Project Backlog

Project title: ReportIt
Team #5
Team members: Juncheng Tang, Yingtong Chen, Liang Zhang, Yuting Guo, Yaoxi Liang

1. **Problem Statement:**

Public concerns appear in all around the world and should be seriously taken care of. However, properly managing and resolving those concerns can be a difficult task. Public service agencies need a convenient and well-managed platform to receive and respond to people's need, and the public needs an integrated platform to express their concern to different agencies.

2. **Background Information:**

Many residents in West Lafayette are facing various annoying problems which they want to report to corresponding departments. The reporting process is full of misunderstanding and frustration for both the residents and department officers since the resources support them this process is limited. Residents may have to waste their whole day in the department for a long time due to the lack of officers, and some of them are even not sure which department to report to. The department officers who receive tons of reports every day may mix reporters' information causing problems. All of our team members are residents in West Lafayette and some of us had experience of dealing with the time-consuming reporting process. Therefore, we aim to develop a web app that make this reporting process simple and organized to assist people who want to report case online with simple narration and clicks as well as department officers who want to organize report documents well online.

3. **Environment:**

We decide to build our front end interface with javascript, css, and html. We will be using external libraries such as Bootstrap, jQuery for UI-optimization. Python Django will be our primary tool to develop the backend. Our choice of database is Postgresql which will store all the reporter and agency information. We will also store the concern reports data in our database. We will use Amazon AWS S3 to store media data like images and videos attachment of the concern reports. Finally we will deploy our application to Heroku.

4. **Functional Requirement:**

Backlog ID	Functional Requirement	Hours	Status
01	As a user, I would like to sign up a reporter/agent account	**5**	**Sprint 1**
02	As a user, I would like to have a profile picture	**3**	**Sprint 1**
03	As a user, I would like to be able to change my password	**5**	**Sprint 1**
04	As a user, I would like to reset password when I forget my password	**2**	**Sprint 1**
05	As a user, I would like to view profile of other users.	**4**	**Sprint 2**
06	As a user, I would like to logout my account securely	**2**	**Sprint 1**
07	As a user, I would like to have my profile page	**3**	**Sprint 2**
08	As a user, I would like to view all the concerns	**4**	**Sprint 2**
09	As a user, I would like to change my profile page	**5**	**Sprint 2**
10	As a user, I would like to view all the concerns I sent	**3**	**Sprint 2**
11	As a user, I would like to see timestamps for every concern.	**4**	**Sprint 2**
12	As an user, I would like to maintain a history events that I have dealt with.	**8**	**Sprint 2**
13	As a user, I would like to look the statistics of the all concerns and agent, including how many reports has been filed, how many reports has been resolved, how much time it takes for each resolved complaint.	**5**	**Sprint 2**
14	As a reporter, I would like to sign up with Facebook account	**4**	**Sprint 2**
15	As a reporter, I would like to sign up with Google+ account	**1**	**Sprint 2**
16	As a reporter, I would like to submit concern to one	**10**	**Sprint 1**

	or multiple community agents		
17	As a reporter, I would like to upload image attachment to the agent	**2**	**Sprint 2**
18	As a reporter, I would like to downvote concerns I think to harm the society	**3**	**Sprint 2**
19	As a reporter, I would like to search for concerns based on keywords	**15**	**Sprint 2**
20	As a reporter, I would like to upvote the concerns I like	**6**	**Sprint 2**
21	As a reporter, I would like to delete my concern	**1**	**Sprint 1**
22	As a reporter, I would like to edit my concern	**5**	**Sprint 1**
23	As a reporter, I would like to mark the best answer for my concern	**4**	**Sprint 2**
24	As an agent, I would like to view all incoming concerns	**3**	**Sprint 2**
25	As an agent, I would like to respond to incoming concerns	**5**	**Sprint 2**
26	As an agent, I would like to be notified for new concerns by email	**6**	**Sprint 2**
27	As an agent, I would like to upload my proof file while signing up for admin to inspect	**5**	**Sprint 2**
28	As an agent, I would like to redirect mis-directed concerns to other agents	**4**	**Sprint 2**
29	As an admin, I would like to verify agent's information to avoid fake agent accounts.	**6**	**Sprint 1**
30	As an admin, I would like to delete spam concerns which have negative impact to the society	**4**	**Sprint 2**
31	As an agent, If two reporter file similar concerns, I would like to mark them as duplicate.	**10**	**Sprint 2**
32	As an agent, I want to add "Resolve" flag for a resolved problem.	**8**	**Sprint 2**

5. **Non-Functional Requirement:**

Security:

Django is a matured python web framework development tool, which is secured enough to protect us from many attacks through website. As a result, we do not need to worry about problems like cross site scripting in our project.

However, we do need to worry about problems on inappropriate usage of our website. For example, we need to make sure that user does not maliciously perform certain operation for too many times and cause any traffic issue in our network server. We will be enforcing such rule with verification email and recaptchas on important operations.

It is also important for us to make sure that no malicious file is attached (and uploaded) to our system. We can enforce the security by limiting the executing privilege of the uploaded files. Since we will not be running any file on our server, it is necessary for us to remind our user of such potential problems.

Lastly, we need to help our user to protect their account by forcing them to set up a strong password.

Scalability:

Currently, our web app provides service for users in West Lafayette. In the future, we will be able to realize the same functionality to users in United States and even all over the world.

Reliability:

We understand the robustness of our web app is crucial for both reporters and agency. Since we will deploy our website on Heroku, host database on Postgresql and save attachments on AWS S3, we will be relying on those platforms for reliability concerns.

