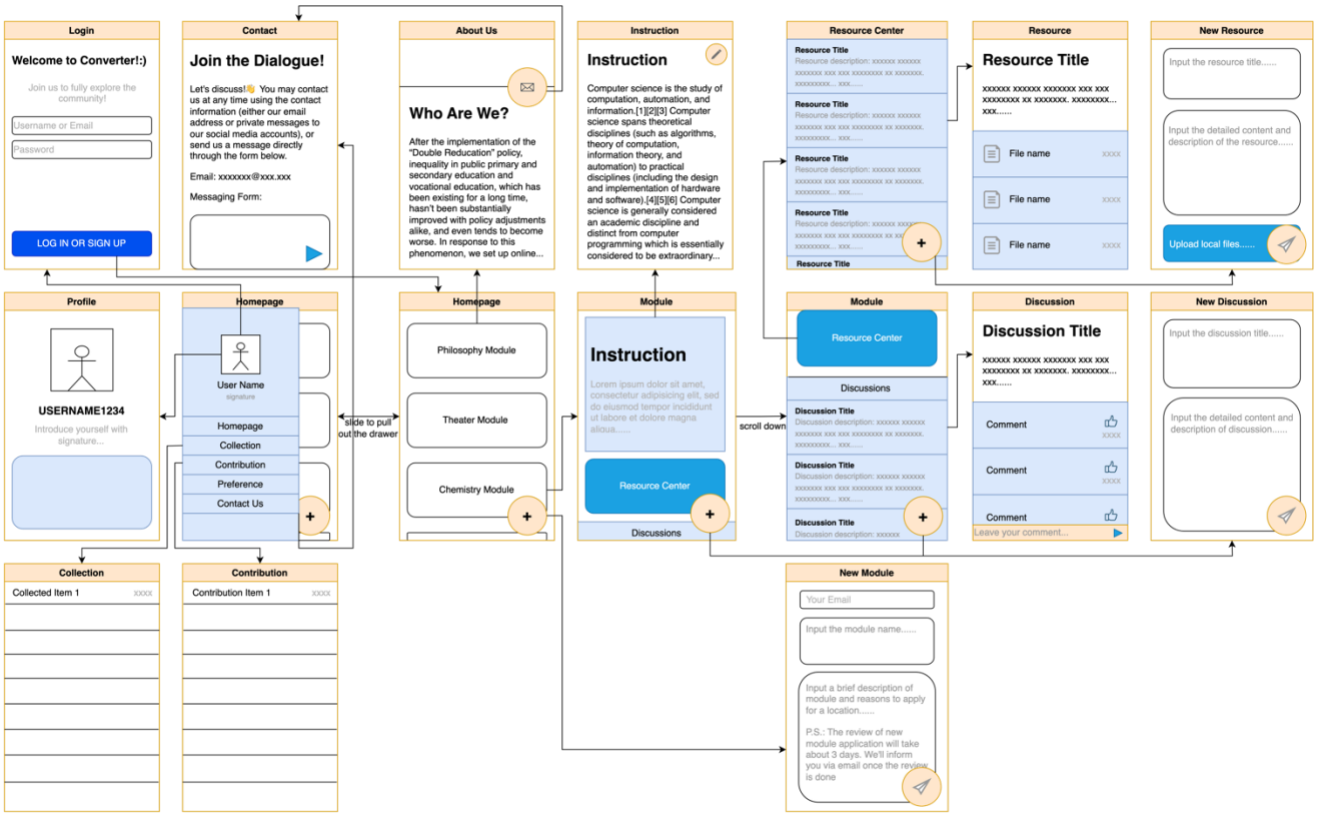
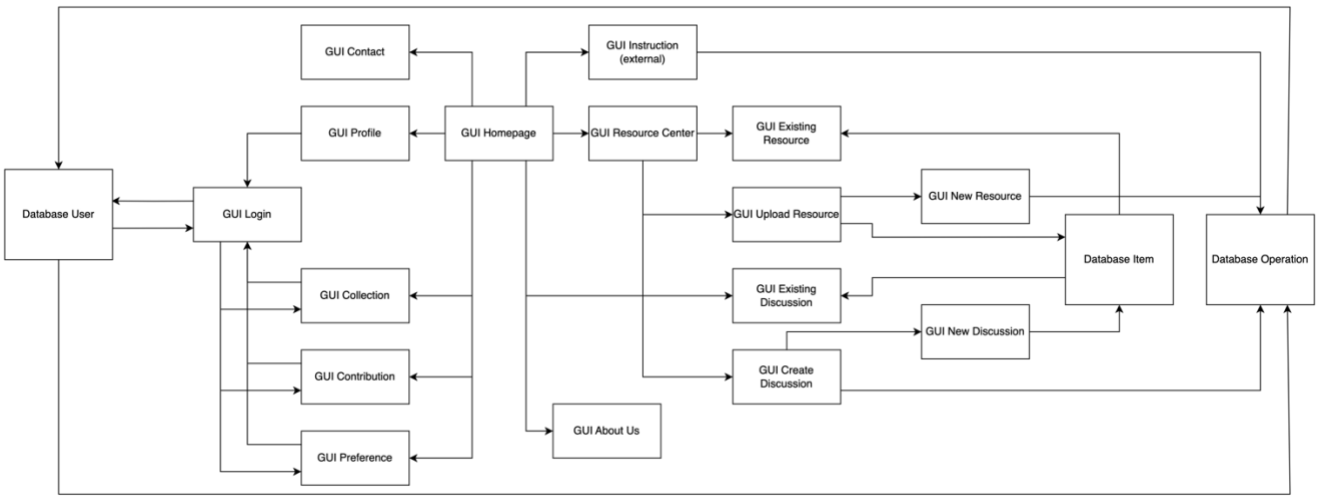
Application Design

Table of contents

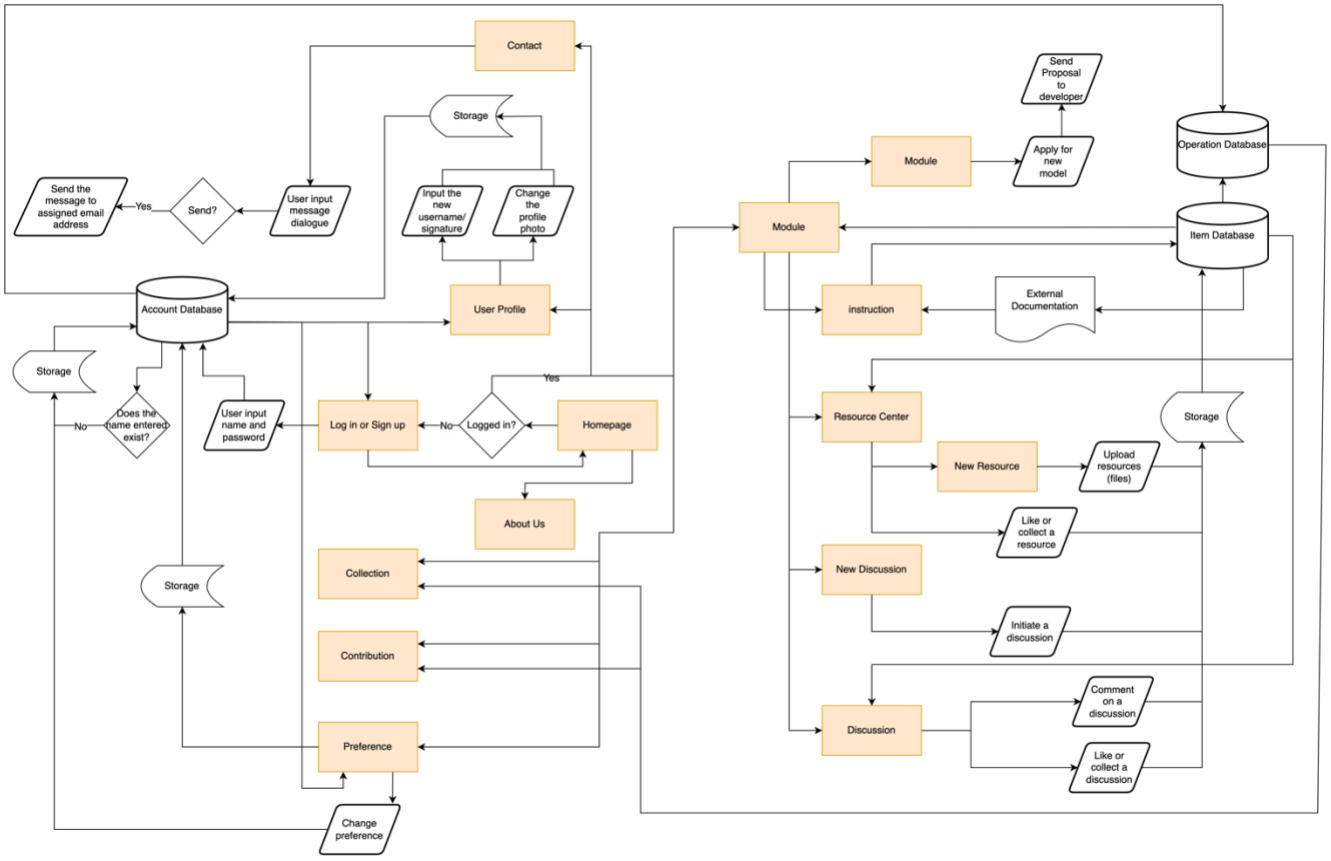
Overall Product Setup (Big Picture)



