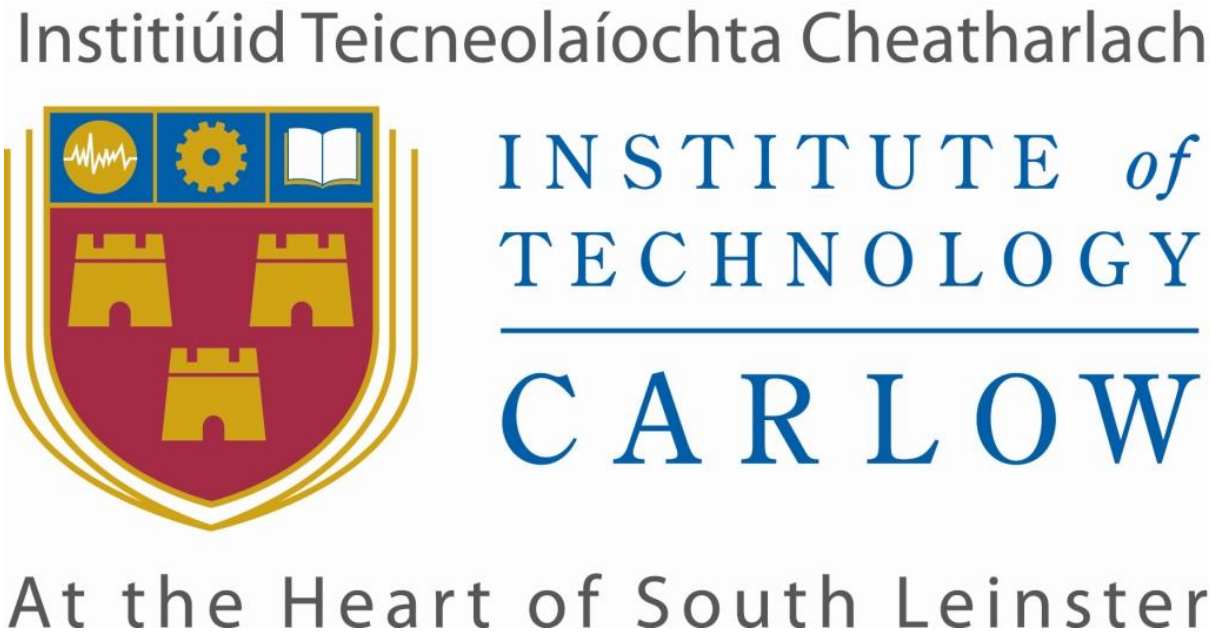
Pro Events

Design Document



Name: Jonathan Finlay

Student Number: C00193379

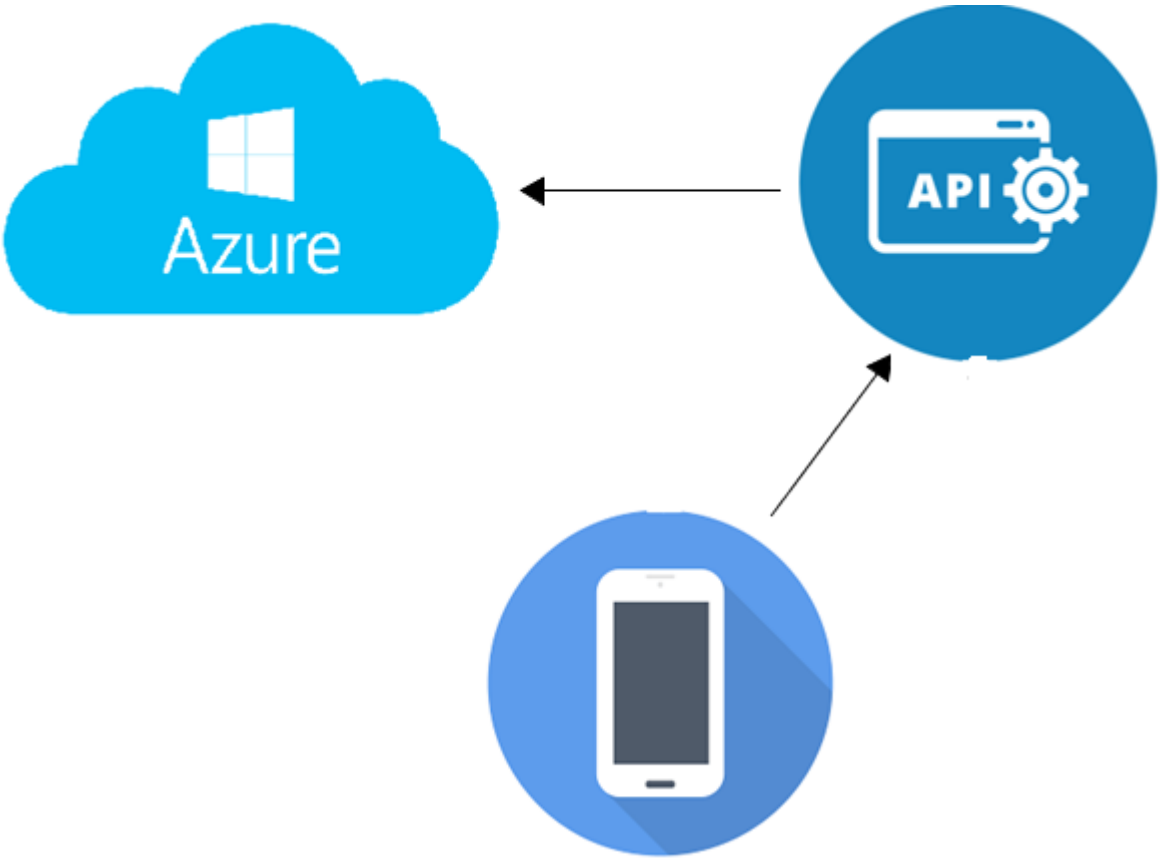
Course: Bachelor of Science (Honours) Software Development

