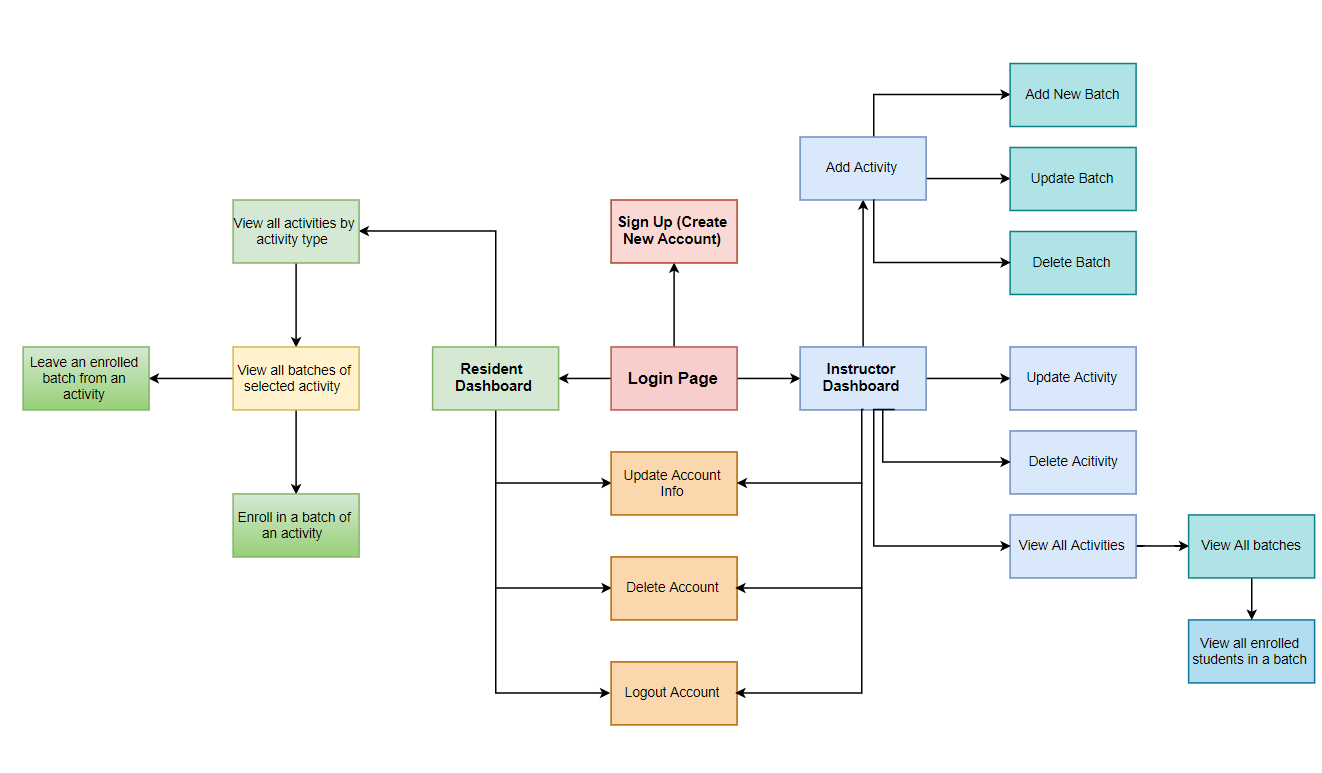
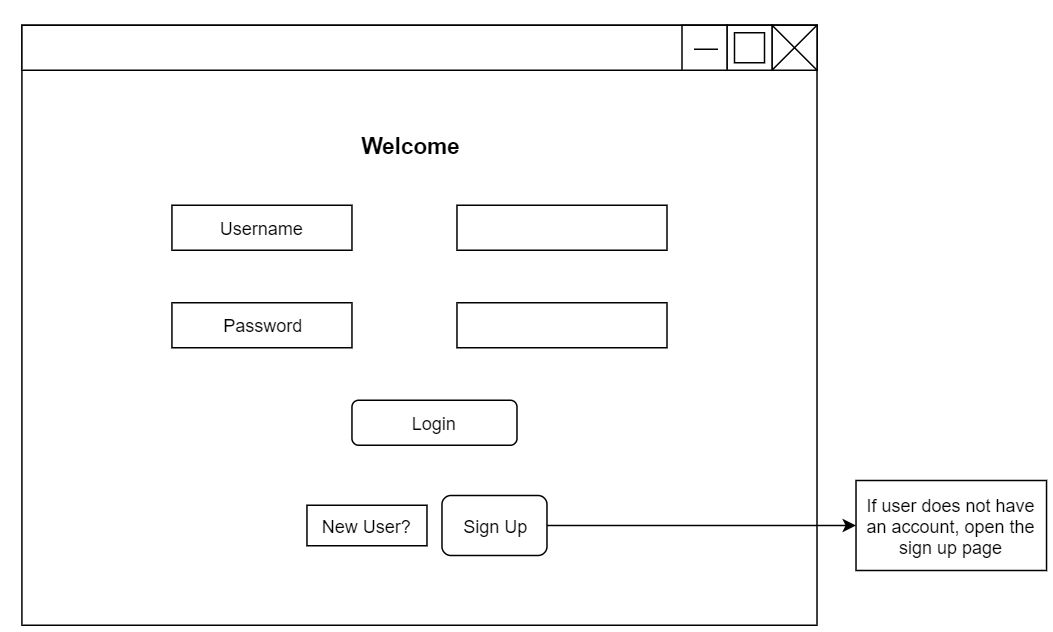
**Design**

