**Use cases**

**For**

**BhaagM otayBhaag app.**

**Group 8**

**Rabee Sohail Malik**

**Ahmed Shahid**

**Dilawar Khan**

**Hassan Ishtiaq**

|  |  |
| --- | --- |
| **Primary Actor** | **Use Cases** |
| User | 1. Login |
| 2. Start GPS tracking |
| 3. Play Music |
| 4. Create Playlist |
| 5. Delete Playlist |
| 6. Create Training Regime |
| 7. Open Training Regime |
| 8. Post on chat-room |
| 9. View past history |
| 10. Create user account |

|  |  |
| --- | --- |
| Use Case ID: | 1 |
| Use Case Name: | Login |
| Actors: | User |
| Description: | User logs in to his account on starting the app. |
| Preconditions: | 1. User account has been created |
| Postconditions: | 1. Server notifies app of successful login 2. User gets access to main menu and past history. |
| Normal Flow: | **1.0 Login to account**   1. User enters username and password on login screen 2. App sends username and password string to server 3. Server queries username and password against stored database. 4. Server notifies app of successful login 5. Server sends unique token to app 6. Main menu opens up |
| Exceptions: | **1.0.E.1 Invalid username/password**   1. User enters username and password on login screen 2. App sends username and password string to server 3. Server queries username and password against stored database. 4. Server notifies app of unsuccessful login. 5. App informs user of invalid username/password and requests user for another username/password   **1.0.E.2 No network connectivity** (at step 1)   1. User enters username and password on login screen 2. No server response within 10 seconds 3. App notifies requests user to check internet connection |
| Includes: | None |
| Priority: | High |
| Frequency of Use: | One use per user on every start of app. |
| Assumptions: | 1. Server is always online. |

|  |  |
| --- | --- |
| Use Case ID: | 2 |
| Use Case Name: | Start GPS tracking |
| Actors: | User |
| Description: | A user starts the GPS tracking of the app before starting workout. |
| Preconditions: | 1. User is logged into his online account |
| Postconditions: | 1. The average speed and distance covered is uploaded to the server. |
| Normal Flow: | **2.0 Start GPS tracking**   1. Before starting workout, the user selects the “Start GPS” button from the main menu. 2. Upon completion of workout, user stops the GPS. |
| Alternative Flows: | **2.1 User pauses the GPS**   1. User pauses the GPS. 2. User resumes the GPS. 3. Upon completion of workout, user stops the GPS. |
| Exceptions: | **2.0.E.1 Phone GPS not accessible** (at step 1)   1. User is notified that the phone GPS does not have a GPS. |
| Includes: | Login |
| Priority: | High |
| Frequency of Use: | One use per user on every start of app. |
| Assumptions: | 1. Assume that if the phone has GPS, the GPS service is always available. |

|  |  |
| --- | --- |
| Use Case ID: | 3 |
| Use Case Name: | Play Music |
| Actors: | User |
| Description: | The user can play music using the app. The music player runs in the background and does not interfere with the rest of the functionality of the app |
| Preconditions: | User is logged into his online account. |
| Normal Flow: | **3.0 Play music**   1. User selects the play music option from the main menu. 2. User selects the song to be played from his collection. 3. User stops the song when done listening to music |
| Alternative Flows: | **3.1 Play music from playlist**   1. User selects the play music option from the main menu. 2. User selects the song to be played from his collection. 3. User pauses the song. 4. User resumes the song. 5. User stops the song when done listening to music. |
| Exceptions: | None |
| Includes: | Login |
| Priority: | Medium |
| Frequency of Use: | Once per app startup |
| Assumptions: | None |

|  |  |
| --- | --- |
| Use Case ID: | 4 |
| Use Case Name: | Create Playlist |
| Actors: | User |
| Description: | User can create a playlist consisting of songs already present in his collection. |
| Preconditions: | Songs exist in user’s collection |
| Postconditions: | A playlist is saved in the phone. |
| Normal Flow: | **4.0 Create new playlist**   1. User selects the create playlist option on main menu 2. App prompts user to enter playlist name. User enters name. 3. App shows user’s song collection. User selects songs by selecting checkbox next to songs to be added. 4. User selects the ‘add to playlist’ button. 5. User selects the ‘save playlist’ button. |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | Login |
| Priority: | Low |
| Frequency of Use: | Approximately 1 per user of app. |
| Assumptions: | None |