Tutor: Hisain Elshaafi

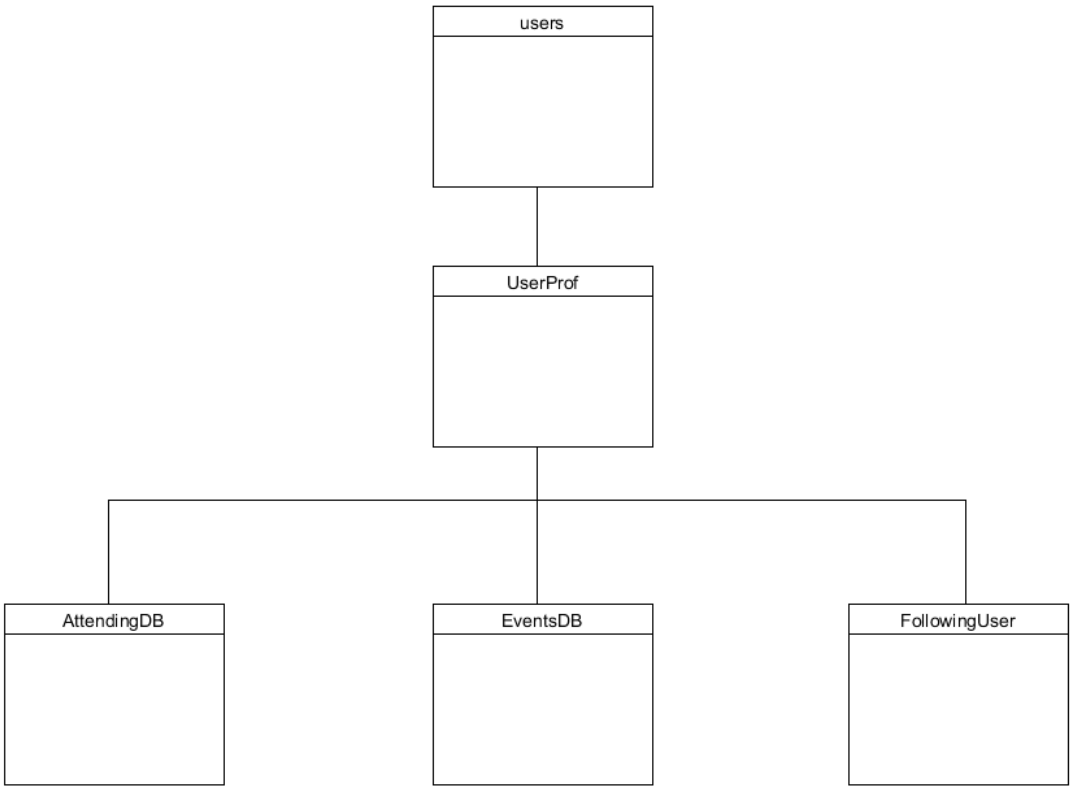
Date: 18-04-18

Contents

System Architecture



Database Structure



Use Cases

Login Menu

Brief Use Case: Login Menu

The user clicks on the login button. The user then enters their username and password. The system checks if the username is in the database. If it isn't, notifies the user that the details entered are incorrect. If they are correct, it compares the password with the encrypted password in the database. If they are a match, the user logs in successfully.

Detailed Use Case: Login Menu

Actors:

User, App, API, Database

Brief Description:

The user clicks on the login button. The user then enters their username and password. The system checks if the username is in the database. If it isn't, notifies the user that the details entered are incorrect. If they are correct, it compares the password with the encrypted password in the database. If they are a match, the user logs in successfully.

Main Success Scenario:

1. User clicks login button.
2. User enters login detail.
3. System verifies details entered with database.
4. System returns details are correct.
5. User logs in successfully.
6. User diverted to Main Menu.

Extensions:

4a. System returns details are incorrect.

4b. System returns user to login screen with message 'Details are incorrect'.

4c. User tries again.

Logout

Brief Use Case: Logout

When the user is on any page, they have the option to logout by pressing the button ‘logout’. This then brings back the menu to login and register.

Detailed Use Case: Logout

Actors:

User, App, API, Database

Brief Description:

When the user is on any page, they have the option to logout by pressing the button ‘logout’. This then brings back the menu to login and register.

Main Success Scenario:

1. The user loads any page.
2. The user clicks on the logout button.
3. The system clears the shared preferences.
4. The user is logged out.
5. The user is redirected to the login/register page.

Extensions:

3a. The system comes across an error.
3b. The system prompts the user of the error.
3c. The system cancels the logout.

Registration

Brief Use Case: Registration

Begins when the user opens the app. The user clicks the register button and a form appears. The user fills in the form with relevant information and clicks the submit button. The account creation request is sent to the API for processing.

Detailed Use Case: Login Menu

Actors:

User, App, API, Database

Brief Description:

Begins when the user opens the app. The user clicks the register button and a form appears. The user fills in the form with relevant information and clicks the submit button. The account creation request is sent to the API for processing.

Main Success Scenario:

1. The system renders the login view for user input.
2. The user clicks on the register button.
3. The login form is dynamically replaced with a registration form.
4. The user fills out the registration form and presses the submit button.
5. The user presses the submit button.
6. The form is sent to the API for validation and creation.
7. The API validates the account creation and notifies the user of successful account creation.
8. The mobile app prompts the user to log in.

Extensions:

4a. The user enters invalid username or password.
4b. The system prompts the user of the error.
4c. The system redirects the user to the registration form.

