Table of Contents

1. Introduction

1.1 Purpose

The purpose of FixIt is to repair and maintain facilities more efficiently than the previous system. FixIt relies on user input to receive notice on reports. Users will be able to report issues which are converted into tickets. After the system screens the tickets for any errors, management will then be able to view these tickets in a customizable virtual map. Finally, management will be able to assign these tickets to their employees. FixIt also has a comprehensive notification system that will quickly notify employees of any new assignments. This enables FixIt to quickly streamline the ticket review process.

1.2 Scope

The FixIt application is only available to UCI students with valid UCInetID. It can be used to report and manage maintenance and repair requests. The main goal of this application is to handle the growing number of requests in a more efficient way. The application shall provide the ability to minimize duplicate reports as well as increase security by having three different types of accounts: UCI affiliates, UCI workers, and facilities managers. Communication between these primary stakeholders is facilitated with the help of notifications.

1. UCI affiliates
    a. Automatic fill in
        i. First/last name
        ii. Student/employee ID
    b. Able to view report status
    c. Provide information
        i. Location
        ii. Pictures of the issue
        iii. Classify the problem
    d. Receive issue-update notifications
2. UCI workers
    a. View assigned issue
        i. Location
        ii. Pictures of the issue
        iii. Issue description
    b. Update issue status which sends a notification to all users involved
3. Facilities management
    a. Schedule regular maintenance and assign tasks to workers which sends a notification to that respective worker
    b. Organized view of the issue
        i. Priority
        ii. Location
        iii. Date
    c. Edit/Add notes to report

1.3 Definition, acronyms, and abbreviations

Term or Acronym	Definition
Facilities management	Assigns each report to the ideal worker
FixIt	A comprehensive notification and report-handling system.
OIT	Office of Information Technology
Student/Faculty	Our primary end users. They report issues by using the application
UCI affiliates	Anyone who is currently operating on the UCI campus and has a valid UCInetID
UCInetID	A unique ID number assigned to anybody operating on the UCI campus such as a student or worker
UCI repair worker	Responds to repair requests from the manager

Refer to Appendix A.1 for additional definitions

1.4 References

See Section Appendix A.2 for models.

1.5 Overview

The app will enable users, facilities managers, and repair workers to harmoniously participate in keeping the campus well-maintained. The app will allow users to quickly file reports of repair issues, provide the tools for managers to prioritize and assign repair tasks, and enable repairers to efficiently handle repairs. Furthermore, this app is guaranteed to be more efficient than the previous system which makes it worth the investment.

2. General description

FixIt will use pre-existing technology in order to expedite a few of its features. For example, it will use WebAuth which is the current system UCI has to authenticate users. WebAuth is important because it tells us what each user is. Each user has their own set of reports and notification settings. Workers will have their own personal assignment list. Lastly, managers will have the most administrative privileges by being able to view all available reports and assign them to their employees.

Another existing function we implemented is GPS. FixIt requires GPS to be on while submitting a report because it allows us to track where the report has been sent. This is vital information because determining where the user is helps us determine where the issue is located. Users can also submit a personal description of where the issue is, but automatically turning on GPS is a much more user-friendly way of obtaining an issue's location.

Lastly, FixIt draws inspiration from social media platforms. Users will be notified automatically whenever their report's status is updated. Repair workers will also be notified whenever they have a new assignment, and managers can receive notifications on incoming reports. This notification system is what makes FixIt handle issues quickly as a result of these automatic reminders.

2.1 Product Perspective

The FixIt system provides a convenient platform for all UCI affiliates including repair workers, students, managers and faculty members to report any facilities issues. The platform makes handling large amount of repair issues easier and efficient. Users will submit reports on issues throughout campus. The application will group all the reported issues and send them as tickets to the UCI Facilities Management Team. The Facilities Management Team will then assign these tickets to their repair workers, and the repair workers will be notified through the FixIt mobile application.

- See Section Appendix A.2.3 for a perspective demonstration

2.2 Product Functions

The FixIt system's main purpose will be to allow for the creation and resolution of tickets that represent ongoing Facilities Management related issues such as broken lights, deteriorated buildings, or any other types of degradation or damage. FixIt will allow for reporters, consisting of UCI students and non-Facilities Management faculty and staff. However, not all users will be given access to the same functions that FixIt offers. Briefly stated, FixIt's functions can be broken down by the type of users that will be using the system as follows:

- Reporters, who can:
  - Submit a report
  - View and edit previously submitted reports
- Facilities Management workers, who can:
  - View and edit reports/issues assigned to them
  - Resolve issues assigned to them

- Facilities Management managers, who can:
    - View and edit any or all reports/issues submitted by reporters
    - Assign issues to their subordinates (i.e. Facilities Management workers)
    - Browse the virtual map that graphically pinpoints ongoing issues
    - Schedule regular maintenance

2.3 User Characteristics

The following are four users of the fixit application.

- Manager
    - They have full access to the system.
    - These are the highest level of users and can manipulate other user's assignments.
- Worker
    - They have slightly limited excess to the system.
    - They can only modify reports that are assigned to them.
- OIT
    - They maintain the system and respond to technical issues
    - They have direct access to the system's data and security
- UCI Students and Faculty
    - These constitute the vast majority of users for the application.
    - They have no control over other user's actions.

2.4 General Constraints

The constraints of the application being developed are listed below:

- The application must be usable by users with disabilities due to UCI policy.
- The application must work on the browsers versions listed below
    - Google Chrome 49+
    - Mozilla Firefox 46+
    - Internet Explorer 11
    - Microsoft Edge 13+
    - Apple Safari 8+
- The application must be secure with login credentials as these login credentials are also used to log into other UCI services.
- The application must be mobile friendly.

2.5 Assumptions

- We are assuming our application will be able to allow users to quickly send and receive reports.
- We are assuming there should be regular maintenance involved that can be scheduled by the manager

- The manager will have special access to the virtual map which will be very useful for viewing and assigning reports.
- The application will mainly be used on Campus
- The application assumes that the user has internet access to send and receive reports

3. Specific Requirement

3.1 Essential Requirements

3.1.1 Functional Requirements

- 3.1.1.1 Functional requirements
    - ID: FR1
    - TITLE: Report Repair Problems
    - DESCRIPTION: Enable users to quickly communicate repair problems to the facilities management.
    - EVENT/USE CASE: Filling out report [A.2.3 Use Case Model]
    - SOURCE: Original requirements from the client
    - FIT CRITERION: The communication of repair problems takes only a few minutes.
    - RATIONALE: The purpose of the system is to save time, and fast communication of problems will help save time.
    - DEPENDENCIES: None
    - SUPPORTING MATERIAL: A.2.3 and A.2.4
    - HISTORY: Included in the original requirements from the case study
- 3.1.1.2 Functional requirements
    - ID: FR2
    - TITLE: Communicate Tasks to Workers
    - DESCRIPTION: Quickly communicate repair and maintenance tasks to workers. Include details such as category and location.
    - EVENT/USE CASE: Send notification of assignment [A.2.3 Use Case Model]
    - SOURCE: Initial elicitation session
    - FIT CRITERION: The communication of tasks to workers takes only a few minutes.
    - RATIONALE: An instant notification of assignment will enable workers to address the problem faster. This aligns with the purpose of saving time.
    - DEPENDENCIES: Assign reports
    - SUPPORTING MATERIAL: A.2.3 and A.2.4
    - HISTORY: Approved in the initial elicitation session. Included in the requirements ever since.
- 3.1.1.3 Functional requirements
    - ID: FR3
    - TITLE: Communicate Task Completion

- DESCRIPTION: Repair workers communicate task completion to facilities management and reporters of the issue.
- EVENT/USE CASE: Declare repair as complete [A.2.3 Use Case Model]
- SOURCE: Initial elicitation session
- FIT CRITERION: The reporter of the issue hears that the repair was completed and the facilities management's database gets updated to show the repair as complete.
- RATIONALE: Declaring an assignment as complete informs the managers of positive work progress and provides users with assurance of job completion.
- DEPENDENCIES: Worker editing assignment
- SUPPORTING MATERIAL: A.2.3 and A.2.4
- HISTORY: Approved in the initial elicitation session. Included in the requirements ever since.

3.1.1.4 Functional requirements

- ID: FR4
- TITLE: Prioritize Reports
- DESCRIPTION: Facilities managers mark some reports as more important than others.
- EVENT/USE CASE: Prioritize reports [A.2.3 Use Case Model]
- SOURCE: Original requirements from the client
- FIT CRITERION: The report can have a priority of 1-5, with 1 being "most urgent" and 5 being "least urgent".
- RATIONALE: Including a field for classifying urgencies saves managers the trouble of manually informing workers of more urgent repairs. It is consistent with the goal of saving time.
- DEPENDENCIES: None
- SUPPORTING MATERIAL: A.2.3 and A.2.4
- HISTORY: Included in the original requirements from the client and accounted for ever since.

3.1.1.5 Functional requirements

- ID: FR5
- TITLE: Minimize Duplicate Reports
- DESCRIPTION: If the software finds multiple reports on the same issue, they shall be merged so the same task is not assigned twice.
- EVENT/USE CASE: Merge duplicate reports [A.2.3 Use Case Model]
- SOURCE: Original requirements from the client
- FIT CRITERION: Reports should be merged if they have the same repair category and are within 50 yards of each other, based on the given coordinates.
- RATIONALE: Some students will report the same issues. If we do not merge duplicate reports, we may accidentally assign the same

job to different repairers. That would waste time, an outcome inconsistent with the goal of the app.

  - DEPENDENCIES: Convert reports to assignments
  - SUPPORTING MATERIAL: A.2.3
  - HISTORY: Included in the original requirements from the client. Later clarified in regards to what we define as a duplicate report.

3.1.2 Non-functional Requirements

3.1.2.1 Usability

- With a huge amount of user involvement, the FixIt application should be as simple as possible. Each part of the system must be understandable for the average user and minimize any user experience needed to use the application. In case of an emergency, a user should be able to report any issue without any problem through the application.

3.1.2.2 Portability

- The FixIt system should be accessible through all common mobile operating systems, including iPhones and Androids. It is important to ensure portability so that users can report facilities issues anywhere around the campus. In case of an emergency issue, all UCI affiliates will be notified through their mobile devices.

3.1.2.3 Security

- The FixIt system handles large amounts of personal data including usernames, passwords, emails, phone numbers, addresses, etc. With increasing numbers of user involvement, it is important to ensure the security of the FixIt system. All information should be encrypted before sending to the database/server and remain private for each respective user.

3.1.2.4 Efficiency

- With a large amount of user involvement and data transmitted per day, it is important to ensure the efficiency of the application. The FixIt system must be able to transmit data at high speeds even with high traffic count. The application should be able to switch from different menus without any lag.