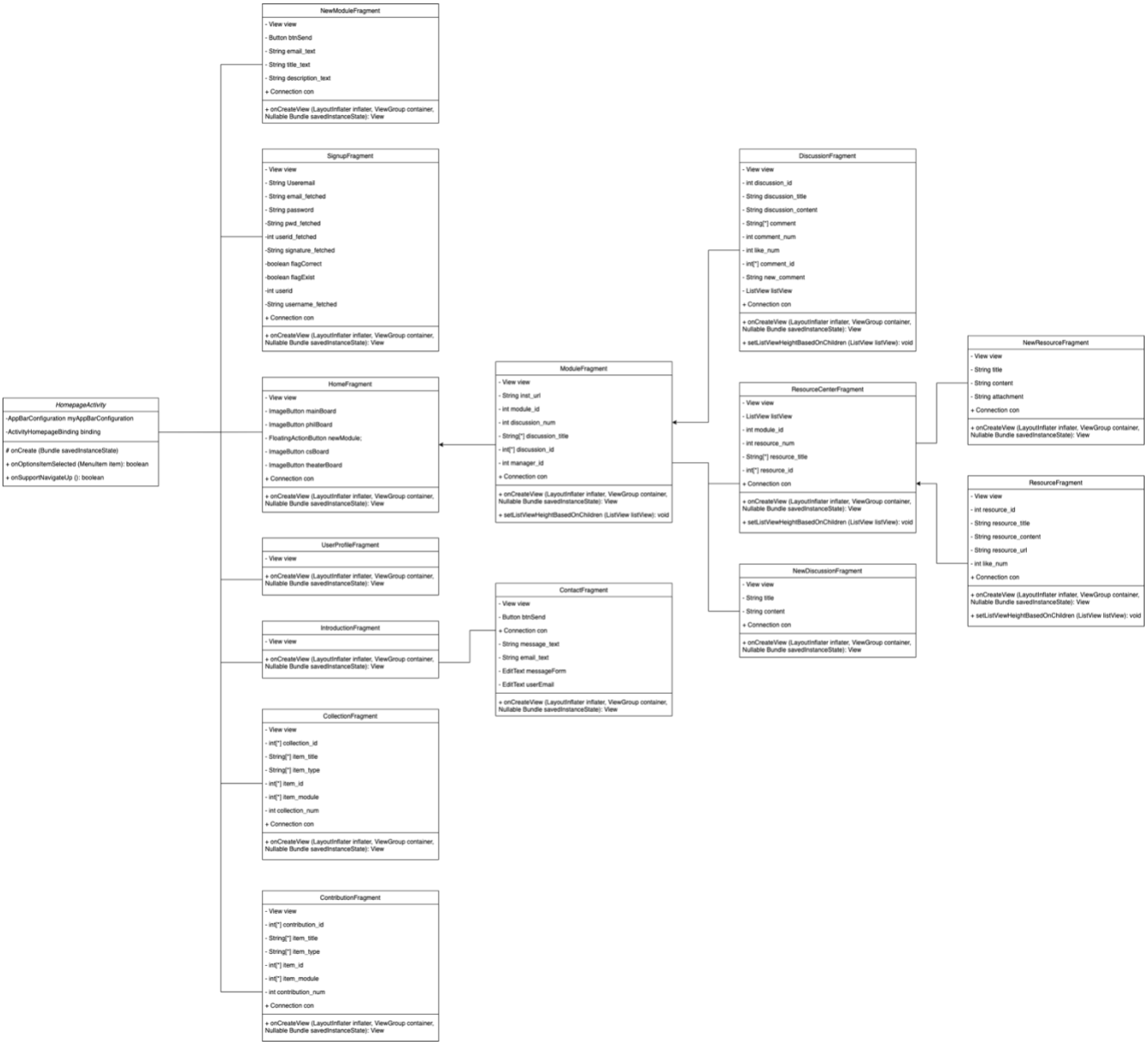
Data Flow Diagram



System Flowchart

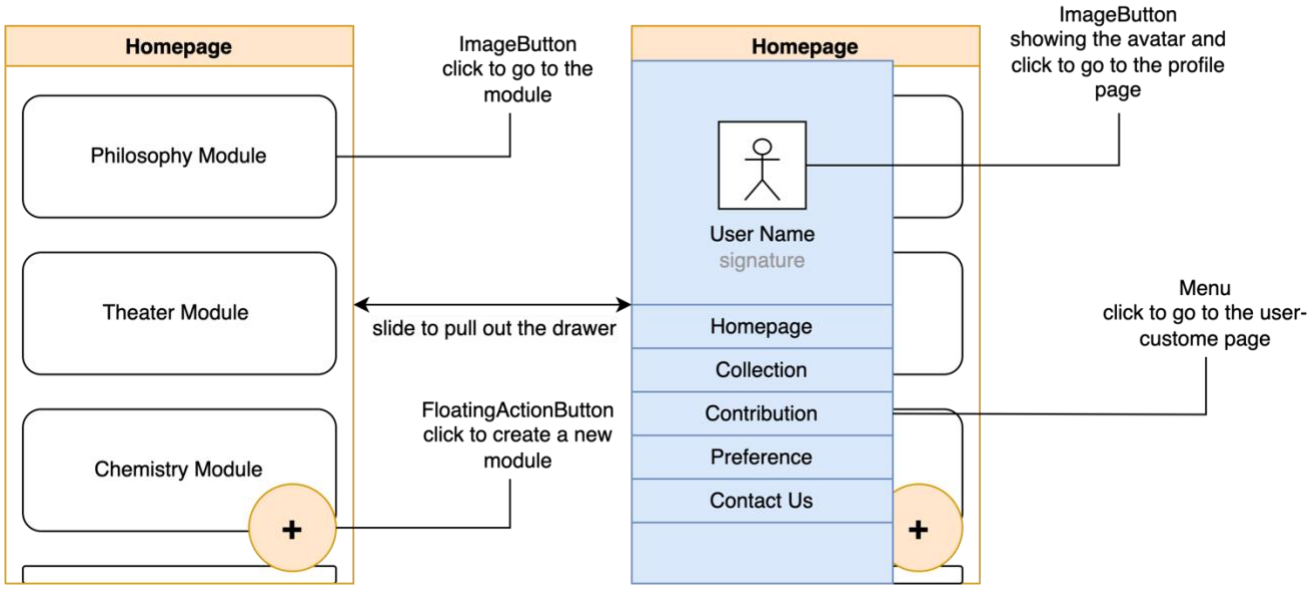


UML Diagram


NewModuleFragment
- View view
- Button btnSend
- String email_text
- String title_text
- String description_text
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
SignupFragment
- View view
- String Useremail
- String email_fetched
- String password
-String pwd_fetched
-int userId_fetched
-String signature_fetched
-boolean flagCorrect
-boolean flagExist
-int userId
-String username_fetched
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
HomepageActivity
-AppBarConfiguration myAppBarConfiguration
-ActivityHomepageBinding binding
onCreate (Bundle savedInstanceState)
+ onOptionsItemSelected (MenuItem item): boolean
+ onSupportNavigateUp (): boolean
HomeFragment
- View view
- ImageButton mainBoard
- ImageButton phillBoard
- FloatingActionButton newModule;
- ImageButton csBoard
- ImageButton theaterBoard
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
UserProfileFragment
- View view
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
IntroductionFragment
- View view
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
CollectionFragment
- View view
- int[*] collection_id
- String[*] item_title
- String[*] item_type
- int[*] item_id
- int[*] item_module
- int collection_num
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
ContributionFragment
- View view
- int[*] contribution_id
- String[*] item_title
- String[*] item_type
- int[*] item_id
- int[*] item_module
- int contribution_num
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
ModuleFragment
- View view
- String inst_url
- int module_id
- int discussion_num
- String[*] discussion_title
- int[*] discussion_id
- int manager_id
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
+ setListViewHeightBasedOnChildren (ListView listView): void
ContactFragment
- View view
- Button btnSend
+ Connection con
- String message_text
- String email_text
- EditText messageForm
- EditText userEmail
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
DiscussionFragment
- View view
- int discussion_id
- String discussion_title
- String discussion_content
- String[*] comment
- int comment_num
- int like_num
- int[*] comment_id
- String new_comment
- ListView listView
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
+ setListViewHeightBasedOnChildren (ListView listView): void
ResourceCenterFragment
- View view
- ListView listView
- int module_id
- int resource_num
- String[*] resource_title
- int[*] resource_id
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
+ setListViewHeightBasedOnChildren (ListView listView): void
NewDiscussionFragment
- View view
- String title
- String content
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
NewResourceFragment
- View view
- String title
- String content
- String attachment
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
ResourceFragment
- View view
- int resource_id
- String resource_title
- String resource_content
- String resource_url
- int like_num
+ Connection con
+ onCreateView (LayoutInflater inflater, ViewGroup container, Nullable Bundle savedInstanceState): View
+ setListViewHeightBasedOnChildren (ListView listView): void

Annotated GUIs

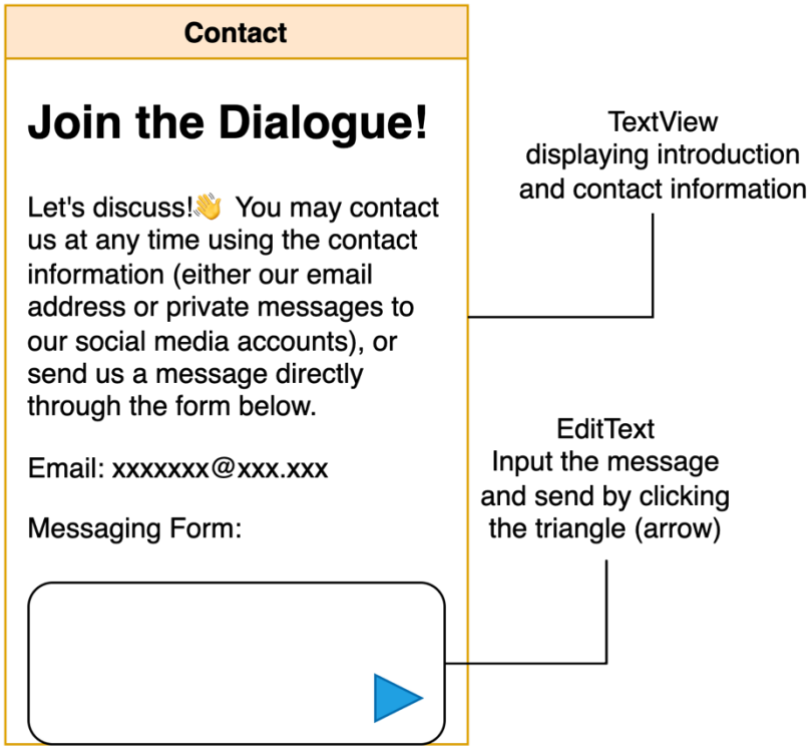
Homepage GUI



The user may either view the main page or the drawer aside. Through the main page, the user can see different modules and go to the module by clicking on each of them; they can also view the contact page and the about us that provides a brief introduction to the app. By clicking the FloatingActionButton, they will go to the module-creation GUI and create a new module by themselves.

