

# Dwell System Requirements Specification

Tefillin Consulting

Leah Bassett, Lydia Browning, Daniel Gibson, Katy O'Malley, Rachel Teal

November 19, 2020

|                                   |           |
|-----------------------------------|-----------|
| <b>1-Introduction</b>             | <b>3</b>  |
| <b>2-User Roles</b>               | <b>3</b>  |
| <b>3-Systems Specifications</b>   | <b>4</b>  |
| 3.1 Non-Functional Requirements   | 4         |
| 3.2-Logical Data Model            | 5         |
| 3.3-Data Dictionary               | 6         |
| 3.4-Accounts                      | 13        |
| 3.5-Modules                       | 20        |
| 3.6-Activities                    | 27        |
| 3.7-Groups                        | 33        |
| 3.8-Friends                       | 42        |
| 3.9 - UX Overview                 | 49        |
| <b>4-Collaborating Systems</b>    | <b>50</b> |
| <b>5-Other Software</b>           | <b>50</b> |
| <b>6-Planned Schedule</b>         | <b>50</b> |
| <b>Appendix</b>                   | <b>51</b> |
| Appendix A- System Response Table | 51        |
| Appendix B- Wireframes            | 56        |

# 1-Introduction

This System Requirements Specification describes in detail the system requirements for a Bible verse memorization web application, Dwell. This system is meant to aid Christians in regularly memorizing scripture. The name of the application is a reference to Colossians 3:16:

“Let the message of Christ dwell among you richly as you teach and admonish one another with all wisdom through psalms, hymns, and songs from the Spirit, singing to God with gratitude in your hearts.”

Dwell will allow new users to choose modules, created and approved by other organizations, in order to further engage in memorization of the Bible. These modules follow a common theme, and the users are given different memorization activities to complete. The data from these memorization activities and the status for each user are stored. This data will help to develop the memorization techniques given to users to enhance their retention of the material. There are multiple rewards and incentives to engage people and to encourage them to continue memorizing scripture.

## 2-User Roles

Dwell will have a few user roles. First there are the businesses who are looking for a memorization tool. These businesses will be able to buy the tool and input their own data into it so that people can start memorizing.

Another type of user are those who create the modules. Modules are created by the company that has purchased usage of the tool. Modules have sorted content for people to memorize. In the case of our business, we will have related verses in a module for people to memorize. The modules will be submitted for approval from those in the company who have authority to approve modules. This way the person or organization running the memorization tool can keep track of what the tool is being used for and can make sure that the content is correct before allowing users to start learning a particular module.

Finally, the last type of user will be those who are looking to memorize data. For our purpose, people will use the tool to memorize scripture. These users can make an account that will keep track of what they have already memorized. The tool will keep track of how someone learns best, so that they get the best chance at retention of information. These users can choose modules that interest them and start memorization through activities. In addition to that users can connect with each other by adding other users as “friends”. More than that these users can collaborate on a module and learn the same material together.

## 3-Systems Specifications

### 3.1 Non-Functional Requirements

#### User Requirements

##### Usability

The system will have a user interface that is clean, modern, and intuitive to users from ages 10 to 99+. When a user accesses new features there will be popup information to guide the user through the use of the feature.

##### Accessibility

The system will be developed first for English speakers, but will be developed in a way conducive to adding multiple foreign languages. The site should be accessible to screen readers and other disability aids.

##### Availability

The system will be available as a full time website, with the occasional complete downtime for updates and maintenance.

##### Documentation and Training

The developers should supply design documents, user manuals, and other helpful documents for the usage and maintenance of this system. The training documents should be specific for the user role, i.e. regular users, administrators, or module creators.

#### System requirements

##### Performance

The system should be responsive and timely, no user should be waiting more than 10s for a process to be completed. The site should be able to support traffic of 500-1000 people by the end of the second year.

##### Capacity

The system will likely have a limited set of users for the first 2 years, 1000 regular users by the end of the second year would be a high goal to reach.

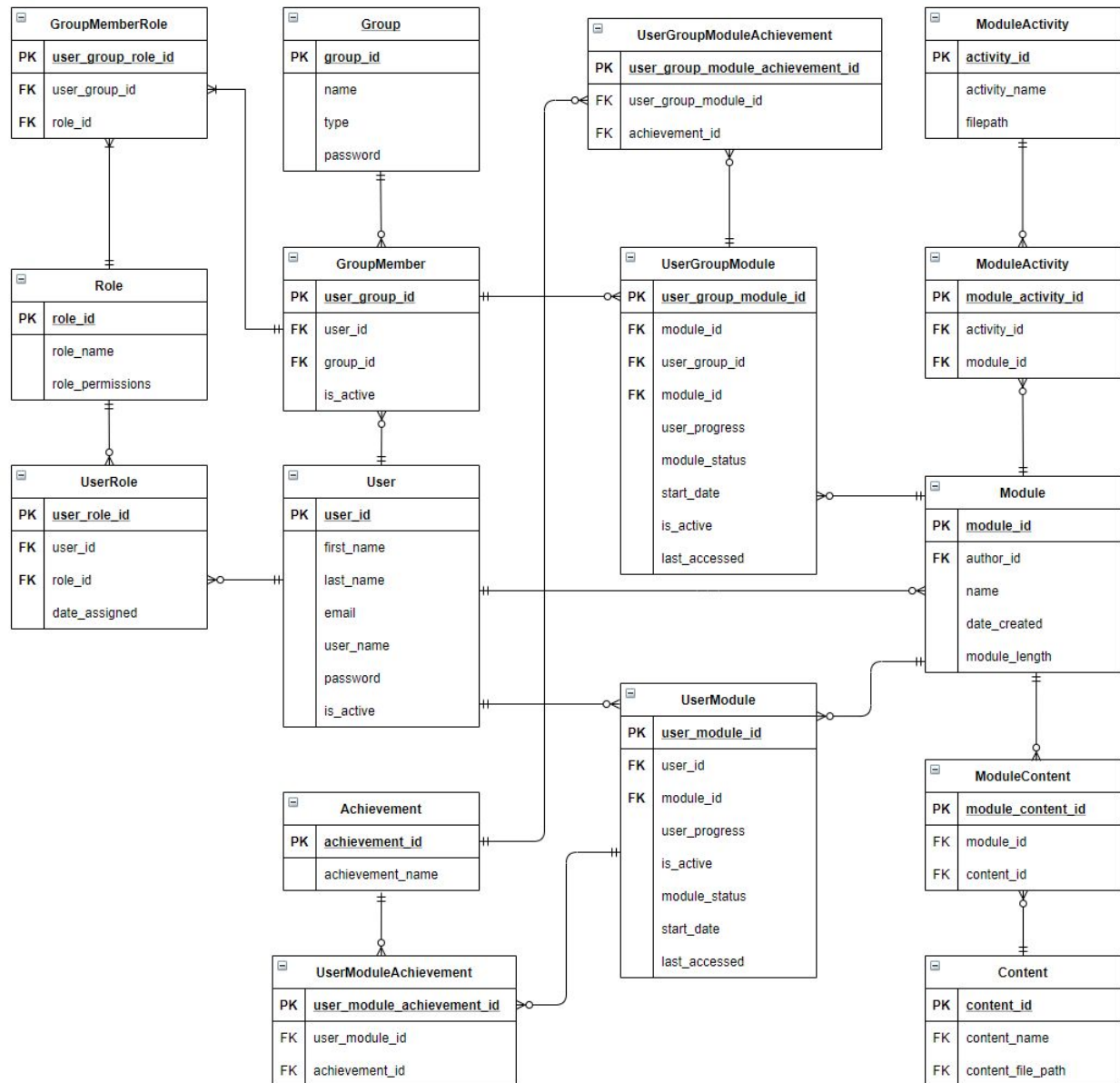
##### Security

User information should be kept private and secured. Regular users will not have access to any information but their own. Administrators will have access to the inner workings of the system.

##### Longevity

The system will remain in full use for 5-10 years, serving as an alternative to a mobile application that would be put into production about 3 years after it. As the user base grows, the website could be adapted to receive donations.

### 3.2-Logical Data Model



### 3.3-Data Dictionary

The Data Dictionary is a description of the entities, attributes and relationships that exist within the system. This is based off of the Logical Process Model is an exhaustive definition of the database used for the system.

**Achievement** - a reward a User receives for making progress within a Module.

#### Attributes

- achievement\_id (PK)
- achievement\_name

#### Relationships

- One-to-Many relationship to UserGroupModuleAchievement
- One-to-many relationship to UserModuleAchievement

**Activity** - a pre-created game or process used to memorize content within a Module

#### Attributes

- activity\_id (PK)
- activity\_name
- filepath - a location on the system server containing the software needed to run the activity

#### Relationships

- One-to-Many relationship to ModuleActivity

**Content** - the data to be memorized within a Module

#### Attributes

- content\_id (PK)
- content\_name
- content\_filepath - a location on the system server containing the data (text) to be memorized

#### Relationships

- One-to-Many relationship to ModuleContent

**Group** - a collection of two or more Users

**Attributes**

- group\_id (PK)
- name
- type - specification of whether the group is a Friend relationship (one-to-one) between users or a larger group

**Relationships**

- One-to-Many relationship to GroupMember

**GroupMember** - link between Group and User showing when a User is in a Group

**Attributes**

- (PK) user\_group\_id
- (FK) user\_id
- (FK) group\_id
- is\_active - boolean value showing whether the User is currently in the group

**Relationships**

- Many-to-One relationship with Group
- One-to-Many relationship with GroupMemberRole
- Many-to-One relationship with User
- One-to-Many relationship with UserGroupModule

**GroupMemberRole** - link between GroupMember and Role showing when a GroupMember is assigned to a specific Role

**Attributes**

- (PK) user\_group\_role\_id
- (FK) user\_group\_id
- (FK) role\_id

**Relationships**

- Many-to-One relationship with GroupMember
- Many-to-One relationship with Role

**Module** - a description of most of the data necessary to make up a Module

**Attributes**

- (PK) module\_id
- (FK) author\_id - the ID of the User who created the Module
- name
- date\_created
- module\_length - the amount of content to be memorized in the Module

**Relationships**

- One-to-Many relationship with ModuleActivity
- One-to-Many relationship with ModuleContent
- Many-to-One relationship with User
- One-to-Many relationship with UserModule
- One-to-Many relationship with UserGroupModule

**ModuleActivity** - link between Module and Activity showing which Activity is a part of which Module

**Attributes**

- (PK) module\_activity\_id
- (FK) activity\_id
- (FK) module\_id

**Relationships**

- Many-to-One relationship to Activity
- Many-to-One relationship to Module



**ModuleContent** - link between Module and Content showing which Content is a part of which Module

**Attributes**

- (PK) module\_content\_id
- (FK) module\_id
- (FK) content\_id

**Relationships**

- Many-to-One relationship to Content
- Many-to-One relationship to Module

**Role** - description of a User's function within the system

**Attributes**

- (PK) role\_id
- role\_name
- role\_permissions - description of the permissions within the system granted by the Role

**Relationships**

- One-to-Many relationship with GroupMemberRole
- One-to-Many relationship to UserRole

**User** - a person with an account in the system

#### **Attributes**

- (PK) user\_id
- first\_name
- last\_name
- email - the user's corresponding email address
- username - a title used to represent the user within the system and to be used for login
- password - a string of alphanumeric as well as special characters that are needed for the user to login to the system
- streak - the number of days in a row the user has completed an activity

#### **Relationships**

- One-to-Many relationship with GroupMember
- One-to-Many relationship with Module
- One-to-Many relationship with UserModule
- One-to-Many relationship with UserRole

**UserGroupModule** - link between GroupMember and Module meant to represent a specific User's progress within a Module being worked on by a larger Group

#### **Attributes**

- (PK) user\_group\_module\_id
- (FK) module\_id
- (FK) user\_group\_id
- (FK) module\_id
- user\_progress - a value representing how much of the Module the User has completed that is updated upon completion of an Activity
- module\_status - a boolean value to show whether the User is currently using the Module
- start\_date - the date the user started working on the Module
- last\_accessed - a date representing the last time the User made progress within the Module

#### **Relationships**

- Many-to-One relationship to GroupMember
- Many-to-One relationship to Module

- One-to-Many relationship to UserGroupModuleAchievement

**UserGroupModuleAchievement** - link between UserGroupModule and Achievement meant to represent an Achievement associated with a UserGroupModule

#### Attributes

- (PK) user\_group\_module\_achievement\_id
- (FK) user\_group\_module\_id
- (FK) achievement\_id

#### Relationships

- Many-to-One relationship to Achievement
- Many-to-One relationship to UserGroupModule

**UserModule** - link between User and Module meant to represent a User's progress within an individually-focused Module

#### Attributes

- (PK) user\_module\_id
- (FK) user\_id
- (FK) module\_id
- user\_progress - a value representing how much of the Module the User has completed that is updated upon completion of an Activity
- module\_status - a boolean value to show whether the User is currently using the Module
- start\_date - the date the user started working on the Module
- last\_accessed - a date representing the last time the User made progress within the Module

#### Relationships

- Many-to-One relationship to Module
- Many-to-One relationship to User
- Many-to-One relationship to UserModuleAchievement

**UserModuleAchievement** - link between UserModule and Achievement meant to represent an Achievement associated with a UserModule

**Attributes**

- (PK) user\_module\_achievement\_id
- (FK) user\_module\_id
- (FK) achievement\_id

**Relationships**

- Many-to-One relationship to Achievement
- Many-to-One relationship to UserModule

**UserRole** - link between User and Role meant to represent a Role given to a User

**Attributes**

- (PK) user\_role\_id
- (FK) user\_id
- (FK) role\_id
- date\_assigned - the date the User was assigned the associated Role

**Relationships**

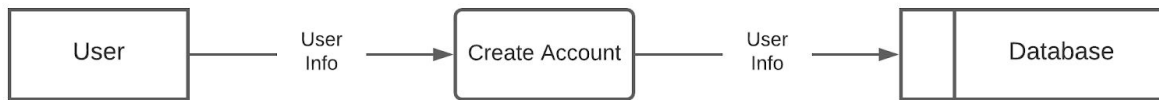
- Many-to-One relationship to Role
- Many-to-One relationship to User

