LoFo

Purdue Lost & Found Online System

Lofo Project backlog

CS408

Team 20: Heng Li, Yiyan Zhou, Yibin Wu, Ketao Chang, Weichu Hu

Problem Statement

It is always a pain to lose stuffs in campus. And it is nearly impossible to find them back after you realized that you lose them. Sometimes people might pick up the lost stuffs and hand it over to lost-and-found but still, the chance to get it back is small. We are building a system to help people to find lost items. In this system, one can post lost items and found items onto the maps and our system can match relative posts. Also we will provide contact information to the one who lost item and the one who found the item.

Background Information

In campus, those who lost their items always have a hard time finding their items. Sometimes they tend to give up due to some difficulties. For example, there is no Lost-and-found office in the some buildings, and in some case, the finders are not able to contact the owner. Our website provides students with a convenient and secure way to find their lost items. They can view information of lost items in a particular area and contact the finder directly. Same for the finders. After a finder finds a lost item, he can log into our system and write a post about the item he found.

Environment

- Language:HTML & CSS, JavaScript
- Framework: AngularJS
- Libraries: Google Map API, Purdue API
- Database: MongoDB

Functional Requirements

Backlog Id	Functional Requirement	Hours	Status
1	As a user, I want to log in through Purdue account.	10	Planned for sprint 1
2	As a user, I want to create a post.	5	Planned for sprint 1
3	As a user, I want to upload the photo in the post.	10	Planned for sprint 1
4	As a user, I want to have description of lost item and the location found/lost in a post.	6	Planned for sprint 1
5	As a user, I want to have my contact information section in a post.	3	Planned for sprint 1
6	As a user, I want to add tags to a post.	3	Planned for sprint 1
7	As a user, I want to edit my post.	5	Planned for sprint 1
8	As a user , I want to delete an ongoing post and my personal contact information.	5	Planned for sprint 1
9	As a user, I want to see the lost and found items pin on a map.	15	Planned for sprint 2
10	As a user, I want to view either the lost or found post on the map.	4	Planned for sprint 2

11	As a user, I want to view items by tags on the map.	3	Planned for sprint 2
12	As a user, I want to view posts by specific created time interval.	3	Planned for sprint 2
13	As a user, I want to see a list of items when I click on a certain building.	3	Planned for sprint 2
14	As a user, I want to see details of a post when I click on a post in the list.	5	Planned for sprint 2
15	As a user, I want to see my own post history.	6	Planned for sprint 1
16	As a user, I want to verify the identity of each other when a lost and found matched.	8	Planned for sprint 2
17	As a user, I want to see all the solved and expired posts.	5	Planned for sprint 2
18	As a user, I want to report a falsely claim.	10	Planned for sprint 2

Non-Functional Requirements

Security

1. As a developer, I would like to keep users' account information private.

2. As a developer, I would like to store the data of expired and solved cases in database.

3. As a developer, I would like to make a post expired if it is not solved after 30 days.

Appearance (UI / UX)

4. As a user, the application will look good on devices of different sizes (auto-layout).

Reliability

5. As a user, the response time should be fast, all searching time should be kept within 10 seconds.

6. As a developer, the system must handle 10 users at a time.

Use Cases

Case 1: Login

Action	**System Response**
1. Enter the username	
2. Enter the password	
3. Click on ‘Signin’ button	4. Verify with Purdue Api, then a dialog shows up to tell the user whether login is successful or not

Case 2: Create a post

Action	**System Response**
1. Click on ‘Create a post’ button in main page	2. A pop-up appears for the user to enter information about the post
3.Input the detail of the post(See case4~7)	
4. Click on ‘Confirm’ button	5. Dialog appears to show that if the post has been created successfully, and the expired date of this post will be set to 30 days after created. The pop-up disappears and the Main Map updates.