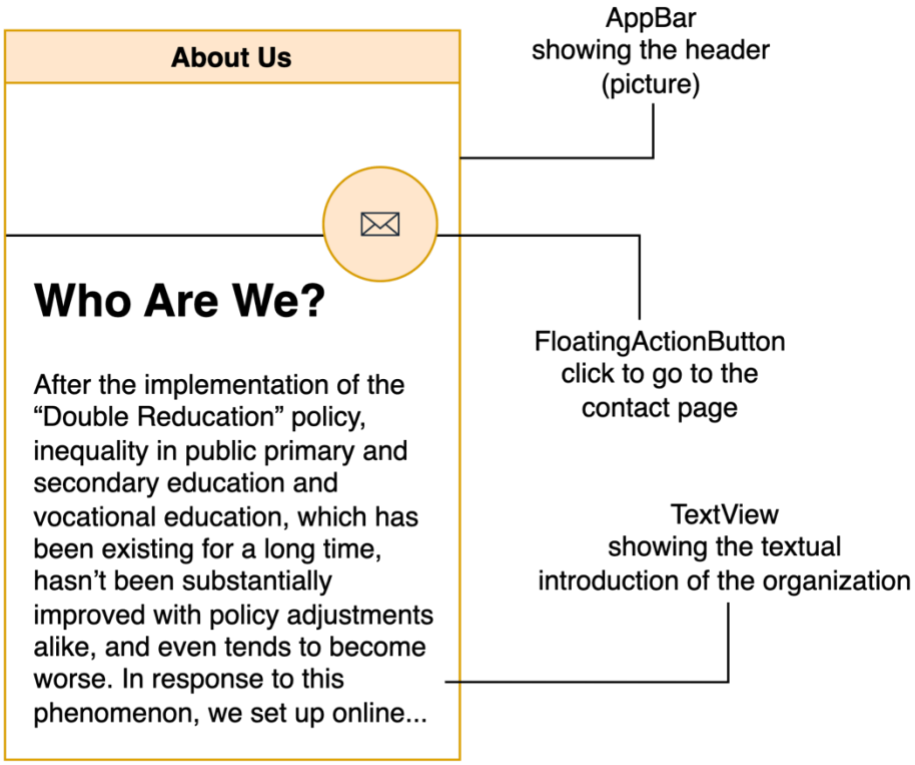
In the drawer, the user can see their customed information, including profile, contribution, collection, and preference. If the user clicks on the drawer and the condition of being logged in is met, the user can go to the respective page; if the condition is not met, they're transmitted to the login page to log in or sign up first.

Contact GUI



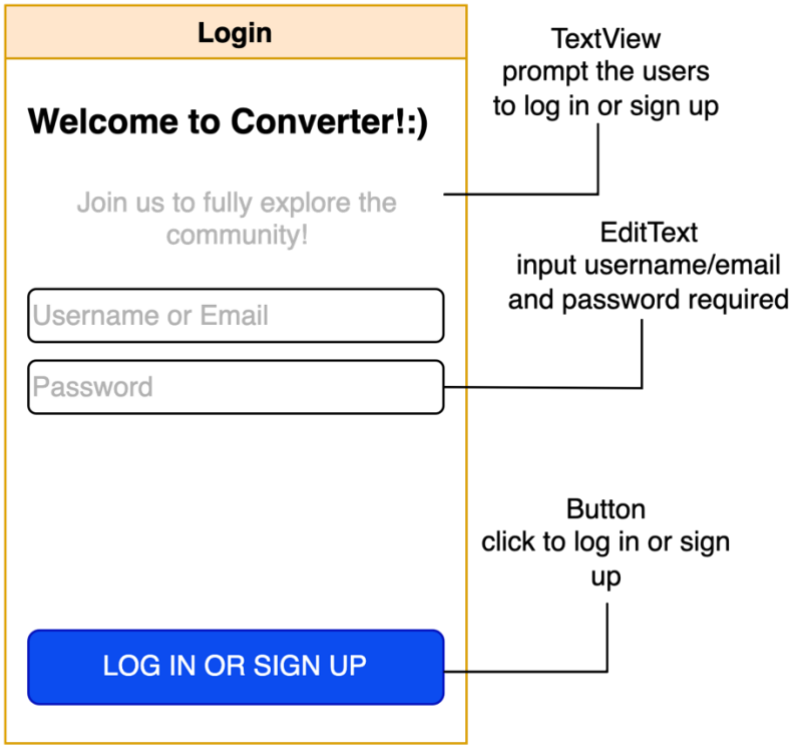
In the Contact GUI, the user can view our slogan and a brief introduction encouraging them to contact the developer; they can copy the contact information (email address) and contact us by themselves; either they may input text in the blank messaging form below and send it inside the application. The text inputted would be received by our database.

About Us GUI



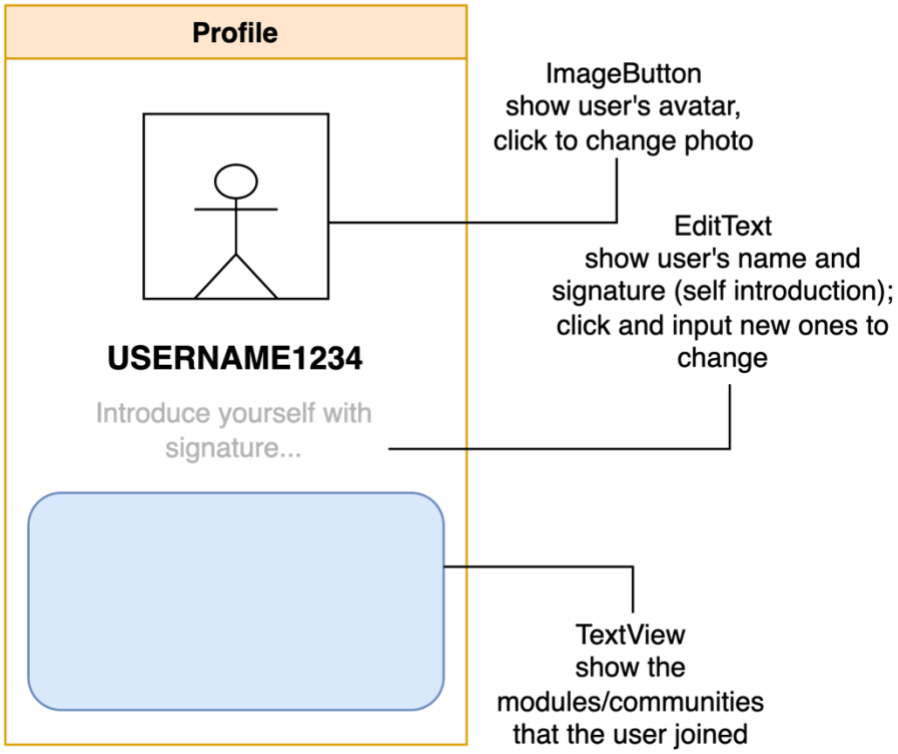
The About Us GUI provides users with static textual contents that introduce the application and the organization. An AppBar on the top will show the logo of the application while the textual contents below introduce the organization in three aspects. The user can also click on the Floating Action Button to go to the contact GUI.

Login GUI



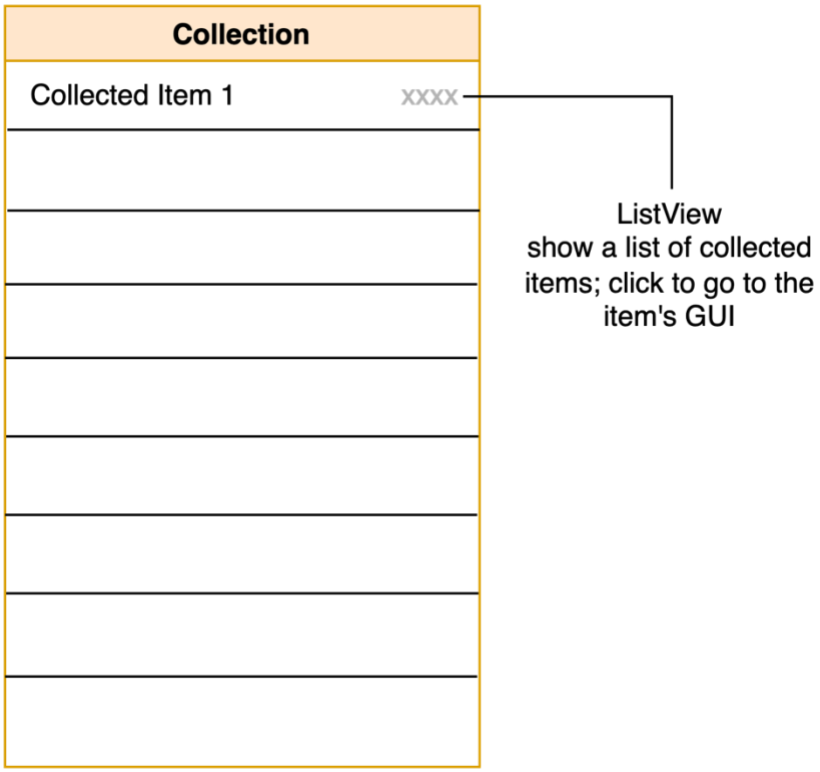
In the Login GUI, the user can either log in or sign up to their account by inputting the username/email as account and the password. They're informed whether the login is successful after clicking the button.

Profile GUI



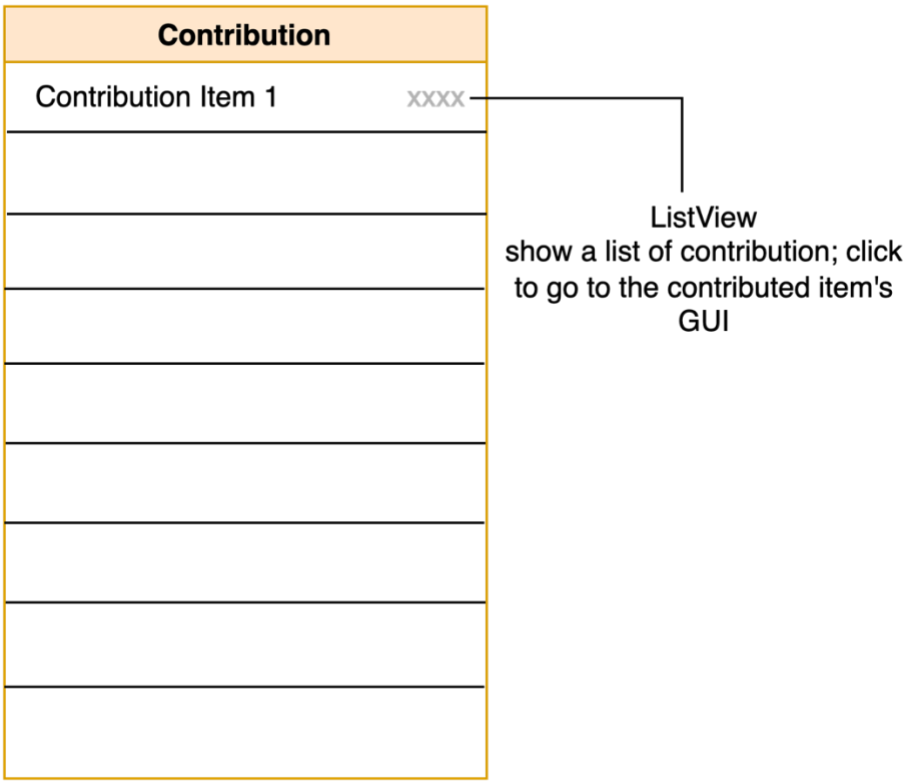
In the Profile GUI, the user can view their current information stored in database: avatar/profile picture, username, self-introduction/signature, and communities or modules joined. They can change their profile picture by clicking it—they may choose photos from their local photo library; they can also change the username and signature by clicking on the texts, and they will be informed by a toasted message if the change is successfully sent to the database. The user can also click “Logout” to log out of the current account.