Edit Profile

Brief Use Case: Edit Profile

The user clicks on the edit profile button on the home/profile page. A form will then open for the user to fill out. Once filled out they save the details where the system validates the form. Once validated, the database will update and the user will be redirected back to the home/profile page.

Detailed Use Case: Edit Profile

Actors:

User, App, API, Database

Brief Description:

The user clicks on the edit profile button on the home/profile home. A form will then open for the user to fill out. Once filled out they save the details where the system validates the form. Once validated, the database will update and the user will be redirected back to the home/profile page.

Main Success Scenario:

1. User clicked the 'Edit Profile' button.
2. System loads the edit profile form.
3. User fills out form.
4. System checks all the fields meet the criteria.
5. System updates the database
6. User redirected back to home/profile page.

Extension:

4a. Details don't meet criteria.
4b. System notifies user where error is.
4c. User fixes mistake.

View Followers

Brief Use Case: View Followers

The user clicks on the ‘View Followers’ button on the home/profile page. The page then loads all the users who clicked follow on the logged in users account. Here they can see their basic details. They have an option to view each users account and follow them too if they’re not already.

Detailed Use Case: Search Bar

Actors:

User, App, API, Database

Brief Description:

The user clicks on the ‘View Followers’ button on the home/profile page. The page then loads all the users who clicked follow on the logged in users account. Here they can see their basic details. They have an option to view each users account and follow them too if they’re not already.

Main Success Scenario:

1. User clicks on the ‘View Followers’ button.
2. Systems redirects user to the view followers page.
3. System loads the details suited towards logged in user.
4. User can browse all results.
5. User can select on any result.
6. User can follow another user.
7. User can view another users profile.

Extension:

3a. System fails to connect to database.
3b. User notified there was a problem.
3c. User retuned back to their last page.

View Profile

Brief Use Case: View Profile(other users)

The user can select on other users details and click the ‘View Profile’ option. The system then loads up their information. The user can then view their name, email, location, category/interest and a short description about them. There is also button to view events they have an interest in. This option is available on all users’ accounts.

Detailed Use Case: Search Bar

Actors:

User, App, API, Database

Brief Description:

The user can select on other users details and click the ‘View Profile’ option. The system then loads up their information. The user can then view their name, email, location, category/interest and a short description about them. There is also button to view events they have an interest in. This option is available on all users’ accounts.

Main Success Scenario:

1. User clicked the ‘View Profile’ button.
2. System loads the users profile form.
3. System displays the information on screen.

Extension:

3a. System failed to load users’ information.
3b. System notifies user there was an error.
3c. User redirected back a page.

My Profile

Brief Use Case: My Profile

The user can select on home button at the top of the screen. When selected, user is redirected to their own account.

Detailed Use Case: My Profile

Actors:

User, App, API, Database

Brief Description:

The user can select on their profile picture or name at the top of the screen. When selected, user is redirected to their own account.

Main Success Scenario:

1. User clicks home button at the top of app.
2. System gathers data.
3. System loads user profile.
4. User can view their own profile

Extension:

2a. System fails to load user profile.
2b. User notified of error and asked to try again.
2c. User returned to Main Menu.

View Events

Brief Use Case: View Events

The user can view an events details by clicking view on the event. Here the system gathers all the information related to the event selected. The user has an option to view other users interested in this event. This option is available whenever viewing an events page.

Detailed Use Case: View Events

Actors:

User, App, API, Database

Brief Description:

The user can view an events details by clicking view on the event. Here the system gathers all the information related to the event selected. The user has an option to view other users interested in this event. This option is available whenever viewing an events page.

Main Success Scenario:

1. User clicked the 'View Events' button.
2. System loads the events details.
3. System displays the information on screen.
4. User can view other users interested in this event.

Extension:

3a. System failed to load events information.
3b. System notifies user there was an error.
3c. User redirected back a page.

Search Events

Brief Use Case: Search Events

The user can access this page from the events page. The user can search for events here based on what category they choose. The number of categories of interests will be limited first, which can be increased in the future. The search will display all events related to that category. The user can then view the events page or follow the event.

Detailed Use Case: Search Events

Actors:

User, App, API, Database

Brief Description:

The user can access this page from the events page. The user can search for events here based on what category they choose. The number of categories of interests will be limited first, which can be increased in the future. The search will display all events related to that category. The user can then view the events page or follow the event.

Main Success Scenario:

1. User clicked 'Search Events' button.
2. Systems loads the search events page.
3. User enters in category to search.
4. System loads all events in database under said category.
5. User can view events details.
6. User can follow the event.
7. User can change the search category.

Extension:

5a. System finds there is no events that match that category.

5b. System notifies the user that there aren't any events related to the category searched.

MyEvents

Brief Use Case: MyEvents

From the events page, users can view events they created. The system loads all events with the creator username matches the logged in username. Here the user can view the events page, which was mentioned above, edit an event, can an event and send out a notification reminder.
Upon the user pressing edit event, they are redirected to a form page.
Upon the user pressing send notification, the system gathers all emails related to the event and sends it to the API. The API sends out an email to all users' following the event reminding them of the details.
Upon the user selecting cancel event, the system sets the status in the database to false. This stops the event from being displayed on the app.

Detailed Use Case: Login Menu

Actors:

User, App, API, Database

Brief Description:

From the events page, users can view events they created. The system loads all events with the creator username matches the logged in username. Here the user can view the events page, which was mentioned above, edit an event, can an event and send out a notification reminder.
Upon the user pressing edit event, they are redirected to a form page.
Upon the user pressing send notification, the system gathers all emails related to the event and sends it to the API. The API sends out an email to all users' following the event reminding them of the details.
Upon the user selecting cancel event, the system sets the status in the database to false. This stops the event from being displayed on the app.