**Design Outline**

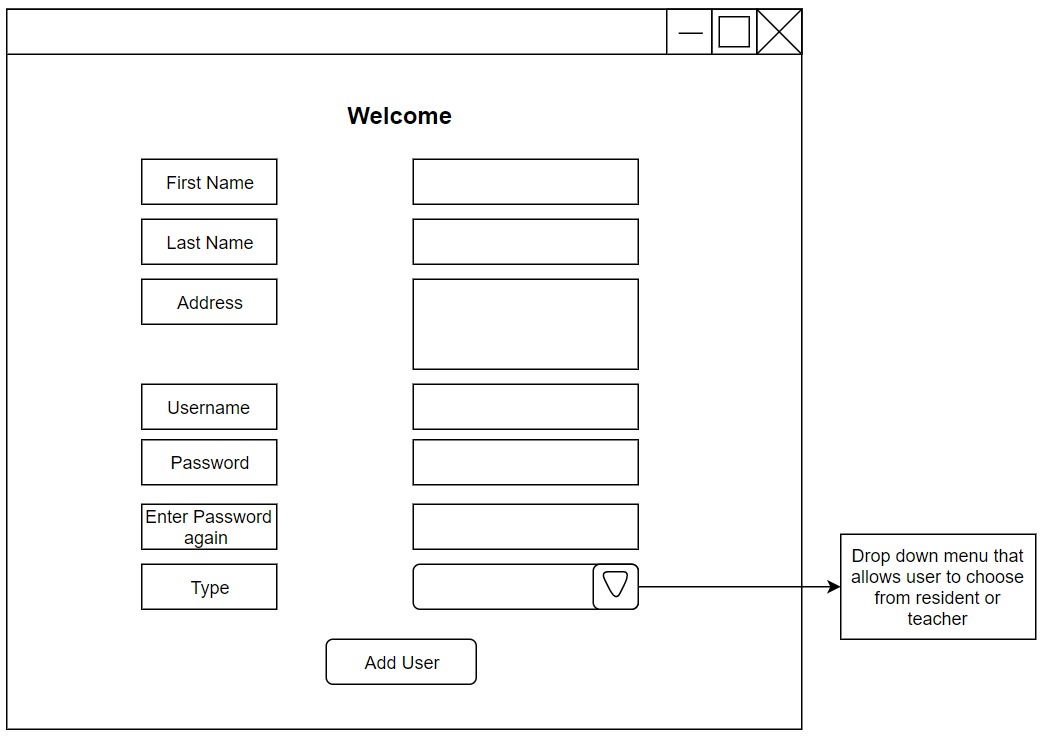


**Screen Layouts**

Welcome screen of the program:

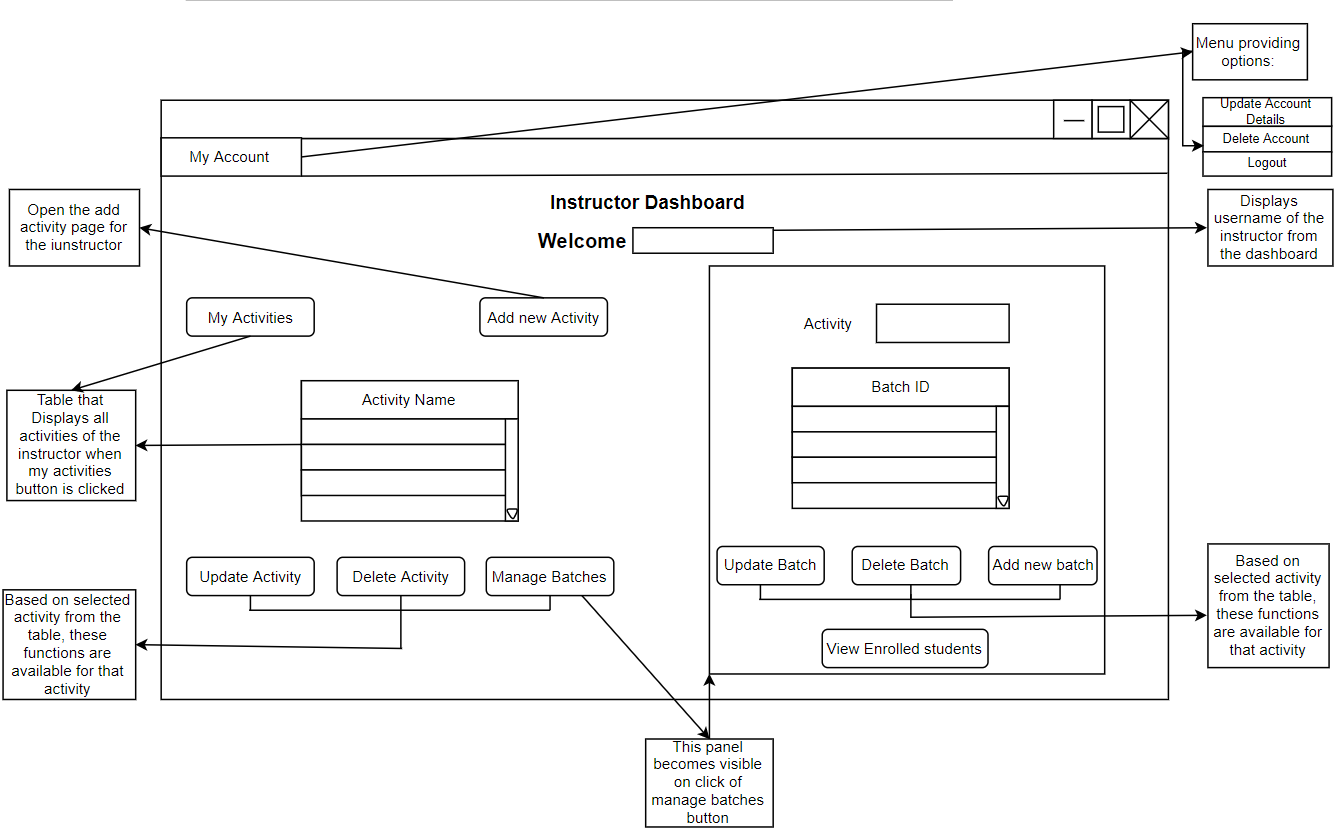


Sign up page to add a new user:



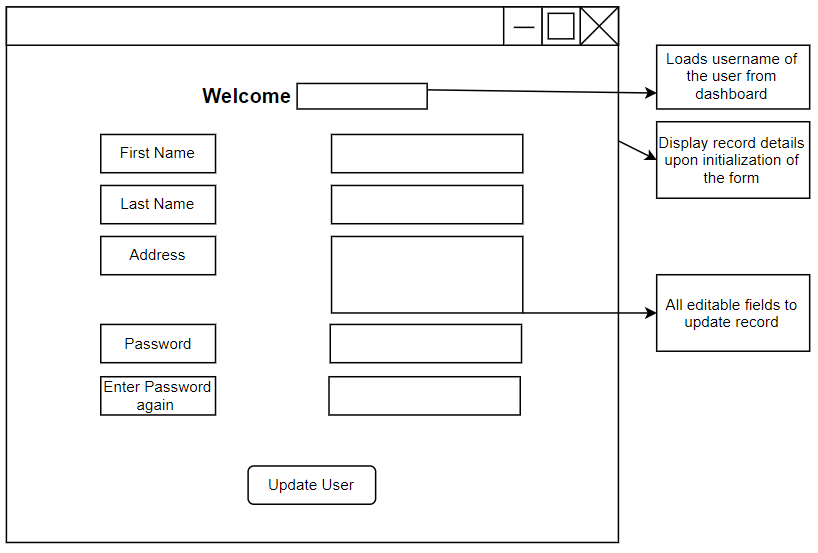
**Instructor segment:**

Instructor dashboard loaded upon login of instructor:

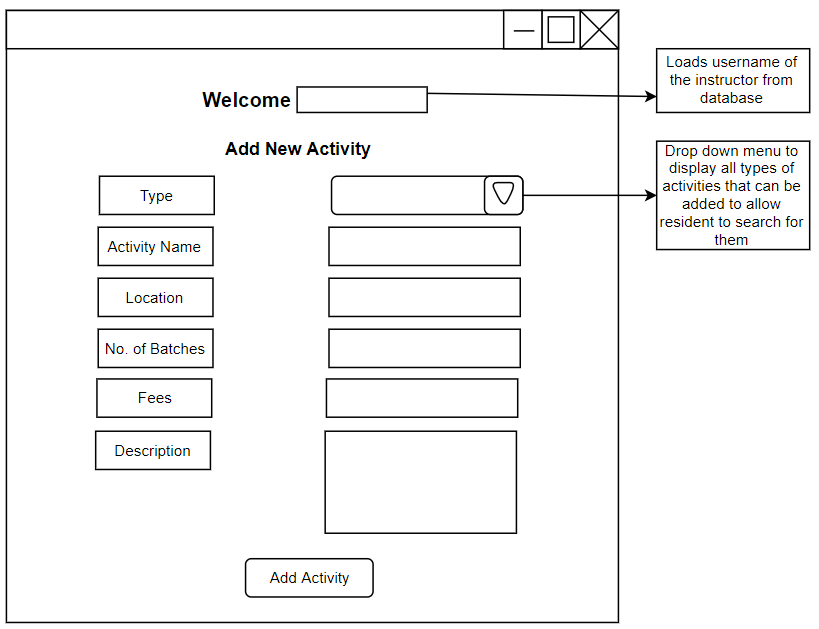


**My Account** - Update user account:

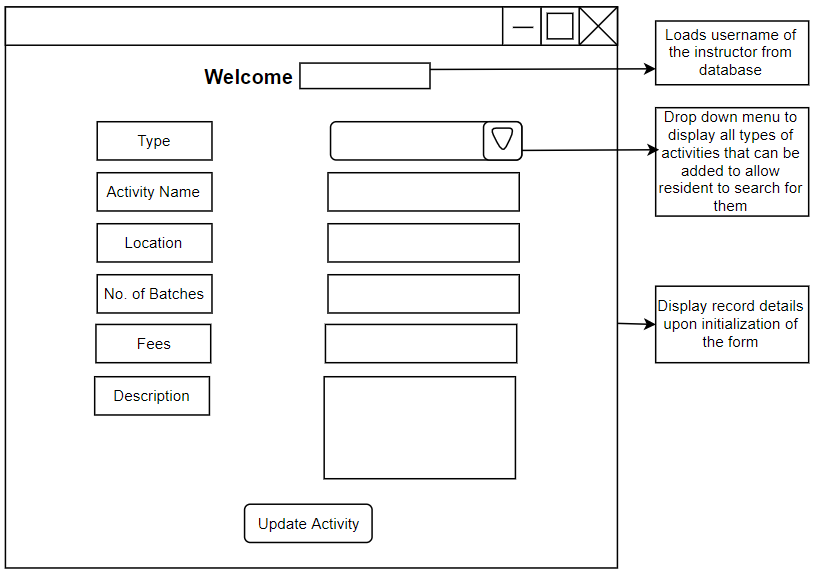
Common for resident as well as instructor



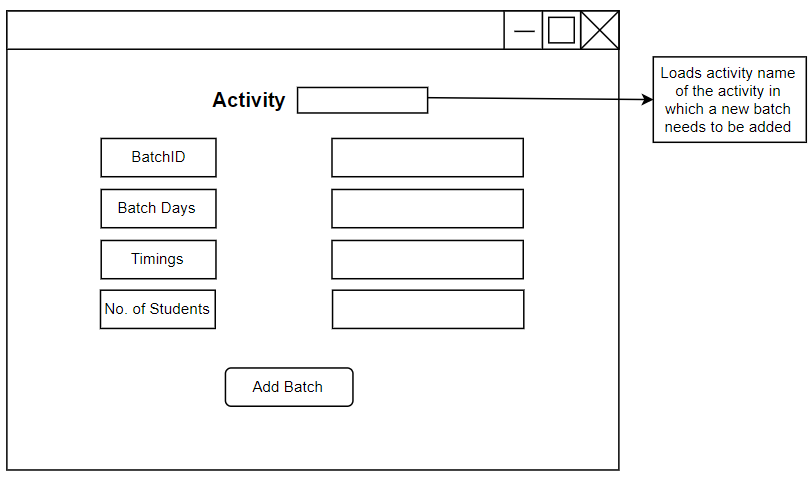
Add new activity by instructor:



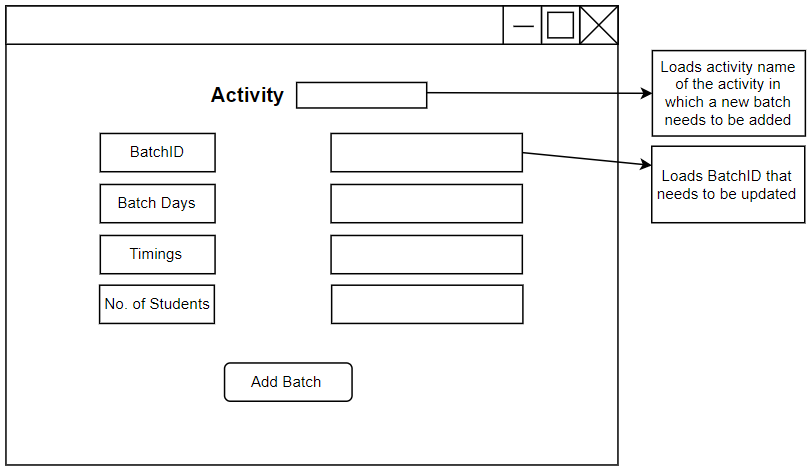
Update Activity by instructor:



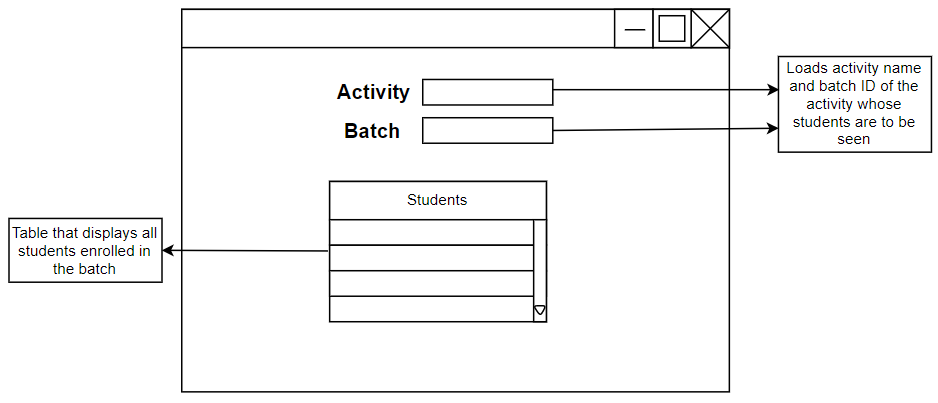
Add new batch for selected activity in the instructor dashboard:



Update batch details of selected activity on instructor dashboard:

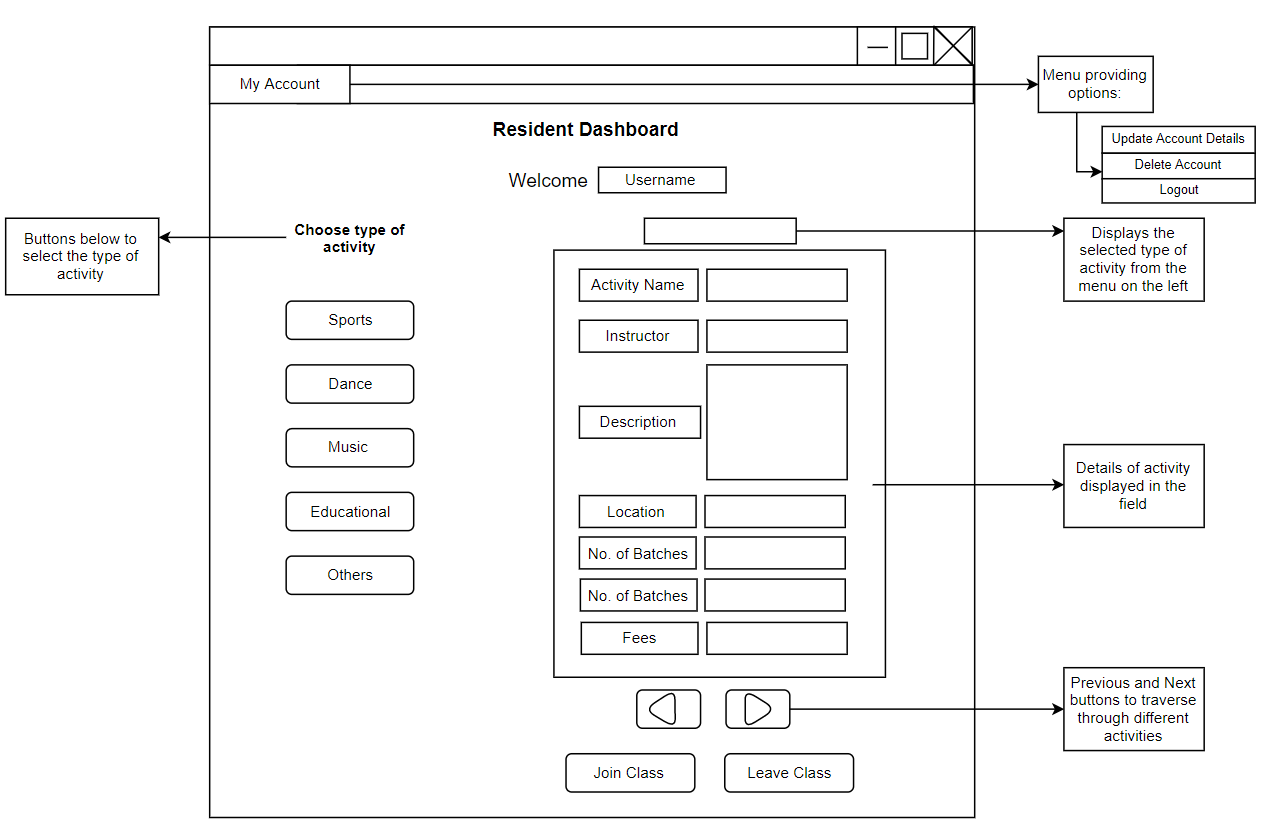


View all enrolled students in the selected batch of activity:

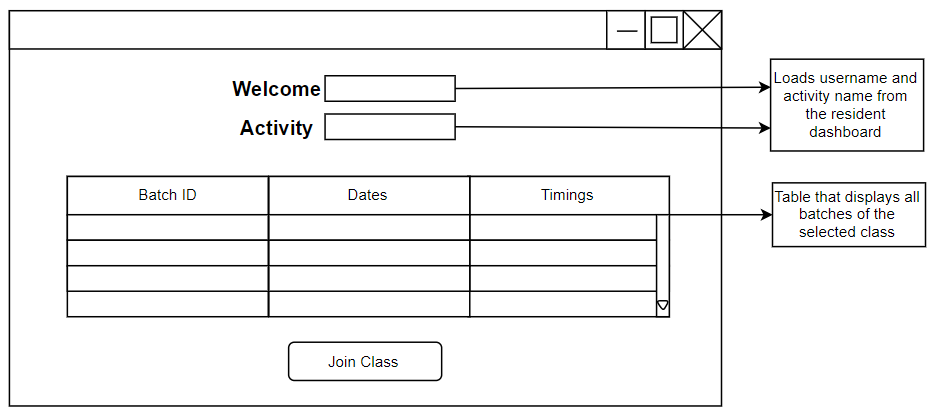


**Resident Segment**

Resident Dashboard loaded upon successful login:



On click of join class display all batches and their details to allow residents to enrol in a particular batch:



**Database:**

To store all required data, a database will be created that will have the following tables and structure.

**User**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Description** | **Type of Key** |
| First Name | Varchar (30) | Stores the first name of user |  |
| Last Name | Varchar (30) | Stores the last name of user |  |
| Address | Varchar (50) | Residential Address of the user |  |
| Username | Varchar(30) | The user can choose a username for himself | Primary Key |
| Password | Varchar(20) | Password of the user |  |
| Password 2 | Varchar(20) | Double entry to ensure it matches password |  |
| Type | Char(1) | “R” for Resident  “T” for Teacher |  |

**Activity Master**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Description** | **Type of Key** |
| Activity Type | Varchar(20) | Activity Type: Can be Sports, Dance, Music, Educational or Others |  |
| Activity Name | Varchar(50) | Name of the class | Primary Key |
| Instructor Name | Varchar(30) | Name of the instructor | Foreign Key |
| Description | Varchar (200) | A brief description of the class |  |
| Location | Varchar(50) | Address of the class (full address) |  |
| No. of Batches | Int | The no. of batches that the teacher wishes to take |  |
| Fees | Float | Fees per month for the class |  |

**Batch Master**

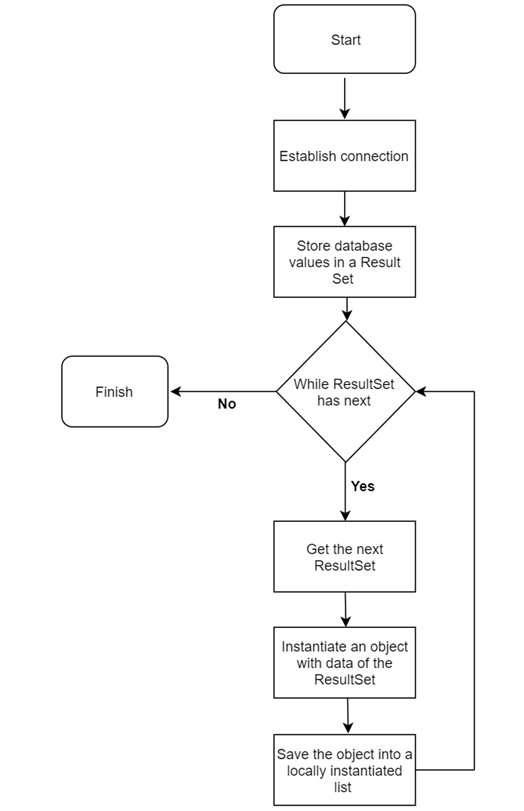
|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Description** | **Type of Key** |
| Activity name | Varchar(20) | Name of activity | Foreign Key |
| Batch ID | Varchar(10) | Unique identity for each batch of an activity | Primary Key |
| Batch Days | Varchar(50) | Days on which the batch is conducted |  |
| Batch Timings | Varchar(50) | Timings of the class |  |
| No. of students | Int | Total number of students in the batch |  |