### 3.4-Accounts

#### Associated Events

|          |        |                                       |  |  |                        |
|----------|--------|---------------------------------------|--|--|------------------------|
| Account1 | User   | Submission of account information     | Account created                          | Account  | Database               |
| Account2 | User   | Request to delete account             | Mark Account Inactive<br><br>Notify user | Account marked inactive<br><br>Account deletion success notification | Database<br><br>User   |
| Account3 | User   | Request to update account information | Account updated                          | Updated account  | Database               |
| Log1     | User   | Submit Username and Password          | Verify Account                           | Verified Account   | Notification to system |
| Log2     | System | Verified Account Notification         | Display User Dashboard                   | Displayed Dashboard  | User                   |
| Log3     | User   | Logs out                              | Account is logged out                    | Logged out user  | User                   |

Fig. 3.4.1 Account1 Data Flow Diagram



## Data Description

User Info = first\_name, last\_name, email, user\_name, password

## Process Specification - Create Account

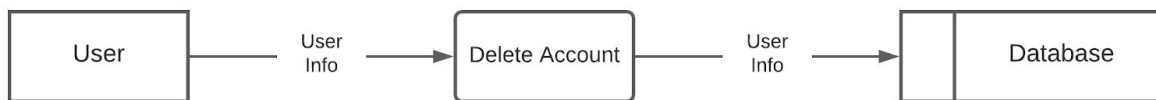
get User Info from User

if User Info valid:

create instance in User with User Info

|                        |   |                     |  |                                       |
|------------------------|---|---------------------|--|---------------------------------------|
| ID                     | Account1  |                     |  |                                       |
| Name                   | Submission of account information                   |                     |  |                                       |
| Primary Actor          | User  |                     |  |                                       |
| Other Actors           | None  |                     |  |                                       |
| Description            | Submitting information to create an account         |                     |  |                                       |
|                        | Actor Action  |                     | System Response                                  |                                       |
| Typical Event Flow     | 1. User submits account information                 |                     | 2. System verifies submitted account information |                                       |
|                        |   |                     | 3. System creates account                        |                                       |
|                        |   |                     |  |                                       |
| Alternative Event Flow | 1. User submits already created account information |                     | 2. System rejects submitted information          |                                       |
|                        |   |                     | 3. System sends User error message               |                                       |
|                        |   |                     |  |                                       |
|                        | Business Focus                                      |                     | System Focus                                     |                                       |
| Before Event           | Trigger   | Submits information | Precondition                                     | User wants account                    |
| After Event            | Conclusion  | Account created     | Postcondition                                    | Database creates new account for User |

Fig. 3.4.2 Account2 Data Flow Diagram



## Data Description

User Info = user\_id

## Process Specification - Delete Account

get User Info from User

using user\_id, update instance in User with user\_id to have is\_active value of false

|                        |   |                              |  |
|------------------------|---|------------------------------|--|
| ID                     | Account2  |                              |  |
| Name                   | Request to delete existing account                  |                              |  |
| Primary Actor          | User  |                              |  |
| Other Actors           | None  |                              |  |
| Description            | User requests to delete existing account            |                              |  |
|                        | Actor Action  |                              | System Response                            |
| Typical Event Flow     | 1. User submits a request to delete current account |                              | 2. System marks account as inactive        |
|                        |   |                              | 3. System notifies User of deleted account |
|                        |   |                              |  |
| Alternative Event Flow | 1. User submits a request to delete current account |                              | 2. System rejects request                  |
|                        |   |                              | 3. System sends User error message         |
|                        |   |                              |  |
|                        | Business Focus                                      |                              | System Focus                               |
| Before Event           | Trigger   | Request to delete an account | Precondition                               |
|                        |   |                              | Existing account                           |
| After Event            | Conclusion  | Account marked as inactive   | Postcondition                              |
|                        |   |                              | Account marked as inactive                 |

Fig. 3.4.3 Account3 Data Flow Diagram



#### Data Description

User Info = user\_id, first\_name, last\_name, email, user\_name, password

#### Process Specification - Update Account Information

get User Info from User

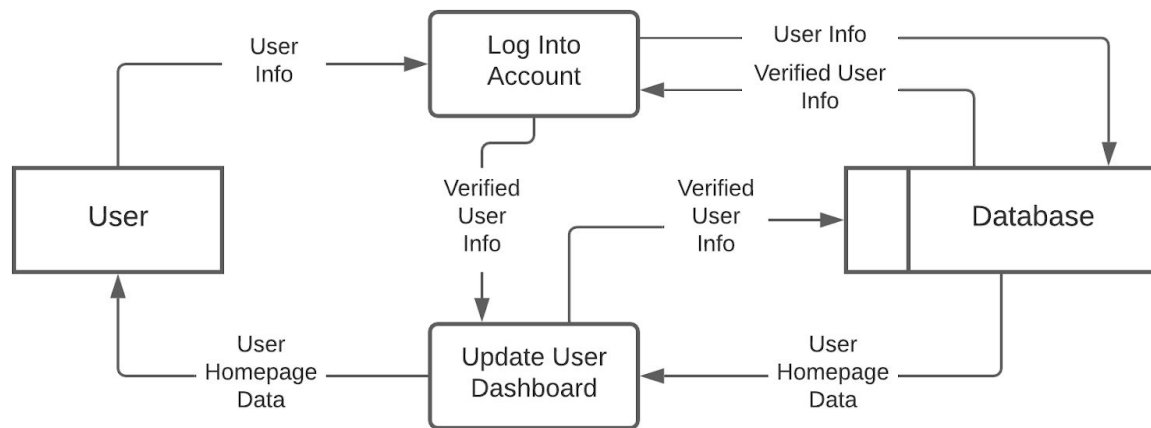
if User Info is valid:

using user\_id, update instance in User with values in User Info

|                        |  |                                       |  |                              |
|------------------------|--|---------------------------------------|--|------------------------------|
| ID                     | Account3                                       |                                       |  |                              |
| Name                   | Update account information                     |                                       |  |                              |
| Primary Actor          | User   |                                       |  |                              |
| Other Actors           | None   |                                       |  |                              |
| Description            | User requests to update account information    |                                       |  |                              |
|                        | Actor Action                                   |                                       | System Response  |                              |
| Typical Event Flow     | 1. User requests to update account information |                                       | 2. System verifies submitted updated account information |                              |
|                        |  |                                       | 3. System updates account                                |                              |
|                        |  |                                       |  |                              |
| Alternative Event Flow | 1. User requests to update account information |                                       | 2. System rejects submitted information                  |                              |
|                        |  |                                       | 3. System sends User error message                       |                              |
|                        |  |                                       |  |                              |
|                        | Business Focus                                 |                                       | System Focus   |                              |
| Before Event           | Trigger  | Request to update account information | Precondition   | User wants to update account |
| After Event            | Conclusion                                     | Account updated                       | Postcondition  | Account updated              |



Fig. 3.4.4 Login1-2 Data Flow Diagram



#### Data Description

User Info = email, password

Verified User Info = user\_id

User Homepage Data = { user\_module\_id }, { user\_group\_module\_id }

#### Process Specification - Log Into Account

get User Info from User

access Users using email

if password matches:

get Verified User Info from Database

send Verified User Info to Update User Dashboard process

#### Process Specification - Update User Dashboard

get Verified User Info from Log Into Account process

use Verified User Info to get User Homepage Data from Database

send User Homepage Data to User

|                        |   |                            |  |                     |
|------------------------|---|----------------------------|--|---------------------|
| ID                     | Log1  |                            |  |                     |
| Name                   | Login to the system                                 |                            |  |                     |
| Primary Actor          | User  |                            |  |                     |
| Other Actors           | None  |                            |  |                     |
| Description            | User submits email and password to login to account |                            |  |                     |
|                        | Actor Action  |                            | System Response                                  |                     |
| Typical Event Flow     | 1. User enters email and password to login          |                            | 2. System verifies submitted account information |                     |
|                        |   |                            | 3. System allows user to login                   |                     |
|                        |   |                            |  |                     |
| Alternative Event Flow | 1. User enters email and password to login          |                            | 2. System rejects submitted information          |                     |
|                        |   |                            | 3. System sends User error message               |                     |
|                        |   |                            |  |                     |
|                        | Business Focus                                      |                            | System Focus                                     |                     |
| Before Event           | Trigger   | Submit account information | Precondition                                     | User wants to login |
| After Event            | Conclusion  | Verify account             | Postcondition                                    | User is logged in   |

|                        |  |                              |               |  |
|------------------------|--|------------------------------|---------------|--|
| ID                     | Log 2                                      |                              |               |  |
| Name                   | Verify account                             |                              |               |  |
| Primary Actor          | User                                       |                              |               |  |
| Other Actors           | None                                       |                              |               |  |
| Description            | User submits information to verify account |                              |               |  |
|                        | Actor Action                               |                              |               | System Response                                  |
| Typical Event Flow     | 1. User enters information to verify       |                              |               | 2. System verifies submitted account information |
|                        |  |                              |               |  |
| Alternative Event Flow | 1. User enters information to verify       |                              |               | 2. System rejects submitted information          |
|                        |  |                              |               | 3. System sends User error message               |
|                        |  |                              |               |  |
|                        | Business Focus                             |                              |               | System Focus                                     |
| Before Event           | Trigger                                    | Verified account information | Precondition  | User wants to verify account                     |
| After Event            | Conclusion                                 | Display user dashboard       | Postcondition | User account verified                            |

Fig 3.4.5 Login3 Data Flow Diagram



## Data Description

User Info = user\_id

## Process Specification - Logout of Account

get User Info from User

access User with User Info

if user\_id valid:

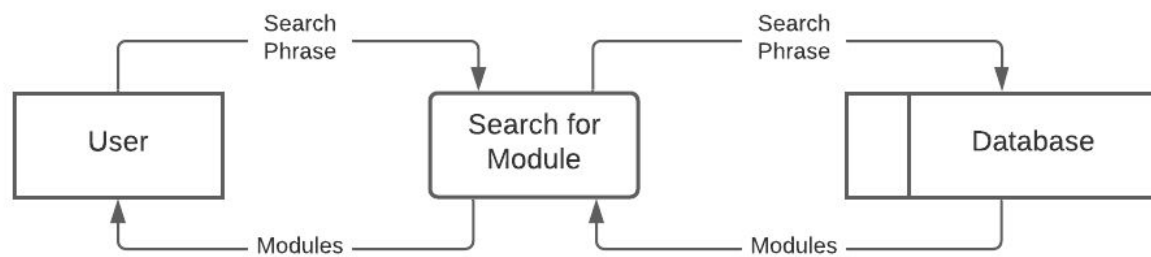
complete Logout process

|                        |  |                       |  |                                  |
|------------------------|--|-----------------------|--|----------------------------------|
| ID                     | Log3                                   |                       |  |                                  |
| Name                   | Log out                                |                       |  |                                  |
| Primary Actor          | User                                   |                       |  |                                  |
| Other Actors           | None                                   |                       |  |                                  |
| Description            | User logs out of account               |                       |  |                                  |
|                        | Actor Action                           |                       | System Response                              |                                  |
| Typical Event Flow     | 1. User requests to log out of account |                       | 2. System verifies request and logs user out |                                  |
|                        |  |                       |  |                                  |
| Alternative Event Flow | 1. User requests to log out of account |                       | 2. System rejects request to logout          |                                  |
|                        |  |                       | 3. System sends user error message           |                                  |
|                        |  |                       |  |                                  |
|                        | Business Focus                         |                       | System Focus                                 |                                  |
| Before Event           | Trigger                                | Logs out              | Precondition                                 | User wants to log out of account |
| After Event            | Conclusion                             | Account is logged out | Postcondition                                | User is logged out               |

### 3.5-Modules

|      |      |                               |   |  |                        |
|------|------|-------------------------------|---|--|------------------------|
| Mod1 | User | Search by keyword for Module  | Display Modules associated with keyword                                     | Displayed Modules                              | User                   |
| Mod2 | User | Select from Displayed Modules | Display Module Details  | Displayed Module and Details                   | User                   |
| Mod3 | User | Subscribe to Selected Module  | Ask User to start or save for later<br><br>User Module relationship created | Displayed Question<br><br>Created Relationship | User<br><br>Database   |
| Mod4 | User | User chooses to start Module  | Display Module Introduction   | Opened Module                                  | User                   |
| Mod5 | User | User chooses to save Module   | Return to search results  | Search results displayed                       | User                   |
| Mod6 | User | Unsubscribe from Module       | Confirm with User<br><br>Delete User Module relationship                    | User Confirmation<br><br>Deleted Relationship  | Server<br><br>Database |

Fig. 3.5.1 Mod1 Data Flow Diagram



### Data Description

Search Phrase = { name }, { author\_id },

Modules = { module\_id, first\_name, last\_name, user\_name, name, date\_created, module\_length, { content\_id, content\_name } }

### Process Specification - Search for Module

get Search Phrase from User

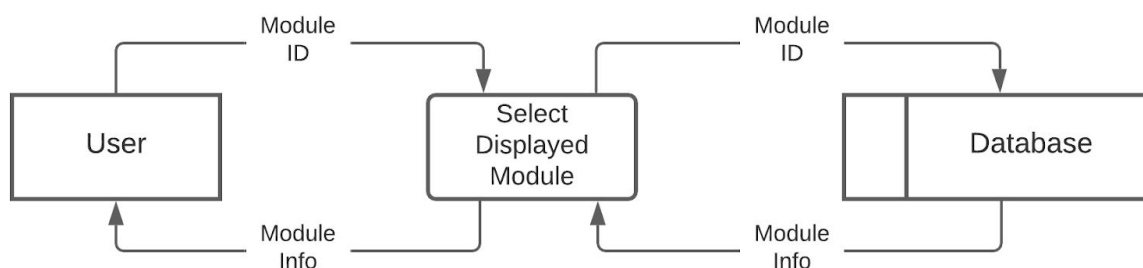
using Search Phrase, find instances in Module with matching name and/or author\_id

for each Module that is found:

send Module to User

|                        |                                    |   |   |
|------------------------|------------------------------------|---|---|
| ID                     | Mod1                               |   |   |
| Name                   | Search module                      |   |   |
| Primary Actor          | User                               |   |   |
| Other Actors           | None                               |   |   |
| Description            | User searches for module           |   |   |
|                        | Actor Action                       |   | System Response                                 |
| Typical Event Flow     | 1. User searches module by keyword |   | 2. System displays module matching the keyword  |
|                        |                                    |   |   |
| Alternative Event Flow | 1. User searches module by keyword |   | 2. System unable to display any related modules |
|                        |                                    |   |   |
|                        | Business Focus                     |   | System Focus                                    |
| Before Event           | Trigger                            | Search by keyword for Module            | Precondition                                    |
|                        |                                    |   | User wants to find modules to begin             |
| After Event            | Conclusion                         | Display Modules associated with keyword | Postcondition                                   |
|                        |                                    |   | System showcases modules for user to choose     |