Case 3: Upload a picture

Action	**System Response**
1. Click on ‘Create a post’	2. A pop-up appears for the user to enter information about the post
3.Click on ‘Choose a picture	4. Image open dialog appears
5. Select a image	
6. Confirm selection	7. Dialog disappears and post content updates with image

Case 4: Input description

Action	**System Response**
1. Click on ‘Create a post’	2. A pop-up appears for the user to enter information about the post
3. User can choose to input his information into the ‘Description’ section, and choose a certain building as the location of item found/lost in the pop-up	
4. Click on ‘Confirm’ button	5. Dialog appears to show that if the post has been created successfully, the pop-up disappears and the Main Map updates.

Case 5: Input contact information

Action	**System Response**
1.Click on ‘Create a post’	2. A pop-up appears for the user to enter information about the post
3.User can choose to input his information into the ‘Contact information’ section of the pop-up	
4.Click on ‘Confirm’ button	5. Dialog appears to show that if the post has been created successfully, the pop-up disappears and the Main Map updates.

Case 6: Add tags to posts

Action	**System Response**
1. Click on ‘Create a post’	2. A pop-up appears for the user to enter information about the post
3. Select the tag checkboxes.	
4. Click on ‘Confirm’ button	5. Dialog appears to show that if the post has been created successfully, the pop-up disappears and the Main Map updates.

Case 7: Edit a post

Action	System Response
1. Click on ‘My history’ button	2. Redirect to history post list
3. Click on ‘Edit’ button beside the desired post.	4. A pop-up appears for the user to enter information about the post
5. Edit the detail of the post(See case 4~7)	
6. Click on ‘Update’ button	5. Dialog appears to show that if the post has been created successfully, and the expired date of this post will be set to 30 days after modified. The pop-up disappears and the Main Map updates.

Case 8: Delete an ongoing post.

Action	System Response
1. Click on the ‘My History’ button.	2. ‘My History’ list appears.
3. Click on the ‘Delete’ button besides the desired post.	4. ‘Confirm’ dialog appears.
4. Click on the ‘Confirm’ or ‘Cancel’ button	5. ‘Confirm’ dialog disappears.

Case 9: See both lost and found items on a map.

Action	System Response
1. Assume the user have signed in.	2. A map generated by post location information with pins appears.

Case 10: See either lost or found items on a map.

Action	System Response
1. Click on the ‘Lost&Found’ filter.	2. ‘Lost&Found’ drop-down menu appears
3. Check either or ‘Lost’ or ‘Found’ option	
4. Click on the ‘Update’ button.	5. Main map shows items of selected option

Case 11: View posts by tags

Action	System Response
1. Click on the ‘Tag’ filter	2. ‘Tag’ drop-down menu appears
3. Check item tags (e.g ‘Umbrella’ and ‘Phone’)	
4. Click on the ‘Update button’	5. Main map shows items of selected options and list shows up

Case 12: View posts by time interval

Action	System Response
1. Click on start date or end date button.	2. A calendar shows up.
3. Select start date or end date.	
4. Click on ‘Update’ button	5. Main map shows items of the specific time interval and list shows up

Case 13: View posts by a certain building

Action	System Response
1. Click on a grouped pin of a certain building	2. The list of items in the specific building shows up

Case 14: See post details

Action	System Response
1. Click on certain post title on the post list	2. Redirect to the the page with post details

Case 15: See post history

Action	System Response
1. Click on ‘My history’ button	2. Redirect to history post list

Case 16: Post solved verify

Action	**System Response**
1. Founder and owner meet together offline	
2. Poster log in	
3. Click on ‘My history’ button	4. Redirect to history post list
5. Click on the post title that matched	6. Redirect to the page with post details
7. Click on post solved button	7. Purdue account verify page show up
8. Another person enter Purdue account information	9. Dialog show up verify success or not. Then, if verify is success, post would mark as solved and move to ‘Expired&Solved’ area with contact information deleted.

Case 17: See expired and solved post

Action	**System Response**
1. Click on ‘Expired&Solved’ button	2. Redirect to expired and solved post list

Case 18: Report falsely claim

Action	**System Response**
1. Click on ‘Report’ button besides a solved post	2. A window pops up asking the user to input more details
3. Input description of the case, then click on ‘Submit’ button	4. A window pops up telling the user submit successful or fail.