CS 2340, Milestone 4

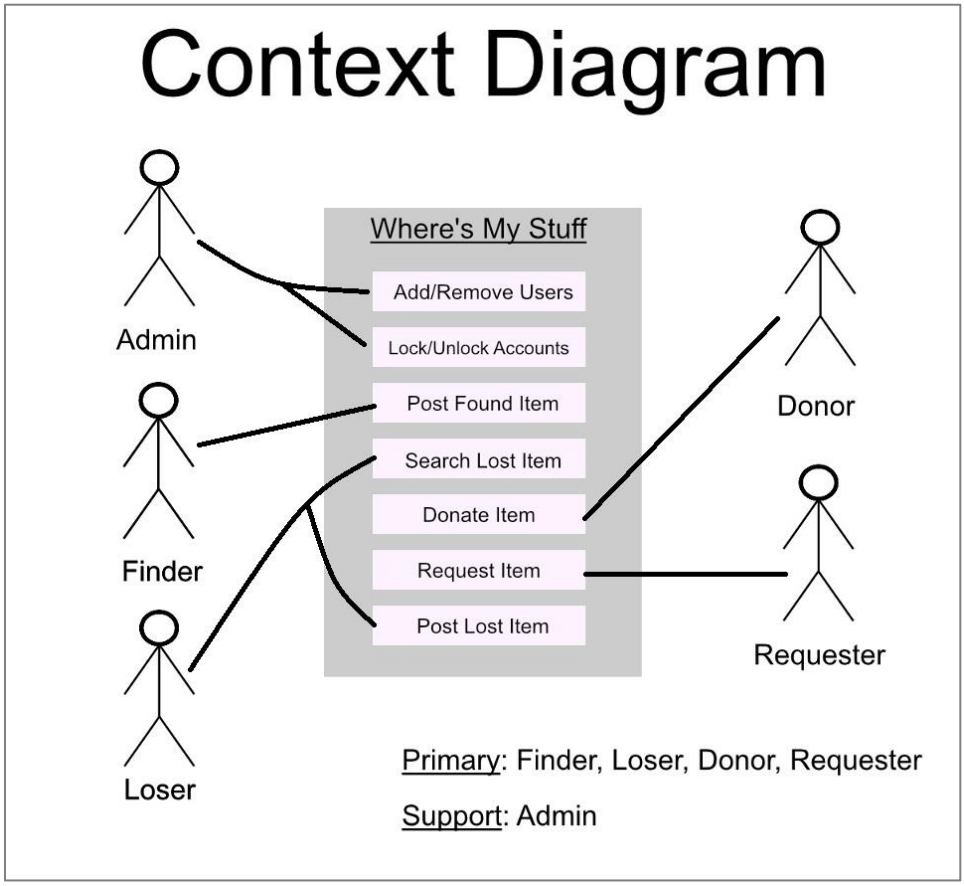
M4 – Project Setup / Login / Logout and User Stories

By: **Group 18**

Section A

Summer Semester 2017

Context Diagram



User Stories

1. **(Simola)** As a first-time user, I want to be able to register on the app, so that I can post lost or found items.
2. **(Matthieu)** As a returning user, I want to be able to login, so that I can check the status of my lost or found items.
3. **(Sharence)** As a user who found something important, I want to be able to report it on the "Found Page", so that people can find it.
4. As a user who lost something, I want to be able to search the "Found Page" for it, so that I can contact the one who found it.
5. As a user who could not find my lost item on the Found page, I want to be able to post on the "Lost Page", so that people can contact me if they found it.
6. **(Alexander)** As an admin, I want to be able to edit the found page and remove items, so that I can remove inappropriate/Spam/old items from the list.
7. **(Ash)** As an admin, I want to be able to look through all the users, so that I can ban ones posting inappropriate thing.
8. As a user who previously posted a found item, I want to be able to remove it from the "Found Page", so that it isn't on the list after the owner got it back.
9. As a user who's seen something found by another user, I want to be able to update its location (if it's a moving item), so people can track it.
10. As a user who needs an item after the disaster, I want to be able to request an item even if I haven't lost it, so that I can get an item I need.
11. As a user who found matching items to my lost item, I want to be able to filter them by date, location, and "status", so that I can find the right one.
12. As a returning user, I want to be able to make donations, so that I can help those in need after a disaster.
13. As a returning user, I want to be able to request an item even if I haven't lost it, so that I can obtain an item I need after a disaster.

Individual User Story Development

User Story 1: Simola Nayak

User Story

"As a first time user, I want to be able to register on the app, so that I can post lost or found items."