Collection GUI



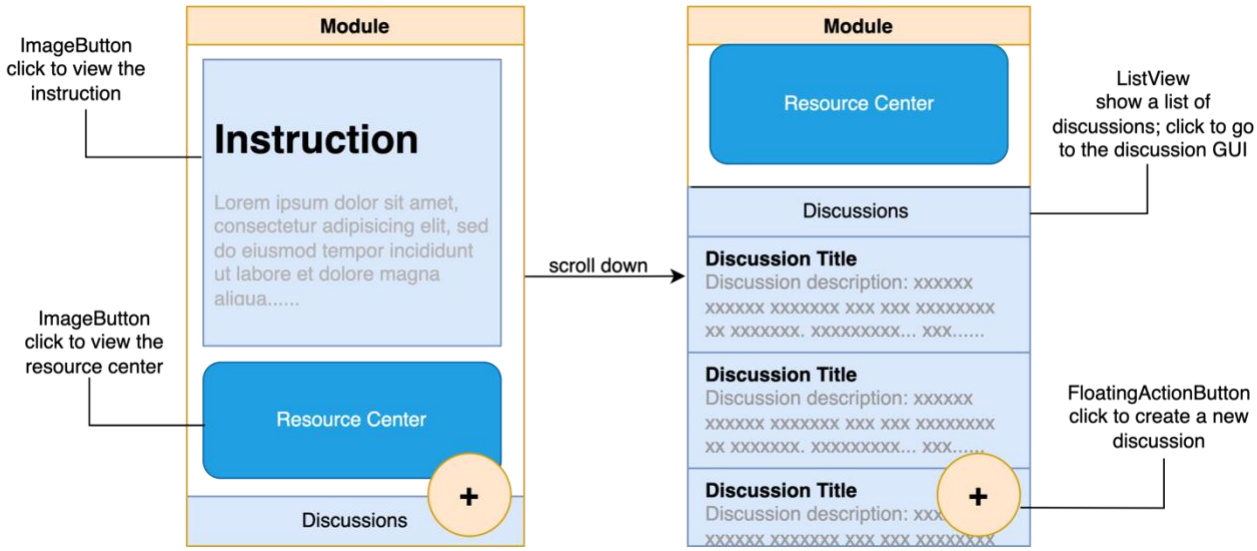
The Collection GUI displays a dynamic list of items (discussion, instruction, or resource) collected by the user logged in with a title and a time when the item is collected. The user can click on each item and will be navigated to the GUI of the item.

Contribution GUI



The Contribution GUI displays a dynamic list of items (discussion, instruction, or resource) that the user logged once contributed to (including editing and commenting) with a title and a time when the contribution is done. The user can click on each item and will be navigated to the GUI of the item.

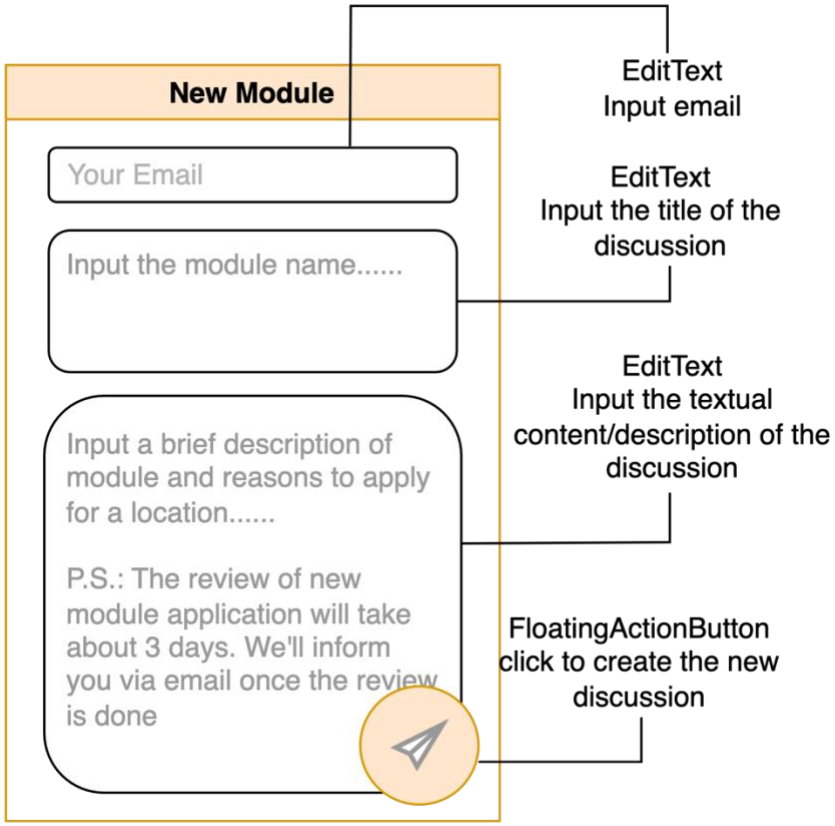
Module GUI



The Module GUI is scrollable and contains mainly three sections: the instruction, resource center, and discussion. The ImageButton of instruction displays a title and a brief introduction to the module, where the user can click on the button and thus be directed to the instruction GUI of the module; the ImageButton below can navigate the user to the Resource Center GUI of the module once the user clicks on it.

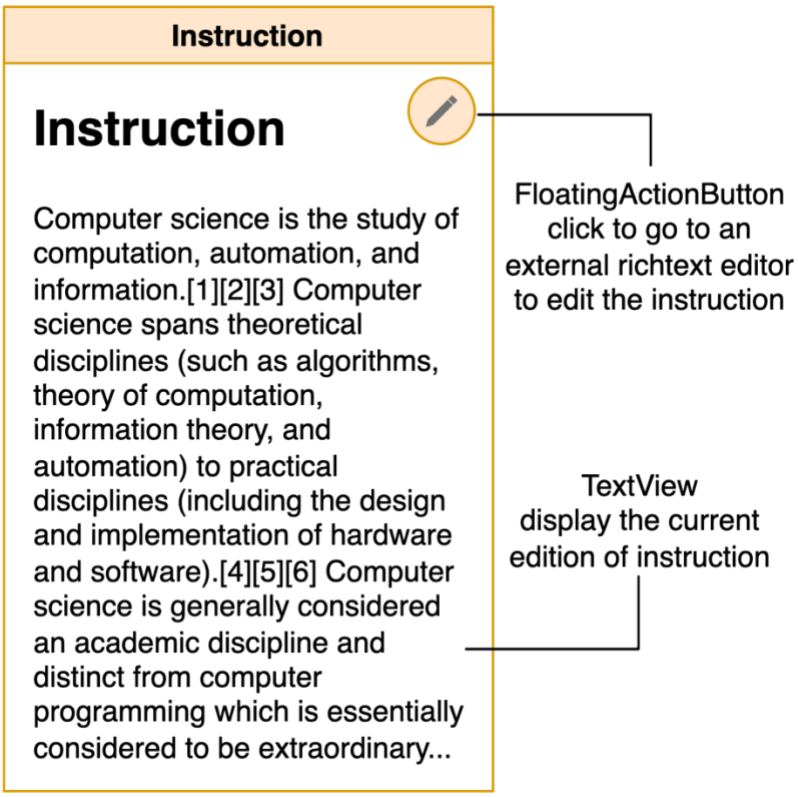
The list of discussions is displayed directly in Module GUI as a ListView, in which each has a title, a brief description, and the time of creation. Similarly, user can click on any of the items in the list to be navigated to the Discussion GUI of the discussion selected. Besides, user can click on the FloatingActionButton on the right bottom that navigates the New Discussion GUI to create a new discussion.

New Module GUI



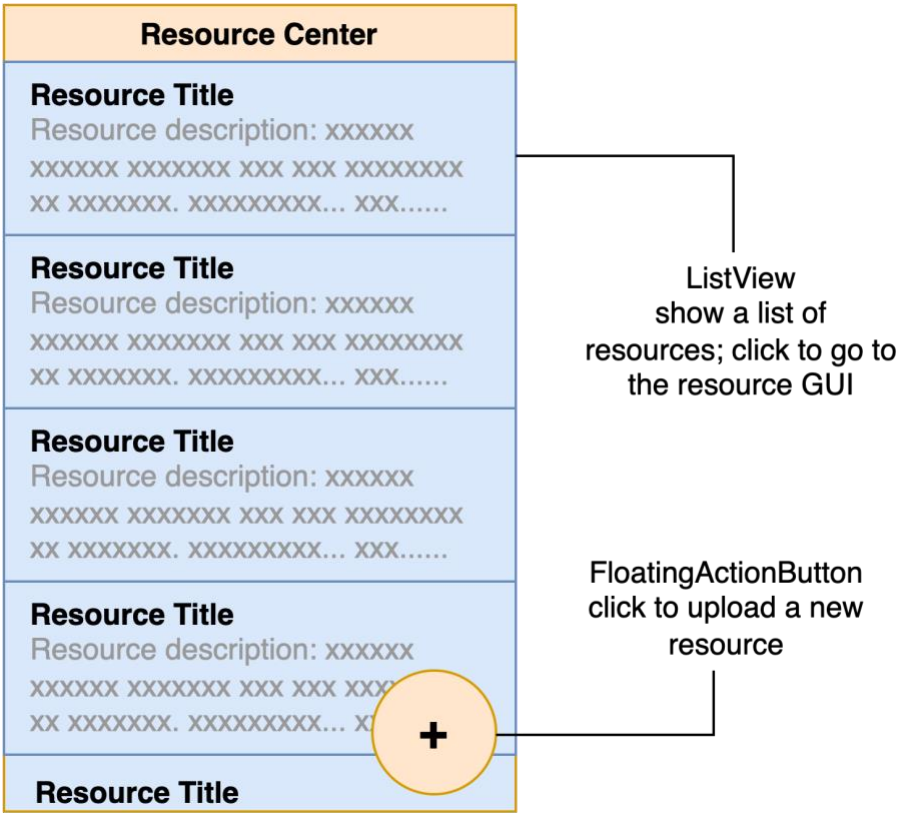
In the New Module GUI, the user can apply to create a new module/community related to a specific field of extracurricular interest. They can input their email, the name of the new module, and a description/proposal to apply for the new module in each of the three EditText widgets. The proposal will be sent to the developer once the user click the FloatingActionButton.