**Batch Transaction**

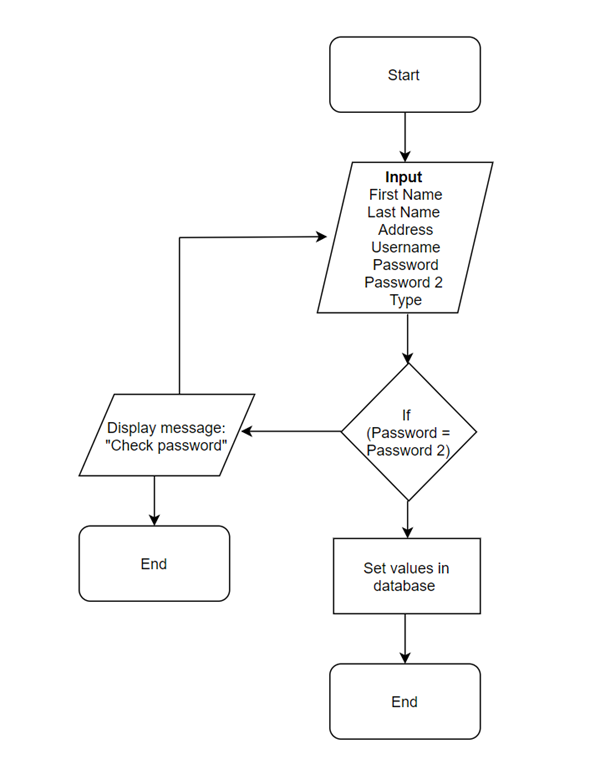
|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Description** | **Type of Key** |
| TID | Int | Transaction ID | Primary Key |
| Username | Varchar(30) | Stores the username of the resident | Foreign Key |
| Activity Name | Varchar(20) | Name of the class | Foreign Key |
| Batch ID | Varchar(10) | Batch no. of the class enrolled in | Foreign Key |

**Flowcharts of the most significant features of the program:**

**Load data from SQL database**

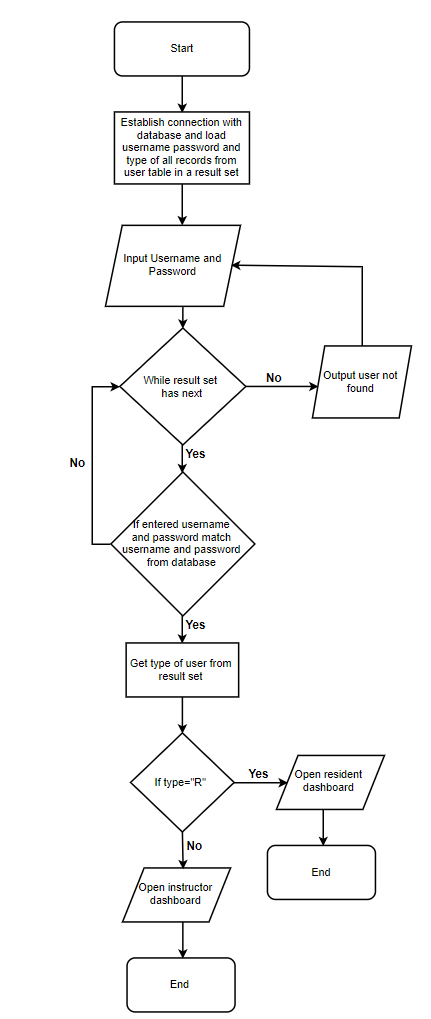
****

**Add a new user:**

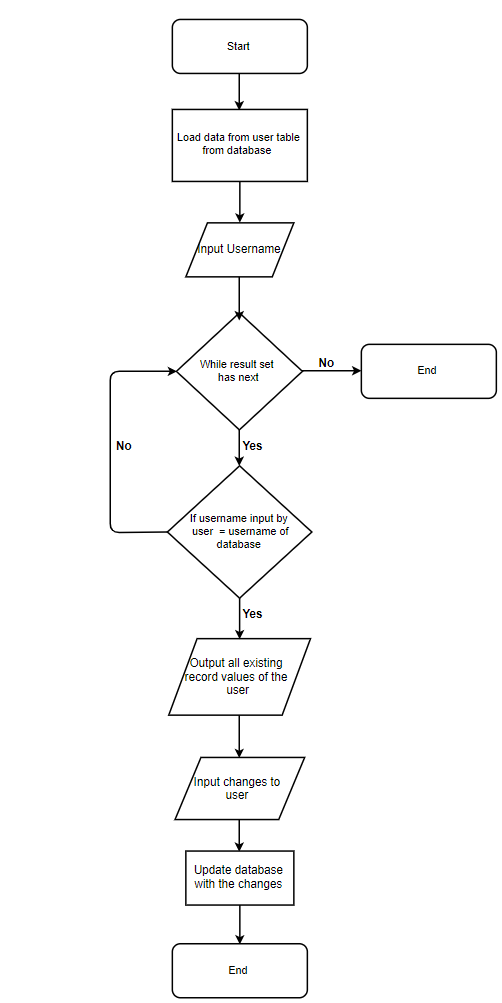
****

A similar flow will be followed when adding a record in any table such as adding an activity or batch. The only difference will be that no password fields will be compared.