Main Success Scenario:

1. The system renders the MyEvents page when the user clicks the button on the 'events' page.
2. The user can browse their upcoming events.
3. The user can click on edit event, which will load the required form.
4. The user can click on send notification, which will send out a reminder email to all lowing this event.
5. The user can cancel the event by clicking the cancel button on the event.

Extensions:

4a. The system already sent a notification.
4b. The system notifies the user.
4c. The system redirects to MyEvents page.

Edit Event

Brief Use Case: Edit Profile

The user clicks on the edit event button on the MyEvents page. A form will then open for the user to fill out. Once filled out they save the details where the system validates the form. Once validated, the database will update and the user will be redirected back to the MyEvents page.

Detailed Use Case: Edit Event

Actors:

User, App, API, Database

Brief Description:

The user clicks on the edit event button on the MyEvents page. A form will then open for the user to fill out. Once filled out they save the details where the system validates the form. Once validated, the database will update and the user will be redirected back to the MyEvents page.

Main Success Scenario:

1. User clicked the 'Edit Event' button.
2. System loads the edit event form.
3. User fills out form.
4. System checks all the fields meet the criteria.
5. System updates the database
6. User redirected back to MyEvents page.

Extension:

4a. Details don't meet criteria.
4b. System notifies user where error is.
4c. User fixes mistake.

Connections

Brief Use Case: Connections

On the Main Menu, the user will select the Connections button. Here they will view connections that they follow. When connected to another user, they can view their profile, which was mentioned above, and they can also unfollow the user if they choose.

Detailed Use Case: Connections

Actors:

User, App, API, Database

Brief Description:

On the Main Menu, the user will select the Connections button. Here they will view connections that they follow. When connected to another user, they can view their profile, which was mentioned above, and they can also unfollow the user if they choose.

Main Success Scenario:

1. User clicks Connections button.
2. User scrolls through Connections list.
3. User can view connections account.
4. User can unfollow account.

Extension:

4a. User notified an error offered.
4b. User redirected back to Main Menu.

Search Connections

Brief Use Case: Search Connections

Begins when the user enters the page from the Connections page. Here the user can search for connections based on a category of interest. The user can scroll through the list of connections and follow them based on the information shown if they choose. They can also change the search whenever they wish.

Detailed Use Case: Search Connections

Actors:

User, App, API, Database

Brief Description:

Begins when the user enters the page from the Connections page. Here the user can search for connections based on a category of interest. The user can scroll through the list of connections and follow them based on the information shown if they choose. They can also change the search whenever they wish.

Main Success Scenario:

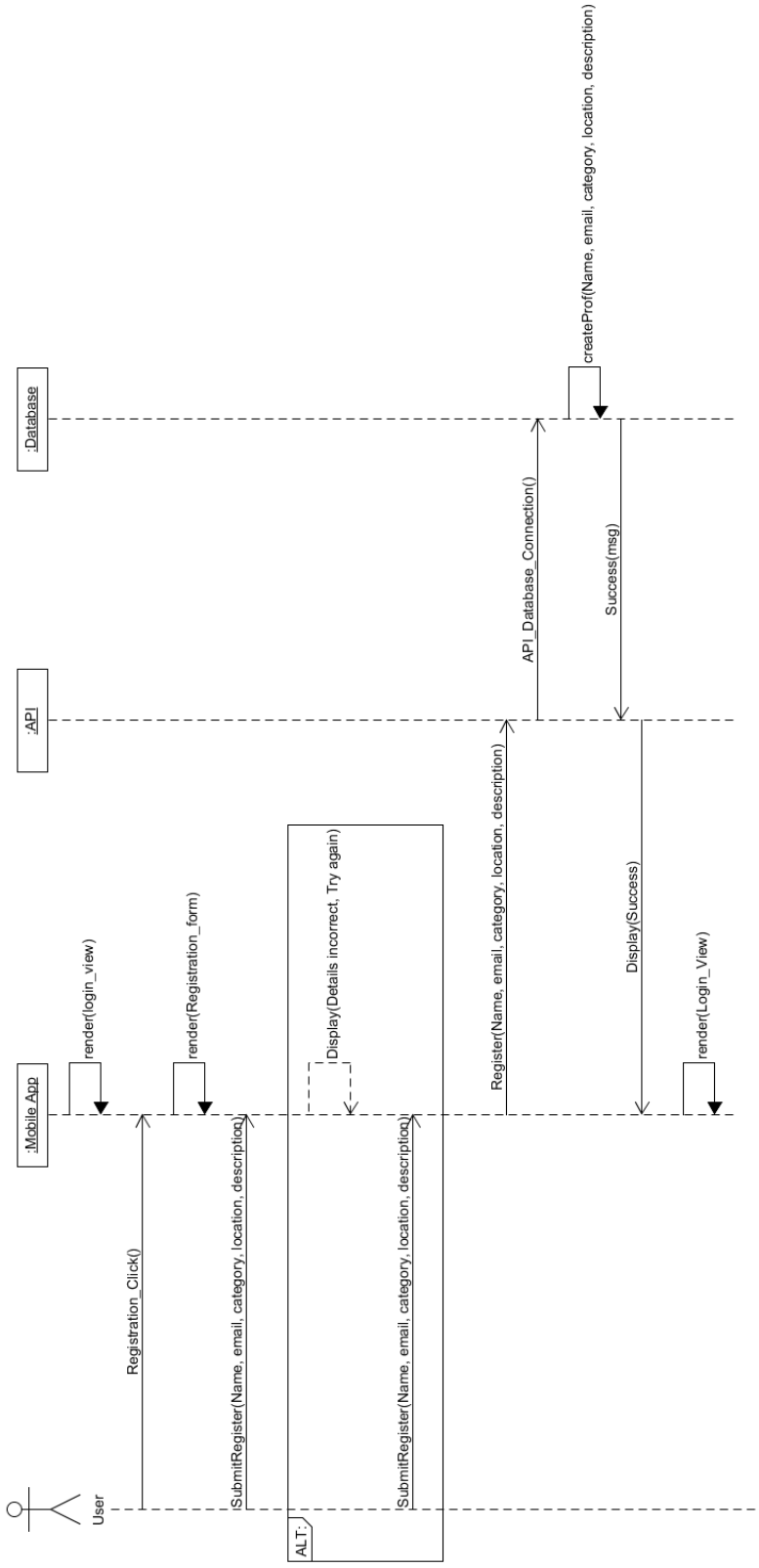
1. User presses the 'Search Connections' button.
2. The system all possible connections related to the search category.
3. User clicks follow on a connection.
4. User changed the search category.

