US02.01.01

As an owner, I want to track people I know. Adding a textual username should be enough.

Use Case 2.1:

This use case describes how the user can add textual usernames to track people.

Actors: Owner(user), App Database

Preconditions: There is an active network connection with the database, User is using the app, User is authenticated

Flow of events:

1) User opens up the app

2) User goes to the friends screen

3) User adds another user by username to tracking list

4) App goes through database to check that it is a valid user

5) The user exists

6) The user now has the other user on the tracking list

Alternate Flows:

* If in step 4 the user doesn’t exist:

4.1) The user doesn’t exist

7.1) App prevents user from adding the other user to the tracking list

8.1) App prompts user of invalid username

9.1) App goes back to …screen

US02.02.01

As an owner, I want to add friends

Use Case 2.2:

This use case describes how the user can add friends through their username

Actors: Owner(user), Other Owner/user, App Database

Preconditions: There is an active network connection with the database, User is using the app, User is authenticated

Flow of events:

1) User opens up the app

2) User goes to friends screen

3) User adds another user by username

4) App goes through database to check that it is a valid user

5) The user exists

6) The request is then sent to the other user for approval

7) The request is approved by the other user

8) Both users are now friends

Alternate Flows:

* If in step 4 the user doesn’t exist:

4.1) The user doesn’t exist

7.1) App prevents user from adding the other user to the tracking list

8.1) App prompts user of invalid username

9.1) App goes back to friends screen

* If in step 7 the user doing the adding cancels the request

7.2) The request is then cancelled by sending user

8.2) The request is then deleted, preventing the other user from responding to it

* If in step 7 the user doing the adding cancels the request

7.3) The request is denied by the other user

8.3) The request is then deleted, and the adding user gets notified of the denial

US02.03.01

As an owner, I want to remove friends

Use Case 2.3:

This use case describes how the user can delete friends through their username/friend list

Actors: Owner(user), Other Owner/user, App Database

Preconditions: There is an active network connection with the database, User is using the app, User is authenticated. Users must be friends.

Flow of events:

1) User opens up the app

2) User goes to friends screen

3) User inputs the username to delete

4) App goes through database to check that it is a valid user and that they’re currently friends

5) The user exists

6) The database then removes that friendship

7) Both users are no longer friends with each other

Alternate Flows:

* If in step 4 the user doesn’t exist:

4.1) The user doesn’t exist

7.1) App prevents user from adding the other user to the tracking list

8.1) App prompts user of invalid username

9.1) App goes back to the friends screen

* If in step 7 the user doing the adding cancels the request

7.2) The request is then cancelled by sending user

8.2) The request is then deleted, preventing the other user from responding to it

* If in step 2 the user goes to friend list instead

2.3) User goes to their friends list screen

3.3) User holds and selects delete on the user

4.3) The database then removes that friendship

5.3) Both users are no longer friends with each other

US02.04.01

As an owner or borrower, I will have a profile where by my contact information and city are recorded.

Use Case #2.4:

This use case describes how the user can record contact information and city in their profile

Actors: Owner/Borrower(user), App Database

Preconditions: There is an active network connection with the database, User is using the app, User is authenticated.

Flow of events:

1) User opens up the app

2) User goes profile screen

3) User selects the edit button

4) User can then modify their profile and city

US02.05.01

As an owner or borrower, I will be able to view the profile of anyone I know ofincluding friends.

Use Case 2.5:

This use case describes how the user can delete friends through their username/friend list

Actors: Owner/borrower(user), Other user, App Database

Preconditions: There is an active network connection with the database, User is using the app, User is authenticated, other user exists

Flow of events:

1) User opens up the app

2) User goes to search screen

3) User inputs the username to view

4) App goes through database to check that it is a valid user

5) The user exists

6) The database then gets their inventory

7) You can then view the inventory

Alternate Flows:

* If in step 4 the user doesn’t exist:

5.1) The user doesn’t exist

6.1) App prevents user from viewing inventory

7.1) App prompts user of invalid username

8.1) App goes back to search screen

* If in step 2 the user goes to friend list instead

2.3) User goes to their friends list screen

3.3) User selects the user

4.3) The database then gets their inventory

5.3) Their inventory is then shown

US03.01.01

As an borrower, I want to search the inventories of my friends.

Use Case 3.1:

This use case describes how the user can search the inventories of their friends

Actors: Borrower(user), App Database, friend

Preconditions: There is an active network connection with the database, User is using the app, User is authenticated, the other user is a friend

Flow of events:

1) User opens up the app

2) User goes to their friends screen

3) User selects on a friend

3) App goes to the friends inventory screen

4) Database shows that users items

US03.01.02

As an borrower, I want to search/browse the inventories of my friends by category.

Use Case 3.2:

This use case describes how the user can search the inventories of their friends by category

Actors: Borrower(user), App Database, friend

Preconditions: There is an active network connection with the database, User is using the app, User is authenticated, the other user is a friend

Flow of events:

1) User opens up the app

2) User goes to their browse screen

3) User selects on a friend

3) App goes to the friends inventory screen

4) Database shows that users items

5) User selects the category wanted

6) View shows only those items with that category

US03.01.03

As an borrower, I want to search/browse the inventories of my friends by textual query..

Use Case 3.3:

This use case describes how the user can search the inventories of their friends with a query

Actors: Borrower(user), App Database, friend

Preconditions: There is an active network connection with the database, User is using the app, User is authenticated, the other user is a friend

Flow of events:

1) User opens up the app

2) User goes to their browse screen

3) User selects on a friend

3) App goes to the friends inventory screen

4) Database shows that users items

5) User searches with a query on the search bar

6) View shows only those items matching that query

US03.02.01

As an owner, any of my publicly shared items will be browseable / searchable by my friends.

Use Case 3.4:

This use case describes how the a friend can view a publicly shared item of yours

Actors: User, App Database, friend

Preconditions: There is an active network connection with the database, User is using the app, User is authenticated, the other user is a friend

Flow of events:

1) Friend opens up the app

2) User goes to their friends screen

3) User selects on you (the user)

3) App goes to your inventory screen

4) Database shows your items that are public