Fig. 3.5.2 Mod2 Data Flow Diagram



### Data Description

Module ID = module\_id

Module Info = { module\_id, author\_id, name, date\_created, module\_length, content\_id, content\_name, content\_filepath }

### Process Specification - Select Displayed Module

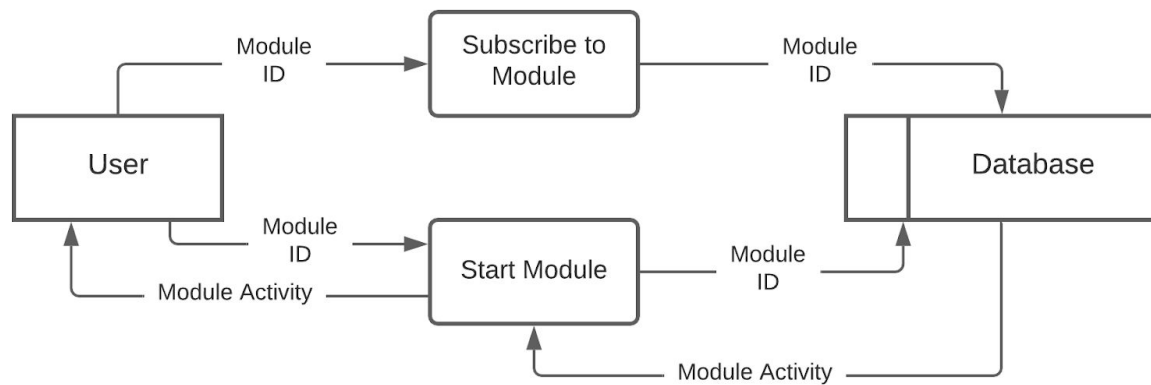
get Module ID from User

using Module ID, get Module Info from database if module\_id values match

send Module Info to User

|                        |   |                               |  |   |
|------------------------|---|-------------------------------|--|---|
| ID                     | Mod2  |                               |  |   |
| Name                   | Select module activity                                  |                               |  |   |
| Primary Actor          | User  |                               |  |   |
| Other Actors           | None  |                               |  |   |
| Description            | User selects module to determine if they want to use it |                               |  |   |
|                        | Actor Action  |                               | System Response                            |   |
| Typical Event Flow     | 1. User selects module                                  |                               | 2. System displays module details          |   |
|                        |   |                               |  |   |
| Alternative Event Flow | 1. User selects module                                  |                               | 2. System unable to display module details |   |
|                        |   |                               | 3. System sends user error message         |   |
|                        |   |                               |  |   |
|                        | Business Focus  |                               | System Focus                               |   |
| Before Event           | Trigger   | Select from Displayed Modules | Precondition                               | User selects module in order to look at details |
| After Event            | Conclusion  | Display Module Details        | Postcondition                              | System displays modules                         |

Fig. 3.5.3 Mod3-5 Data Flow Diagram



#### Data Description

Module ID = module\_id, user\_id

Module Activity = module\_id, { content\_id, content\_filepath }, { activity\_id, filepath }

#### Process Specification - Subscribe to Module

get Module ID from User

create instance of UserModule using user\_id, module\_id

#### Process Specification - Start Module

get Module ID from User

using module\_id, get Module Activity from Database

send Module Activity to User

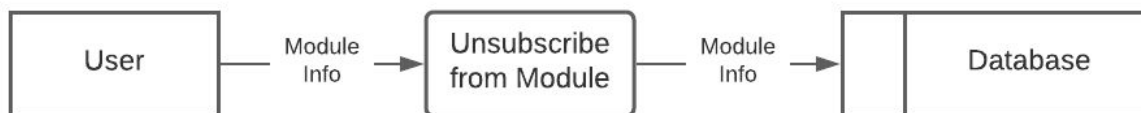
|                        |   |  |  |  |
|------------------------|---|--|--|--|
| ID                     | Mod3  |  |  |  |
| Name                   | Subscribe to Module   |  |  |  |
| Primary Actor          | User  |  |  |  |
| Other Actors           | None  |  |  |  |
| Description            | User subscribes to module, and they can either to start or save for later |  |  |  |
|                        | Actor Action  |  | System Response  |  |
| Typical Event Flow     | 1. User subscribes to module  |  | 2. System verifies subscription                              |  |
|                        |   |  | 3. System allows user to start or save for later             |  |
|                        |   |  |  |  |
| Alternative Event Flow | 1. User subscribes to module  |  | 2. System unable to verify module relationship with the user |  |
|                        |   |  | 3. System sends user error message                           |  |
|                        |   |  |  |  |
|                        | Business Focus  |  | System Focus   |  |
| Before Event           | Trigger   | Subscribe to Selected Module                       | Precondition   | User wants to subscribe                    |
| After Event            | Conclusion  | Ask user to start or save and relationship created | Postcondition  | System allows user to start or save module |

|                        |  |                              |  |                            |
|------------------------|--|------------------------------|--|----------------------------|
| ID                     | Mod4   |                              |  |                            |
| Name                   | Start module   |                              |  |                            |
| Primary Actor          | User   |                              |  |                            |
| Other Actors           | None   |                              |  |                            |
| Description            | After subscribing to module, user decides to start module right away |                              |  |                            |
|                        | Actor Action   |                              | System Response                                |                            |
| Typical Event Flow     | 1. User chooses to start the module                                  |                              | 2. System displays module introduction to user |                            |
|                        |  |                              |  |                            |
| Alternative Event Flow | 1. User chooses to start the module                                  |                              | 2. System unable to display module to user     |                            |
|                        |  |                              | 3. System sends user error message             |                            |
|                        |  |                              |  |                            |
|                        | Business Focus   |                              | System Focus                                   |                            |
| Before Event           | Trigger  | User chooses to start Module | Precondition                                   | User wants to start module |



|                        |  |                             |  |   |
|------------------------|--|-----------------------------|--|---|
| ID                     | Mod5   |                             |  |   |
| Name                   | Save module for later  |                             |  |   |
| Primary Actor          | User   |                             |  |   |
| Other Actors           | None   |                             |  |   |
| Description            | After subscribing to module, user decides to save module for later |                             |  |   |
|                        | Actor Action   |                             | System Response                                      |   |
| Typical Event Flow     | 1. User chooses to save module for later                           |                             | 2. System saves module and returns to search results |   |
|                        |  |                             |  |   |
| Alternative Event Flow | 1. User chooses to save module for later                           |                             | 2. System unable to save module                      |   |
|                        |  |                             | 3. System sends user error message                   |   |
|                        |  |                             |  |   |
|                        | Business Focus   |                             | System Focus   |   |
| Before Event           | Trigger  | User chooses to save Module | Precondition   | User wants to save module for later               |
| After Event            | Conclusion   | Return to search results    | Postcondition  | System saves module and returns to search results |

Fig. 3.5.4 Mod6 Data Flow Diagram



#### Data Description

Module Info = user\_id, module\_id, user\_module\_id

#### Process Specification - Unsubscribe from Module

get Module Info from User

using user\_module\_id, set module\_status to False

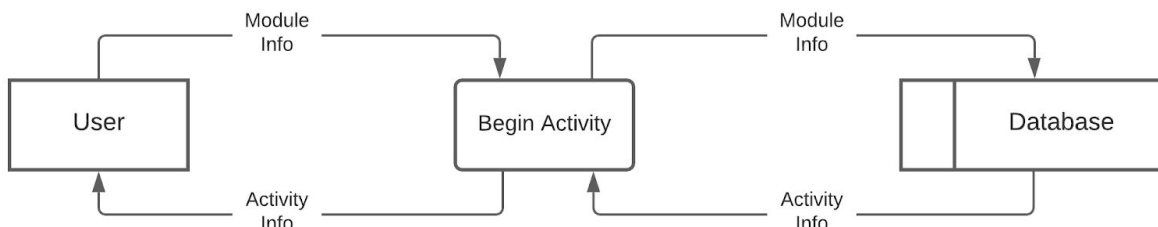
|                        |                                    |  |   |   |
|------------------------|------------------------------------|--|---|---|
| ID                     | Mod6                               |  |   |   |
| Name                   | Unsubscribe from module            |  |   |   |
| Primary Actor          | User                               |  |   |   |
| Other Actors           | None                               |  |   |   |
| Description            | User unsubscribes from a module    |  |   |   |
|                        | Actor Action                       |  | System Response   |   |
| Typical Event Flow     | 1. User unsubscribes from a module |  | 2. System allows user to unsubscribe after confirmation   |   |
|                        |                                    |  | 3. System deletes relationship                            |   |
|                        |                                    |  |   |   |
| Alternative Event Flow | 1. User unsubscribes from a module |  | 2. System unable to allow user to unsubscribe from module |   |
|                        |                                    |  | 3. System sends user error message                        |   |
|                        |                                    |  |   |   |
|                        | Business Focus                     |  | System Focus  |   |
| Before Event           | Trigger                            | Unsubscribe from Module                          | Precondition  | User wants to unsubscribe from module   |
| After Event            | Conclusion                         | Confirm with user and delete module subscription | Postcondition   | System deletes user-module relationship |

### 3.6-Activities

|          |      |  |  |  |                                  |
|----------|------|--|--|--|----------------------------------|
| Act1     | User | Clicking on Activity                                   | Begin Activity   | Active Activity session  | User                             |
| Act2*    | User | Answers Questions                                      | Check Answer   | Verified answer  | User                             |
| Act3     | User | Exit Activity  | Save Activity Progress   | Saved Activity State   | Database                         |
| Act4     | User | Complete Activity                                      | Update Module Progress<br><br>Save Activity Progress<br><br>If first activity of the day, update streak<br><br>Notify user of updated streak | Updated Module Progress<br><br>Saved Activity Progress<br><br>Updated Streak<br><br>Notification | Database<br><br><br><br><br>User |
| Rewards1 | User | Activity Completed or some achievement requirement met | Award User<br><br>Notify User  | Updated User Stat<br><br>Display Achievement   | Database<br><br>User             |

\*Act2 does not have a logical process model because its logic is contained in the activity process

Fig 3.6.1 Data Flow Diagram Act1



### Data Description

Module Info = { user\_module\_id }, { user\_group\_module\_id }

Activity Info = activity\_id, filepath

### Process Specification

get Module Info from User

if Module Info has user\_module\_id:

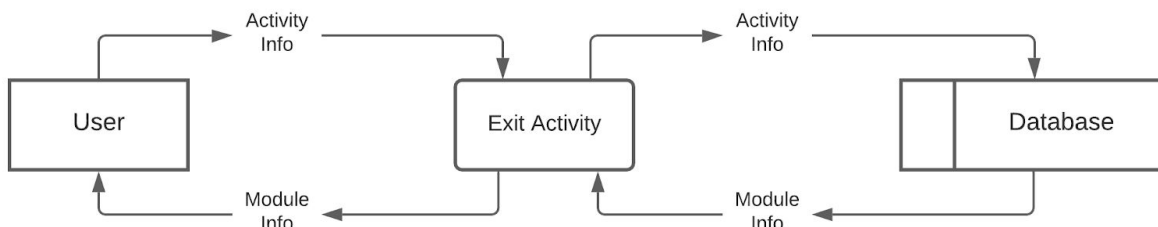
get module\_id from UserModule

get activity\_id and filepath from Module across ModuleActivity

|                        |  |                      |  |                              |
|------------------------|--|----------------------|--|------------------------------|
| ID                     | Act1   |                      |  |                              |
| Name                   | Working on activity                                    |                      |  |                              |
| Primary Actor          | User   |                      |  |                              |
| Other Actors           | None   |                      |  |                              |
| Description            | User clicks on activity to begin and complete activity |                      |  |                              |
|                        | Actor Action   |                      | System Response  |                              |
| Typical Event Flow     | 1. User clicks on activity                             |                      | 2. System verifies request and allows user to work on activity |                              |
|                        | 3. User begins working on activity                     |                      |  |                              |
|                        |  |                      |  |                              |
| Alternative Event Flow | 1. User clicks on activity                             |                      | 2. System rejects request                                      |                              |
|                        |  |                      | 3. System sends user error message                             |                              |
|                        |  |                      |  |                              |
|                        | Business Focus   |                      | System Focus   |                              |
| Before Event           | Trigger  | Clicking on activity | Precondition   | User wants to begin activity |
| After Event            | Conclusion   | Begin activity       | Postcondition  | User begins activity         |

|                        |                             |                          |   |  |
|------------------------|-----------------------------|--------------------------|---|--|
| ID                     | Act2                        |                          |   |  |
| Name                   | Activity work               |                          |   |  |
| Primary Actor          | User                        |                          |   |  |
| Other Actors           | None                        |                          |   |  |
| Description            | User is working on activity |                          |   |  |
|                        | Actor Action                |                          | System Response                                       |  |
| Typical Event Flow     | 1. User answers question    |                          | 2. System checks answer correctness and notifies user |  |
|                        |                             |                          | 3. System records answer                              |  |
|                        |                             |                          |   |  |
| Alternative Event Flow | 1. User answers question    |                          | 2. System rejects request to answer                   |  |
|                        |                             |                          | 3. System sends user error message                    |  |
|                        |                             |                          |   |  |
|                        | Business Focus              |                          | System Focus  |  |
| Before Event           | Trigger                     | Answers questions        | Precondition  | User begins answering activity questions |
| After Event            | Conclusion                  | Check and record answers | Postcondition   | System records response                  |