3.1.3 External Interface Requirements

This section will provide a few visual examples of the user interface and also detail all other points of input and output within the application.

3.1.3.1 User Interface Prototype

Login:

All users first view the login page when they enter the application. They will login using their UCInet ID and password.

Timeline:

Once logged in, regular users will view a timeline with the reports they've created and options to edit or create a new report. Employees will view a similar timeline but with reports that have been assigned to them, see Figure 1 below. All user types will have a settings cog in the top right corner.

Settings:

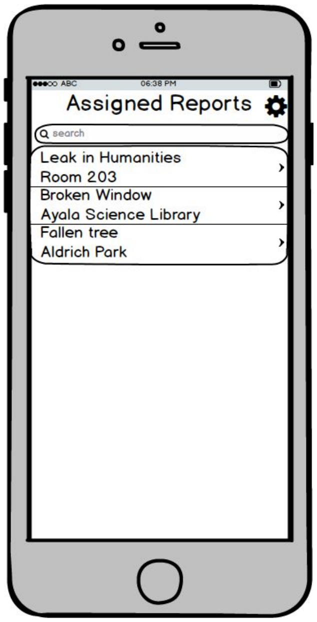
All users will be able to access the settings tab (Figure 2). This tab will let users change preferences like the notifications they'd like to receive and also if they'd lack their location to be used by app.

Create Report:

Regular users will be able to access the Create Report tab. This tab will have text fields for a description of the report, a title, and any notes. It will also have an area for selecting the location of the incident or using the user's current location.

Edit Report:

Edit report has the same interface as the Create Report tab. However the text fields are already filled with the report information that was previously filled out, and the user can alter these fields if they'd like.

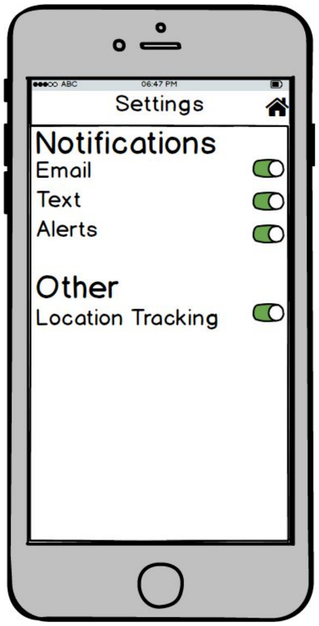


Leak in Humanities
Room 203

Broken Window
Ayala Science Library

Fallen tree
Aldrich Park

Figure 1



Notifications
Email ●
Text ●
Alerts ●

Other
Location Tracking ●

Figure 2

3.1.3.2 Inputs and Outputs

- Incoming data consists of user data: report details, locations, settings changes
- Outgoing data consists of report updates: Emails and texts

3.1.3.2.2 Input: Data / Output: Data Storage (database)
ITEM NAME: Report data
DESCRIPTION OF PURPOSE: The application is constantly receiving user submitted reports. These reports will not be stored on the user's device, rather they will be stored on a server in a database.
SOURCE INPUT:
1. User inputs data into either the Create Report form or Edit Report form.
DESTINATION OUTPUT: Database, could be any type of database (MySQL, noSQL). No preferences as of now.
UNITS OF MEASURE: bytes
TIMING: 1-2 reports every 10 minutes
RELATIONSHIPS: User [has as] Device [has a] Input form [has a] Database
SCREEN FORMATS: Mobile devices, desktops, tablets, and laptops
WINDOW FORMATS: Mobile devices, desktops, tablets, and laptops
DATA FORMATS: Text, Location(tuple of doubles)

3.1.3.3 Hardware Interfaces

There is no specified hardware required besides either a computer, tablet, or smartphone. The application will be designed to work on web and also will have an app.

3.1.3.4 Software Interfaces

The application will interact with a database, sending it report data. The application will also interact with a phone GPS software to be used when a user has turned on their location settings.

3.1.4 Logical Data Model

User:

UCInetID	VARCHAR(20)
Password	VARCHAR(20)
Email	VARCHAR(20)
Phone	VARCHAR(10)
SettingsID	VARCHAR(20)

FOREIGN KEY (SettingsID) REFERENCES Settings (settingsID)

Reporter:

UCInetID	VARCHAR(20)

FOREIGN KEY (UCInetID) REFERENCES User (UCInetID)

FacilitiesRepairer

UCInetID	VARCHAR(20)
EmployeeID	VARCHAR(8)
Title	VARCHAR(30)

FOREIGN KEY (UCInetID) REFERENCES User (UCInetID)

FacilitiesManager

UCInetID	VARCHAR(20)
EmployeeID	VARCHAR(8)
Title	VARCHAR(30)

FOREIGN KEY (UCInetID) REFERENCES User (UCInetID)

Report:

ReportID	VARCHAR(20)
UCInetID	VARCHAR(20)
CurrentLocation	VARCHAR(50)
PhoneNumber	VARCHAR(10)
Description	VARCHAR(400)

ImageData	VARCHAR(20)
TimeReported	VARCHAR(20)

FOREIGN KEY (UCInetID) REFERENCES User (UCInetID)

Issue:

IssueID	VARCHAR(20)
UCInetID	VARCHAR(20)
Title	VARCHAR(40)
Category	VARCHAR(40)
Status	BOOL
ReportDupNum	INT
Priority	INT

FOREIGN KEY (UCInetID) REFERENCES
FacilitiesRepairer(UCInetID)

Settings:

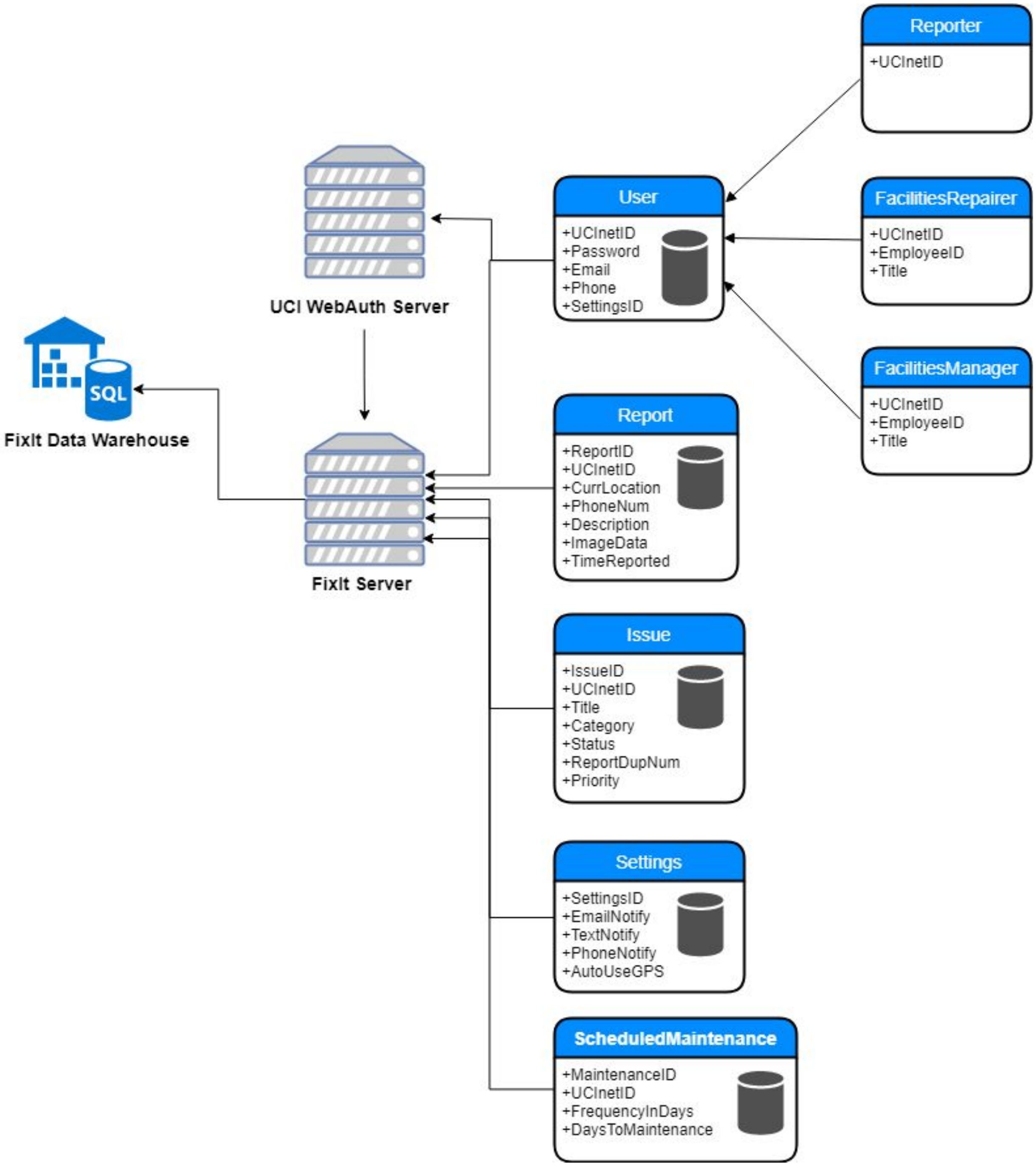
SettingsID	VARCHAR(20)
EmailNotify	BOOL
TextNotify	BOOL
PhoneNotify	BOOL
AutoUseGPS	BOOL

ScheduledMaintenance:

MaintenanceID	VARCHAR(20)
UCInetID	VARCHAR(20)
FrequencyInDays	INT
DaysToMaintenance	INT

FOREIGN KEY (UCInetID) REFERENCES FacilitiesManager(UCInetID

Logical Data Model Diagram:



3.1.5 Design Constraints

3.1.5.1 Hardware Constraints

FixIt should be able to run on both personal computers, be it desktop or laptop, as well as mobile devices such as smartphones or tablets. All devices running FixIt should have at least the following specifications:

- 32-bit or 64-bit dual-core CPU
- 2 GB RAM
- 20 GB main memory space
- Ethernet or Wireless adapter for Internet connectivity
- Dedicated or discrete graphics card with 2 GB VRAM

3.1.5.2 Software Constraints

Since FixIt may be ran on either a personal computer or mobile device, it should be supported by the following operating systems:

- Windows (7 or above)
- MacOS (Yosemite or above)
- Android (5.0 or above)
- iOS (6.1.6 or above)

FixIt should be able to be run on all major Internet browsers (always the latest version), if run on a personal computer, such as:

- Mozilla FireFox
- Google Chrome
- Safari
- Opera
- Microsoft Edge
- Internet Explorer

3.2 Extension Requirements

3.2.1 Functional Requirements

- The system should be able to allow users to edit and view their own reports. Managers should be able to edit and view all reports, and workers should be able to edit and view assigned reports
- The system should be able to validate the user in order to determine if they are a manager, worker, or end user in order to grant that person special privileges.
- The user should be able to turn their GPS location on and off, but in the beginning it will be on unless they turn it off manually.
- A user should be able to contact OIT if they need any help accessing the app or submitting a report

- The system should be able to automatically determine if a report is a duplicate to minimize duplicate reports
- The manager should have access to a virtual map to manage reports easier and schedule regular maintenance.