Instruction GUI



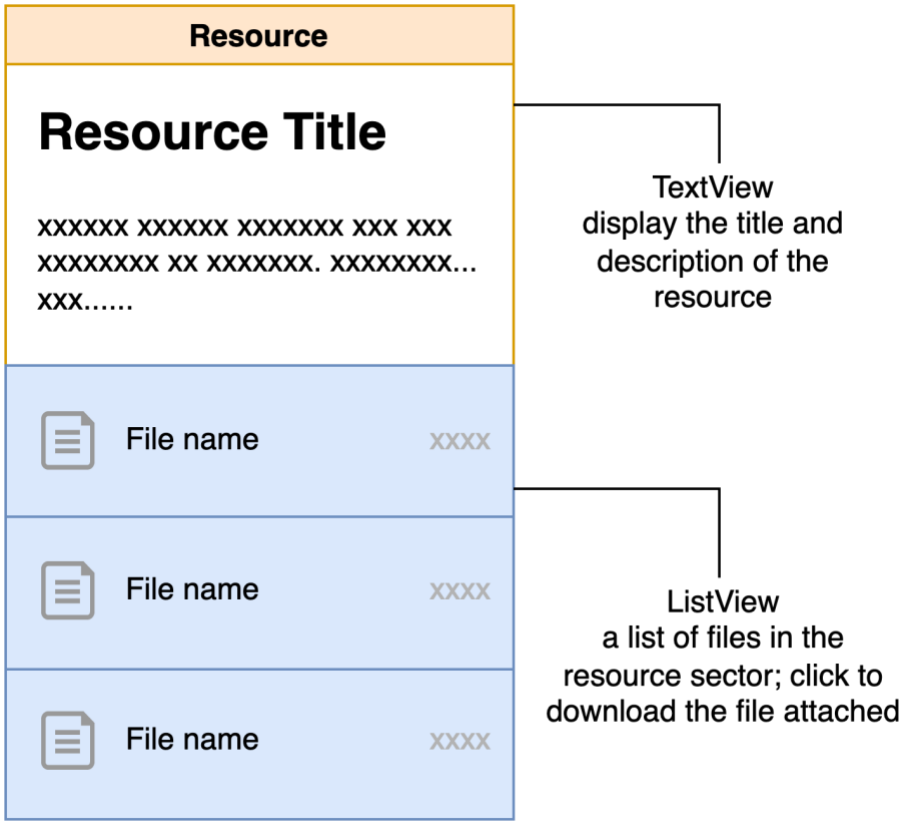
The Instruction GUI display the instruction to start up with the extracurricular field of the module with Text. The user can scroll to view the textual contents. If they want to edit the instruction, they can click the FloatingActionButton which navigates them to an external rich text editor.

Resource Center GUI



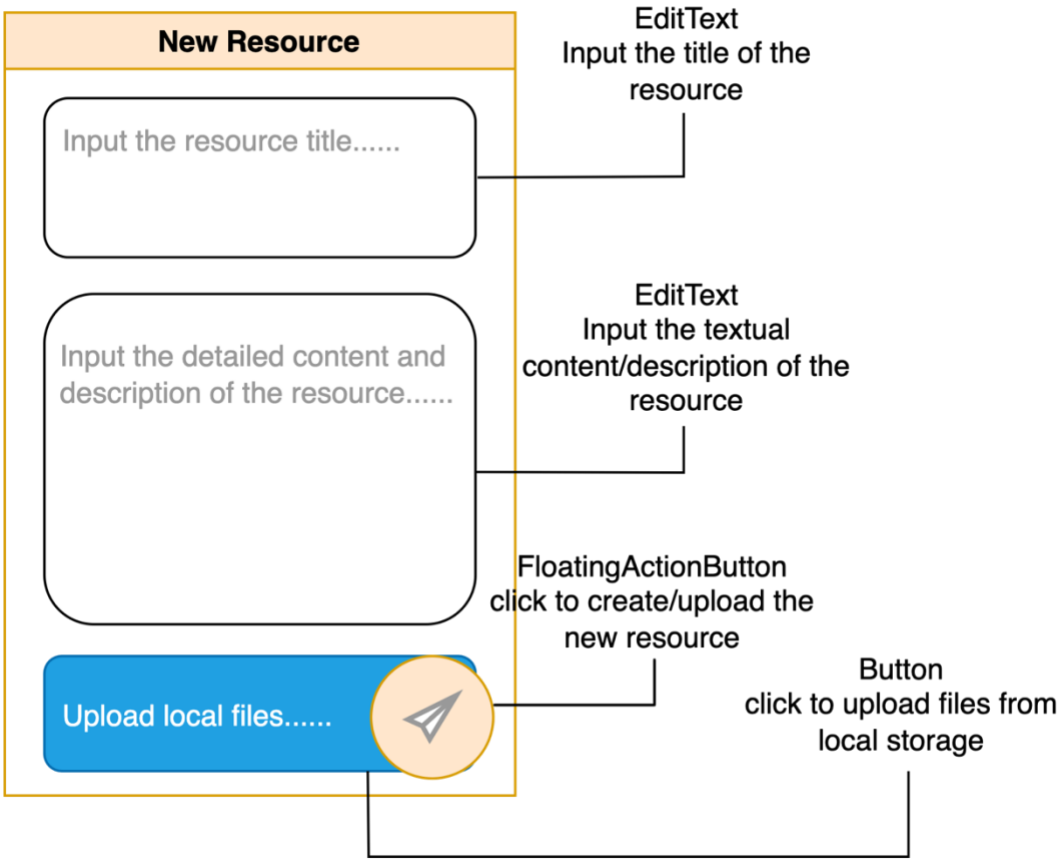
The Resource Center GUI displays a list of resources uploaded in this module/community. Each item in the list shows a title and a brief description. The user can click on the item and go to Resource GUI of the resource selected.

Resource GUI



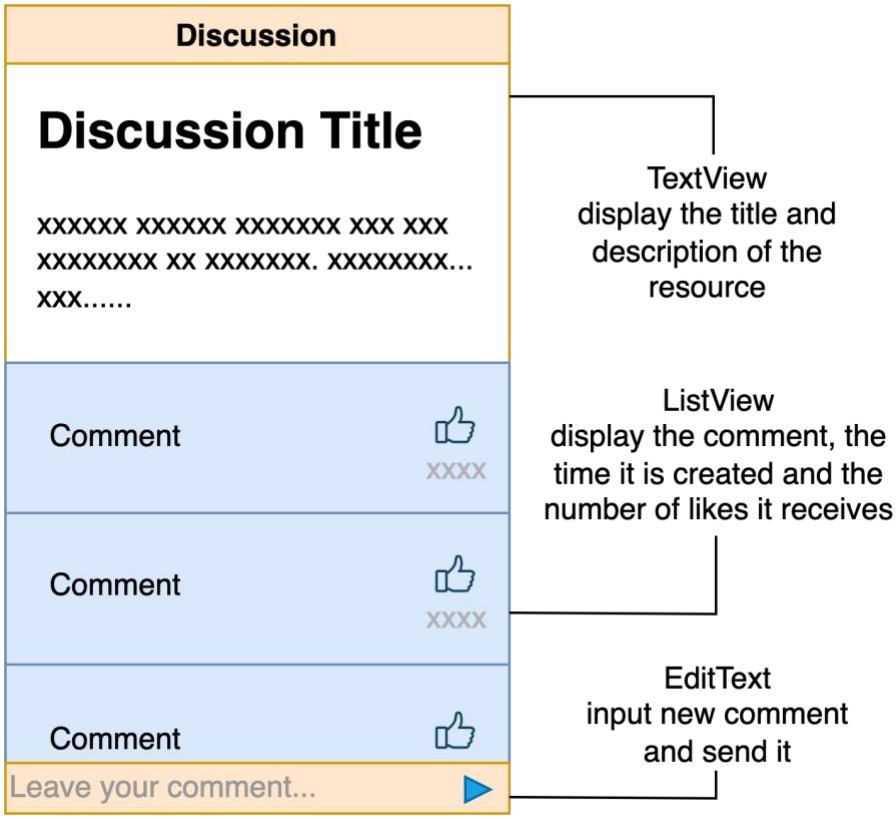
The Resource GUI allows users to view the resource in detail. The user may see the title and description in TextView; they may then see resource files uploaded in a ListView below; they can click to open and download the file.

New Resource GUI



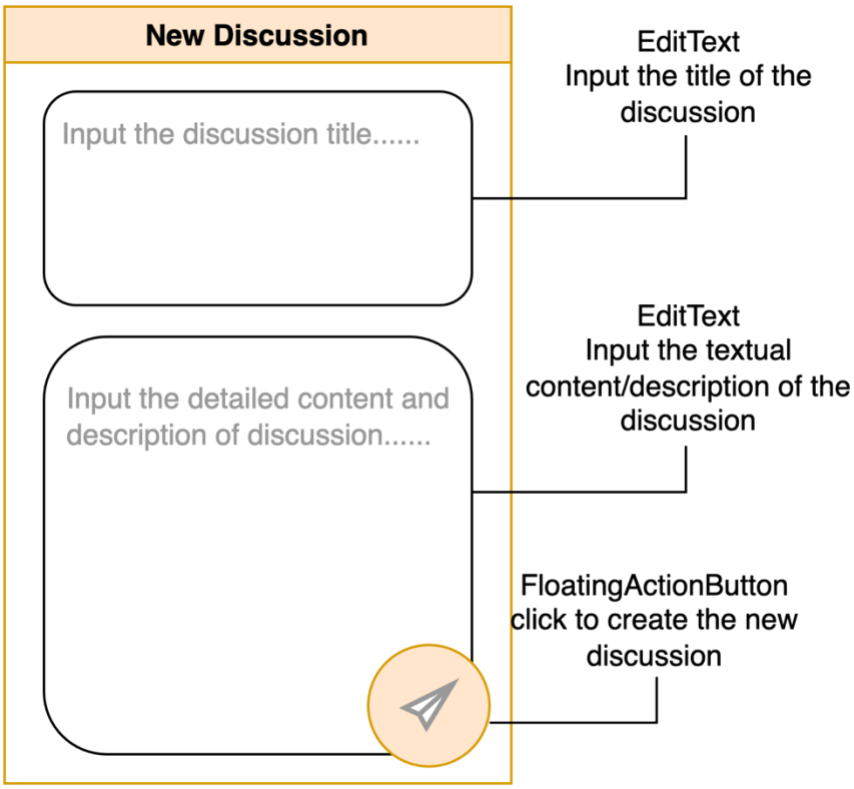
The user can create a new resource section and upload local files through the New Resource GUI. The user may input the title of resource and a description or explanation through the two EditText widgets; they can upload local files by clicking the blue button in the bottom, which will navigate to browse the files on the mobile phone; finally, they may create and upload the resource by clicking the FloatingActionButton which saves the information to database.