Fig 3.6.2 Data Flow Diagram Act3



#### Data Description

Activity Info = activity\_id, module\_activity\_id, user\_id

Module Info = module\_id, { user\_module\_id }, { user\_group\_module\_id }

#### Process Specification - Exit Activity

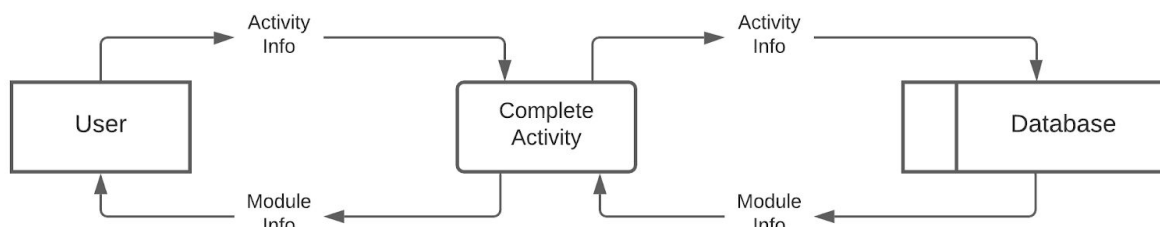
get Activity Info from User

get Module Info from Database using module\_activity\_id and user\_id

send Module Info to User

|                        |                        |                        |                                    |                       |
|------------------------|------------------------|------------------------|------------------------------------|-----------------------|
| ID                     | Act3                   |                        |                                    |                       |
| Name                   | Exit activity          |                        |                                    |                       |
| Primary Actor          | User                   |                        |                                    |                       |
| Other Actors           | None                   |                        |                                    |                       |
| Description            | User exits activity    |                        |                                    |                       |
|                        | Actor Action           |                        | System Response                    |                       |
| Typical Event Flow     | 1. User exits activity |                        | 2. System saves progress           |                       |
|                        |                        |                        |                                    |                       |
| Alternative Event Flow | 1. User exits activity |                        | 2. System unable to save progress  |                       |
|                        |                        |                        | 3. System sends user error message |                       |
|                        |                        |                        |                                    |                       |
|                        | Business Focus         |                        | System Focus                       |                       |
| Before Event           | Trigger                | Exit activity          | Precondition                       | User wants to logout  |
| After Event            | Conclusion             | Save activity progress | Postcondition                      | System saves progress |

Fig 3.6.3 Data Flow Diagram Act4



#### Data Description

Activity Info = activity\_id, module\_activity\_id, user\_id

Module Info = module\_id, { user\_module\_id }, { user\_group\_module\_id }, streak

#### Process Specification - Exit Activity

get Activity Info from User

get Module Info from Database using module\_activity\_id and user\_id

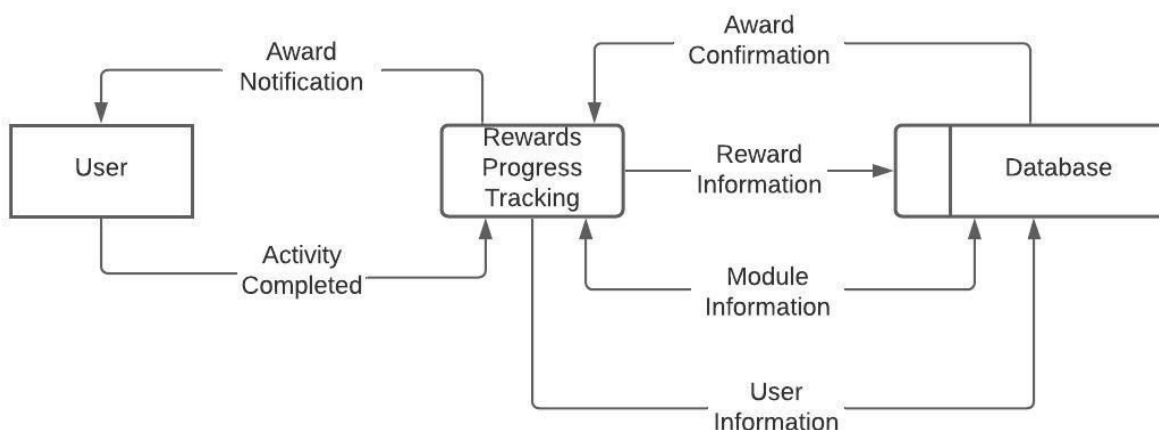
if streak value in User has not been updated today:

    increase streak value and add value to Module Info

send Module Info to User

|                        |                            |  |   |
|------------------------|----------------------------|--|---|
| ID                     | Act4                       |  |   |
| Name                   | Completed activity         |  |   |
| Primary Actor          | User                       |  |   |
| Other Actors           | None                       |  |   |
| Description            | User completes activity    |  |   |
|                        | Actor Action               |  | System Response                             |
| Typical Event Flow     | 1. User completes activity |  | 2. System saves and updates module progress |
|                        |                            |  | 3. System updates streak and notifies user  |
|                        |                            |  |   |
| Alternative Event Flow | 1. User completes activity |  | 2. System is unable to save                 |
|                        |                            |  | 3. System streak is unable to save          |
|                        |                            |  | 4. System sends user error message          |
|                        |                            |  |   |
|                        | Business Focus             |  | System Focus                                |
| Before Event           | Trigger                    | Update module progress and save            | Precondition                                |
| After Event            | Conclusion                 | Updated module progress and saved activity | Postcondition                               |
|                        |                            |  | User finishes module                        |
|                        |                            |  | System saves progress                       |

Fig. 3.6.4 Reward1 Data Flow Diagram

**Data Definition**

User Information = user\_id

Reward Information = achievement\_id

**Process Specification**

User completes an Activity and has met the requirements for an Achievement

The Reward Information and User Information are used to create a new

UserModuleAchievement Record

The Database confirms creation of the award

The User is notified of their Award

|                        |   |  |
|------------------------|---|--|
| ID                     | Rewards1  |  |
| Name                   | Rewards for completion  |  |
| Primary Actor          | User  |  |
| Other Actors           | None  |  |
| Description            | After user completes activity or requirement, they are rewarded |  |
|                        | Actor Action  | System Response  |
| Typical Event Flow     | 1. User completes activity or achieves requirement              | 2. System notifies user, and rewards user              |
|                        |   | 3. System updates user database                        |
|                        |   |  |
| Alternative Event Flow | 1. User completes activity or achieves requirement              | 2. System unable to be updated in order to reward user |
|                        |   | 3. System sends user error message                     |
|                        |   |  |



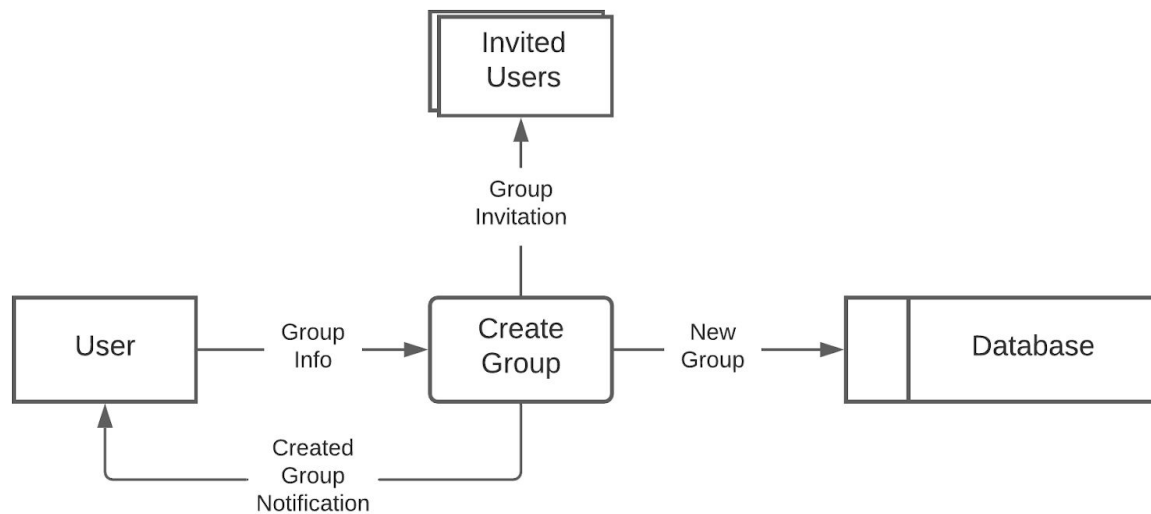
|              | Business Focus |   | System Focus  |   |
|--------------|----------------|---|---------------|---|
| Before Event | Trigger        | Activity completed or achievement requirement met | Precondition  | User has completed a module or activity |
| After Event  | Conclusion     | Award and notify user                             | Postcondition | System rewards user on completion       |

### 3.7-Groups

|        |      |   |  |  |   |
|--------|------|---|--|--|---|
| Group1 | User | User selects create group option at module signup | User presented with form to fill out information-admins, group name, members to invite | User presented with form   | User                                    |
| Group2 | User | Submits group signup form                         | Validate and create group  | Group created<br><br>Notify user and give them a password<br><br>Invite listed members | Database<br><br>User<br><br>Other users |
| Group3 | User | Accept/Decline group Invitation                   | Create relation on acceptance<br><br>Notify user on success                            | User joined group<br><br>User notified   | Database<br><br>User                    |
| Group4 | User | Chooses join                                      | User   | Displayed  | User                                    |

|        |      |                    |  |   |                                    |
|--------|------|--------------------|--|---|------------------------------------|
|        |      | group in dashboard | presented with password box                      | password box  |                                    |
| Group5 | User | Enters password    | Password checked for activation                  | Password is accepted or denied<br><br>User notified<br><br>On acceptance, user added to group | Server<br><br>User<br><br>Database |
| Group6 | User | User leaves group  | Records updated<br><br>Notification sent to user | User left<br><br>User notified  | Database<br><br>User               |

Fig. 3.7.1 Group1-2 Data Flow Diagram



#### Data Description

Group Info = name, { user\_id, role\_id }, { user\_id, role\_id }

New Group = group\_id, name, type, password. { user\_id, role\_id }

Created Group Notification = group\_id, name, password

Group Invitation = user\_id, role\_id, group\_id, password

#### Process Specification - Create Group

get group info from User

if group info is valid:

    create new instance of Group

    for each admin in group info:

        create new instance of GroupMember, GroupMemberRole

    for each user in invited users:

        create new instance of GroupMember, GroupMemberRole

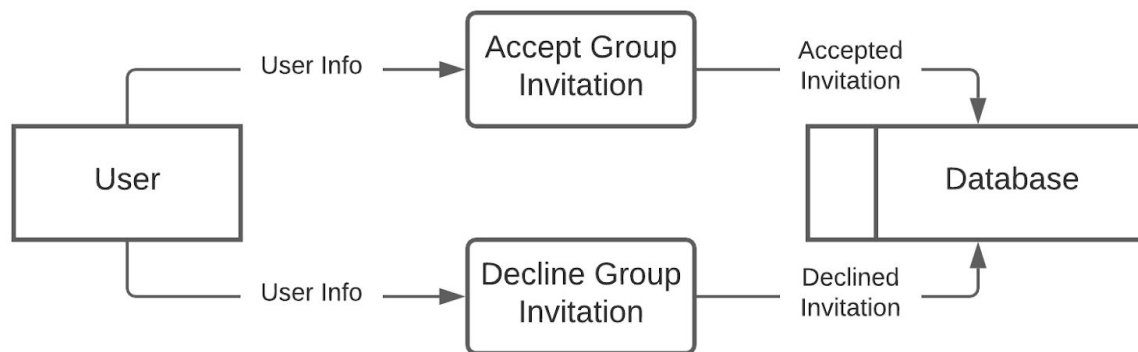
    return to user created group notification

|                        |   |   |  |  |
|------------------------|---|---|--|--|
| ID                     | Group1  |   |  |  |
| Name                   | User selects create group                         |   |  |  |
| Primary Actor          | User  |   |  |  |
| Other Actors           | None  |   |  |  |
| Description            | User selects create group option at module signup |   |  |  |
|                        | Actor Action                                      |   | System Response  |  |
| Typical Event Flow     | 1. User selects the create group option           |   | 2. System verifies request to create a group           |  |
|                        |   |   | 3. System displays a form for user to fill and submit  |  |
|                        |   |   |  |  |
| Alternative Event Flow | 1. User selects the create group option           |   | 2. System rejects request to create a group for module |  |
|                        |   |   | 3. System sends User error message                     |  |
|                        |   |   |  |  |
|                        | Business Focus                                    |   | System Focus   |  |
| Before Event           | Trigger   | User selects to create group option               | Precondition   | User wants to create group for module                                    |
| After Event            | Conclusion  | User is displayed with a form to create the group | Postcondition  | User is displayed with the form to fill-out and submit to create a group |

|                    |   |  |   |  |
|--------------------|---|--|---|--|
| ID                 | Group2  |  |   |  |
| Name               | Submits group sign-up form  |  |   |  |
| Primary Actor      | User  |  |   |  |
| Other Actors       | Other users   |  |   |  |
| Description        | User submits group sign-up form to get verified and to have a group created |  |   |  |
|                    |   |  | System Response   |  |
| Typical Event Flow | 1. User submits completed group form with accurate information              |  | 2. System verifies submitted group form                             |  |
|                    |   |  | 3. System create accounts   |  |
|                    |   |  | 3. System creates group   |  |
|                    |   |  | 4. System notifies user the group is created                        |  |
|                    |   |  | 5. System notifies other users that they have been added to a group |  |
|                    |   |  |   |  |

|                        |  |                              |  |  |
|------------------------|--|------------------------------|--|--|
| Alternative Event Flow | 1. User submits completed group form with inaccurate information |                              | 2. System rejects submitted group form |  |
|                        |  |                              | 3. System sends User error message     |  |
|                        |  |                              |  |  |
|                        | Business Focus   |                              | System Focus                           |  |
| Before Event           | Trigger  | Submits completed group form | Precondition                           | User submits a completed group form    |
| After Event            | Conclusion   | Group created                | Postcondition                          | Group created and other users notified |

Fig. 3.7.2 Group3 Data Flow Diagram



#### Data Description

User Info = user\_id, group\_id

Accepted Invitation = user\_id, group\_id, role\_id

Declined Invitation = user\_id, group\_id

#### Process Specification - Accept Group Invitation

get User Info from User

using User Info, delete from Database any instance in GroupMember and GroupMemberRole

using User Info, add instance in GroupMember and GroupMemberRole with Role of Member