3.2.2 Non-functional Requirements

- The application shall support Windows, Android and iOS.
- The application should be usable by users of all technical backgrounds.
- The application shall be robust and should have almost no failures for managers or workers.
- The user reports shall update for Facilities Management every minute.

Appendix

A.1 Glossary

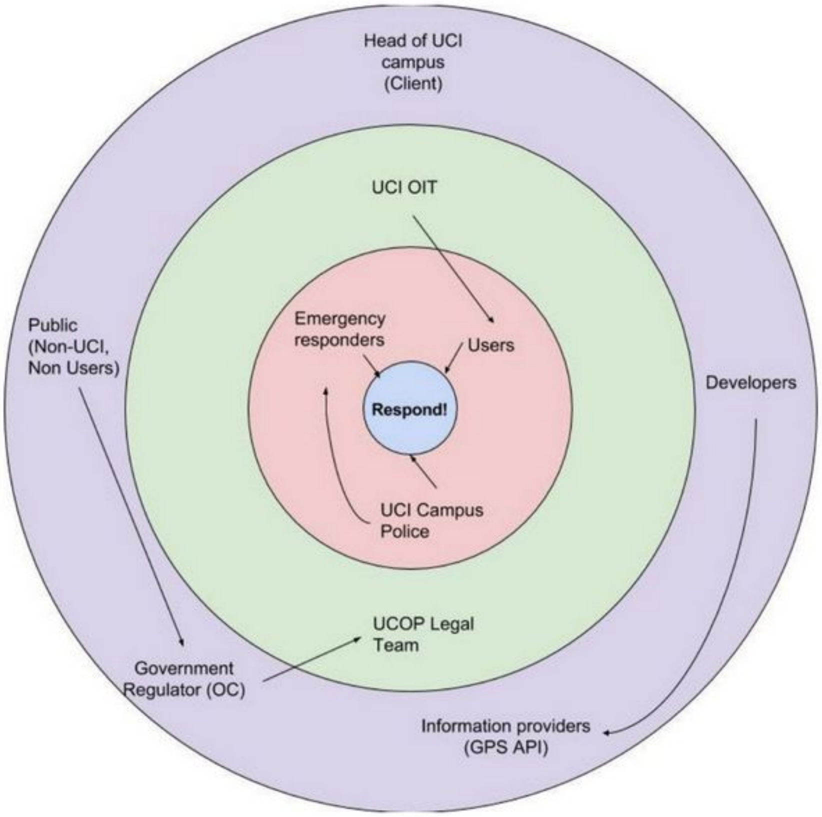
Term or Acronym	Definition
Facilities management	Assigns each report to the ideal worker
FixIt	A comprehensive notification and report-handling system.
FR	Functional requirement
GPS	Global Positioning System which is used in our application to track where the user submitted the report
OIT	Office of Information Technology
SRS	System Requirements Specification
Student/Faculty	Our primary end users. They report issues by using the application
UCI	University of California, Irvine.
UCI affiliates	Anyone who is currently operating on the UCI campus and has a valid UCInetID
UCInetID	A unique ID number assigned to anybody operating on the UCI campus such as a student or worker

UCI repair worker	Responds to repair requests from the manager
WebAuth	Used to authenticate user and determine if they are a student/faculty, repair worker, or manager. Depending on who they are, they receive certain privileges.

A.2 Analysis Models (Stakeholder, Goal, Usage, Domain)

A.2.1Stakeholder/ Onion model

Onion Stakeholder Model


Head of UCI campus (Client)
UCI OIT
Emergency responders
Users
Respond!
UCI Campus Police
Public (Non-UCI, Non Users)
Developers
UCOP Legal Team
Government Regulator (OC)
Information providers (GPS API)

Stakeholder Analysis

UCI faculty(teachers, TAs etc)

a Motivation: The faculty form another large set of users. Given that this system is user dependent, we want the faculty to be able to use the app.

b Authority: The faculty have relatively low authority compared to the OIT or maintenance workers, since they may only submit, edit, and track reports. However, they are still important because our app relies on their reports to notify maintenance workers of issues on campus.

c Relation to other stakeholders: They will send reports to the UCI Facilities Management upper management, who in turn will assign the tasks to their workers.

d Level of expertise: They should have the expertise to speak English and use apps on their phones.

e Their expectations: They should expect to be able to submit, edit, and track their reports. The reports should also be done in an orderly and timely manner.

f Your expectations of them: We expect them to submit reports on any important issues on campus, and they should be as detailed as possible when describing a report.

g Location/availability: We expect the faculty to be located mostly on campus in their allocated office hours.

Students

ivation: Students will be the most common users of our app, and this is a very user-dependent system. Therefore, students are an important stakeholder group since they comprise the majority of user-submitted reports.

b Authority: Lowest authority of all our stakeholders, but they will have the ability to submit, edit, and track the status of their reports.

c Relation to other stakeholders: They will be able to send reports that will be received by the UCI facilities management upper management, who in turn will assign the tasks to the UCI facilities management workers.

d Level of expertise: They should be technologically competent enough to use a basic smartphone app.

e Their expectations: Their expectations for this system is that they will be able to track and edit their report. Furthermore, their reports should be handled in a timely manner.
f Your expectations of them: We should expect our students to not abuse the system by sending dummy reports, and for them to report issues as accurately as possible.
g Location/availability: We expect them to be located mostly on campus. If they are not on campus, they should specify the location of their issue. We also expect them to be available to confirm if a report has succeeded or not.

UCI Facilities Management (UFM) upper management
a Motivation: The upper management is important because they will be the ones assigning reports to his/her workers.
b Authority: They will have the highest authority of any user because they will have the ability to assign and modify any existing reports.
c Relation to other stakeholders: The upper management will have to interact with all the other stakeholders. They would interact with workers by assigning them tasks, and if necessary the students and faculty can contact them if they need to resolve a serious issue.
d Level of expertise: They should know their workers well enough to assign the right task to the right person, and they must be able to assign tasks as quickly and efficiently as possible.
e Their expectations: They should expect the app to be able to easily track reports. Furthermore, since the upper management will be assigning tasks to their workers, they should expect the report to be as detailed as possible. That way they will know the best person for a certain task.
f Your expectations of them: We should expect the upper management to assign issues to their employees, and for them to make sure that the tasks are completed before any deadlines.
g Location/availability: We expect them to be located in the office where they can watch over everyone, and they should be able to respond immediately when anything important arises.

OIT
a Motivation: The Office of Information Technology is the stakeholder that will be responsible for maintaining and supporting the application. Therefore they're important for making sure our system runs smoothly.
b Authority: They will have higher authority than a general user for this application. This is because their authority can be based on how much

permissions they need during maintenance of the application. If necessary, they should be able to take control of the system to solve any technical problems.
c Relation to other stakeholders: OIT will listen to the demands of the UFM upper management and UCI administration. If the students/staff and maintenance workers report any bugs, the OIT will try their best to fix them.
d Level of expertise: Needs to have a complete understanding of all of the application's features and architecture. Should understand the most common technical problems and how to fix them.
e Their expectations: The system will be feasibly maintainable, and the users can be trusted to not abuse the system by overloading the servers or sending faulty information.
f Your expectations of them: We expect them to help users having trouble with accessing the app, and they should also maintain the app to make sure it is running smoothly at all times.
g Location/availability: They should be accessible through phone or email. We can expect them to be available quickly during their open hours, which can range from the early morning to night.

UCI Administration
a Motivation: The UCI administrator's job is to talk to investors who are willing to fund our project. They are very important for making sure that the project gets enough funds to succeed, so they act as an intermediary between the project managers and investors.
b Authority: They have the second highest priority because they are the ones managing the money. They still need to listen to some of the demands of the investors, but overall they will be making sure the project is done on time and properly funded.
c Relation to other stakeholders: they would interact with investors to secure funds for our project. Furthermore, they would survey the UCI maintenance staff and students to make sure that their demands are satisfied as well.
d Level of expertise: We can expect them to have good negotiation skills to secure funds for our project.
e Their expectations: They should expect that the project will be done in a timely manner with the budget given.
f Your expectations of them: We expect them to be clear on how many funds can be used to develop our project.
g Location/availability: They should be available directly at their office in specified times for discussions, or they should be contactable through email. We should

expect them to be available when major deadlines are coming to make sure the project is progressing smoothly for their tastes.

Investors

a Motivation: Investors will be financially responsible for the development and maintenance of the system. They will be communicating with UCI administration constantly to get details about the project.
b Authority: Investors are one of the highest authority among the stakeholders as they are funding the entire project. Their funding support is necessary and important to compensate each part of the development project. Without investors, any development stage will become impossible.
c Relation to other stakeholders: They will be interacting with the administration so that they can allocate funds to different stages of the project.
d Level of expertise: They are not required to have any level of skills or knowledge as they do not responsible for any part of the development of the project.
e Their expectations: Investors will expect their investment allocated in a way that is efficient. They expect their fundings will support the whole development process and the team will develop a satisfying project.
f Your expectations of them: We expect investors communicate with UCI administrators to understand the project and their needs so they can support them financially.
g Location/availability: We do not expect them to be available at times. The most important thing is to discuss the details of the project with UCI administration and to know their needs.

Non-UCI maintenance workers

a Motivation: Unlike the UCI faculty, staff and students, Non-UCI maintenance workers will not be part of end users for the application. They will be getting the details on issues reported on the application during their work hours while contracted by UCI to work on reported issues. However, they are still important for any large or emergency issues that would require more workers to resolve.
b Authority: Non-UCI maintenance workers will have the shared authority over how the application handles the resolution process.
c Relation to other stakeholders: They will be in contact with the UCI Facilities Management upper management to get contracts to solve reported issues.
d Level of expertise: Their level of expertise would not affect the application since they will neither be a part of development nor a part of the user-base.

e Their expectations: Non-UCI maintenance workers should expect that our application makes sharing and accessing reports even easier.. They should also expect for there to be more frequent reports since there are more contributors. Furthermore, they should also expect that the reports assigned to them are clear and detailed.
f Your expectations of them: We expect them to be available to take the reports from UCI Facilities management and to do the necessary tasks to solve it.
g Location/availability: We do not expect them to be available on campus at all times. They will be contracted when needed for specific issues reported on the application.