Extension:

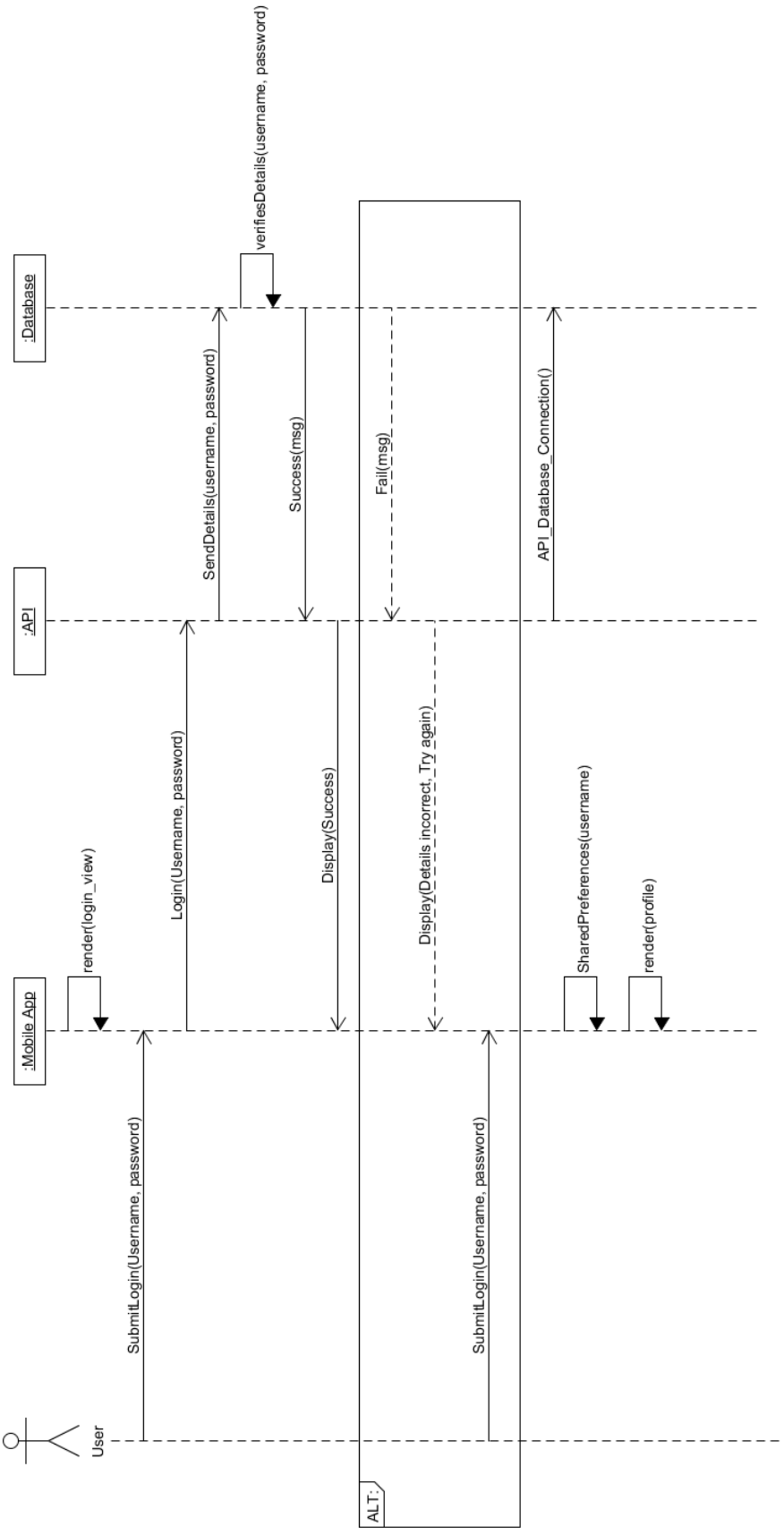
4a. User already following connection.
4b. The system notifies the user that they are already following the connection.
4c. User returned to 'Search Connection' page.