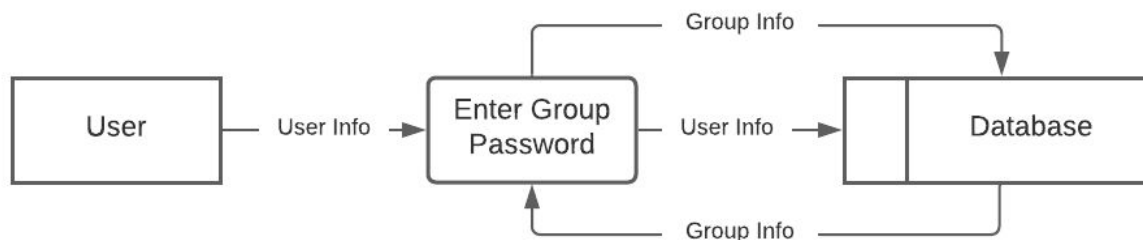
#### Process Specification - Decline Group Invitation

get User Info from User

using User Info, delete from Database any instance in GroupMember and GroupMemberRole

|                        |   |  |   |  |
|------------------------|---|--|---|--|
| ID                     | Group3  |  |   |  |
| Name                   | Accept or decline group invitation                            |  |   |  |
| Primary Actor          | User  |  |   |  |
| Other Actors           | Other user  |  |   |  |
| Description            | Accepting or declining the invitation to be a part of a group |  |   |  |
|                        | Actor Action  |  | System Response   |  |
| Typical Event Flow     | 1. User accepts group invitation                              |  | 2. System notifies other user that their invite has been submitted    |  |
|                        |   |  | 3. System confirms the addition to the group                          |  |
|                        | 1. User declines group invitation                             |  | 2. System notifies other user that their invitation has been declined |  |
|                        |   |  |   |  |
| Alternative Event Flow | 1. User accepts group invitation                              |  | 2. System experiences an error  |  |
|                        |   |  | 3. System sends User and other user an error message                  |  |
|                        | 1. User declines group invitation                             |  | 2. System experiences an error  |  |
|                        |   |  | 3. System fails to send error message                                 |  |
|                        |   |  |   |  |
|                        | Business Focus  |  | System Focus  |  |
| Before Event           | Trigger   | User accepts/declines invitation                     | Precondition  | User receives invitation to join a group       |
| After Event            | Conclusion  | User officially added to group or removed from group | Postcondition   | User is officially added or removed from group |

Fig. 3.7.3 Group4-5 Data Flow Diagram



#### Data Description

User Info = user\_id, password

Group Info = group\_id, password

#### Process Specification - Enter Group Password

get User Info from User

get Group Info from Database using password in User Info

if group exists with password:

send User Info, Group Info to Database

create Instance in GroupMember, GroupMemberRole using User Info, Group Info, with Role of Member

|                        |  |                        |   |                                 |
|------------------------|--|------------------------|---|---------------------------------|
| ID                     | Group4                                 |                        |   |                                 |
| Name                   | Chooses join group on dashboard        |                        |   |                                 |
| Primary Actor          | User                                   |                        |   |                                 |
| Other Actors           | None                                   |                        |   |                                 |
| Description            | User chooses to join group             |                        |   |                                 |
|                        |  | System Response        |   |                                 |
| Typical Event Flow     | 1. User chooses to join group          |                        | 2. System presents user with password box |                                 |
|                        | 3. User is presented with password box |                        |   |                                 |
|                        |  |                        |   |                                 |
| Alternative Event Flow | 1. User chooses to join group          |                        | 2. System cannot verify group             |                                 |
|                        |  |                        | 3. System sends User error message        |                                 |
|                        |  |                        |   |                                 |
|                        | Business Focus                         |                        | System Focus                              |                                 |
| Before Event           | Trigger                                | User joins group       | Precondition                              | User wants to join group        |
| After Event            | Conclusion                             | User is presented with | Postcondition                             | User is presented with password |

|  |  |              |  |     |
|--|--|--------------|--|-----|
|  |  | password box |  | box |
|--|--|--------------|--|-----|

|                        |   |                                 |                                       |                            |
|------------------------|---|---------------------------------|---------------------------------------|----------------------------|
| ID                     | Group5                                    |                                 |                                       |                            |
| Name                   | Enters password                           |                                 |                                       |                            |
| Primary Actor          | User                                      |                                 |                                       |                            |
| Other Actors           | Server                                    |                                 |                                       |                            |
| Description            | User enters password to join a group      |                                 |                                       |                            |
|                        |   |                                 | System Response                       |                            |
| Typical Event Flow     | 1. User enters and submits password       |                                 | 2. System verifies submitted password |                            |
|                        |   |                                 | 3. System adds user to group          |                            |
|                        |   |                                 |                                       |                            |
| Alternative Event Flow | 1. User enters and submits wrong password |                                 | 2. System rejects password            |                            |
|                        |   |                                 | 3. System sends User error message    |                            |
|                        |   |                                 |                                       |                            |
|                        | Business Focus                            |                                 | System Focus                          |                            |
| Before Event           | Trigger                                   | User enters password            | Precondition                          | User enters password       |
| After Event            | Conclusion                                | Group member verified and added | Postcondition                         | User is added to the group |



Fig. 3.7.4 Group6 Data Flow Diagram



## Data Description

User Group Info = user\_id, group\_id

## Process Specification - Leave Group

get User Group Info from User

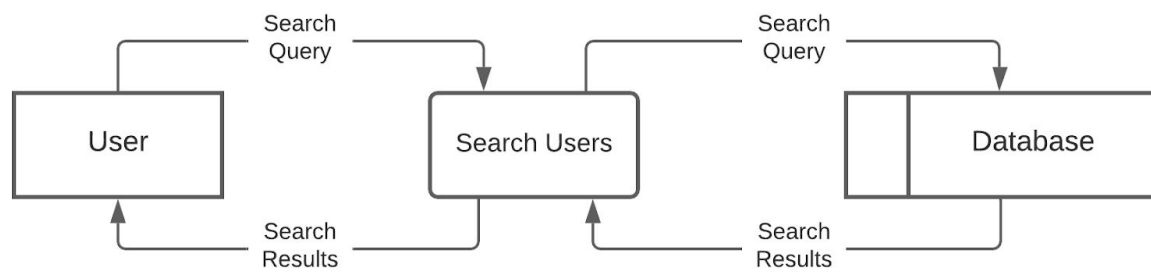
update Instance in GroupMember using user\_id, group\_id and set is\_active to false

|                        |  |                     |   |                           |
|------------------------|--|---------------------|---|---------------------------|
| ID                     | Group6                                 |                     |   |                           |
| Name                   | User leaves group                      |                     |   |                           |
| Primary Actor          | User                                   |                     |   |                           |
| Other Actors           | None                                   |                     |   |                           |
| Description            | User leaves group and updates database |                     |   |                           |
|                        |  |                     | System Response                               |                           |
| Typical Event Flow     | 1. User leaves group                   |                     | 2. System removes member from group           |                           |
|                        |  |                     |   |                           |
| Alternative Event Flow | 1. User leaves group                   |                     | 2. System experienced an error                |                           |
|                        |  |                     | 3. System is unable to remove user from group |                           |
|                        |  |                     |   |                           |
|                        | Business Focus                         |                     | System Focus                                  |                           |
| Before Event           | Trigger                                | User leaves group   | Precondition                                  | User wants to leave group |
| After Event            | Conclusion                             | User has left group | Postcondition                                 | User has left group       |

### 3.8-Friends

|         |      |                                      |  |  |                             |
|---------|------|--------------------------------------|--|--|-----------------------------|
| Friend1 | User | User searches for friend's username  | Present user with search results with search results   | Displayed search results                           | User                        |
| Friend2 | User | Select user from display results     | User information shown   | Displayed user information                         | User                        |
| Friend3 | User | Send friendship request              | Notify potential friend  | Notification sent                                  | Potential friend            |
| Friend4 | User | Accept or decline friendship request | <p>Friendship confirmed or rejected - notify original requester</p> <p>If confirmed, update database records</p> | <p>Friendship</p> <p>Updated records</p>           | <p>User</p> <p>Database</p> |
| Friend5 | User | End Friendship                       | <p>Mark friendship inactive</p> <p>Notify user</p>   | <p>Friendship Deactivated</p> <p>User notified</p> | <p>Database</p> <p>User</p> |

Fig. 3.8.1 Friend1 Data Flow Diagram



### Data Description

Search Query = username

Search Results = { user\_id, username }

### Process Specification - Search Users

get Search Query from User

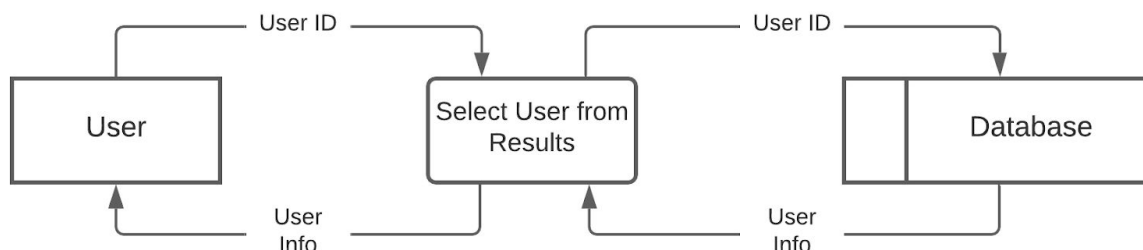
using username, get Search Results from Database where username matches

send Search Results to User

|                        |                                       |                                     |  |
|------------------------|---------------------------------------|-------------------------------------|--|
| ID                     | Friend1                               |                                     |  |
| Name                   | Search for friend by username         |                                     |  |
| Primary Actor          | User                                  |                                     |  |
| Other Actors           | Username that matches search          |                                     |  |
| Description            | User searches for friend by username  |                                     |  |
|                        | Actor Action                          |                                     | System Response  |
| Typical Event Flow     | 1. User searches for another username |                                     | 2. System displays usernames that match the search results |
|                        |                                       |                                     | 3. System updates user database                            |
|                        |                                       |                                     |  |
| Alternative Event Flow | 1. User searches for another username |                                     | 2. System unable to find matches to the search             |
|                        |                                       |                                     | 3. System sends user error message                         |
|                        |                                       |                                     |  |
|                        | Business Focus                        |                                     | System Focus   |
| Before Event           | Trigger                               | User searches for friend's username | Precondition   |
|                        |                                       |                                     | User has completed a module or activity                    |

|             |            |  |               |                                   |
|-------------|------------|--|---------------|-----------------------------------|
| After Event | Conclusion | Present user with search results with search results | Postcondition | System rewards user on completion |
|-------------|------------|--|---------------|-----------------------------------|

Fig. 3.8.2 Friend2 Data Flow Diagram



#### Data Description

User ID = user\_id, username

User Info = user\_id, first\_name, last\_name, user\_name, is\_active

#### Process Specification - Select User from Results

get User ID from User

get User Info from Database using user\_id

if is\_active is true:

send User Info to User

else:

send error: User Info not found to User

| ID                     | Friends2   |   |
|------------------------|--|---|
| Name                   | User selection   |   |
| Primary Actor          | User   |   |
| Other Actors           | Other users  |   |
| Description            | After searching for username and system retrieving matching results, the user is able to choose and display other users' information |   |
|                        | Actor Action   | System Response                                   |
| Typical Event Flow     | 1. User selects one of the username matches  | 2. System displays user with the user information |
|                        |  |   |
| Alternative Event Flow | 1. User selects one of the username matches  | 2. System unable to display user information      |

|              |                |                                  |  |
|--------------|----------------|----------------------------------|--|
|              |                |                                  | 3. System sends user error message               |
|              |                |                                  |  |
|              | Business Focus |                                  | System Focus                                     |
| Before Event | Trigger        | Select user from display results | Precondition<br>User given matching user results |
| After Event  | Conclusion     | User information shown           | Postcondition<br>System displays user details    |

Fig. 3.8.3 Friend3 Data Flow Diagram



## Data Description

Friend Request Info = { user\_id }, role\_id

## Process Specification - Send Friend Request

from User get Friend Request Info

create instance in Group with type of Friendship, save group\_id

for each user\_id in Friend Request Info:

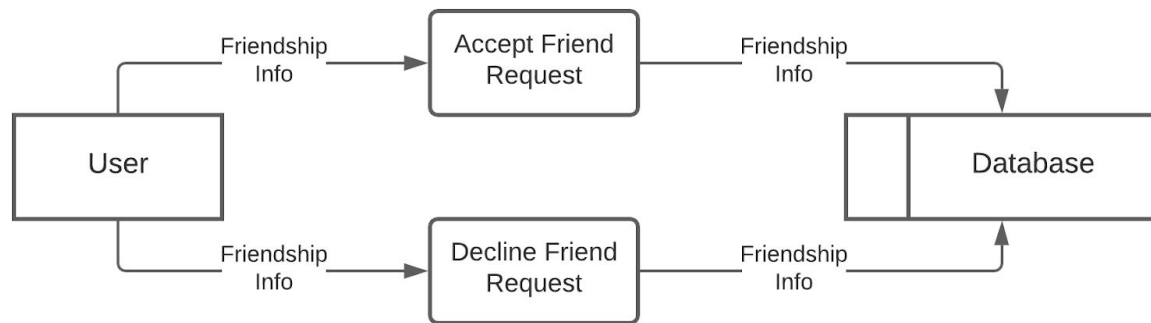
create instance in GroupMember using user\_id and group\_id, save user\_group\_id

create instance in GroupMemberRole using user\_group\_id and role\_id

|                        |                                  |   |
|------------------------|----------------------------------|---|
| ID                     | Friends3                         |   |
| Name                   | Friendship request               |   |
| Primary Actor          | User                             |   |
| Other Actors           | Other users                      |   |
| Description            | User requests friendship         |   |
|                        | Actor Action                     | System Response                         |
| Typical Event Flow     | 1. User sends friendship request | 2. System notifies user of request      |
|                        |                                  |   |
| Alternative Event Flow | 1. User sends friendship request | 2. System unable to send friend request |
|                        |                                  | 3. System sends user error message      |
|                        |                                  |   |
|                        | Business Focus                   | System Focus                            |

|              |            |                         |               |                                       |
|--------------|------------|-------------------------|---------------|---------------------------------------|
| Before Event | Trigger    | Send friendship request | Precondition  | User is displayed user details        |
| After Event  | Conclusion | Notify potential friend | Postcondition | System sends notification for request |

