#### LoginController

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *doGet()* | *Handles the HTTP <code>GET</code> method.* |
| *2* | *doPost()* | *Handles the HTTP <code>POST</code> method.* |

#### UserDAO

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *getUser()* | *Method used to check if an account record existed in database and get all data of that account.Inputs are username and password.Return an <Code>model.User</Code> represents the user’s data* |

#### DBContext

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *getConnection* | *Method used to connect to the database. Return a <Code>java.sql.Connection</Code> represents a connection to the database.* |

Select \* from [User] where (username=? OR email=?) AND password=?

#### EditController

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *doGet()* | *Handles the HTTP <code>GET</code> method.* |
| *2* | *doPost()* | *Handles the HTTP <code>POST</code> method.* |

#### UserDAO

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *getUser()* | *Method used to check if an account record existed in database and get all data of that account.Inputs are username and password.Return an <Code>model.User</Code> represents the user’s data* |

#### DBContext

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *getConnection* | *Method used to connect to the database. Return a <Code>java.sql.Connection</Code> represents a connection to the database.* |

Select \* from [User] where (username=? OR email=?) AND password=?

#### EditController

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *doGet()* | *Handles the HTTP <code>GET</code> method.* |
| *2* | *doPost()* | *Handles the HTTP <code>POST</code> method.* |

#### UserDAO

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *editProfile()* | *Method used to let admin update the data of that account.Inputs are admin details.Update those to database.* |

#### DBContext

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *getConnection* | *Method used to connect to the database. Return a <Code>java.sql.Connection</Code> represents a connection to the database.* |

update[User] set

[email] = ?,

[fullname] = ?,

[phone] = ?,

[address] = ?

where [id] = ?

Login

● Function trigger: Click “Login” on pop up home page

● Function description:

o Actor: Admin , User

o Users that have not yet registered cannot log in. They must contact the education unit to create a new account.

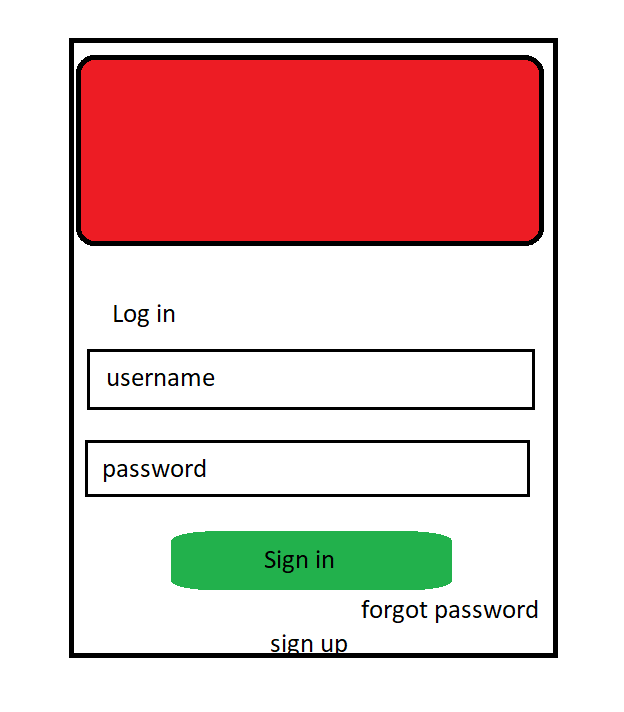
● Function detail:

o Users shall login to the system by using the username and password which is registered.

o Users shall be able to use system functionalities and be directed to the formal screen after successful login.

o The Users will receive “errors” to the login screen if the username or password is invalid.

● Screen layout:



View profile

● Function trigger: Dash board page click “ Profile”

● Function description:

o Actor: Admin

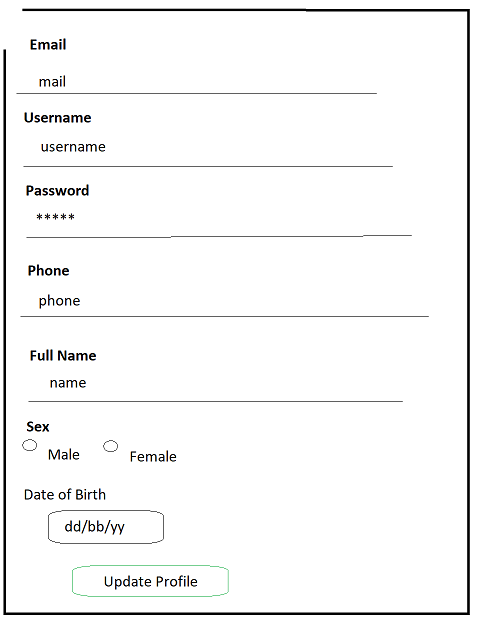
o Admin can view account’s profile

● Function detail:

o The profile of Admin account will display on profile screen

o Admin can view all those details

● Screen layout:



View profile

● Function trigger: From Profile page

● Function description:

o Actor: Admin

o Admin can update profile need to

● Function detail:

o There is a list of data of admin account.

o Admin can check and rewrite what they want

o Click the “Update” button to save the new details

o Notice update successfully

● Screen layout:

Graphical user interface, text

Description automatically generated