System Sequence Diagrams

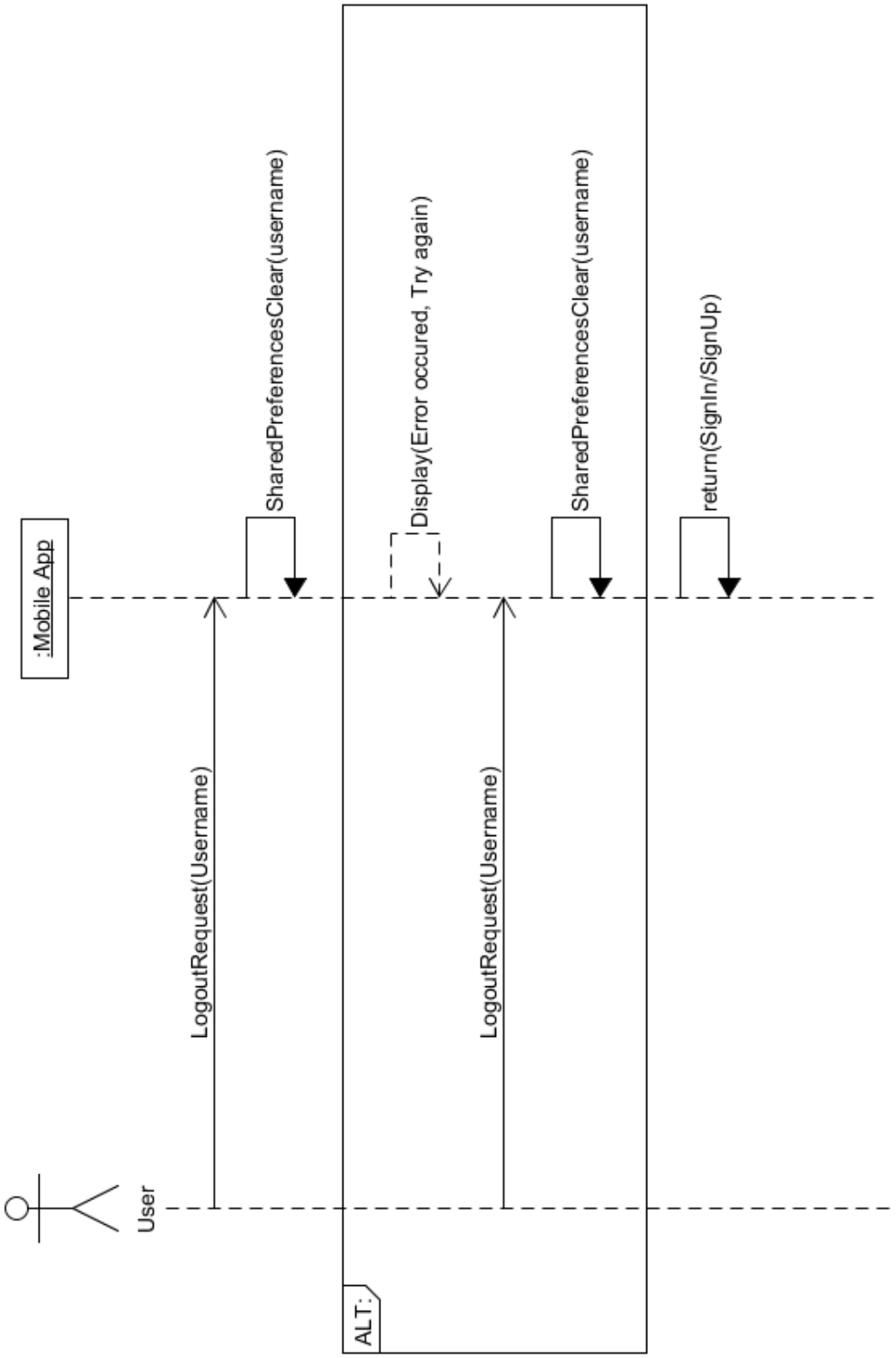
Registration



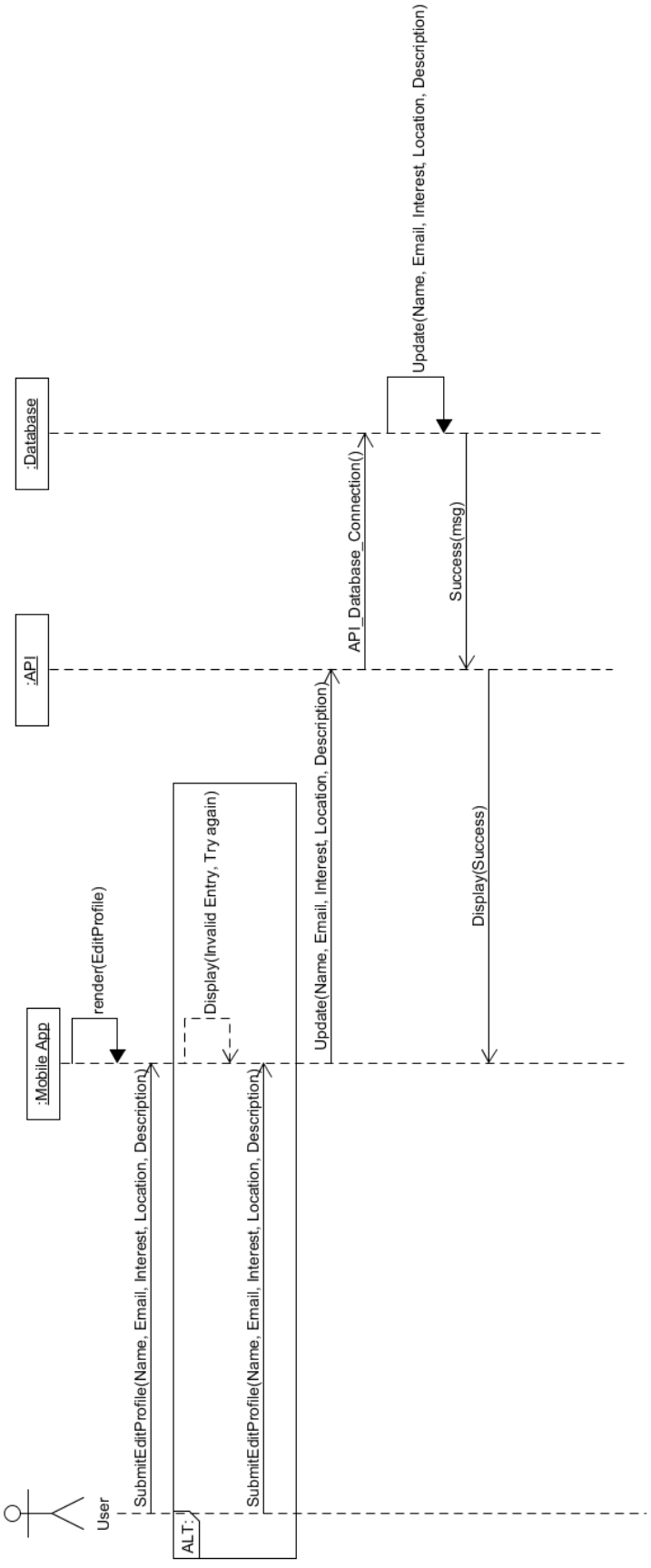
Login


:Database
:API
:Mobile App
User
verifiesDetails(username, password)
SendDetails(username, password)
Success(msg)
Fail(msg)
API_Database_Connection()
render(login_view)
Login(Username, password)
Display(Success)
Display(Details incorrect, Try again)