UCI Facilities Management (UFM) workers
a Motivation: Along with UCI faculty, staff, and students, UCI facilities management workers will be part of the primary end-user base, especially since their entire work schedules will revolve around FixIt. Due to their high dependency on FixIt, it is paramount that we receive timely input from the facilities team's workers.
b Authority: UCI facilities management workers will have authority over issue resolutions within the FixIt app as they will be the primary responders.
c Relation to other stakeholders: UCI facilities workers will be assigned access by upper-level management (i.e. upper-level management will give appropriate access to new-hires). Also, facilities workers may be in contact with the UCI-affiliated users who submit an issue on the FixIt app. As a result, the workers will be the resolvers of the issues posed by UCI faculty, staff, and students.
d Level of expertise: UCI facilities workers will need a higher-level knowledge of FixIt than other users. They will need to know how to fully navigate the app, update issues, resolve issues, etc. As a result, there should be some documentation provided to expedite the time it takes to become proficient at using FixIt.
e Their expectations: Facilities workers should expect any level of issue-influx within the FixIt system. There may be a high level of reports submitted by users that facilities workers should be able to process. They should expect the app to function with few, if any bugs, with little down-time as well.
f Your expectations of them: We expect the facilities management workers to use the FixIt app as per the guidelines that will be stated after the production of the app. We also expect them to report any bugs, exploits, or even feedback regarding FixIt.

g Location/availability: The expectation is that facilities management workers will primarily be on the UCI campus, although since UCI owns property outside the premises of the main campus, their locations may not necessarily be central in one designated area. Facilities management workers should be available to the extent of their existing guidelines, regulations, and work schedules and should treat the FixIt app as being a part of their work routine.

Developers of FixIt:

a.Motivation: They are the ones developing the core parts of FixIt, so it is their responsibility to make sure the application is running all its required tasks properly. It is also important that they finish parts of the projects before deadlines.

b. Authority: They will have some authority in deciding whether or not a function should be required in the application, depending on how difficult a function may be to implement, the payoff might not be as worth it.

c. Relation to other stakeholders: They will directly work with the UCI administrators overseeing our project to make sure that their demands are properly satisfied.

d. Level of expertise: They should be very efficient in programming applications for multiple platforms. This is because our device should be usable for the majority of the populace in order to succeed.

e. Their expectations: They should expect to get compensated for the work they have done with the proper funds. Furthermore, they should also be given certain permissions to the UCI database to test whether or not the application is working. That way the application will be less faulty in launch.

f. Your expectations of them: We should expect them to develop the core functions of FixIt, such as having a visualization map and allowing users and workers to edit and track reports. They should finish parts of the project before deadlines as well.

g. Location/availability: We should expect them to be available online or in person on campus mostly through the day. It is important that they are readily available for most of the day to make sure that any demands are met successfully.

A.2.2 Goal model

DIAGRAM 1:

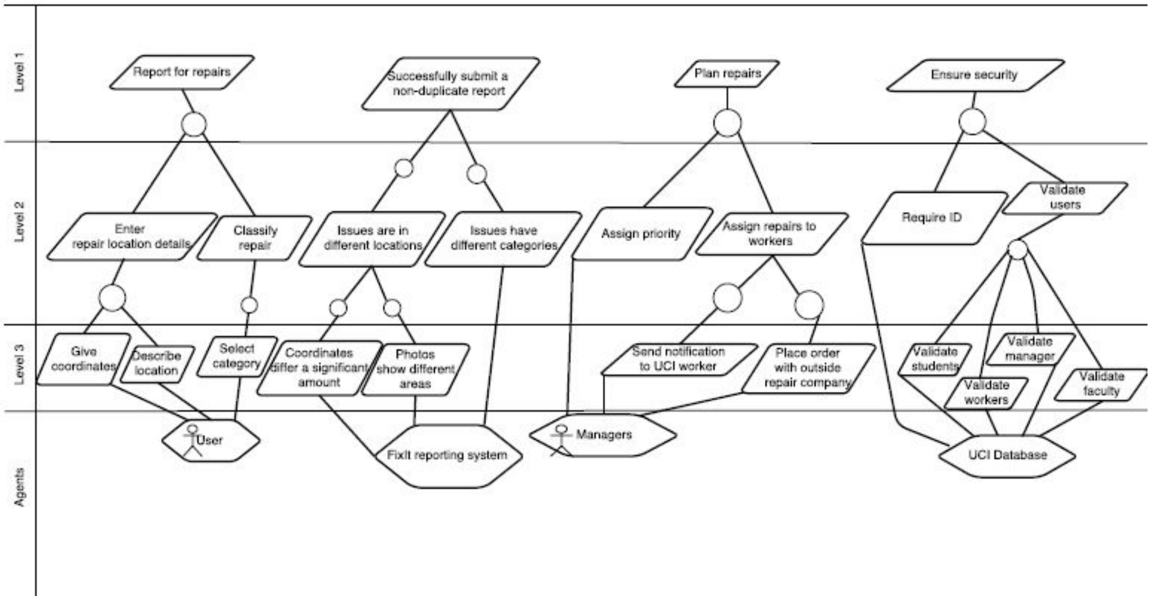

Level 1
Level 2
Level 3
Agents
Report for repairs
Successfully submit a non-duplicate report
Plan repairs
Ensure security
Enter repair location details
Classify repair
Issues are in different locations
Issues have different categories
Assign priority
Assign repairs to workers
Require ID
Validate users
Give coordinates
Describe location
Select category
Coordinates differ a significant amount
Photos show different areas
Send notification to UCI worker
Place order with outside repair company
Validate students
Validate manager
Validate workers
Validate faculty
User
FixIt reporting system
Managers
UCI Database

DIAGRAM 2:

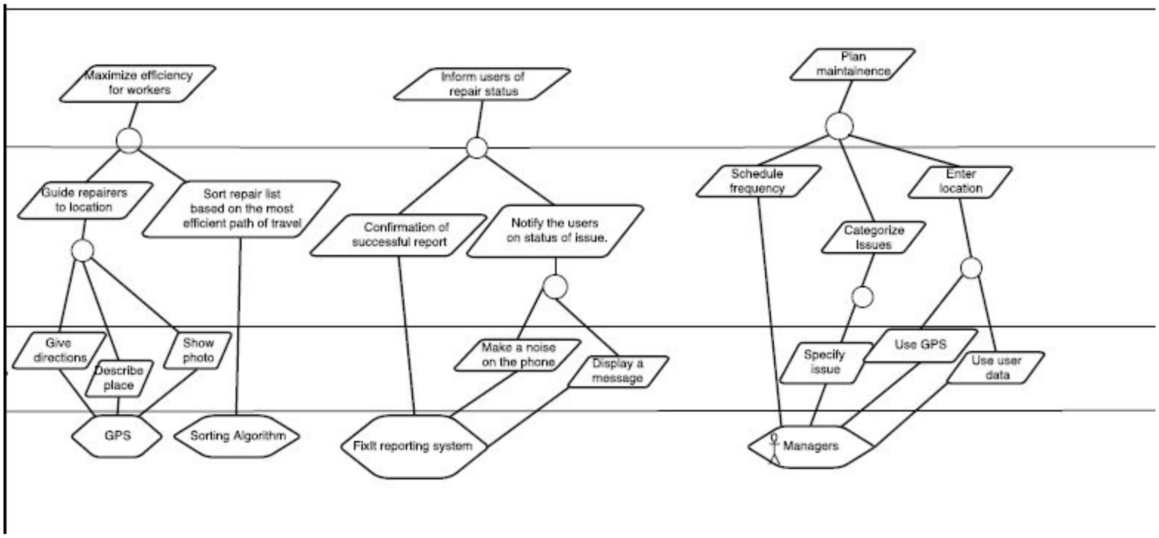
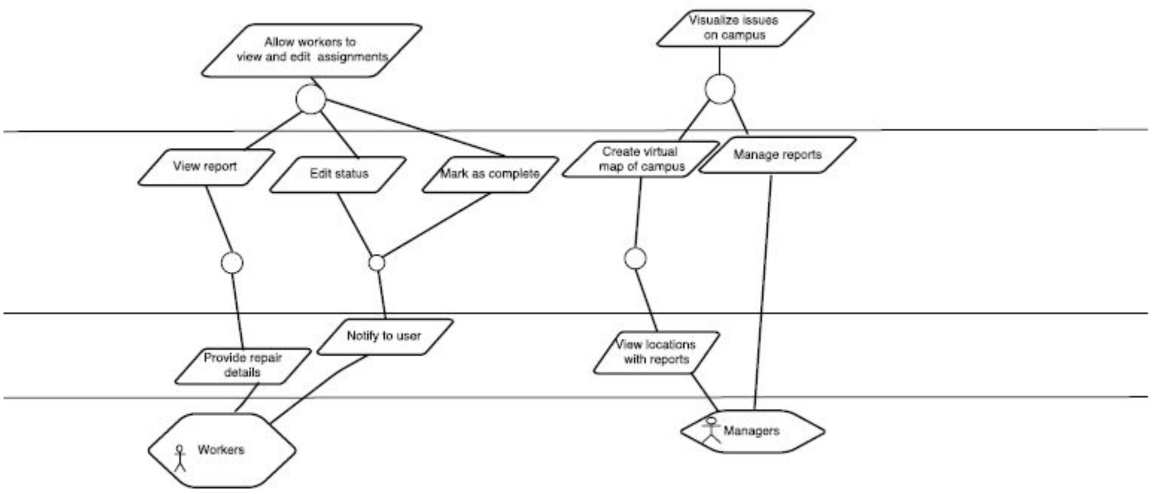

Maximize efficiency for workers
Inform users of repair status
Plan maintainence
Guide repairers to location
Sort repair list based on the most efficient path of travel
Confirmation of successful report
Notify the users on status of issue.
Schedule frequency
Categorize Issues
Enter location
Give directions
Describe place
Show photo
Make a noise on the phone
Display a message
Specify issue
Use GPS
Use user data
GPS
Sorting Algorithm
FixIt reporting system
Managers

DIAGRAM 3:


Allow workers to view and edit assignments
Visualize issues on campus
View report
Edit status
Mark as complete
Create virtual map of campus
Manage reports
Provide repair details
Notify to user
View locations with reports
Workers
Managers

Goal Annotations
Team Lucky

Level 1 Goals:

Goal: Report for repairs
Definition: If an issue occurs, users should be able to report any maintenance issues that require attention, provided that the user fills out the required information such as the location details and repair needed.
Type: Achieve goal
Priority: Medium
Source: Interview with the other team on 1/17
Q: What are the general steps a user must currently take to successfully submit a report?
A: The general steps are as follows:

1. User will start a new report
2. They will fill in location details if GPS coordinates are not given
3. They will describe the issue, so that workers can understand the severity of the issue
4. Pictures, and urgency level of the situation are all optional.

