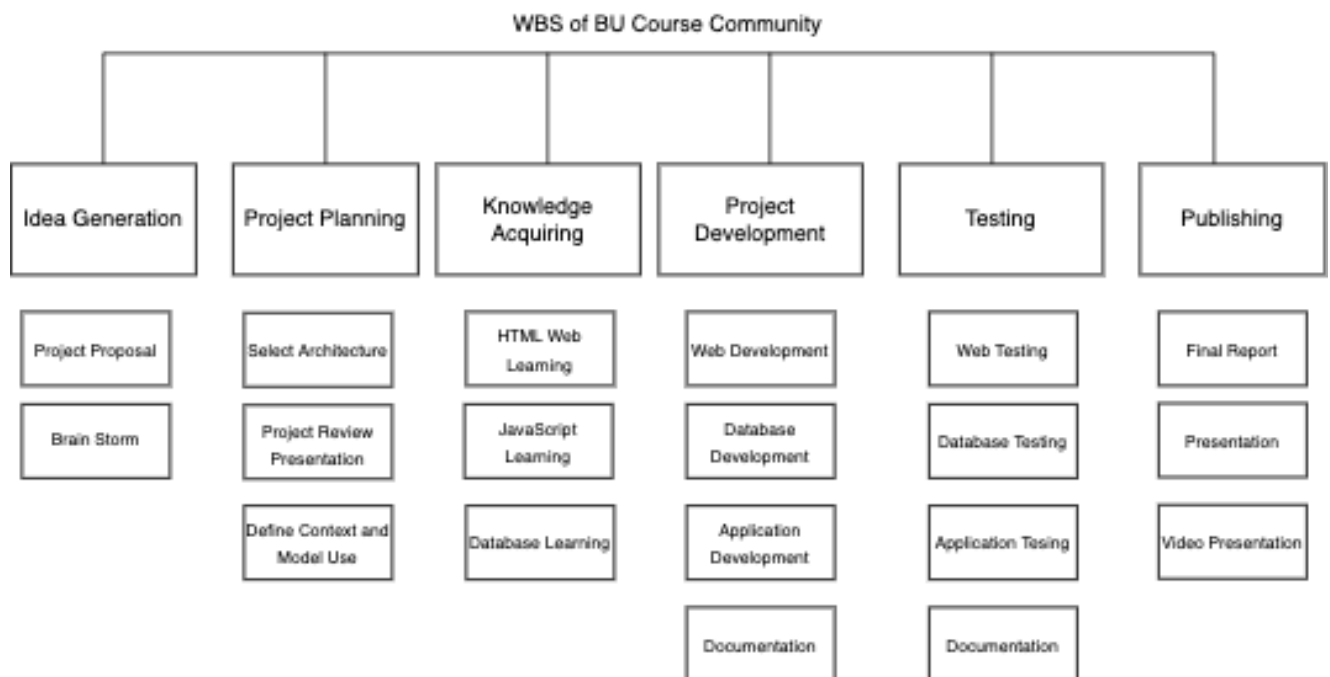
<https://getbootstrap.com/>

Django & Mysql reference

<https://docs.djangoproject.com/en/4.0/>

<https://dev.mysql.com/doc/>

Project WBS




Graph #6: Project WBS

Task Assignment Matrix


<u>Task</u>	<u>Task Owner</u>	<u>Support</u>
User Web Development (Front end)	Hao Chen	Qintian Huang
Database Design (Back end)	Lingwen Deng	Chengjie Gu
Project Proposal	Hao Chen	Lingwen Deng, Qintian Huang
Report	Chengjie Gu	Lingwen Deng
PPT Presentation	Qintian Huang	Hao Chen, Qintian Huang, Lingwen Deng
Video Presentation	Hao Chen, Qintian Huang, Chengjie Gu, Lingwen Deng	

Graph #7: Task Assignment Matrix


Sample of commits

 Commits on Jun 26, 2022


Add files via upload


 Duckyliam committed 3 hours ago

Verified





f579908




 Commits on Jun 24, 2022

modify backend code


 dlwen committed 3 days ago




f45cebfb





upload backend code

 dlwen committed 3 days ago





90fb3c9




 Commits on Jun 19, 2022

6.19 hqt frontend update


 qthuang committed 7 days ago




f738486




6.19 hqt updated


 qthuang committed 7 days ago




1e7cb12




6.19 hqt updated


 qthuang committed 7 days ago




f0231d1





6.19 hqt

 qthuang committed 7 days ago




4f6e919




 Commits on Jun 12, 2022


Add files via upload

 Danny731 committed 14 days ago

Verified



f75b2fb



Graph #8: Git Commit Example Screenshot