SharedPreferences(username)
render(profile)
SubmitLogin(Username, password)
SubmitLogin(Username, password)
ALT:

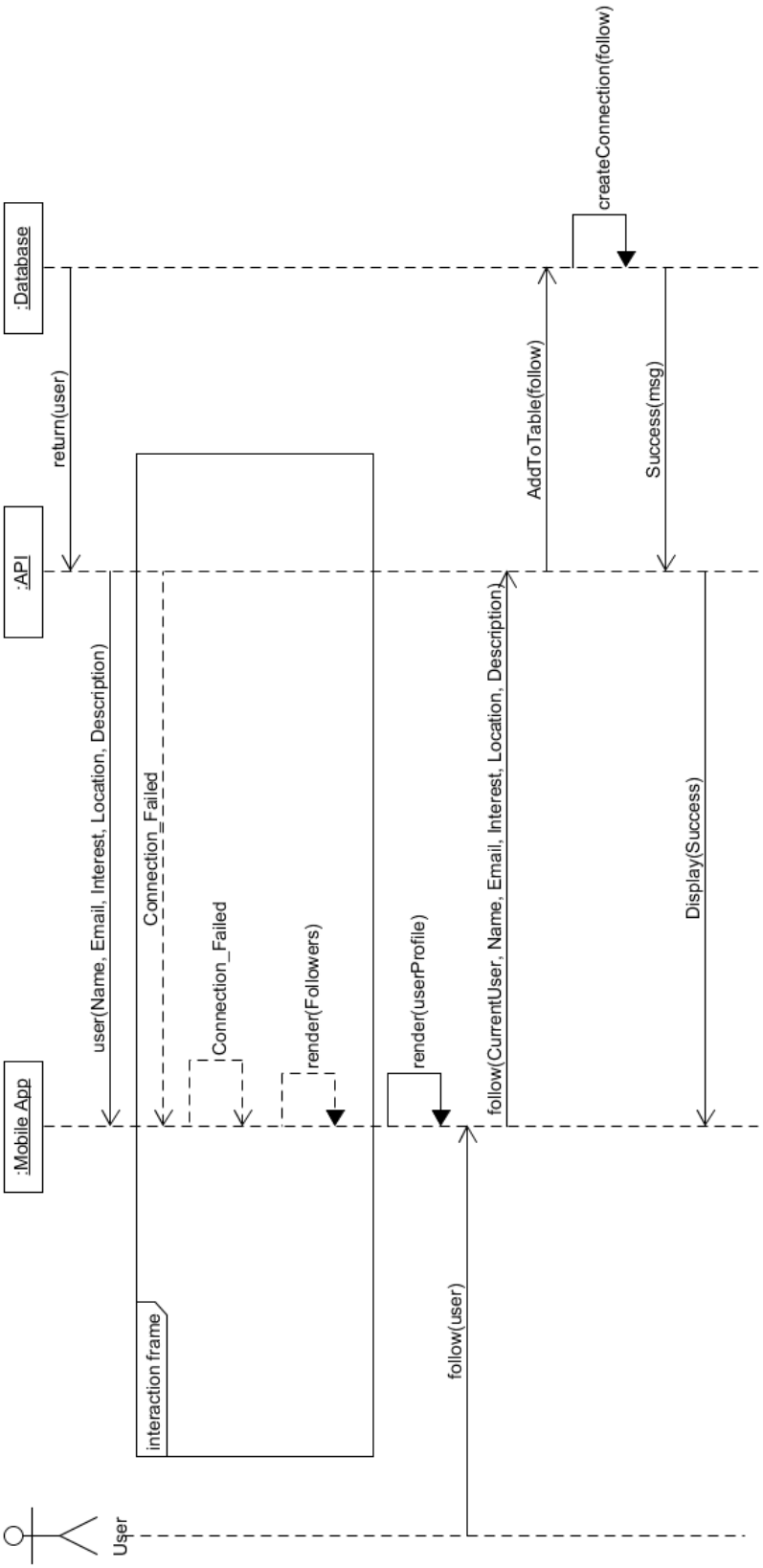
Logout


:Mobile App
User
SharedPreferencesClear(username)
Display(Error occured, Try again)
SharedPreferencesClear(username)
return(SignIn/SignUp)
LogoutRequest(Username)
LogoutRequest(Username)
ALT:

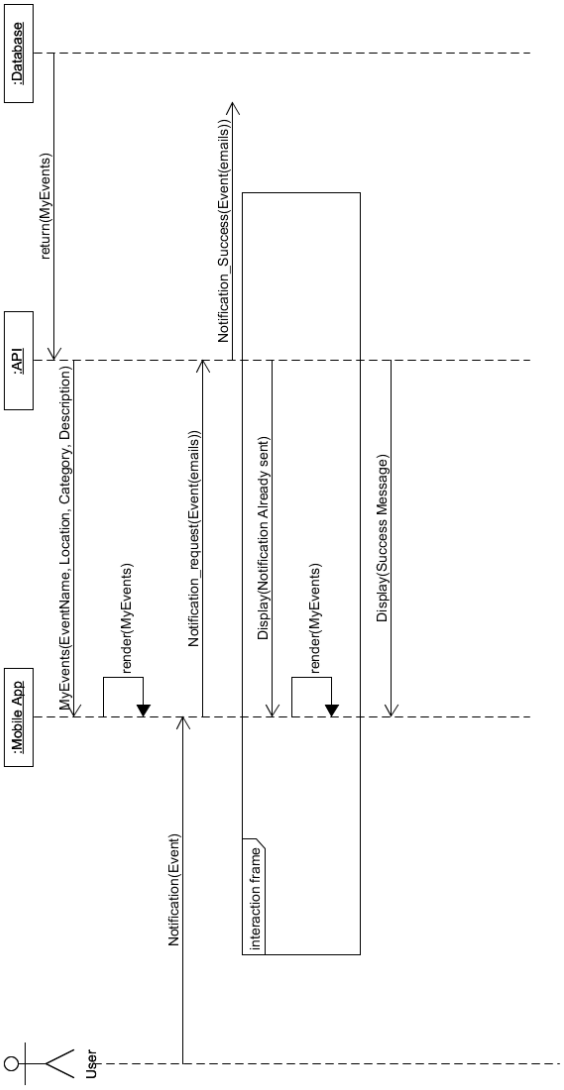
Update Profile



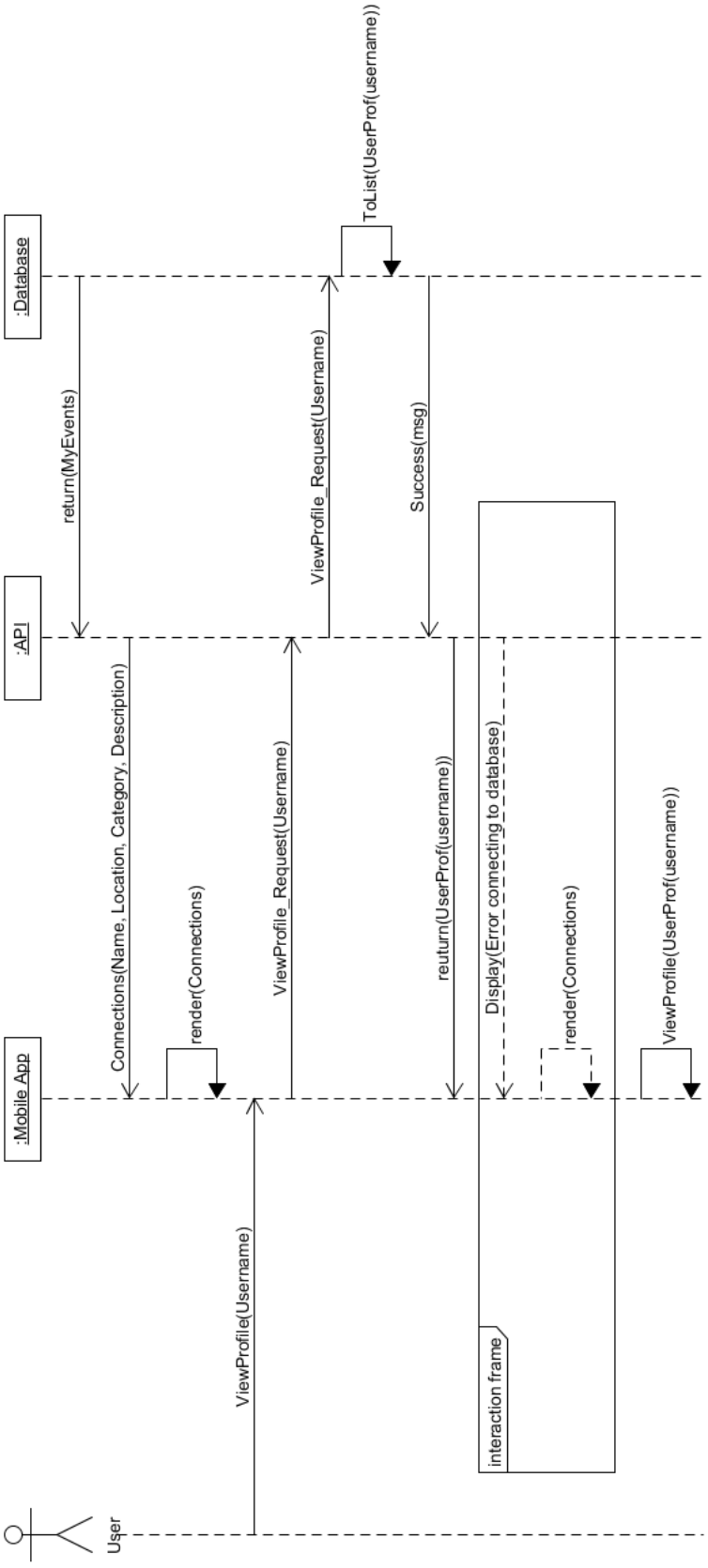
View Profile



MyEvents


:Database
:API
:Mobile App
User
return(MyEvents)
MyEvents(EventName, Location, Category, Description)
render(MyEvents)
Notification_request(Event(emails))
Notification_Success(Event(emails))
Display(Notification Already sent)
render(MyEvents)
Display(Success Message)
interaction frame
Notification(Event)

Connections



Search Connections