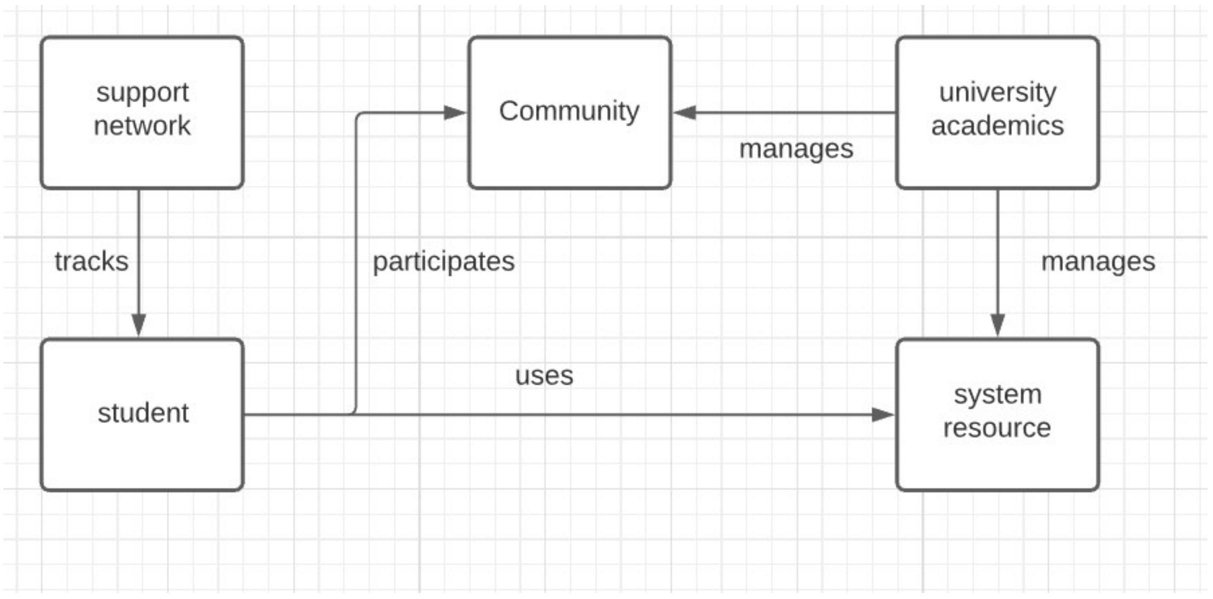
1. General analysis of the system architecture

According to the specification, the main usages of the system can be described as:

1. Student users will use the resources of the system to interact with other students, mainly through study activities, with implementation of gamification. Alternatively, student users can interact with each other through a student community
2. Staff users will participate with the student community by managing the study related content, alternatively they can manage system resources
3. Wellbeing/Welfare staff can monitor the ongoing wellbeing of students, without the need for direct participation with the student community.

This is illustrated in the diagram below:



2. Analysis of the system resource design (gamification)

As a core part of the student community, the system needs to implement a mechanism which allows student users to interact with each other using virtual resources. The concept of gamification provides a way to merge study behaviour with game mechanics. However, by definition, gamification refers to apply a whole set of gaming mechanism to the study behaviour rather than just simply adding a reward feature to the original system.

3. Technical Analysis of implementing a track system for support network

As an initial idea, student users can update their wellbeing status proactively through a welfare questionnaire. This can then be viewed by the corresponding support network of that student user, which requires a space to store the information given by the user. The required data will be held in a database. Every time the student answers another welfare quiz, new data on their wellbeing status will be added into the database. The support network can then perform operations to read this data. Alternatively, a user interface could be provided to allow both student and the support web to perform these operations, which in this case will be a website, as specified.

4. Technical Analysis of implementing a student community

Specified features of a student community:

1. Student user can post new study notes / resources and modify / delete them
2. Student user can update / share their wellbeing status (including achievements)
3. Staff user can modify study notes / resources / challenges once posted
4. Student user can set challenges for other users to attempt – most likely in the form of quizzes
5. Student user can build a relationship with other student users through discussions in the forums

To implement feature 2 and 5, there must exist a space to store corresponding data of all student users; meanwhile to implement feature 1/3/4, there must exist another space, this time to store data corresponding to all posts in a community. These two requirements can also be fulfilled using a database, which enables users with appropriate privileges to write / read the data. Alternatively, same to the previous section, a user interface will be needed for both student and the staff users to perform these operations, which in this case will be a website, as specified.

5. Technical Analysis of implementing a gamified mechanism for student users

Based on the analysis above, we think it is fair to combine an existing game mechanism to a quiz system, such as a card game. The requirements of implementing this are:

1. Asynchronous communications between different student users
2. Efficient mechanisms to display the game field
3. Animations on web pages
4. Appropriate storage space for the current game's data
5. Appropriate questions resources

<u>To fulfil requirement 1:</u>

After doing some research, we have found that Django (a python framework used for developing web applications) has a feature to implement asynchronous communications between different clients, which can be easily implemented using channels.

<u>To fulfil requirement 2:</u>

From our research, we have found that Vue (a JavaScript framework) will wrap data into “components”, like the concept of objects in Object-Oriented Programming. Thus, allowing us to easily display many copies of the same object on our web pages. This in turn means we can construct the hand / card table / cards of both players as components, supporting a live, and therefore constantly changing, webpage as is required.

<u>To fulfil requirement 3:</u>

Vue also supports a “transition” tag to manage the animations of a html tag, we have also found a CSS animation library called “animate.css”.

<u>To fulfil requirement 4:</u>

As our server is being run with Python, it seemed the natural choice to also run the game. This will reduce the risk of individuals cheating as the game data will be kept secure.

<u>To fulfil requirement 5:</u>

We have decided to use the university past paper storage system as our question source, as all questions stored were written (and thus approved) by staffs. As students already have access to these papers, this will not reduce the integrity of future exams. This is because the questions are unlikely to appear in future exams.