**Login user and open resident or instructor dashboard based on login**

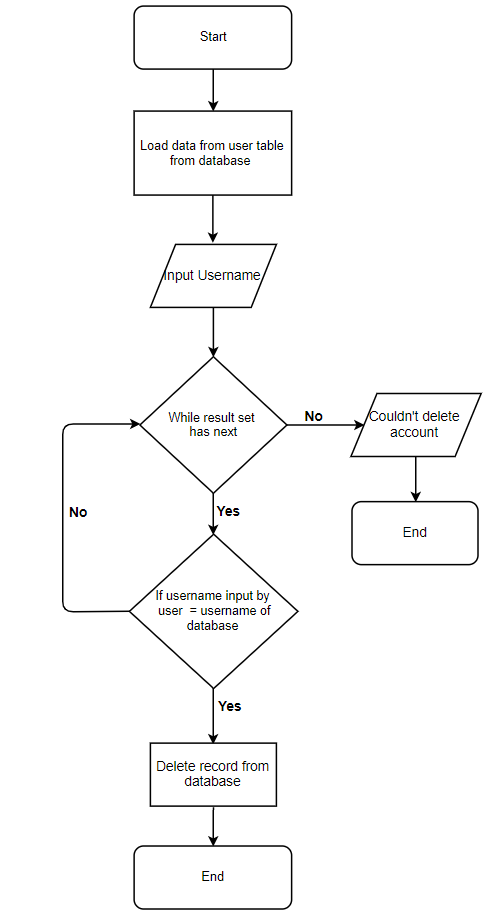


**Update user details in user table:**



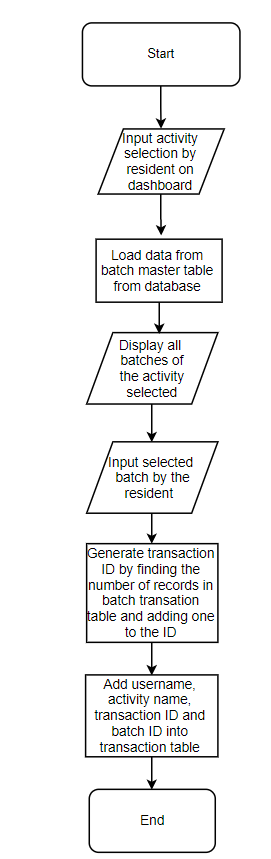
A similar flow will be followed to update activity and batch details from their respective tables in database.

**Delete User:**



The same process will be followed for deleting activity and batch as well from activity master and batch master table respectively.

**Enrol in a batch of an activity**



**Test Plan**

|  |  |
| --- | --- |
| **Criteria of Success** | **Test Method** |
| GUI based desktop application and provide a friendly interactive software for the users | Check if all inputs and outputs are GUI based. Try different inputs to verify the functionality of the product and ensure that there are no run time errors. Ensure navigation in the program in smooth. |
| Separate interfaces for teachers and students | Make sure the user can access information only based on their login account. Ensure that a user’s login restricts their actions. |
| Allow users to create an account | Add users of both types and check if an account is being created successfully for each user. Try adding normal, abnormal and extreme account details to check if all checks are functioning |
| Allow users to update and delete their account | Perform tests to update user details for both types of users and delete a few users and check if their account is deleted from the database. |
| Allow users to logout from their account | Check if data is being dropped by the software for the current user if a user logs out. Check this by logging out from both type of users. |
| Allow instructors to create activities | Try adding different classes and check database if the entry took place or not. |
| Instructors should be able to update and delete activities | Perform tests to check if data is being properly updated in the database. Delete activities and check if record is deleted from database. |
| Instructors should be able to create batches for their activities | Add different batches for different activities by various instructors to check if proper entries are being added into the database. Use extreme, normal and abnormal data for entries |
| Allow instructors to update and delete batches | Try changing non-changeable attributes to ensure that they are non-editable. Update different batches and delete batches to verify if they are updated or deleted from database. |
| Allow instructors to view all their activities and batches | After adding different activities and batches, ensure that all records are being displayed when the instructor wishes to view all of them. |
| In case a batch is full, disable residents to enrol in the batch | Check whether instructor is able to disallow further residents for enrolling in a class if the batch is full |
| View all enrolled students in a batch of a selected activity | Login from teacher’s account and view residents and check database if all residents are being displayed. |
| Allows residents to see all activities by activity type | Check for user profile and see activities of all different types to ensure the user can view all these activities. Check database and verify that all records are visible |
| Allow residents to enrol in classes | Try enrolling different residents in classes providing their details. Ensuring that these details are added onto the database. |
| Allow residents to leave activities previously joined | Operate the exit button for different residents and check database for deletion of their record. |
| Have database that stores information of user credentials | Add different entries into user table of the database and check whether password and re-password fields are same. Check if retrieval of information from the table is possible. |
| Have database that stores information of all activities | Try to add, delete and retrieve activities from the table. Check if only the allowed types of activities are added to the table and all information is correct. |
| Have database that stores information about all batches | Add different batch data to the table and whether all data is accurate. Ensure that links with activity and user table are working properly. |
| Have database to store data about all enrolled residents in activities | Ensure that the Transaction ID is properly generated for each resident enrolment into a batch. Check whether all connections to other tables are functioning and all data is correct. Retrieve information from the table to ensure access to the database. |