Goal: Successfully submit a non-duplicate report #1
Definition: Duplicate reports shall be minimized by primarily comparing location and repair category for similarities between reports, while also considering variables such as time frame and description.
Type: Maintain
Priority: High
Source: Interview with the other team on 1/17
Q: How does your system currently determine which reports are duplicates?
A: Two or more events that have the same location, category of emergency, time frame, and description of event.

Goal: Plan repairs #2
Definition: UCI upper management will be able to assign priority to an issue and assign repairs to their workers.
Type: Achieve
Priority: Medium
Source: Interview with the other team on 1/17
Q: Who can assign issues to employees? Managers, any employee, contractors, etc.?
A: The cleaning administration (managers) handles the assignment and picks people to deal with the issues.

Goal: Ensure security #3
Definition: To ensure security, all users need to enter their UCI ID. The system will be able to take that data and determine if they are a student, worker, manager, or faculty member. Depending on who they are, they will always be granted certain privileges and priorities.
Type: Soft goal
Priority: High
Source: Interview with the other team on 1/17
Q: How should the system to verify the students' information?
A: Cross referencing the student directory if the UCI ID is in the database and if the given name and email match the ID.

Goal: Maximize efficiency for workers #4
Definition: The app would help workers determine the location of the issue, and it will route the most efficient path of travel to save workers' time.

Type: Soft goal
Priority: Medium
Source: Interview with the other team on 1/17
Q: If the site of a needed repair is in a building, how would you normally specify the location in the building such that a repair person knows exactly where to go?
A: It can be specified by the user-inputted data if applicable. Otherwise, the GPS will try to pinpoint the location as accurately as possible.

Goal: Inform users of repair status #5
Definition: Users should first receive confirmation that they successfully submitted a report. Afterwards, users should be able to receive frequent status notifications on the report they submitted.
Type: Achieve goal
Priority: Medium
Source: Interview with other team on 1/17
Q: How should the issue reporter be notified when their issue has been updated (email, in-app notification, text)?
A: They should be notified by an in-app notification through a text bubble pop-up.

Goal: Plan maintenance #6
Definition: They will be able to schedule frequency and location for maintenance. They can categorize what maintenance should be done such as non-regular and regular maintenance. Regular maintenance will be based on what was reported as most common during the last year.
Type: Achieve goal
Priority: High
Source: Interview with other team on 1/17
Q: What maintenance and repair issues have people most frequently reported in the past year?
A: Broken pipes, ground is unstable, electricity outages, clogged bathrooms, and technological failures.

Goal: Allow workers to update an issue's status #7
Definition: To speed up the process of repairs, workers should be always able to update an issue so that those involved can see the current status of a repair.
Type: Achieve

Priority: High
Source: Interview with the other team 2/2
Q: Why is it important for workers to be able to update the status of an issue for others to see?
A: It is important for the worker to constantly update the issue, so that all parties affected can know the status of the issue.

Goal: Visualize issues on campus #8
Definition: UCI upper management can navigate a visualization map of the campus, which will assist them when planning repairs. Users can edit, add notes, and design plans for repairs.
Type: Achieve
Priority: Low
Source: Interview with the other team on 2/2
Q: How will a visualization map aid the user in planning maintenance for reports?
A: They will have a better view of what issues are available, and they can also sort, edit, add notes, and design plans to address these issues.

Level 2 Goals:

Goal: Enter repair location details
Parent Goal: Report for repairs
Definition: Before sending in a report, the app will always check if the user has GPS. If so, it takes in coordinates, otherwise the user must describe the issue in more detail by providing the location.
Type: Achieve
Priority: High
Source: Interview with other team on 1/17
Q: What are the general steps a user must currently take to successfully submit a report?
A: The general steps are as follows:

1. User will start a new report
2. They will fill in location details if GPS coordinates are not given
3. They will describe the issue, so that workers can understand the severity of the issue
4. Pictures, and urgency level of the situation are all optional.

Goal: Classify repairs
Parent Goal: Report for repairs
Definition: Users should be allowed to classify the repair needed from a set of categories
Type: Achieve goal
Priority: Medium
Source: Interview with other team on 1/17
Q: What are the general steps a user must currently take to successfully submit a report?
A: The general steps are as follows:

1. User will start a new report
2. They will fill in location details if GPS coordinates are not given
3. They will describe the issue, so that workers can understand the severity of the issue
4. Pictures, and urgency level of the situation are all optional.

Goal: Issues are in different locations
Parent Goal: Successfully submit a non-duplicate report
Definition: If the coordinates of an issue differ from another issue significantly and have different photos, then then we know that the reports are in a different location, which signify that it is not a duplicate issue.

Type: Maintain
Priority: High
Source: Interview with other team on 1/17
Q: How does your system currently determine which reports are duplicates?
A: Two or more events that have the same location, category of emergency, time frame, and description of event.

Goal: Issues have different categories
Parent Goal: Successfully submit a non-duplicate report
Definition: Different categories allow our sorting algorithm to determine if a report is a duplicate.
Type: Soft goal
Priority: High
Source: Interview with other team on 1/17
Q: How does your system currently determine which reports are duplicates?
A: Two or more events that have the same location, category of emergency, time frame, and description of event.

Goal: Assign priority
Parent Goal: Plan repairs
Definition: The upper management should be able to determine how important an issue is based on user data. They can then assign priority to those issues for their workers to resolve.
Type: Achieve goal
Priority: Medium
Source: Interview with other team on 1/31
Q: How are the report being prioritize, who is prioritizing them, or are there an algorithm?
A: The UCI facilities management services team will determine which reports to prioritize and they will prioritize the reports where lives are in danger, such as a dangerous gas leak, a potential flood, a building falling apart, etc. The managers of the management services will determine which reports to prioritize and the application will not do the prioritizing. There will also be an option for users to report whether a situation is dire and requires immediate assistance or not.

Goal: Assign repairs to workers

Parent Goal: Plan Repairs
Definition: After a report is submitted, the manager will be able to assign repairs to the ideal worker. They can send notifications to that worker, and can even assign repairs to workers not from UCI.
Type: Maintain goal
Priority: Low
Source: Interview with other team on 1/17
Q: Who can assign issues to employees? Managers, any employee, contractors, etc.?
A: The cleaning administration (managers) handles the assignment and picks people to deal with the issues.

Goal: Require ID
Parent Goal: Ensure security
Definition: User will need to provide their UCI ID credentials to access the reporting system.
Type: Maintain goal
Priority: High
Source: Interview with other team on 1/17
Q: How should the system to verify the students' information?
A: Cross referencing the student directory if the UCI ID is in the database and if the given name and email match the ID.

Goal: Validate users
Parent Goal: Ensure security
Definition: Check who the user to see if they are a student, or someone with higher priority such as a worker, manager, or faculty member.
Type: Maintain goal
Priority: High
Source: Interview with other team on 1/17
Q: How should the system to verify the students' information?
A: Cross referencing the student directory if the UCI ID is in the database and if the given name and email match the ID.

Goal: Guide repairers to location
Parent Goal: Maximize efficiency for workers

Definition: The repairers will know the exact location of the issue. They will be given the directions, description, and photo of the place..
Type: Maintain goal
Priority: High
Source: Interview with other team on 1/17
Q: If the site of a needed repair is in a building, how would you normally specify the location in the building such that a repair person knows exactly where to go?
A: There will be an option to call the UCI affiliates and the users will be able to specify which building/room need the repair in the description of the event.

Goal: Sort repair list based on the most efficient path of travel
Parent Goal: Maximize efficiency for workers
Definition: The repair list should be listed in a way that will produce an efficient day-by-day workflow for the repair process.
Type: Soft goal
Priority: Medium
Source: Interview with other team on 2/2
Q: How can FixIt optimize the workflow procedures for pending repairs?
A: It will generate a list from highest to lowest priority by using the FixIt sorting algorithm which sorts based on location and priority to increase efficiency. The worker can then save time by going to each of suggested locations on the list in order.

Goal: Confirmation of successful report
Parent Goal: Inform users of repair status
Definition: The user will be getting a confirmation that his/her report has been submitted successfully
Type: Achieve goal
Priority: Low
Source: Interview with other team on 2/1
Q: How should users be notified that their report has been successfully submitted?
A: They should receive a message saying that they have successfully submitted a report.

Goal: Notify the users on status of the issue
Parent Goal: Inform users of repair status
Definition: After a task is being mark as complete, the user (reporter) will be notify that the issue has been repaired by a noise and message.

Type: Achieve goal
Priority: Medium
Source: Interview with other team on 1/17
Q: How should the issue reporter be notified when their issue has been updated (email, in-app notification, text)?
A: They should be notified by in-app notification through a text bubble pop-up.

Goal: Schedule frequency
Parent Goal: Plan maintenance
Definition: Determine which issues require regular maintenance, such that they should be scheduled automatically by the app
Type: Achieve
Priority: High
Source: Interview with other team on 1/17
Q: What maintenance and repair issues have people most frequently reported in the past year?
A: Broken pipes, ground is unstable, electric failing, clogged bathrooms, and technological failures.

Goal: Categorize issues
Parent Goal: Plan maintenance
Definition: Specify the type of repair that needs to be done on the issue.
Type: Soft goal
Priority: Low
Source: Interview with other team on 2/2
Q: How would we categorize issues to determine what repair they need?
A: We would have several categories that the user can choose from. They can check all options that apply to the issue, such as: broken pipe, broken projector, broken light, clogged bathroom, and other technological failures.

Goal: Enter location
Parent Goal: Plan maintenance
Definition: Any issue must have its GPS coordinates or/and location described from the user data.
Type: Achieve goal
Priority: Medium

Source: Interview with other team on 1/31

Q: Some applications are unable to provide GPS but can still access the Internet; how would
your application deal with this problem?

A: The application should be able to use maps with a collaboration with an outside company and
providing GPS coordinates should be optional for the users. Users will be able to describe the
location.

Goal: View report

Parent Goal: Allow workers to update an issue's status

Definition: Facilities Management workers should be allowed to view any unresolved issues.

Type: Achieve goal

Priority: High

Source: Interview with other team on 2/2

Q: Who can view unresolved reports?

A: Workers and those who submitted the report can view unresolved reports. Users who submitted the report can request it to be opened again.