Discussion View GUI



The Discussion GUI allows users to view the resource in detail. The user may see the title and description in TextView; they may then see comments (along with the number of likes and the time it is created) under the discussion in a ListView below; they can click the EditText at the bottom to input new comments and send the comment by clicking the small icon of icon.

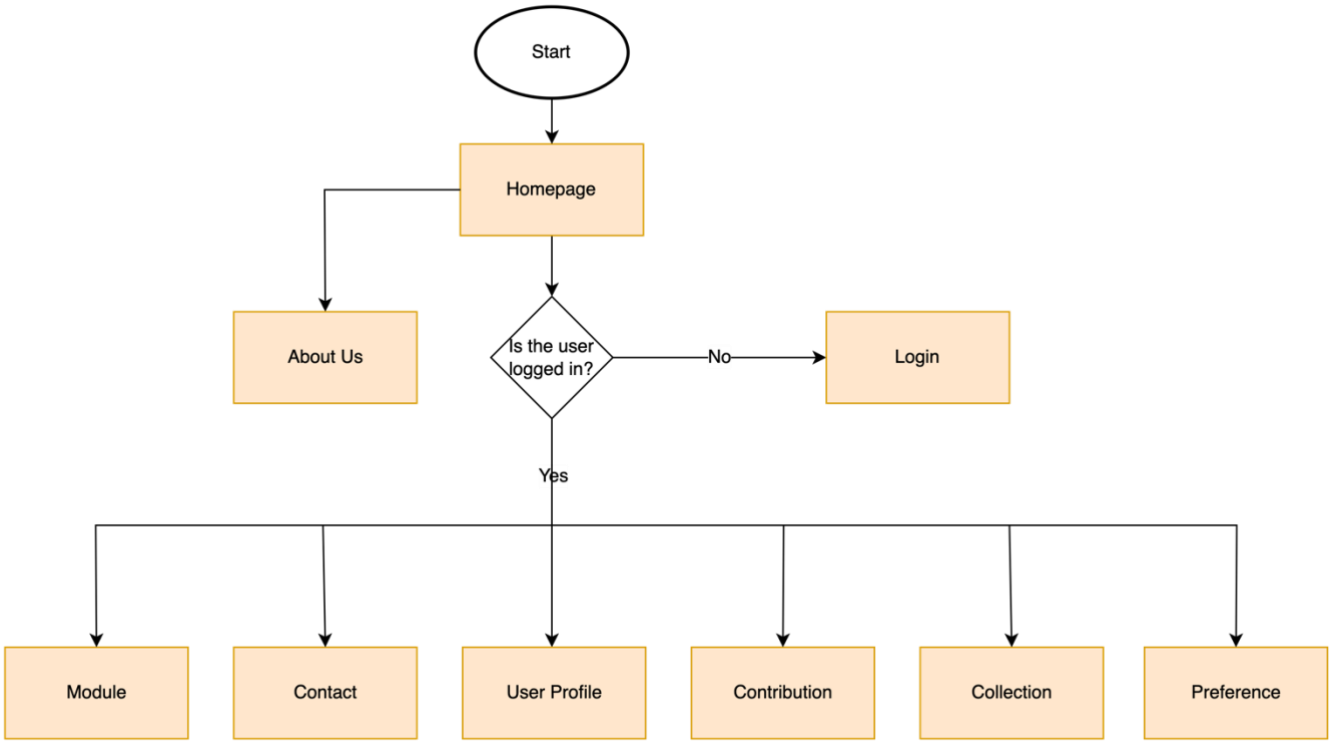
New Discussion GUI



The user can create a new discussion through the New Discussion UI. The user may input the title of resource and a description or explanation through the two EditText widgets; they may create the discussion by clicking the FloatingActionButton with icon of a paper plane which saves the information to the cloud database.

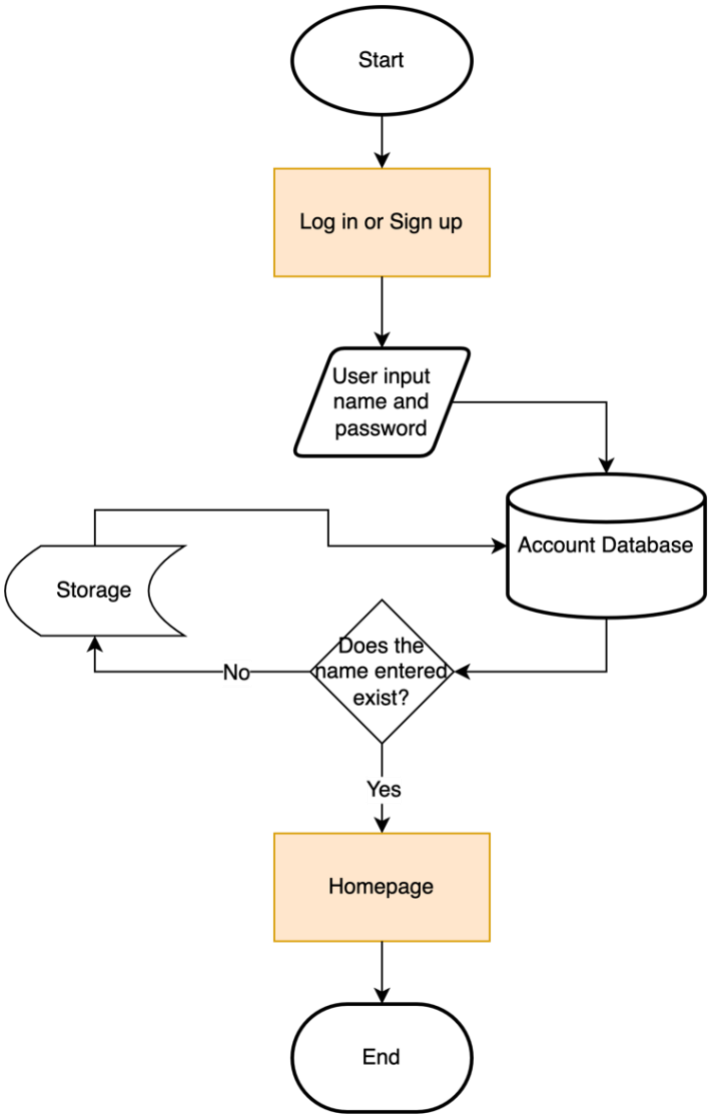
Flowcharts

Homepage



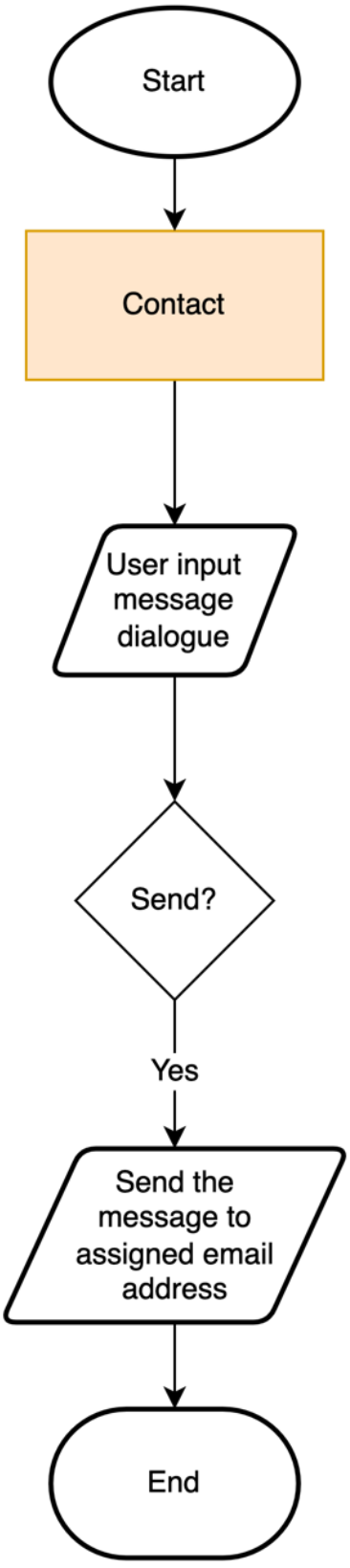
The user will view the homepage and various available widgets to navigate to. Essentially, besides the About US GUI that can be directly directed to, a condition of being logged in is required to be met in order to enter other GUIs like Module, Contact, and User Profile; if the user isn't currently logged in, instead of the GUI they select, they'll be directed to the Login GUI.

Login



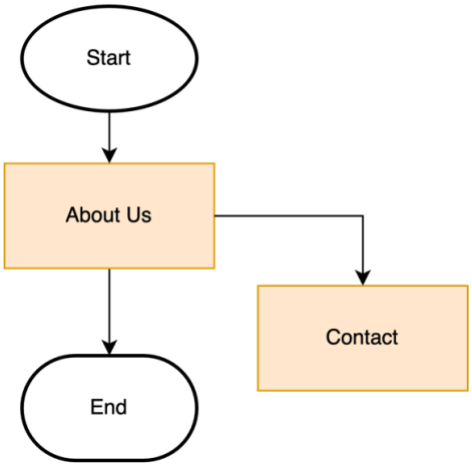
The input is the username/email and the password that the user types in. If the username inputted exists in the database, the user will be successfully logged in; if not, the name and password will be stored in the database and the user is signed up. The user will be directed to the Homepage after login or signup.