Fig. 3.8.4 Friend4 Data Flow Diagram



#### Data Description

Friendship Info = { user\_id }, group\_id, role\_id

#### Process Specification - Accept Friend Request

get Friendship Info from User

for user\_id in Friendship Info:

delete instance from GroupMemberRole using user\_group\_id

delete instance from GroupMember using user\_group\_id

create instance of GroupMember using user\_id and group\_id, save user\_group\_id

create instance of GroupMemberRole using user\_group\_id and role\_id

#### Process Specification - Decline Friend Request

get Friendship Info from User

for user\_id in Friendship Info:

delete instance from GroupMemberRole using user\_group\_id

delete instance from GroupMember using user\_group\_id

|               |                                |
|---------------|--------------------------------|
| ID            | Friends4                       |
| Name          | Accepts or declines friendship |
| Primary Actor | User                           |
| Other Actors  | None                           |

|                        |   |                                      |   |   |
|------------------------|---|--------------------------------------|---|---|
| Description            | After user completes activity or requirement, they are rewarded |                                      |   |   |
|                        | Actor Action  |                                      | System Response                               |   |
| Typical Event Flow     | 1. Requested user accepts or declines friendship                |                                      | 2. System updates users relationship          |   |
|                        |   |                                      |   |   |
| Alternative Event Flow | 1. User completes activity or achieves requirement              |                                      | 2. System unable to update users relationship |   |
|                        |   |                                      | 3. System sends user error message            |   |
|                        |   |                                      |   |   |
|                        | Business Focus  |                                      | System Focus                                  |   |
| Before Event           | Trigger   | Accept or decline friendship request | Precondition                                  | User has completed a module or activity |
| After Event            | Conclusion  | Friendship confirmed or rejected     | Postcondition                                 | System rewards user on completion       |

Fig. 3.8.5 Friend5 Data Flow Diagram



## Data Description

Friendship Info = group\_id

## Process Specification

get Friendship Info from User

for GroupMember with group\_id:

set is\_active in GroupMember to false

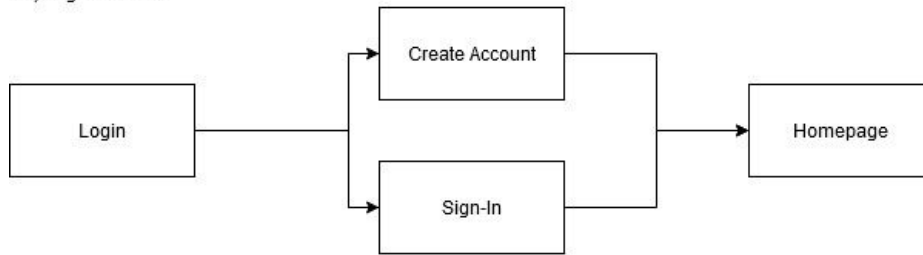
|                    |  |  |
|--------------------|--|--|
| ID                 | Friends5                               |  |
| Name               | End friendship                         |  |
| Primary Actor      | User                                   |  |
| Other Actors       | Other users                            |  |
| Description        | User ends friendship with another user |  |
|                    | Actor Action                           | System Response  |
| Typical Event Flow | 1. User end friendship                 | 2. System notifies user and marks relationship as inactive |

|                        |                        |  |  |                              |
|------------------------|------------------------|--|--|------------------------------|
| Alternative Event Flow |                        |  | 3. System updates user database                  |                              |
|                        |                        |  |  |                              |
|                        | 1. User end friendship |  | 2. System unable to be updated user relationship |                              |
|                        |                        |  | 3. System sends user error message               |                              |
|                        | Business Focus         |  | System Focus                                     |                              |
| Before Event           | Trigger                | End Friendship                           | Precondition                                     | User wants to end friendship |
| After Event            | Conclusion             | Mark friendship inactive and notify user | Postcondition                                    | System ends friendship       |

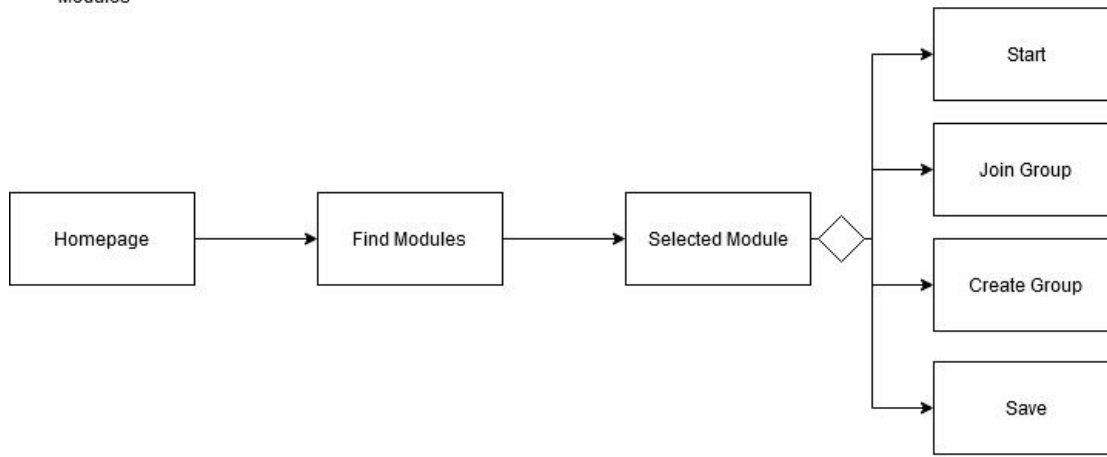


## 3.9 - UX Overview

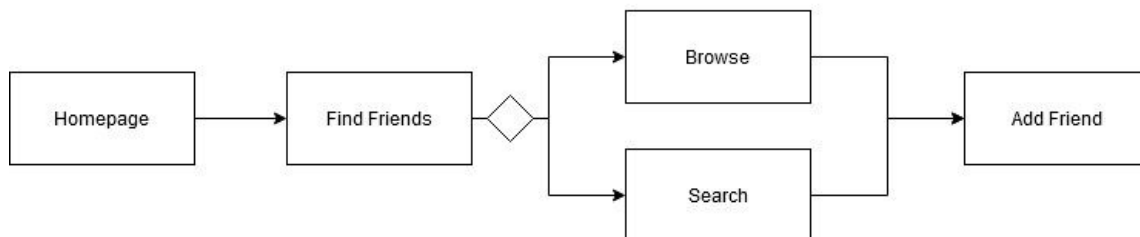
### 1.) Login-Process



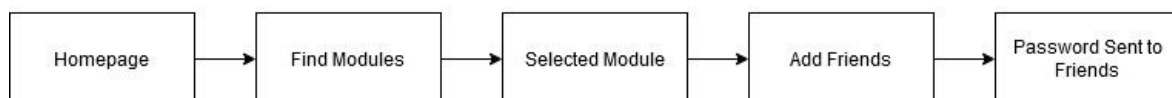
### 2.) Homepage to Modules



### 3.) Homepage to Friends



### 4.) Homepage to Create Group



## 4-Collaborating Systems

The goal of Dwell is to provide a free Bible memorization application. However, we would like to offer people a few different ways to support the cause. These could be things like prayer, submitting modules to be approved, and possible financial support. Dwell must be able to accept and securely store information for users making donations. To complete these financial interactions, the system's records must be available to authorized accountants. If the accountants use their own computerized financial system, it would be beneficial for Dwell to automatically pass information about the transaction to it. Depending on the financial system, this could be done in a daily generated report that the system can parse or in a direct data transfer connection.

## 5-Other Software

Dwell will make use of a relational database with a SQL interface. We believe that the open source MySQL server MariaDB would be a good option for this, as it is free and reliable.

Should the customers prefer to use a web hosting platform or a cloud database, we would be happy to evaluate options for you. Our system design is flexible and would be able to take advantage of the possibilities of cloud services. We are interested in creating a system that you are excited about and can easily use.

## 6-Planned Schedule

### 12 Week Plan

| Milestone                              | Hours |
|--|-------|
| Finalize and Set up Database Structure | 18    |
| Secure Accounts System                 | 16    |
| Store and Access Modules and Data      | 32    |
| Memorization Activities                | 90    |
| Memorization Plan                      | 30    |
| User Analytics/Progress Tracker        | 100   |
| Rewards/Motivational System            | 30    |
| Social Elements/Groups                 | 50    |
| Aesthetics                             | 20    |

# Appendix

## Appendix A- System Response Table

| ID       | Source | Trigger                               | Response                                 | Major Output   | Ext. Destination       |
|----------|--------|---------------------------------------|--|--|------------------------|
| Account1 | User   | Submission of account information     | Account created                          | Account  | Database               |
| Account2 | User   | Request to delete account             | Mark Account Inactive<br><br>Notify user | Account marked inactive<br><br>Account deletion success notification | Database<br><br>User   |
| Account3 | User   | Request to update account information | Account updated                          | Updated account  | Database               |
| Log1     | User   | Submit Username and Password          | Verify Account                           | Verified Account   | Notification to system |
| Log2     | System | Verified Account Notification         | Display User Dashboard                   | Displayed Dashboard  | User                   |
| Log3     | User   | Logs out                              | Account is logged out                    | Logged out user  | User                   |
| Act1     | User   | Clicking on Activity                  | Begin Activity                           | Active Activity session  | User                   |
| Act2     | User   | Answers                               | Check                                    | Verified   | User                   |

|      |      |                               |  |  |                              |
|------|------|-------------------------------|--|--|------------------------------|
|      |      | Questions                     | Answer<br><br>Record Answer  | answer<br><br>Data   | Database                     |
| Act3 | User | Exit Activity                 | Save Activity Progress   | Saved Activity State   | Database                     |
| Act4 | User | Complete Activity             | Update Module Progress<br><br>Save Activity Progress<br><br>If first activity of the day, update streak<br><br>Notify user of updated streak | Updated Module Progress<br><br>Saved Activity Progress<br><br>Updated Streak<br><br>Notification | Database<br><br><br><br>User |
| Mod1 | User | Search by keyword for Module  | Display Modules associated with keyword  | Displayed Modules  | User                         |
| Mod2 | User | Select from Displayed Modules | Display Module Details   | Displayed Module and Details   | User                         |
| Mod3 | User | Subscribe to Selected Module  | Ask User to start or save for later<br><br>User Module relationship created  | Displayed Question<br><br>Created Relationship   | User<br><br>Database         |

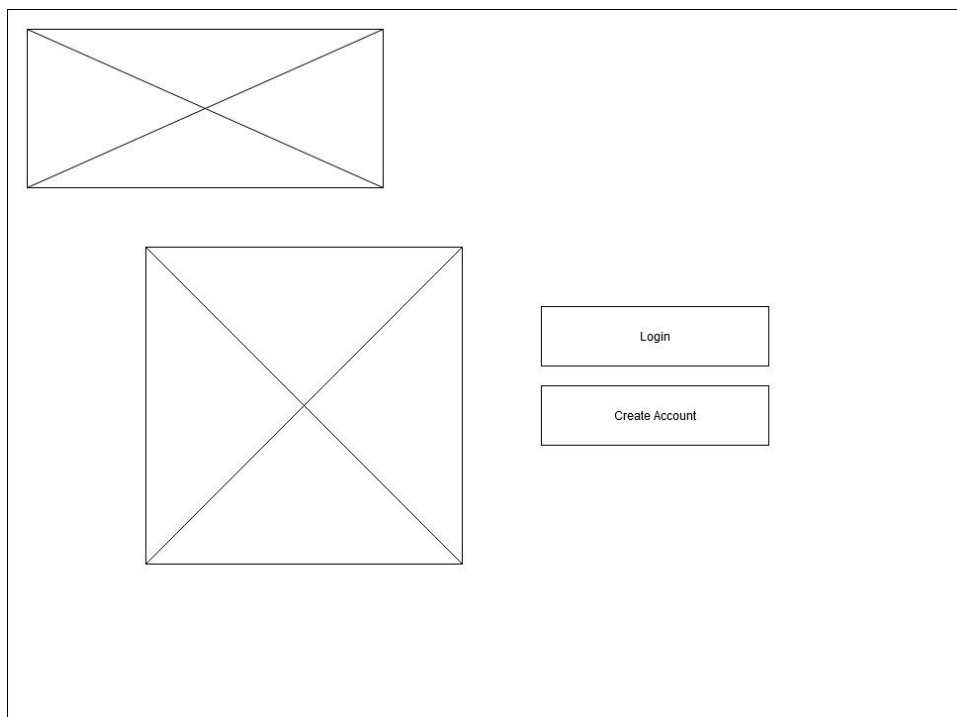
|          |      |  |  |  |                        |
|----------|------|--|--|--|------------------------|
| Mod4     | User | User chooses to start Module                           | Display Module Introduction                                      | Opened Module  | User                   |
| Mod5     | User | User chooses to save Module                            | Return to search results<br><br>User module relationship created | Search results displayed<br><br>Created relationship | User<br><br>Database   |
| Mod6     | User | Unsubscribe from Module                                | Confirm with User<br><br>Delete User Module relationship         | User Confirmation<br><br>Deleted Relationship        | Server<br><br>Database |
| Rewards1 | User | Activity Completed or some achievement requirement met | Award User<br><br>Notify User                                    | Updated User Stat<br><br>Display Achievement         | Database<br><br>User   |
| Friend1  | User | User searches for friend's username                    | Present user with search results with search results             | Displayed search results                             | User                   |
| Friend2  | User | Select user from display results                       | User information shown   | Displayed user information                           | User                   |
| Friend3  | User | Send friendship request                                | Notify potential friend  | Notification sent                                    | Potential friend       |
| Friend4  | User | Accept or decline friendship                           | Friendship confirmed or rejected -                               | Friendship   | User                   |

|         |      |   |  |  |   |
|---------|------|---|--|--|---|
|         |      | request   | notify original requester<br><br>If confirmed, update database records                 | Updated records  | Database                                |
| Friend5 | User | End Friendship                                    | Mark friendship inactive<br><br>Notify user  | Friendship Deactivated<br><br>User notified  | Database<br><br>User                    |
| Group1  | User | User selects create group option at module signup | User presented with form to fill out information-admins, group name, members to invite | User presented with form   | User                                    |
| Group2  | User | Submits group signup form                         | Validate and create group  | Group created<br><br>Notify user and give them a password<br><br>Invite listed members | Database<br><br>User<br><br>Other users |
| Group3  | User | Accept/Decline group Invitation                   | Create relation on acceptance<br><br>Notify user on success                            | User joined group<br><br>User notified   | Database<br><br>User                    |

|        |      |                                 |  |   |                                    |
|--------|------|---------------------------------|--|---|------------------------------------|
| Group4 | User | Chooses join group in dashboard | User presented with password box                 | Displayed password box  | User                               |
| Group5 | User | Enters password                 | Password checked for activation                  | Password is accepted or denied<br><br>User notified<br><br>On acceptance, user added to group | Server<br><br>User<br><br>Database |
| Group6 | User | User leaves group               | Records updated<br><br>Notification sent to user | User left<br><br>User notified  | Database<br><br>User               |