Goal: Edit status

Parent Goal: Allow workers to update an issue's status

Definition: When a worker updates a report, all relevant parties will be notified including the original reporter, the worker the issue has been assigned to, and any other workers who may have worked on the issue in the past.

Type: Achieve goal

Priority: High

Source: Interview with other team on 2/2

Q: What should happen when a worker, whether it is the one assigned the issue or not, updates a ticket?

A: When the worker updates a ticket, all parties involved will receive a notification of the status.

Goal: Mark as complete

Parent Goal: Allow workers to update an issue's status

Definition: Marking a report as complete does the following: notifies the original reporting user that the issue has been resolved, takes the report out of the system, and archives the report for future searches, thus reducing clutter.
Type: Achieve goal
Priority: Medium
Source: Interview with other team on 2/1 and 2/2
Q1: What should happen to a report once it has been resolved?
A1: It should be archived in case it needs to be reopened again by the sender.
Q2: Will the users get a response when a report is closed by the worker?
A2: The users should get a notification when a report has been deemed completed by the worker.

Goal: Create virtual map of campus
Parent Goal: Visualize issues on campus
Definition: All the reports with their locations can be seen with a broader view using the campus virtual map.
Type: Achieve goal
Priority: Medium
Source: Assumption
According to the interviews, the client seems to be prioritize worker efficiency. We believe a virtual map of the campus will give upper management a better idea on how to assign reports.

Goal: Manage reports
Parent Goal: Visualize issues on campus
Definition: The reports will be sorted for better visualization depending on the user's inputted parameters. For example, reports can be sorted by general area or priorities.
Type: Achieve goal
Priority: Medium
Source: Assumption
To increase the workflow of the workers, different ways of managing reports, by location, priority, data, can make the worker feel like they are in control of that app. Giving workers more control over the app makes them more efficient.

Level 3 Goals:

Goal: Give coordinates
Parent Goal: Enter repair location details
Definition: With the GPS coordinates, the workers will be able to pinpoint exactly where the issue is, to make the workflow more efficient.
Type: Achieve goal
Priority: Medium
Source: Interview with other team on 1/31
Q: Some applications are unable to provide GPS but can still access the Internet; how would
your application deal with this problem?
A: The application should be able to use maps with a collaboration with an outside company and

providing GPS coordinates should be optional for the users. Users will be able to describe the
location.

Goal: Describe location
Parent Goal: Enter repair location details
Definition: To increase efficiency for the workers, if a location is hard to find or if it is within the building, it will be stated and described.
Type: Soft goal
Priority: Medium
Source: Interview with other team on 1/31
Q: Some applications are unable to provide GPS but can still access the Internet; how would
your application deal with this problem?
A: The application should be able to use maps with a collaboration with an outside company and
providing GPS coordinates should be optional for the users. Users will be able to describe the
location.

Goal: Select Category
Parent Goal: Classify repair
Definition: Users will be able to select from a list of categories that is best suited for the situation.
Type: Achieve goal
Priority: Medium
Source: Interview with other team on 1/17
Q: "What are the general steps a user must currently take to successfully submit a report?
A:

1. User will start a new report
2. They will fill in location details if GPS coordinates are not given
3. They will describe the issue, so that workers can understand the severity of the issue
4. Pictures, and urgency level of the situation are all optional.

Goal: Coordinates differ a significant amount

Parent Goal: Issues are in different locations
Definition: If the coordinates of multiple reports differ significantly, the app will consider the issues as being distinct/unique.
Type: Maintain
Priority: High
Source: Interview with other team on 1/17
Q: How does your system currently determine which reports are duplicates?
A: Two or more events that have the same location, category of emergency, time frame, and description of event.

Goal: Photos show different areas
Parent Goal: Issues are in different locations
Definition: To minimize work for repairers, a picture of the issue will be shown, so the designated worker can be prepared before they arrive on-site.
Type: Soft goal
Priority: Medium
Source: Interview with other team on 1/17
Q: Is there any information, besides a photo, classification, and GPS coordinates, that a user should provide on a reported issue?
A: There will be an option to further describe the reports.

Goal: Send notification to UCI worker
Parent Goal: Assign repairs to workers
Definition: If a new issue is added to the workers' worklist, the worker will receive a notification within the app.
Type: Achieve goal
Priority: Low
Source:
Q: Should the Facilities Management employees get a notification or alert when a new issue is assigned to them?
A: Yes, they should get a notification when a new issue is assigned to them and they should be able to see their active assignments.

Goal: Place order with outside repair company
Parent Goal: Assign repairs to workers

Definition: If the issue requires special tools/skills, the order will be place with an outside repair company.
Type: Achieve goal
Priority: Medium
Source: Interview with other team on 1/17
Q: Take the outside facilities organization UCI does the most business with. How often do
workers from that company come in to make repairs?
A: Outside facilities are only contacted if it's a specialized issue that UCI affiliates are unable to handle such as fixing unstable ground.

Goal: Validate students
Parent Goal: Validate users
Definition: After a student types in their UCI ID, the app should be able to confirm the user is a student.
Type: Achieve goal
Priority: High
Source: Interview with other team on 1/17
Q: What information must students provide when they create accounts?
A: UCI ID, name, and UCI email.
*Gap in information. Needed to ask a question about validation with the other group

Goal: Validate workers
Parent Goal: Validate users
Definition: After a worker types in their UCI ID, the app should be able to confirm the user is a repair worker.
Type: Achieve goal
Priority: High
Source: Interview with other team on 1/17
Q: What information must students provide when they create accounts?
A: UCI ID, name, and UCI email.
*Gap in information. Needed to ask a question about validation with the other group

Goal: Validate manager
Parent Goal: Validate users

Definition: After a manager types their UCInetID, the app should be able to lookup and determine that the user is in fact a manager with higher access rights, versus a regular student/user.
Type: Achieve goal
Priority: High
Source: Interview with other group on 1/17
Q: How should the system to verify the students' information?
A: Cross referencing the student directory if the UCI ID is in the database and if the given name and email match the ID.
*Gap in information. Needed to ask a question about validation with the other group

Goal: Validate faculty
Parent Goal: Validate users
Definition: After typing in a faculty user's UCInetID, the app should be able to validate the user is a faculty user.
Type: Achieve goal
Priority: High
Source: Interview with other group on 1/17
Q: How should the system to verify the students' information?
A: Cross referencing the student directory if the UCI ID is in the database and if the given name and email match the ID.
*Gap in information. Needed to ask a question about validation with the other group

Goal: Give direction
Parent Goal: Guide repairers to location
Definition: The app should be able to guide the workers to the location of the issue.
Type: Soft goal
Priority: Medium
Source: Assumption
To increase the efficiency for the workers, we believe this feature is necessary. If a worker is not familiar with an area, the app should guide the worker to the correct place.

Goal: Describe place
Parent Goal: Guide repairers to location
Definition: Should describe where the location is and what needs to be fixed in case the GPS coordinates are not available.

Type: Soft goal
Priority: Medium
Source: Interview with other team on 1/17
Q: Is there any information, besides a photo, classification, and GPS coordinates, that a user should provide on a reported issue?
A: There will be an option to further describe the reports.

Goal: Show photo
Parent Goal: Guide repairers to location
Definition: An optional photo can be provided if users want to pinpoint an issue.
Type: Soft goal
Priority: Low
Source: Interview with other team on 1/17
Q: Is there any information, besides a photo, classification, and GPS coordinates, that a user should provide on a reported issue?
A: There will be an option to further describe the reports.

Goal: Make a noise on the phone
Parent Goal: Notify the users on status of issue
Definition: Helps users know if an issue has been resolved
Type: Achieve goal
Priority: Low
Source: Interview with other team on 1/17
Q: Is there any information, besides a photo, classification, and GPS coordinates, that a user should provide on a reported issue?
A: There will be an option to further describe the reports.

Goal: Display a message
Parent Goal: Notify the users on status of issue
Definition: Gives context of the status of the issue
Type: Achieve goal
Priority: Low
Source: Interview with other team on 1/17
Q: Is there any information, besides a photo, classification, and GPS coordinates, that a user should provide on a reported issue?
A: There will be an option to further describe the reports.

Goal: Specify an issue
Parent Goal: Categorize
Definition: Manager will analyze issues and try to understand how to describe what repairs might be needed.
Type: Soft goal
Priority: Medium
Source: Interview with other team 2/2
Q: Why is it important that the upper management check what the issue is before assigning it to their workers?
A: They should know what issue it is, so they can assign it to the correct worker who knows how to handle it.

Goal: Use GPS
Parent Goal: Enter location
Definition: Managers will use the GPS coordinates if they think the report was in the right location.
Type: Achieve goal
Priority: Medium
Source: Interview with other team 2/2
Q: How will the manager determine whether to use the provided GPS coordinates or the user description of the issue to determine where the repair needs to be done?
A: They will use their best judgment on the issue. If it appears that the user did not present the right GPS coordinates for the issue, then they will read the user data for the location and vice versa.

Goal: Use user data
Parent Goal: Enter location
Definition: Managers will use the user data if they think the GPS coordinates provided were inaccurate, or if they were not submitted.
Type: Achieve goal
Priority: Medium
Source: Interview with other team 2/2
Q: How will the manager determine whether to use the provided GPS coordinates or the user description of the issue to determine where the repair needs to be done?