Contact



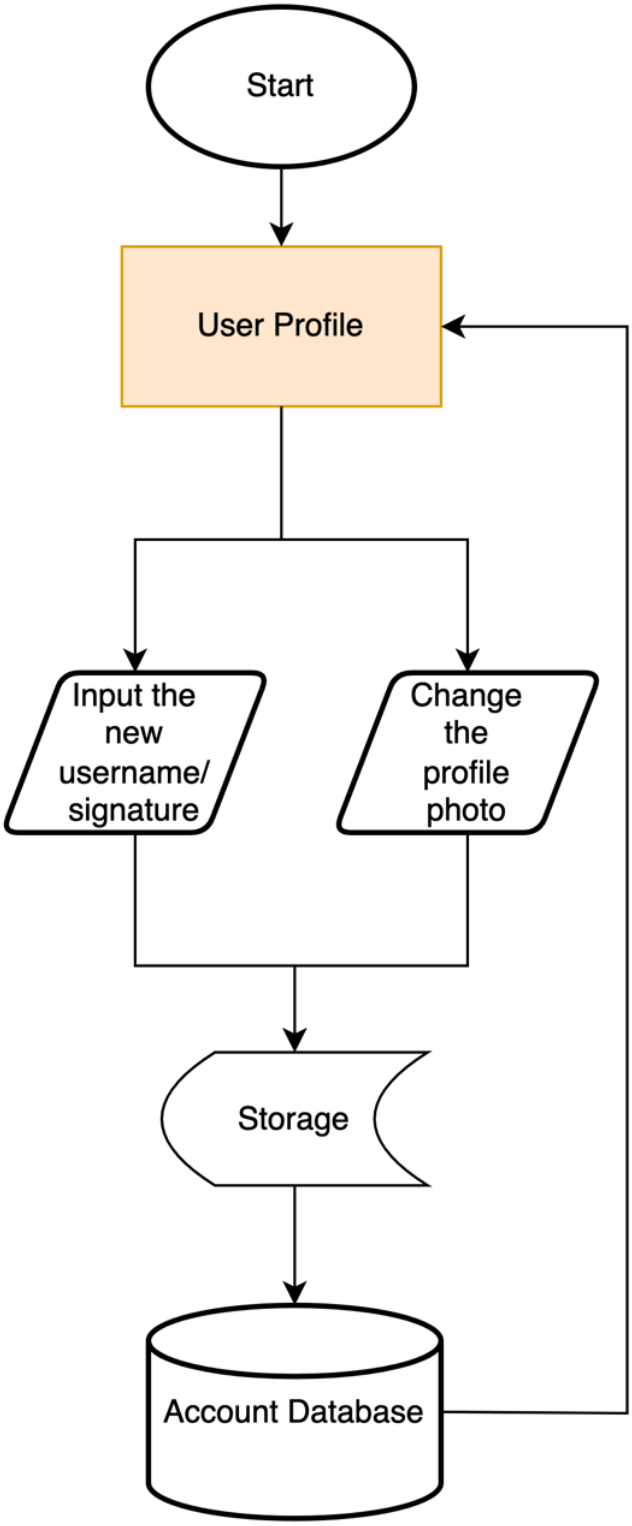
The input is the message dialogue that the user types in. Once the user clicks the button of sending the message, it will be sent to the developer's email.

About Us



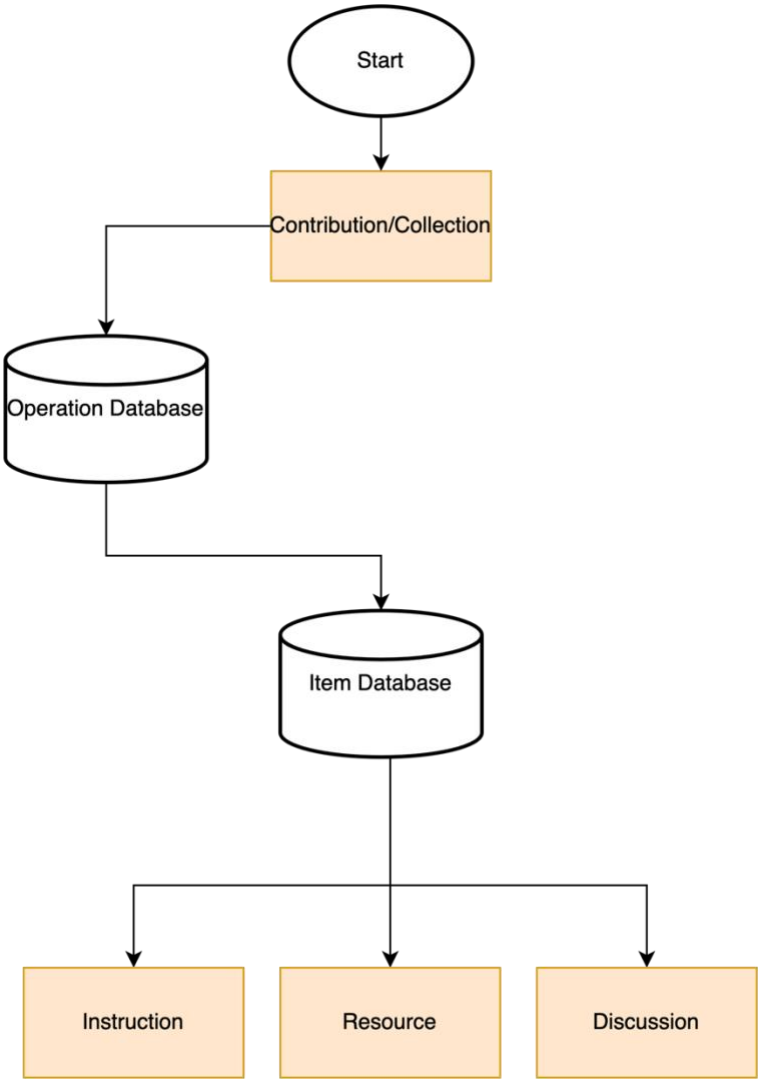
The About Us GUI follows a simple logic of displaying fixed, textual contents behind the application/about the developer. It also directs users to Contact GUI.

User Profile



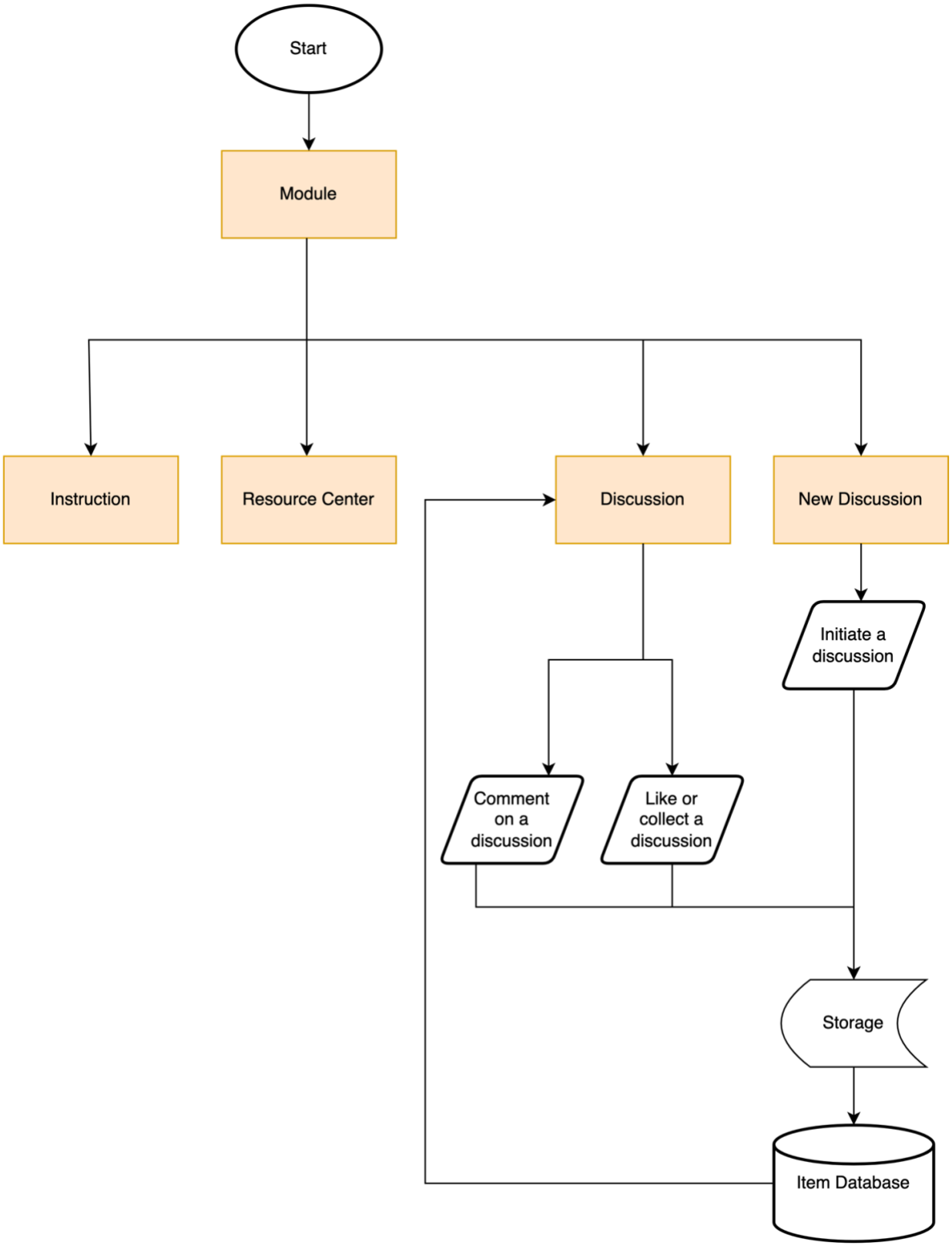
The Input are the username and signature that the user types in or the new profile photo that the user uploads. Once the inputs are done, they're sent to the account database to be stored, which will alter the display of the User Profile GUI.

Contribution/Collection



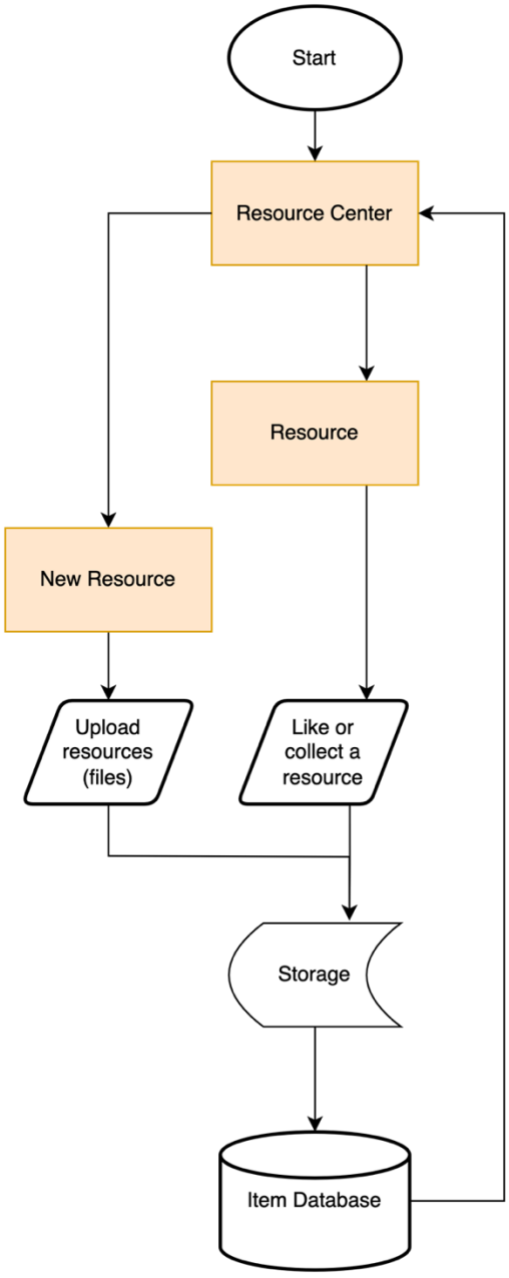
Contribution and Collection GUI share almost the same flowchart logic. Once the GUI is opened, data are fetched from the operation database to display. When items on the menu is clicked, the data are fetched from item database with addresses fetched from the operation database before; user will then be directed to new GUIs (instructions, resource, discussion), depending on what the user clicks on.