Work Tasks

- Create new account based on the following text fields:
    - Username
    - Password
    - Location/Hometown
- Verify that the username is not already taken, and suggest another similar one
- that isn't (if the user suggests one that is).
- Create an option to gather information from social media for a faster sign-up.
- Allow users to access the new-user sign in screen from the existing users screen in two ways:
    - After passing in a nonexistent username
    - Through a message for new users to sign up

Acceptance Scenarios

<u>How the user gets there:</u>

GIVEN: The user goes to the login screen for returning users.

WHEN: The user tries to pass a nonexistent email address or username into the login, and proceeds with it (presses OK).

THEN: System prompts user with the message "Sorry, that username doesn't exist."

Along with that comes the option to create a username: "Sign up here."

WHEN: The user clicks on said link.

THEN: The user goes to the new sign up activity, except the field for a new username is pre-filled (but still editable).

WHEN: The user clicks on a link to the new user sign up activity (says

something like "New users sign up here").

THEN: The user goes to the new user sign up activity.

GIVEN: The very first boot of the app.

WHEN: The user goes through the tutorial.

THEN: Allow the user to sign up (default) or (SEE RETURNING USER) go to the sign-in screen for returning users.

WHEN: The user skips the tutorial (or exits at any point).

THEN: Allow the user to sign up (default) or (SEE RETURNING USER) go to the sign-in screen for returning users.

How the user signs up:

GIVEN: The user lands on the sign-up page.

WHEN: The user has already passed in a user name before landing on the sign-up page.

THEN: The user name field is already filled but the user can edit it and proceed with the password and contact info.

WHEN: The user tries to pass in a user name that's already taken.

THEN: Suggest a new user name that appends a number to the end of what the user initially passed in.

WHEN: The user passes in and confirms an invalid password.

THEN: Stay on the page, clear the password and confirmation fields, display a message outlining the password requirements and put a red outline around the box.

WHEN: The user passes in a valid (non-taken) username and password.

THEN: Go to the page for the next action, which is a choice between reporting a missing item, reporting a sighting of an item, and donating.

User Story 2: Matthieu Capuano

User Story

"As a returning user, I want to be able to login, so that I can check the status of my lost or found items."

Work Tasks

- Implement login activity (done)
- Implement check of username and password when logging in
    - Requires implementation of database that stores all usernames and passwords as key-value pairs (separate user story)
    - Check username from database of valid usernames
    - Check password from database for that specific username
- Implement option on the user's page/account to allow him to view:
    - His/her found items
    - His/her lost items
- Implement viewable list of **lost** items. Each item, while in the list, should show:
    - Name
    - Short description
    - Date posted
    - Author who posted it
    - Contact of author
    - Location (after Google maps implementation)
    - Show the user's lost items at the top or differentiate it in some way
- Implement viewable list of **found** items. Each item, while in the list, should show:
    - Same as above
- Items in the lost and found item lists should be "tap-able" so more info can be seen

Acceptance Scenarios

<u>Scenario 1: The user tries to log in correctly</u>

- GIVEN: A user is viewing the login page for the app
    - AND the user has an account

- WHEN: The user enters a correct username and password for his account
- THEN: The user is logged in and can see the main page of the app from which he can access everything else (including his/her page)

Scenario 2: The user tries to log in incorrectly

- GIVEN: A user is viewing the login page for the app
- WHEN: The user enters an incorrect/no username or password
- THEN: The app gives a warning saying that the username and/or password is/are incorrect

Scenario 3: A user tries to view the "Lost Items" (and "Found Items") page

- GIVEN: A user is viewing the lost items page
    - AND there are enough items on the page that it can be scrolled up and down
- WHEN: The user slides his/her finger up or down the list
- THEN: The list scrolls up or down correctly
    - Same acceptance scenario for the found items page

Scenario 4: A user tries to view the details of an item

- GIVEN: A user is viewing the lost/found items page
    - AND there are items in the list with more details to view
- WHEN: The user taps on an item
- THEN: The app switches screen to show more details about the item

Done Done Criteria

- All acceptance criteria are met
    - Must have been tested several times by other developers
- Code must also be reviewed by at least one other developer
- Code has been tested on multiple android emulators
    - And/or multiple android phones
    - And/or multiple computers (in case the code accidentally contains parts specific to the original coder's settings)

User Story 3: Sharence Solomero

User Story

"As a user who found something important, I want to be able to report it on the 'Found Page', so that the owner can find it."