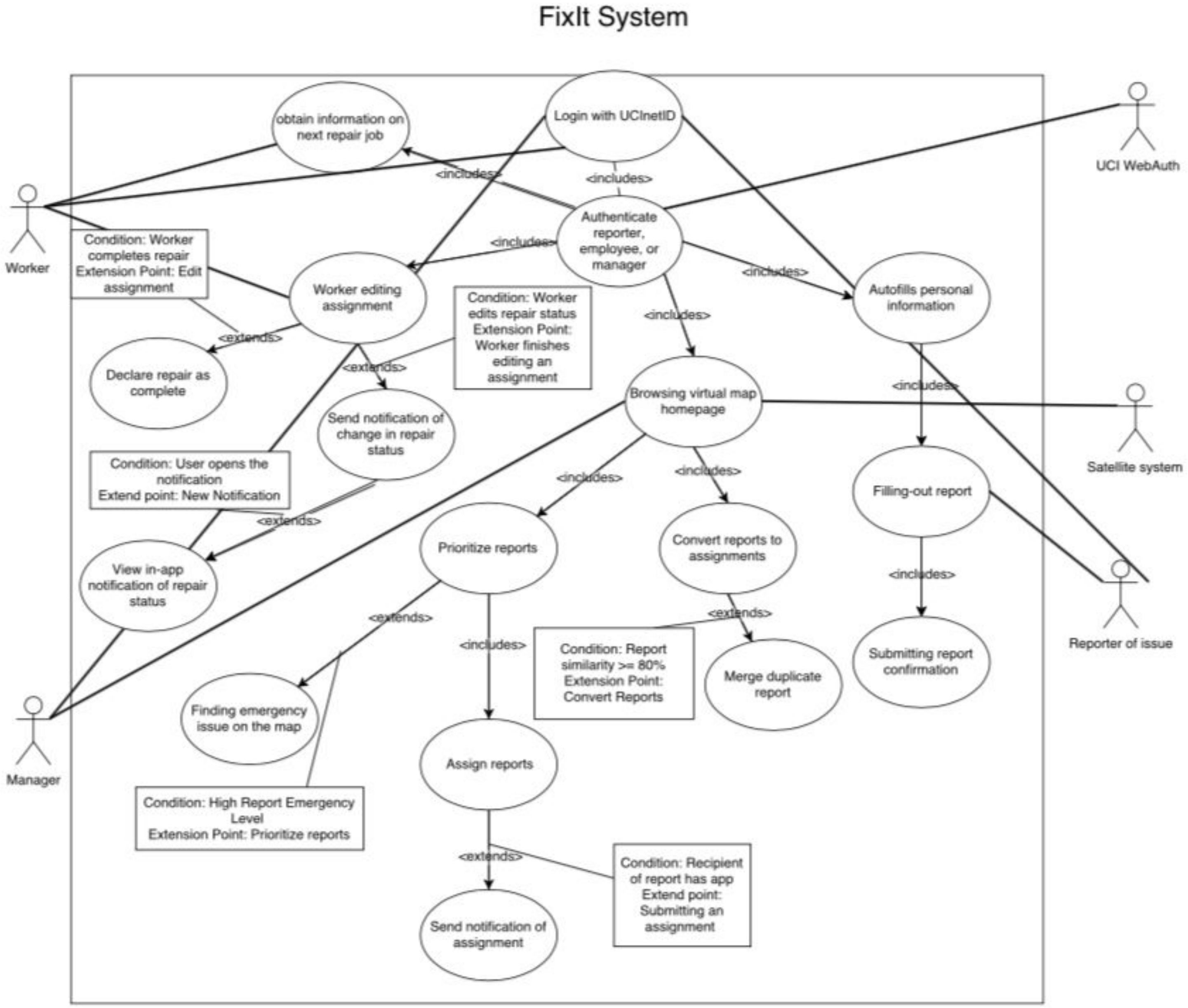
A: They will use their best judgment on the issue. If it appears that the user did not present the right GPS coordinates for the issue, then they will read the user data for the location and vice versa.

Goal: Provide repair details
Parent Goal: View report
Definition: Tells the worker what they have to do in their assignment
Type: Achieve goal
Priority: High
Source: Interview with other team on 2/2
Q: Why is it important that the upper management check what the issue is before assigning it to their workers?
A: They should know what issue it is, so they can assign it to the correct worker who knows how to handle it.

Goal: Notify to user
Parent Goal: Edit status
Definition: Notifies user whenever the worker edits the issue to know how it is being handled.
Type: Achieve goal
Priority: Low
Source: Interview with other team on 1/17
Q: How should the issue reporter be notified when their issue has been updated (email, in-app notification, text)?
A: They should be notified by in-app notification through a text bubble pop-up.

Goal: View locations with reports
Parent Goal: Create virtual map of campus
Definition: Helps the manager see exactly where each report is, so they can plan accordingly.
Type: Achieve goal
Priority: Low
Source: Interview with other team on 2/2
Q: How will a visualization map aid the user in planning maintenance for reports?
A: They will have a better view of what issues are available, and they can also sort, edit, add notes, and design plans to address these issues.

A.2.3 Use case model


FixIt System
obtain information on next repair job
Login with UCInetID
UCI WebAuth
<includes>
<includes>
Worker
Authenticate reporter, employee, or manager
Condition: Worker completes repair Extension Point: Edit assignment
<includes>
Worker editing assignment
<includes>
Autofills personal information
Condition: Worker edits repair status Extension Point: Worker finishes editing an assignment
<extends>
<includes>
Declare repair as complete
<extends>
Send notification of change in repair status
Browsing virtual map homepage
<includes>
Satellite system
Condition: User opens the notification Extend point: New Notification
<includes>
<includes>
Filling-out report
<extends>
Prioritize reports
Convert reports to assignments
View in-app notification of repair status
<includes>
<extends>
<extends>
Reporter of issue
<includes>
Condition: Report similarity >= 80% Extension Point: Convert Reports
Merge duplicate report
Submitting report confirmation
Manager
Finding emergency issue on the map
Assign reports
Condition: High Report Emergency Level Extension Point: Prioritize reports
<extends>
Condition: Recipient of report has app Extend point: Submitting an assignment
Send notification of assignment

List of features for our short description:

Virtual map feature

Reporting feature

Edit reports feature

Assign reports feature

Validate user feature

Assign priority feature

Report list feature

Notification feature

Change status feature

Autofill Personal Information

Author: Richard Lai

Priority: Low priority; an optional feature that enhances the user experience.

Criticality: Low criticality; this use case is just for user convenience. It does not affect the operation of the application since users can manually enter information.

Source: *Report for repairs* in goal model

Short Description: When a user logs into the system with his/her UCInetID, the system will obtain information from the UCI database to fill out all the information (birthday, email, phone number, etc.) that is on file.

Goal(s):

1. Improve efficiency when reporting issues
2. Reduce errors when reporting issues

Primary Actor: UCI database

Secondary Actors: UCI affiliates, UCI repair workers

Preconditions: Users log into the application/website with his/her UCInetID.

Success End Condition: The system automatically fills the user's information with no errors.

Failed End Condition: The system fails to fill the user's information automatically, or the system fills the user's information incorrectly.

Trigger: UCI affiliates login into the application/website with his/her UCInetID. When the user chooses to report a problem, the app has a report template to fill in. The system automatically fills in any extra details of the report such as birthday, email, and phone number when the report is submitted.

Main Success Scenario:

1. The user starts the Fixit application and types in his/her UCInetID and password.
2. The user clicks on "Report a problem" on the homepage of the application.
3. The application then searches for the user's information from the UCI database.
4. The system then displays the user's information automatically on the screen.

Alternative Scenario:

1. The user starts the Fixit application.
2. The user forgets his/her UCInetID.
3. The user then clicks on "Forgot username or password".
4. The user types in his/her UCInetID and password.
5. The user clicks on "Report a problem" on the homepage of the application.
6. The application then searches the user's information from the UCI database entry that corresponds to his/her UCInetID.
7. The system then displays the user's information automatically on the screen.

Exception Scenarios:

1. The user starts the Fixit application and types in his/her UCInetID and password.
2. The user clicks on "Report a problem" on the homepage of the application.
3. The application then searches the for user's information from the UCI database.
4. The system says they are unable to find the user's information.
5. As a result, the system requests the users to fill in the personal information manually due to lack of back-end user information.

Relationships to other use cases:

1. The use case is included by the **Authenticate reporter, employee, or manager** use case since the system needs to know the type of users and verify their information before autofilling.
2. This use case includes the **Filling out report** use case because the user must fill out the necessary information before the system autofills the rest of the report.

Supplementary Information(2 sentences minimum): All UCI affiliates (Student, Facilities employee, or Facilities manager) should have UCInetIDs to login to the system. They should all have their personal information (UCInetID, email, phone number, address, etc.) in the UCI database.

Open Issues: We are not sure if there should be a “Forgot username” option for those who forgot their UCInetIDs. Also, the information stored in the UCI database may contain errors. As a result, we are unsure whether a warning should be displayed on the application, asking users to check their information to make sure it is error-free.

Filling Out Report

Author: Aditya Gupta

Priority: High Priority; The entire application works around the user’s ability to report issues.

Criticality: High Criticality; Without the ability to fill out reports, there will be no indication for managers or workers about issues.

Source: *Report for repairs* in goal model

Short Description: When a user notices something is broken or is in need of maintenance/fixing, he/she can open the app, provide information on the issue, and send it to the repair management. This will all be done through the **Reporting Feature** within the FixIt app.

Goal(s):

1. Easily submit reports on issues around the campus
2. Indicate issues to Facilities Management and view detailed reports
3. Indicate pending action

Primary Actor: Reporters

Secondary Actors: All FixIt Users

Preconditions: Users must sign in with a UCInetID account and have knowledge of the issue they wish to report.

Success End Condition: When a report is successfully filled out, the **Notification Feature** generates a successful submission notification and the status changes to ‘Reported’.

Failed End Condition: The notification did not show, which means the report did not submit successfully due to the incomplete or incorrect information.

Trigger: User logs in to the application with UCINet ID and tries to fill out all the required information. After user submits the report, the application verifies all the fields and information and decides if the submission is successful.

Main Success Scenario:

1. User opens the application

2. User logs in with UCINetID. Any other personal information will be autofilled by the system.
3. User will see an empty report template which they can fill out with the appropriate information to report a new issue.
4. User completes the whole report with the correct information and submits the report
5. User receives a notification that confirms his/her report has been successfully submitted.

Alternative Scenario:

1. User logs in to the application
2. User will see an empty form needed to be filled with appropriate information to report a new issue
3. User leaves a few required fields in the form and tries to submit the report
4. User notices that the submission did not go through and fills out the remaining information. Afterwards, the user submits the report again.
5. User receives a notification that confirms his/her report has been successfully submitted

Exception Scenarios:

1. User logs into the application
2. User will see an empty form needed to be filled with appropriate information to report a new issue
3. User does not provide the location of the issue and tries to submit the report
4. User notices that the submission did not go through and does not want to provide the location as he or she is unsure of the location.
5. User quits the application and the issue is not reported.

Relationships to other use cases:

1. This use case is included by the **Autofill personal information** use case because once an user tries to login, their credentials will be checked by the application and the personal information will be filled out.
2. This use case includes the **Submitting report confirmation** use case because the users will need a confirmation to show that they have successfully submitted a report.

Supplementary Information: This use case may be applicable to all the users, including managers and workers. Anyone who notices anything that needs fixing around the campus can become a reporter and may fill out a report.

Open Issues: Only reporters with a UCINetID are able to log into the application. Even if a person is on campus and notices something worth reporting, they will need a UCINetID to do so.

Submitting report confirmation

Author: Edward Fu

Priority: Low priority; an additional feature to give feedback to the user

Criticality: Medium criticality; without any feedback or confirmation on the submission, users will not be able to tell if the report has been submitted successfully.