|  |  |
| --- | --- |
| Use Case ID: | 5 |
| Use Case Name: | Delete Playlist |
| Actors: | User |
| Description: | User deletes an existing playlist. |
| Preconditions: | Playlist already exists. |
| Postconditions: | The playlist is deleted from the app. |
| Normal Flow: | **5.0 Create playlist**  1. The user selects the delete playlist option from the menu.  2. The user selects the playlist to be deleted from a list and selects the ‘delete’ button. |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | Login |
| Priority: | Low |
| Frequency of Use: | Approximately once a month per user |
| Assumptions: | None |
| Use Case ID: | 6 |
| Use Case Name: | Create Training Regime |
| Actors: | User |
| Description: | User can Create a daily training Regime. The user will be notified of training regime. |
| Preconditions: | User is logged into the system. |
| Postconditions: | The Training Regime |
| Normal Flow: | **6.0 Create Training Regime**   1. From the main menu the user selects the create training regime option. 2. The app then asks the user to select time period in days for training regime. 3. The app asks the user for the daily target. 4. The app asks the user of time to be notified. 5. The Regime is uploaded to the server. 6. Server sends confirmation that the regime has been created. 7. Once the regime is created the user is notified at the specified time. |
| Alternative Flows: | None |
| Exceptions: | **6.0.E.1 Server is not online**  1. If server does not respond for 5 seconds then the regime is not created.  2. User Notified that the regime was not created successfully.  3. User prompted to try again later.  **6.0.E.2 Overlapping Regime**  1. Regime that is sent to the server is overlapping with an existing regime.  2. Server notifies the app that the Regime is overlapping.  3. App asks user to overwrite or delete the new Regime.  4. Server Sent request to overwrite if overwrite is selected.  5. Server overwrites the regime and notifies the app.  6. App notifies the user. |
| Includes: | Login |
| Priority: | Medium |
| Frequency of Use: | Once per month per user on average |

|  |  |
| --- | --- |
| Use Case ID: | 7 |
| Use Case Name: | Open Training Regime |
| Actors: | Patron, Payroll System |
| Description: | An existing Training Regime is viewed by the user. |
| Preconditions: | 1. User is logged in.  2. Training Regime Exists. |
| Postconditions: | None |
| Normal Flow: | View Training Regime  1. User Selects the view Training Regime option from the main menu.  2. App sends request to server to display training regime.  3. Server sends Regime to the app.  4. App displays the Training Regime. |
| Alternative Flows: | None |
| Exceptions: | **7.0.E.1 Server is offline**  1. If server does not respond for 5 seconds then the regime is not displayed.  2. User Notified that the regime was not found.  3. User prompted to try again later. |
| Includes: | Login |
| Priority: | Medium |
| Frequency of Use: | Once per day per user on average. |

|  |  |
| --- | --- |
| Use Case ID: | 8 |
| Use Case Name: | Post on chat-room |
| Actors: | User |
| Description: | User views chat, and is able to post on chat room. |
| Preconditions: | User is logged in. |
| Postconditions: | Message posted is stored in the server. |
| Normal Flow: | **8.0 Post on Chat-Room**  1. User Selects the chat-room option from the main menu.  2. The app sends message to server to display chat messages.  3. Server receives request and sends chat history to the app.  4. App displays chat messages.  5. User enter text into the text box.  6. Message sent to the server.  7. Message Saved on server. |
| Includes: | Login |
| Priority: | Medium |
| Frequency of Use: | Once per day per user on average |

|  |  |
| --- | --- |
| Use Case ID: | 9 |
| Use Case Name: | View History |
| Actors: | User |
| Description: | User views his past workout history which includes average running speed, calories burnt and distance covered. |
| Preconditions: | User has used the app at least once in the previous month. |
| Postconditions: | None |
| Normal Flow: | **9.0 View History**  1. The user selects the view history option from the menu.  2. The user is presented with both the weekly history and monthly history. |
| Alternative Flows: | None |
| Exceptions: | None |
| Includes: | Login |
| Priority: | Low |
| Frequency of Use: | Approximately once a week per user |
| Assumptions: | None |

|  |  |
| --- | --- |
| Use Case ID: | 10 |
| Use Case Name: | Create user account |
| Actors: | User |
| Description: | A first time user creates a new account, which is necessary for the use of the app. |
| Preconditions: | None. |
| Postconditions: | A user account is created. The relevant username and password is saved in the online database. |
| Normal Flow: | **10.0 Create user account**  1. The user selects the create account button from the login menu.  2. The user enters the desired username and password.  3. The account is created and the relevant username and password is added in the online database. |
| Alternative Flows: | None |
| Exceptions: | **10a Username or password already in use.**  1. The user enters the desired username  2. The online server checks the username database for duplicates. Server sends error message to app.  3. The user is prompted to enter a new username. |
| Includes: | Login |
| Priority: | Low |
| Frequency of Use: | Approximately once a month per user |
| Assumptions: | None |