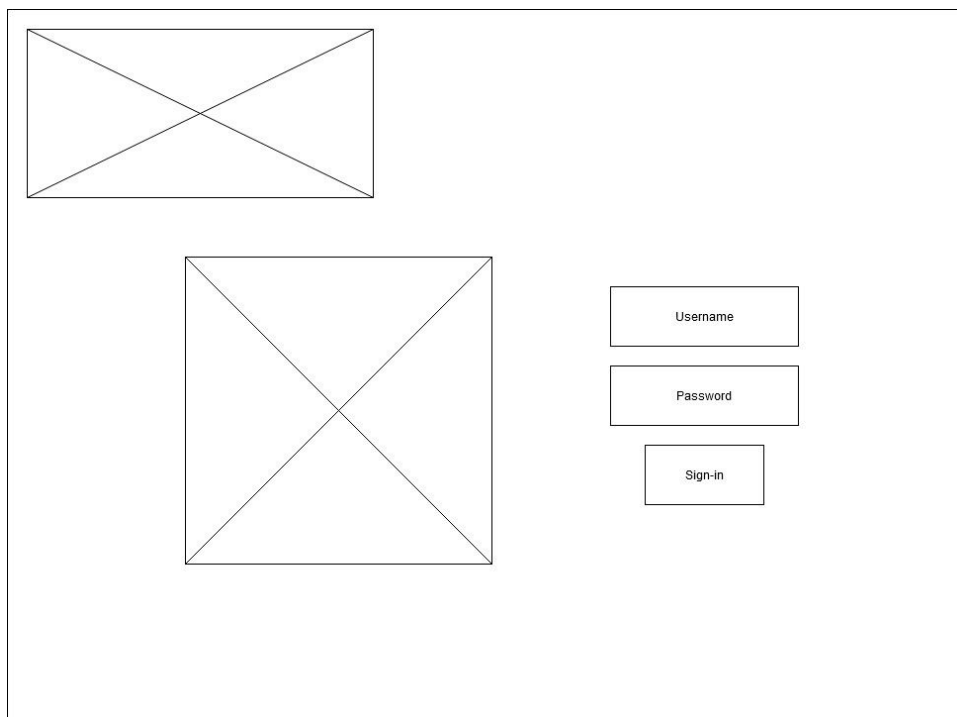
## Appendix B- Wireframes

Login/Create Account Page



A wireframe diagram of a login page layout. It features a rectangular header area at the top left containing a rectangle with an 'X' inside. Below the header is a large square area, also containing a rectangle with an 'X' inside. To the right of the square are two stacked rectangular buttons. The top button is labeled 'Login' and the bottom button is labeled 'Create Account'.

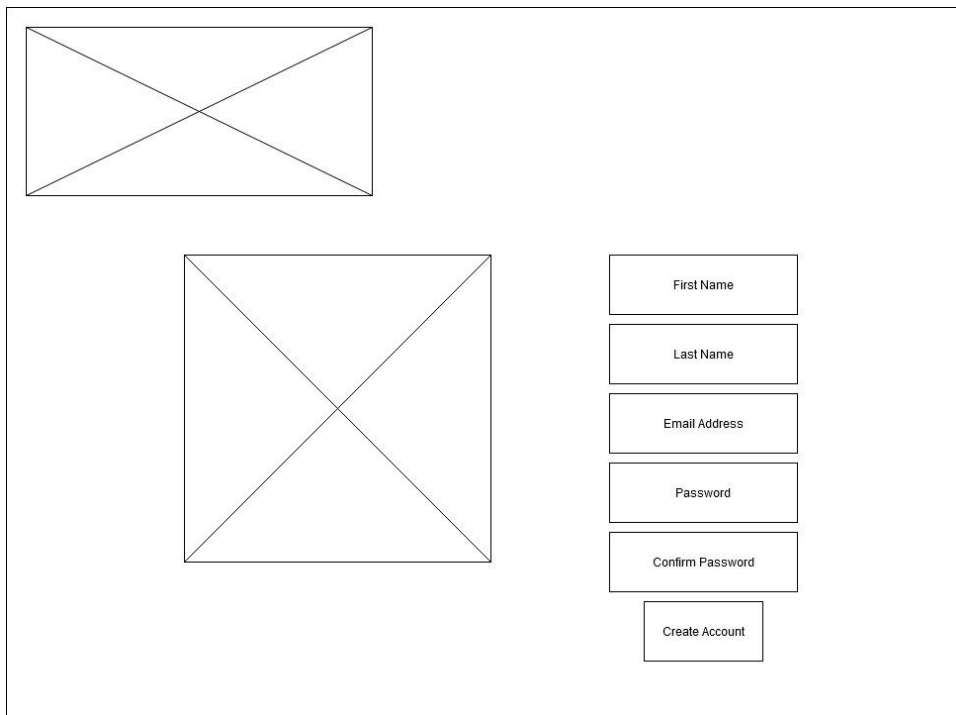
Login Page



A wireframe diagram of a create account page layout. It features a rectangular header area at the top left containing a rectangle with an 'X' inside. Below the header is a large square area, also containing a rectangle with an 'X' inside. To the right of the square are three stacked rectangular input fields. The top field is labeled 'Username', the middle field is labeled 'Password', and the bottom field is labeled 'Sign-in'.

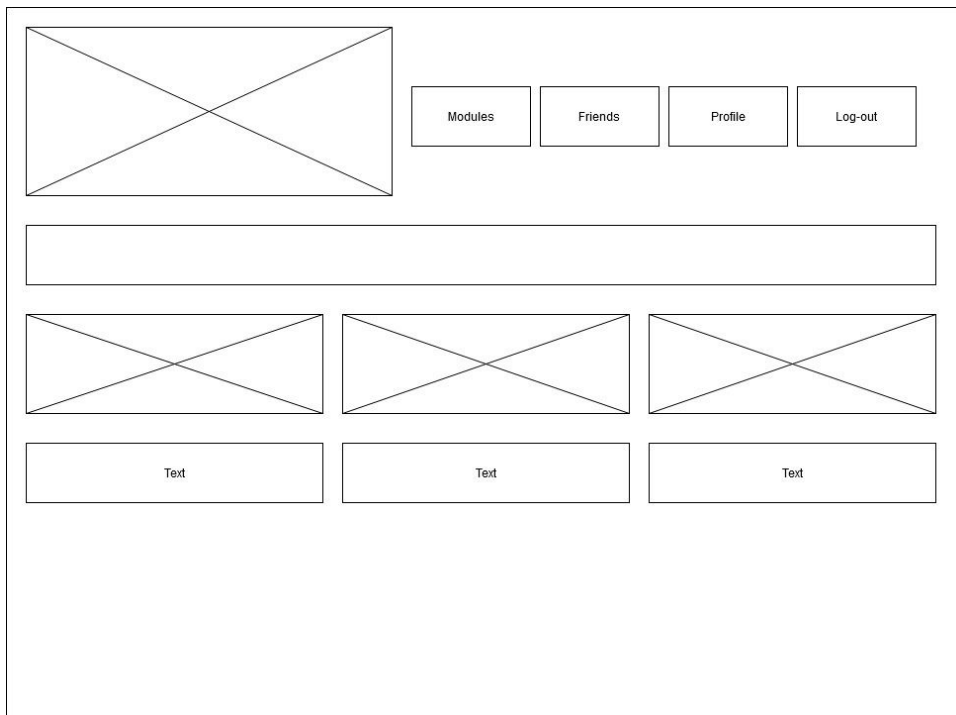
Create Account





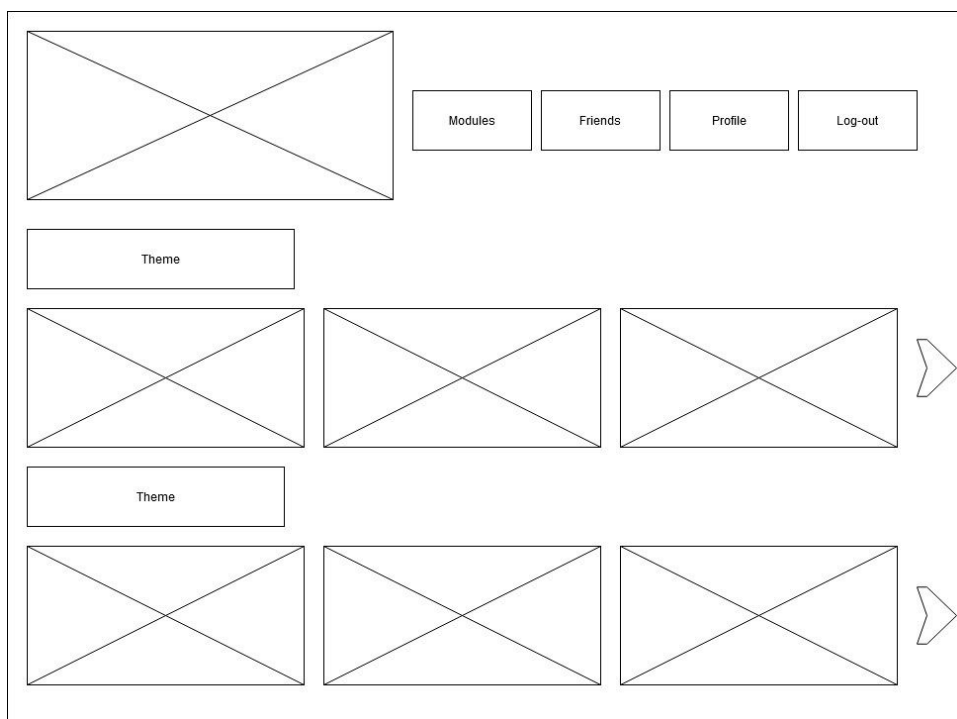
A user registration form layout. It features a small rectangular placeholder with an 'X' in the top left corner. Below it is a larger square placeholder with an 'X'. To the right of the square placeholder is a vertical stack of five input fields labeled 'First Name', 'Last Name', 'Email Address', 'Password', and 'Confirm Password'. Below these fields is a 'Create Account' button.

## User Dashboard

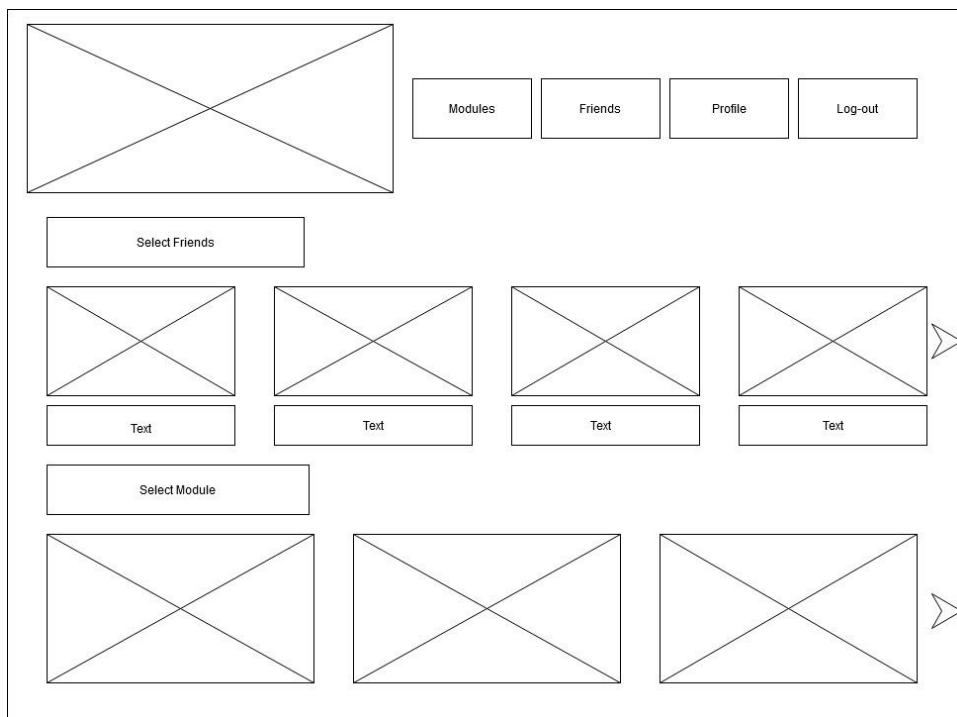


A user dashboard layout. It includes a large rectangular placeholder with an 'X' in the top left corner. To its right is a horizontal row of four buttons labeled 'Modules', 'Friends', 'Profile', and 'Log-out'. Below the placeholder and buttons is a long, empty rectangular box. Underneath this box is a row of three rectangular placeholders, each with an 'X'. Below these placeholders is a row of three text input fields, each labeled 'Text'.