Work Tasks

- Implement Login and Registration
- Create found page that can hold a list of items
- Allow user to add an item to the list
- Each item should have its own page when selected with specific details
- Code UI

Acceptance Scenarios

Scenario 1:

Given Robert finds a watch at a park bench and wants to report it

And Robert is able to login user for the Where's My Stuff app

When Robert adds an item to the found page

Then the found page list should contain the item Robert just posted

Scenario 2:

Given Robert finds a watch at a park bench and wants to report it

And Robert forgets his account password and tries to login more than 3 times

When Robert tries to login again

Then his account is locked and he is not able to report lost item until Admin unlocks his account

Done Done Criteria

- User able to post to a found page and data is retained and can be viewed by another user
- Login and Registration implemented (Locking an account not required)
- UI can be easily understood
- Code reviewed by at least two team members

User Story 6: Alexander Wilkins

User Story

"As an admin, I want to be able to edit the found page and lost page and be able to remove items, so that I can remove inappropriate/Spam/old items from the list."

Work Tasks

- Allow Admin to view found page
- Allow Admin to view lost page
- Verify that user is an Admin
- Allow Admin to view a given found item's information including post date, description, the user who posted, etc.
- Allow Admin to view a given lost item's information including post date, description, the user who posted, etc.
- Give Admin the ability to view both the found page and the lost page in an editable format (one that regular users are not privy to)
- Give Admin the ability to remove an item from either list
- Either a lost or found item
- Give the Admin the ability to save changes made to the found or lost page and update that said page for all users

Acceptance Scenarios

<u>Scenario 1: An Admin wants to remove an item from the found page</u>

GIVEN there are items posted on the found page

And the user wishing to remove an item is signed in as an Admin

WHEN the Admin requests to edit the found page

And deletes remove an item

And saves changes

And exits editor

THEN the item is removed from the found page for the Admin

And the item is removed from the found page for all users

Scenario 2: An Admin wants to remove an item from the lost page

GIVEN there are items posted on the lost page

And the user wishing to remove an item is signed in as an Admin

WHEN the Admin requests to edit the lost page

And deletes remove an item

And saves changes

And exits editor

THEN the item is removed from the lost page for the Admin

And the item is removed from the lost page for all users

Scenario 3: An Admin wants to remove an item from the found page or lost page but doesn't save changes

GIVEN there are items posted on the lost page or found page

And the user wishing to remove an item is signed in as an Admin

WHEN the Admin requests to edit the lost page/found page

And deletes an item

And does not save changes

And exits editor

THEN the item is not removed from the page for both the Admin and the users

Scenario 4: An Admin wants to remove an item from either page but changes their mind before saving

GIVEN there are items posted on the lost page / found page

And the user wishing to remove an item is signed in as an Admin

WHEN the Admin requests to edit the lost page/ found page

And deletes an item

And changes their mind

And reverses change through an undo button

And saves changes

And exits editor

THEN the item is not removed from the page for both the Admin and the users

Scenario 5: An Admin wants to remove an item from either page but changes their mind after saving

GIVEN there are items posted on the lost page / found page

And the user wishing to remove an item is signed in as an Admin

WHEN the Admin requests to edit the lost page/ found page

And deletes an item

And saves changes

And exits editor

And changes their mind

THEN the item is removed from the page for both the Admin and the users

And the data must be reentered into the system

Scenario 6: A user who isn't signed in as an Admin wants to remove an item

GIVEN there are items posted on the lost page/ found page

And the user is not signed in as an Admin

WHEN the user wants to remove an item

THEN a message will pop up telling the user to sign in as an Admin to use this feature

Or there won't be an edit button (or functionality) currently viewable to them

And the item won't be removed from the page for both the Adin and the users

Done Done Criteria

- Acceptance Criteria is met
- Code is reviewed by another team member
- Feature is tested for accessibility

User Story 7: Ash Bhimasani

User Story

"As an admin, I want to be able to look through all the users, so that I can ban ones posting inappropriate things."

Work Tasks:

- Administrator must be able to:
    - Flag posts that are deemed irrelevant or negative to the community. This ability is available to a normal user as well in order to have a self policing community.
    - Delete posts in a similar fashion.
- Create an menu button all posts that displays options. One of the options will be to flag a post for an administrator to review the content of the post.

Acceptance Scenarios:

Scenario 1: How user interact with inappropriate content

GIVEN: The user is successfully logged in and is displayed a feed of lost items.

WHEN: The user sees inappropriate, offensive, or irrelevant postings.

THEN: The user opens the post and selects “Flag Post”

WHEN: User clicks on “Flag Post”, the post is marked for review by administrators

Scenario 2: How administrator interact with flagged post

GIVEN: Administrator is presented list of Flagged posts.

WHEN: Admin clicks on arbitrary inappropriate post.

THEN: Admin is presented option to delete post

WHEN: Admin clicks on Delete Post

THEN: Post is removed from network

WHEN: Admin clicks on Flagged Post but not inappropriate, offensive, or irrelevant.

THEN: Admin is presented option to remove Flag on post

WHEN: Admin clicks on Remove Flag

THEN: Post is no longer flagged and remains in network