6. **Use Cases:**

Case#1 - Reporter Sign Up	
Action	**System Response**
1. Reporter click “Reporter sign up”	2. Reporter sign up page appears
3. Reporter enter signup user information (email required)	
3. Reporter enter invalid information	4. Do not allow to submit form and show error message
5. Reporter click “submit”	6. “Verification email sent” Dialogue appears and verification email is sent to user’s email, then direct to the login page.
7. Reporter open mailbox and open verification email and click on verification link.	8. Redirect user to main page

Case#2 - Agent Sign Up	
Action	**System Response**
1. Agent click “Sign up”	2. Agent sign up page appears
3. Agent enter signup user information (email required)	
4. Agent upload the proof file as attachment.	5. Proof file has added in the database and a notification will sent to admin.
6. Agent click “submit”	7. “Verification email sent” Dialogue appears and verification email is sent to user’s email
8. Agent click “OK”	9. Redirect user back to login page
10. Agent open mailbox and open verification email and click on verification link.	11. Redirect user to profile page

Case#3 - Sign up with third party	
Action	**System Response**
1. User click Sign up with Google, Facebook	2. Direct to third party signup site
3. User enter signup user information	4. After authorizing user identification, pop up “please enter email” dialog
5. User enter email	
6. User click “submit”	7. “Verification email sent” Dialogue appears
8. User click “OK”	9. Redirect user back to login page
10. User open mailbox and open verification email and click on verification link.	11. Redirect user to profile page

Case#4 - Edit Profile	
Action	**System Response**
1. User click “My profile”	2. User profile page appears
3. User click “Edit”	4. “Edit profile page” appears
5. User make changes to the fields (such as nickname, password, picture etc.)	
6. User click “submit”	7. Confirm dialog appears
8. User click “Confirm”	9. Redirect user to user profile page

Case#5 - Log in and log out	
Action	**System Response**
1. User click “Sign in”	2. Sign in page appears
3. User enter username and password	
4. User click “login”	5. User redirect to profile page
6. User click “logout”	7. User redirect to logged out page

Case#6 - Submit concern	
Action	**System Response**
1. Reporter click “Submit Concern”	2. Submit concern page appears
3. Reporter enter concern text	4. Concern textbox is filled with text
5. Reporter click attach file button (if necessary) and attach file	6. Attach files has attached with concern
7. Reporter select agents to report to	8. Agents selected
6. Reporter ready to submit	7. User being prompted for recaptchas
8. Reporter click on recaptchas and then click submit	9. Notification email sent to target agents
10. Agent receive notification email about the report	

Case#7 - View all concern	
Action	**System Response**
1. User click “View Concern” button	2. Concern page appears with all the submitted information
3. Reporter click on the button “Upvote Concern”	4. Pop up “Upvote Successful” dialogue, and vote number raises
5. Reporter click on the button “Downvote Concern”	6. Pop up “Upvote Successful” dialogue, and vote number decreases
7. User click on “Close”	8. “Upvote Successful” dialogue closed

Case#8 - Respond to concern	
Action	**System Response**
1. Agent select concern to respond	2. Concern is selected
3. Agent click “Respond”	4. Display response page
5. Agent fill in response, and attach additional files (if necessary)	
6. Agent click “submit”	7. Agent redirected to concern page

Case#9 - View profile	
Action	**System Response**
1. User click on the name of the reporter/agent/yourself	2. Pop up a profile tag/window
3. User click on close profile button	4. Close Profile Page tag/window

Case#10 - View related concern	
Action	**System Response**
1. Agent click “View Concern”	2. Concern Page appears with all the concerns directed to this agent
3. Reporter click “View Sent Concerns”	4. Concern Page appears with all the concerns sent by this reporter

Case#11 - Edit related concern	
Action	**System Response**
1. Reporter click “delete concern” in concern page	2. Pop up a confirmation window
3. Reporter click “confirm delete”	4. Concern is deleted from database, and the page is refreshed
5. Reporter click “edit concern” in concern page	6. Redirect to concern edit page
7. Reporter click “confirm edit”	8. Redirect to concern page with edited concern

Case#12 - Reset Password	
Action	**System Response**
1. User click on “forgot password” before login	2. Redirect to forgot password page

3. User fill in email information	4. Reset password link sent to user's email
5. User login the mailbox and click on the reset password link	6. Link directs user to reset password page
7. User filling in email and enter new password	8. The password is successfully reset.

Case#13 - Search Concerns	
Action	**System Response**
1. User put search keywords in search box in view concern page and click "search"	2. Display fuzzy search concern results in order
3. User click on one of the result concern	4. Redirect user to concern page

Case#14 - Concern Redirect	
Action	**System Response**
1. Agent find a misdirected concern and then select this concern, and click "re-direct" button	2. Pop up concern redirected window
3. Agent select agents from dropbox to redirect and click "redirect"	4. Re-assign the concern to the target agent

Case#15 - Admin Inspect	
Action	**System Response**
1. When a concern is downvoted too many times, admin deletes this concern	2. This concern disappears from our database

Case#16 - Access invalid network address	
Action	**System Response**
1. Agent/Reporter enters an invalid address	2. Display 403 page

Case#17 - View History	
Action	**System Response**
1. User click a specific concern and in this concern page, click “view history” button	2. Display concern edit history in a pop-up box

Case#18 - View Statistics	
Action	**System Response**
1. User click a specific concern and in this concern page, click “view Statistic” button	2. Display this concern statistic in a pop-up box
3. User click an agent profile and in this profile page, click “view statistics” button	4. Display agent statistics in a pop-up box

Case# 19 - Concern Resolved Status	
Action	**System Response**
1. As an agent click “Resolved” button	2. Change the concern status from “Unresolved” to “Resolved”
3.As a report owner click “Reject Solution” button	4. Change the concern status from “Resolved” to “Unresolved”