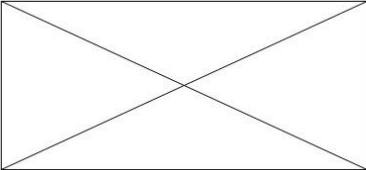
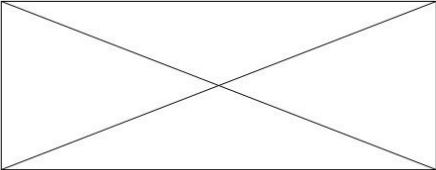
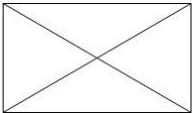
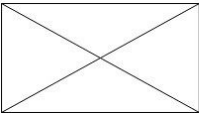
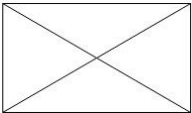
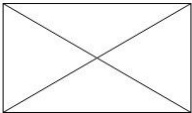
## Find a Module



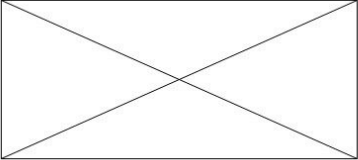
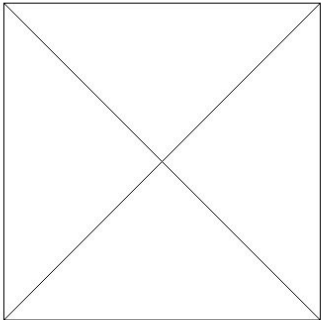
Make a group

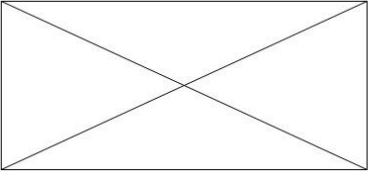
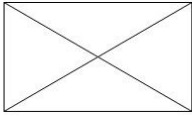
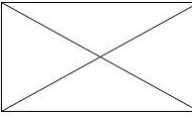
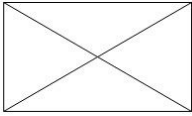
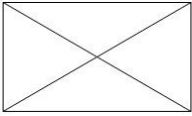
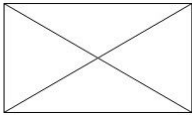
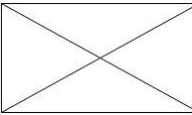
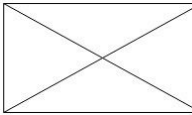
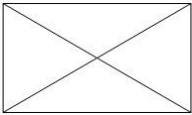


Group Modules Page

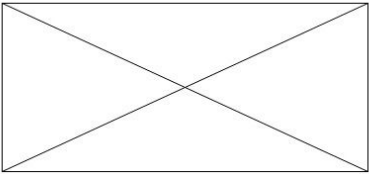
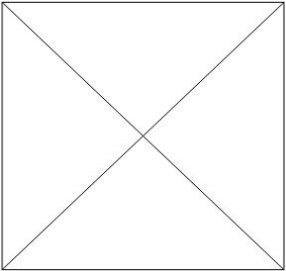
|   |   |  |   |
|---|---|--|---|
|  | <div>Modules</div> <div>Friends</div> <div>Profile</div> <div>Log-out</div>       |  |   |
|  | <div>Text</div>   |  |   |
| <div>Group Members</div>  |   |  |   |
|  |  |  |  |
| <div>Text</div>   | <div>Text</div>   | <div>Text</div>  | <div>Text</div>   |

### Directly Join a Group

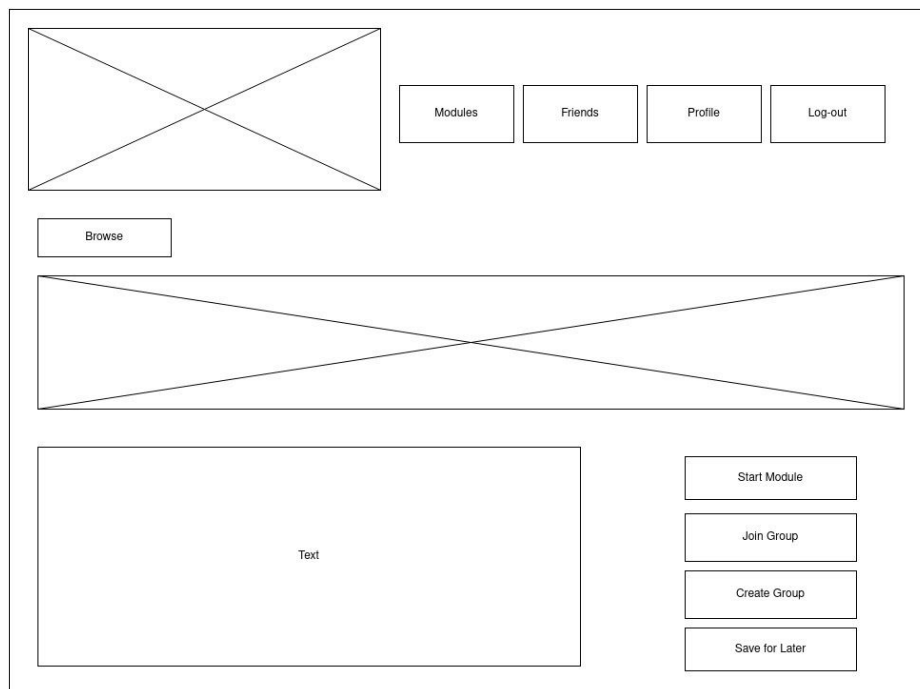
|   |   |
|---|---|
|   |   |
|  | <div>Group Password</div> <div>Submit</div> |

|   |   |  |   |                         |
|---|---|--|---|-------------------------|
|  | <a href="#">Modules</a>   | <a href="#">Friends</a>  | <a href="#">Profile</a>   | <a href="#">Log-out</a> |
|   | <input type="text" value="Search Friends"/>                                       |  |   |                         |
|  |  |  |  |                         |
| <input type="text" value="Text"/>   | <input type="text" value="Text"/>   | <input type="text" value="Text"/>  | <input type="text" value="Text"/>   |                         |
|  |  |  |  |                         |
| <input type="text" value="Text"/>   | <input type="text" value="Text"/>   | <input type="text" value="Text"/>  | <input type="text" value="Text"/>   |                         |

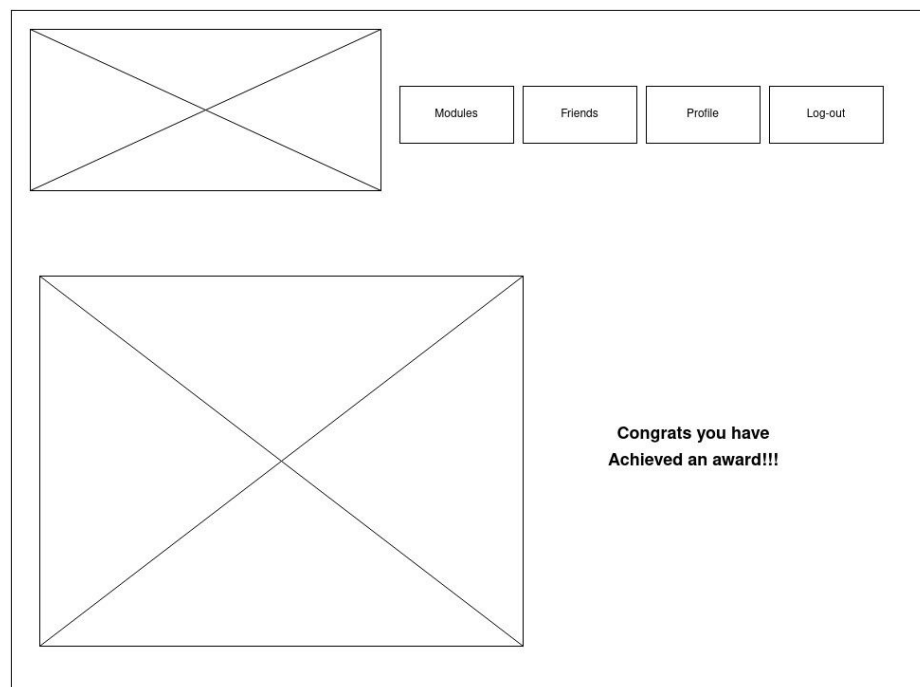
## Profile Page

|  |   |                         |                         |                         |
|--|---|-------------------------|-------------------------|-------------------------|
|  | <a href="#">Modules</a>   | <a href="#">Friends</a> | <a href="#">Profile</a> | <a href="#">Log-out</a> |
|  |  |                         |                         |                         |
| <input type="text" value="Name"/> ▼  |   |                         |                         |                         |
| <input type="text" value="Status"/> ▼  |   |                         |                         |                         |
| <input type="text" value="Edit Account"/> ▼  |   |                         |                         |                         |
| <input type="text" value="Settings"/> ▼  |   |                         |                         |                         |

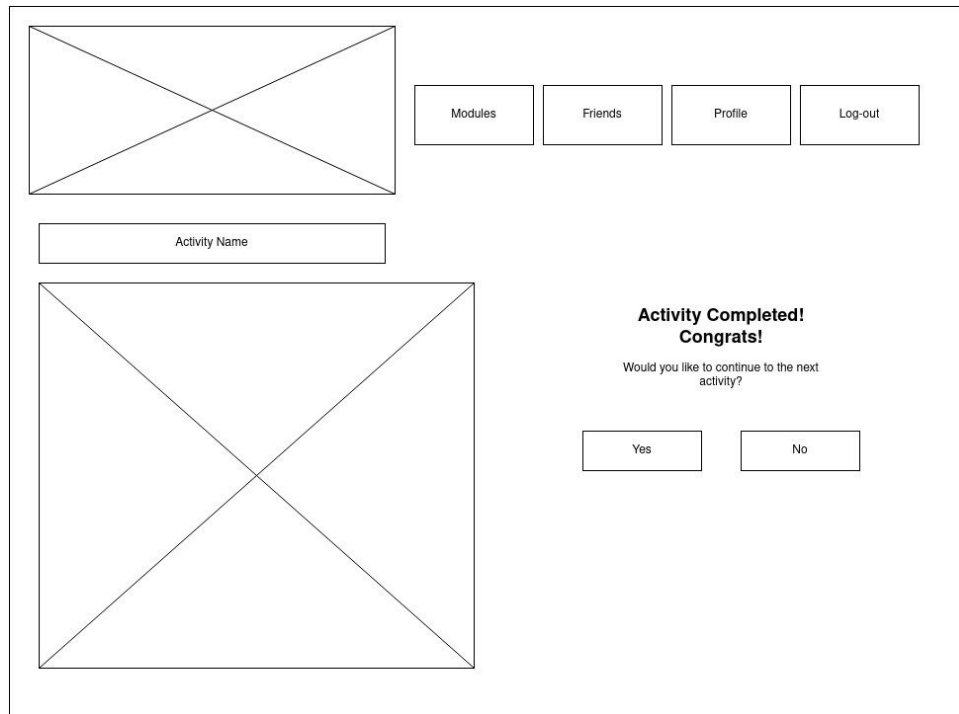
## Select A Module



Acheivement earned



Activity completed



Activity Completed! Congrats!

Would you like to continue to the next activity?

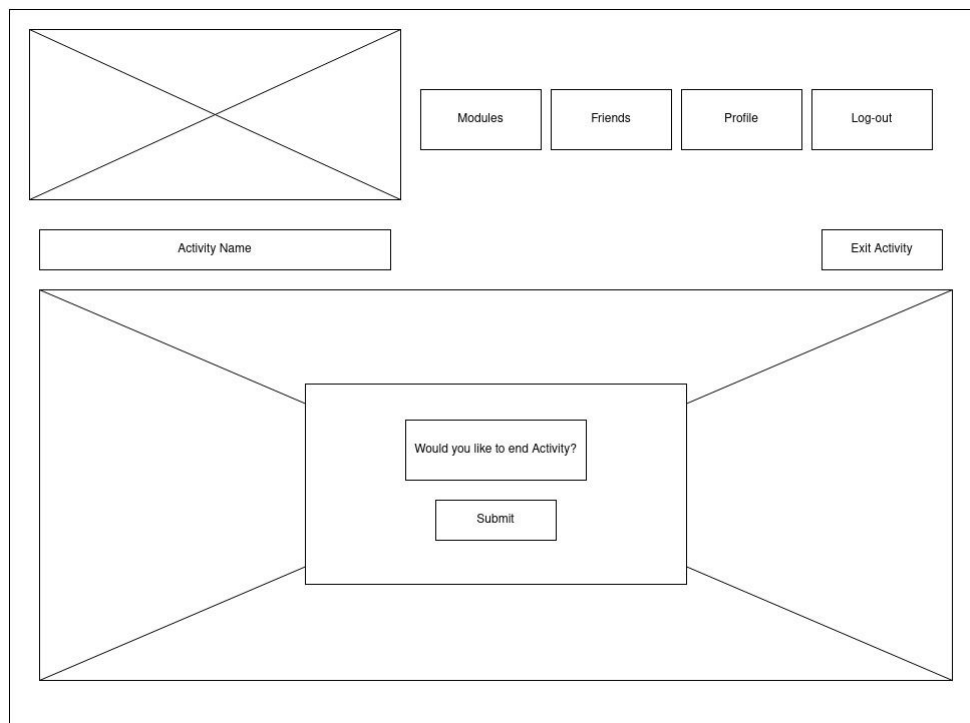
Yes No

Activity Name

Modules Friends Profile Log-out

This screen displays a large rectangular area with a diagonal cross, likely a placeholder for an image or video. To the right of this area, there is a section titled "Activity Completed! Congrats!" followed by a question: "Would you like to continue to the next activity?". Below the question are two buttons: "Yes" and "No". Above the large rectangular area, there is a smaller rectangular area with a diagonal cross, and a label "Activity Name". To the right of the "Activity Name" label, there are four buttons: "Modules", "Friends", "Profile", and "Log-out".

### Exit Activity



Would you like to end Activity?

Submit

Activity Name

Exit Activity

Modules Friends Profile Log-out

This screen displays a large rectangular area with a diagonal cross, likely a placeholder for an image or video. In the center of this area, there is a smaller rectangular area containing the question "Would you like to end Activity?" and a "Submit" button below it. To the right of the large rectangular area, there is a button labeled "Exit Activity". Above the large rectangular area, there is a smaller rectangular area with a diagonal cross, and a label "Activity Name". To the right of the "Activity Name" label, there are four buttons: "Modules", "Friends", "Profile", and "Log-out".

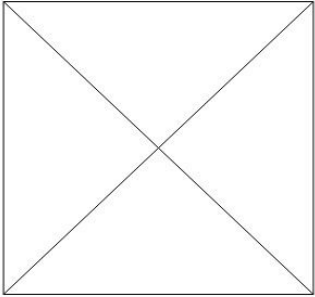
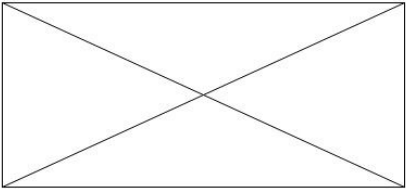
### Activity in Progress

Wireframe of a web page layout. The page contains a placeholder for a profile picture (a rectangle with an 'X' inside) in the top left. To its right are four navigation buttons: "Modules", "Friends", "Profile", and "Log-out". Below the profile picture placeholder is a text input field labeled "Activity Name". The main content area is a large rectangle with an 'X' inside, indicating a placeholder for content.

### Select an Activity

Wireframe of a web page layout for selecting an activity. The page contains a placeholder for a profile picture (a rectangle with an 'X' inside) in the top left. To its right are four navigation buttons: "Modules", "Friends", "Profile", and "Log-out". Below the profile picture placeholder is a text input field labeled "Activities". Below the "Activities" input field is a grid of six placeholder boxes (rectangles with an 'X' inside) arranged in two rows of three, representing a selection of activities.

## Delete Account



Modules

Friends

Profile

Log-out

Delete Account

Submit