Module


Start
Module
Instruction
Resource Center
Discussion
New Discussion
Initiate a discussion
Comment on a discussion
Like or collect a discussion
Storage
Item Database

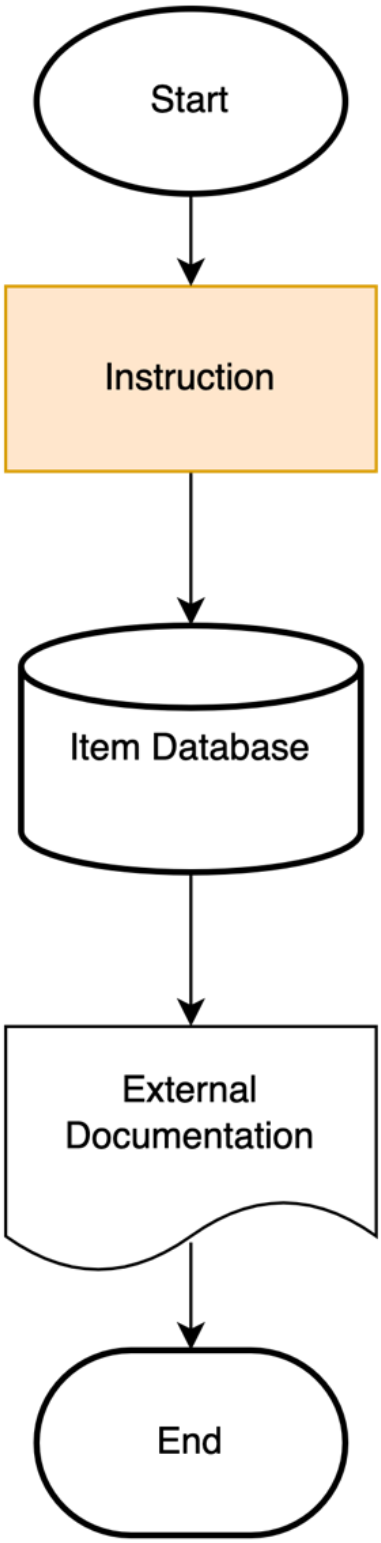
The user can go to different GUIs (Instruction, Resource Center, Discussion, and New Discussion) by clicking on different widgets. In a discussion, the user can input textual comments or likes, which will be stored in the item database; in the New Discussion, the user, can initiate a discussion by inputting information of it, which will also be stored in the item database.

Resource Center



The user can go to the New Resource GUI to create new resources: the input is the textual description of the resource and files attached; they're all sent and stored in the item database. The users may also view the Resource GUI, in which they can like or collect a resource, and their operations are being stored in the item database, which will alter the display of Resource Center GUI.

Instruction



The information URL displayed in instruction GUI is first fetched from the item database, so that the user can be navigated to an external documentation containing the instructional contents.

Database and Tables

User Detail

Store information of each user

Field Name	Data Type	Description
User ID	Integer	The ID of the user
User Name	Varchar	The custom name of the user
Password	Varchar	The custom password of the user
Email	String	The email that the user uses to sign up
Phone Number	Integer	The phone number that the user uses to sign up
Birthday	String	The birthday of the user
Grade Level	Integer	The grade level of the user
Signature	Varchar	The signature or self introduction of the user
Communities	List	The communities/modules that the user joins
Avatar	String (URL)	The user’s profile picture

Module Item

Store information of each module

Field Name	Data Type	Description
Module ID	Integer	The ID of the module
Manager ID	Integer	The user ID of the module's manager
Title	Varchar	The title of the module
Cover Picture	String	The cover picture of the module
Introduction	String	The brief introduction shown on the module
Heat Index	Integer	An index showing how popular the module is
Discussion Number	Integer	Calculate how many discussions there are in the module
Resource Number	Integer	Calculate how many discussions there are in the module

Resource Item

Store information of each resource

Field Name	Data Type	Description
Resource ID	Integer	The ID of the resource
Module ID	Integer	The ID of the module that the resource belongs to
Publisher ID	Integer	The user ID of the one who publishes the resource
Create Time	String	The time when the resource is uploaded
Title	Varchar	The title of the resource
Content	String	The textual description of the resource
Attachment	String (URL)	The URL of the resource file attached
Like Number	Integer	The number of likes that the resource received
Collect Number	Integer	The number of times that the resource is collected

Discussion Item

Store information of each discussion

Field Name	Data Type	Description
Discussion ID	Integer	The ID of the discussion
Module ID	Integer	The ID of the module that the discussion belongs to
Publisher ID	Integer	The user ID of the one who publishes the discussion
Create Time	String	The time when the discussion is created
Title	Varchar	The title of the discussion
Content	String	The textual content of the discussion (the prompt)
Comment Number	Integer	The number of comments that the discussion receives
Like Number	Integer	The number of likes that the discussion receives
Collect Number	Integer	The number of times that the discussion is collected

Comment Item

Store information of each comment

Field Name	Data Type	Description
Comment ID	Integer	The ID of the comment
Discussion ID	Integer	The ID of the discussion that the comment belongs to
Publisher ID	Integer	The user ID of the one who writes the comment
Create Time	String	The time when the comment is made
Content	String	The textual content of the comment
Like Number	Integer	The number of likes that the comment receives
Collect Number	Integer	The number of times that the comment is collected

Contribution Operation

Store information of each contribution done

Field Name	Data Type	Description
Contribution ID	Integer	The ID of the contribution operation
Contribution Type	Varchar	The type of the contribution: instruction, resource, discussion, or comment
Item ID	Integer	The ID of the contribution
User ID	Integer	The user ID of the one who makes the contribution
Module ID	Integer	The ID of the module that the contribution belongs to
Create Time	String	The time when the contribution is made
Title	Varchar	The title of the contribution displayed

Collection Operation

Store information of each collection done

Field Name	Data Type	Description
Collection ID	Integer	The ID of the collection operation
Collection Type	Varchar	The type of the item collected: instruction, resource, discussion, or comment
Item ID	Integer	The ID of the item collected
User ID	Integer	The user ID of the one who collects
Module ID	Integer	The ID of the module that the collection belongs to
Create Time	String	The time when the collection is made
Title	String	The title of the collection displayed

Test Plan

Sr.No.	Success Criteria's	Nature of Test	Expected Result
1	There should be a homepage, where users can browse modules (cover pages) of different learning communities. The homepage should have a drawer navigating users to other functional pages.	1. Unit Test: Scroll the GUI and click on each of the modules and the floating button; slide aside to pull out the drawer and click on widgets in the drawer. 2. Validation Test: Click on the drawer's menu when the user is currently logged in and not logged in.	1. The user is directed to the About Us and different modules by clicking on them. By clicking the floating button, user will go to the module-creation GUI. The user can pull out the drawer and see a menu of customed information, including profile, contribution, collection, and preference. 2. The user can go to respective GUIs by clicking on items in the drawer's menu if the current status is logged in; if not, the user is directed to the Login GUI.
2	There should be a login page where users can either log in with existing accounts or sign up a new one: it validates whether the user's information inputted is matched with database storage	1. Unit Test: Enter the correct username/email and password that exist in the database. 2. Validation Test: Enter a mismatched pair of username/email	1. Successfully logged in with a toast message and is directed to the homepage. 2. Receive a toast message noticing "login failed"; or successfully signed up and logged in

	for logging-in, or for signing-up, if the user's email is existing in the database, and displays an error message.	and password; also try entering a pair of username/email and password that doesn't exist in the database.	with a toast message, directed to the homepage.
3	There should be a profile page showing personal information of users logged in.	Unit Test: Change the profile picture, username, and signature.	Information is successfully changed with a toast message and is displayed.
4	There should be an About-Us page introducing the origin, idea, and future aims of the application.	Unit Test: Scroll the page and click on the floating button.	The page is scrollable and clicking on the floating button will direct to the Contact GUI.
5	There should be a contact page showing the contact information of developer; it may also have a direct messaging form for users to send feedback within the application.	Unit Test: Scroll the page; input texts in the messaging box and click on the send button.	The page is scrollable, and the developer's email should receive the message from the user.
6	There should be a new module page where the user can send message forms to developers to propose new interest-based community modules.	Unit Test: Input message and send the proposal form by clicking the button	The developer receives email with full details of proposal form, along with the email address of the proposer for future reference

7	There should be a community homepage for each interest module/field, where users can locate the editorial, video tutorial, resource center, and discussion board.	Unit Test: Scroll the page to view different sectors; click on the instruction, resource center, discussion, and the floating button to see changes.	The page is scrollable, and the user is navigated to respective GUIs when clicking on different widgets.
8	There should be a resource center page displaying a scrollable list of resources of different types (files, links, and folders); folders may have their own pages displaying a list of items in it.	Unit Test: Scroll the page and click on items in the list.	The page is scrollable and the user is directed to a GUI showing detailed information about the resource once the selected resource item is clicked.
9	There should be a page for each discussion, where users can like and comment under the discussion.	1. Unit Test: Scroll and view the discussion; try to input textual comments and send by clicking the button. 2. Validation Test: Click the sending button without inputting anything.	1. The page is scrollable and the user can send comments that will be displayed under the discussion. 2. Receive a toast message showing “comment can’t be empty.”
10	There should be a collection page where users logged in can find shared resources and discussions they collected.	Unit Test: Scroll and view the list; click on each of the items displayed.	The page is scrollable and the user will be directed to respective GUIs when clicking on different items in the collection list.

11	There should be a contribution page exhibiting records of user sharing, editing, or supporting the community since the account is created.	Unit Test: Scroll and view the list; click on each of the items displayed.	The page is scrollable and the user will be directed to respective GUIs when clicking on different items in the contribution list.