Source: *Inform users of repair status* objective from the goal model

Short Description(how is it made and where is it found): The submitting report confirmation will be shown right after a user submits a report successfully. It is essentially offering informative feedback for better interface design and occurs in the **Notification feature.**

Goal(s):

1. Offer informative feedback to the user
2. Avoid unfinished submission

Primary Actor: Email system

Secondary Actors: Reporter

Preconditions: The use need to fill-out the report completely before submitting it to get a submit report confirmation

Success End Condition: A report confirmation message will be shown on the FixIt app after a report has been submitted successfully.

Failed End Condition: The message did not show, which means the report did not submit successfully due to incomplete or incorrect information.

Trigger: User logs in to the application with their UCInetID and fills out all the required information. After hitting submit, the application will give a message that the report has been submitted.

Main Success Scenario:

1. User logs into the application
2. User will see a template of the report and all the requirements that are necessary to be filled in.
3. User completes the whole report with correct information.
4. User receives a confirmation message.

Alternative Scenario:

1. User logs in to the application.
2. User will see a template of the report and all the requirements that are necessary to be filled in.

3. User completes the whole report but with some incorrect information.
4. User receives a confirmation message.

Exception Scenarios:

1. User did not fill in all the required information.
2. User clicks the submit button.
3. The message did not show due to incomplete information.
4. The application takes the user back to the information page.

Relationships to other use cases:

1. This use case is included by the **Filling-out report** use case, because the user will have to fill out the report completely before he/she can get a confirmation message back from the application.

Supplementary Information: The user can easily check if the report has been successfully submitted by getting information feedback from the application. If the report is submitted successfully, the application will jump to the next page with a green check mark and text that indicates the report has been submitted. If the submission is unsuccessful, the application will stay at the same page and tell the user the reason for the failure.

Open Issues: Human error can be a bit of a problem when it comes to submitting the report. The application can only tell if the report has been filled out, but the information that is given from the reporter might be incorrect.

Merge duplicate report

Author: Edward Fu

Priority: High priority; essential to make the app for efficient

Criticality: High criticality; without merging the duplicate reports, managers will need to take extra time looking through all the files to avoid assigning the same task to two different workers

Source: *Successfully submit a non-duplicate report* objective from the goal model

Short Description: Merging duplicate reports will happen after a user submits a report. Everything will be happen in the **Assign priority feature.**

Goal(s):

1. Avoid double assign task
2. Optimize process of assigning tasks

Primary Actor: System algorithm

Secondary Actors: Manager

Preconditions: User must successfully fill out a report that has the same/similar description, location, and picture, as an existing report on file.

Success End Condition: The duplicate reports will be merged into a single report.

Failed End Condition: Mistake in the algorithm wrongly analyzes two different issues as the same issue.

Trigger: Reporter successfully submitted a report that has already been submitted by another user. The application determines and merges duplicate reports by the location, description, and picture.

Main Success Scenario:

1. User fills in the same location
2. User uploads the same picture, with similar description
3. User receives a confirmation
4. FixIt application analyze a similar report has been reported
5. Compare two reports
6. Finalize it is a duplicate report
7. Merge both report into one

Alternative Scenario:

1. User fills in the same location
2. User uploads the same picture, with similar description
3. User receives a confirmation
4. User manually indicates it is a duplicate report
5. Merge the new report with the old report

Exception Scenarios:

1. User uploads a picture, pinpoint his/her location and a description
2. The algorithm can not find similar report in the database
3. The report gets create as a new report

Relationships to other use cases:

1. This use case is extended by the **Convert report to assignment** use case because merging duplicate reports must be done before assignments. Also, by merging duplicate reports, it minimizes the amount of reports in the database, which can make managing reports easier and more time efficient.

Supplementary Information: The manager can clearly see how many duplicate reports have been submitted regarding an issue by the number next to the report. Managers can then use that number to determine the importance of the situation.

Open Issues: Programs are not known to be good at recognizing. The inputs from the user can greatly affect the algorithm of the application.

Prioritize Reports

Author: Edward Chen

Priority: Medium priority; Essential for executing reports that need immediate attention.

Criticality: Medium criticality; Without a way to prioritize reports, important reports may be overlooked in favor of reports that could have waited later.

Source: *Plan repairs* objective from the goal model

Short Description: This use case is generated whenever the manager must sort the reports before assigning the reports with the highest priority. It can be found in the **Assign Priority feature.**

Goal(s):

- Reduce time spent on unimportant reports and more time on important ones.
- Handle emergency reports smoothly and efficiently

Primary Actor: Manager

Secondary Actors: Reports, virtual map feature

Preconditions: Manager is logged into the UCI Facilities Management system to access the reports. He has already opened up the virtual map and has selected a building which generates a list of reports for him to sort.

Success End Condition: The manager correctly sorts the reports from highest to lowest priority.

Failed End Condition: The manager ignores an emergency report, or doesn't sort the reports at all.

Trigger: The manager logs in to the UCI Facilities Management system. He clicks a building using the virtual map feature and a list of issues pops up.

Main Success Scenario:

1. The manager clicks a building on the virtual map and opens up a list of reports
2. He opens the top report which says that the carbon monoxide alarm needs to be replaced
3. The next few reports mention regular maintenance such as replacing a water filter. However, that is regular maintenance which is done automatically.
4. He decides the reports are already sorted and assigns the carbon monoxide alarm report to one of his employees.

Alternative Scenario:

1. The manager clicks a building on the virtual map and opens up a list of reports.
2. He opens the top report which says that there is a broken light in DBH building.
3. He sees a report in the middle which says the power is out in the EH building.
4. He decides to put the middle report on the top of the priority list instead.

Exception Scenarios:

1. The manager clicks a building on the virtual map and opens up a list of reports.
2. He opens the top report which says that the printer in the library is out of ink.
3. He opens the report in the middle which says that a window is broken in the SSH building.
4. He decides to handle the printer issue first even though the broken window is more important.

Relationships to other user cases:

1. This use case extends the **Finding emergency issues** use case because occasionally very serious issues such as a broken pipe are reported.
2. This use case includes the **Assign reports** use case because assigning reports usually defaults to the highest priority issues on the top of the list.
3. This use case is included by the **Browsing virtual map homepage** use case because to start sorting reports the manager has to log in to the homepage.

Supplementary Information: The sorting feature is easy to use. Users can drag reports up and down the priority list. The top of the priority list is the most important, while the bottom is the least important.

Open Issues: There may also be reports that will be left unfinished if higher priority reports keep taking up the top of the list. Furthermore, the sorting feature is still prone to human error.

<u>Assign Reports</u>

Author: Edward Chen

Priority: High priority; Essential for resolving reports by giving them to workers.

Criticality: Medium; Converting reports to assignments helps workers handle issues quickly, but it is not completely necessary.

Source: *Plan repairs* from the goal model

Short Description: This use is generated when the manager wants to assign a particular report, and it is found in our **FixIt assign reports feature.**

Goal(s):

- Reduce need for workers to manually receive reports
- Achieve a very proactive response environment to emergency reports

Primary Actor: Manager

Secondary Actors: List of Reports

Preconditions: Manager is logged into the UCI Facilities Management system to access the reports.

Success End Condition: The manager assigns the task to the right person.

Failed End Condition: The manager sends the assignment to the wrong person.

Trigger: The manager logs in to the UCI Facilities Management system. He clicks a building using the virtual map feature and a list of issues pops up.

Main Success Scenario:

1. The manager clicks a building on the virtual map and opens up a list of reports
2. He reads through the first report and determines that it has all the correct information.
3. He sends a notification to all his employees, and one of them responds.
4. He sends the assignment to that employee.

Alternative Scenario:

1. The manager clicks a building on the virtual map and opens up a list of reports.
2. He opens the top report and sees that the report must be attended to immediately: The lights are out in the SH building!
3. He sends a notification to all his employees and waits five minutes, but none of them respond.
4. He decides to call an outside repair company instead.
5. He tells them the issue, and they send a repairman immediately.

Exception Scenarios:

1. The manager clicks a building on the virtual map and opens up a list of reports.
2. He opens the top report which says that a tree branch is blocking the path of students in Ring Road.
3. He accidentally sends the task to the cleaning staff instead of the gardening staff.
4. The manager must cancel the assignment and resend it again to the proper recipient.

Relationships to other user cases:

1. This use case extends the **Send in-app notification of assignment or change in repair status** use case because the manager assigns tasks to his employees and it gives notifications on their phones if they have the app. Note that non-UCI repairers do not have the app.

2. This use case is included by the **Prioritize Reports** use case because sorting the list must be done before assigning the top report of the list.

Supplementary Information: UCI is well affiliated with outside repair companies in case an issue goes horribly wrong. Assignments are sent through a notification system.

Open Issues: We are not sure what constitutes a strong enough emergency that would warrant assign an issue to a repairman from an outside company. Also, the time it takes for employees to respond to the issue may be shorter than the time it takes for an outside repairman to handle the issue.

Login with UCInetID

Author: Saif Kayani

Priority: High priority; essential for authenticating all users on the FixIt system.

Criticality: High criticality; without a dedicated UCInetID login service, we would have to implement our own login system, which is costly and error-prone.

Source: *Ensure security* from the goal model

Short Description: The login prompt is displayed when the user opens the FixIt application, whether on the mobile app or web page. It is essentially a virtual security checkpoint to authenticate the user and determine their respective access rights within the FixIt system. Logging in is part of the **validate user feature**.

Goals:

- Ensure a secure system with all users being kept “in check”
- Enforce access rights within the system

Primary Actor: End-user (either reporter, Facilities worker, or Facilities manager)

Secondary Actors: UCI WebAuth service

Preconditions: Application must be initiated, either by opening the mobile app on a mobile device or accessing the web page on a browser.

Success End Condition: The end-user is able to successfully log in and is led to their respective user dashboard.

Failed End Condition: The end-user is unable to successfully log in, either due to entering incorrect credentials or not having student/employee status.

Trigger: The end-user initiates the FixIt application by either opening the mobile app or navigating to the FixIt web page. After being prompted for their UCInetID and password, the user enters their credentials and presses/clicks the “Login” button.

Main Success Scenario:

1. The user starts the FixIt application to then be prompted to enter their login details.
2. The user enters their UCInetID and password.
3. Behind-the-scenes, FixIt verifies the credentials.
4. Upon successful login, the user is led to their dashboard, which will vary depending on their access rights (i.e. whether they are a reporter, worker, or manager).

Alternative Scenario:

1. The user starts the FixIt application to then be prompted to enter their login details.
2. The user forgets their UCInetID and password and resets it through OIT's password reset tool.
3. User re-enters their UCInetID and new password.
4. FixIt verifies the credentials.
5. Upon successful login, the user is led to their dashboard, which will vary depending on their access rights (i.e. whether they are a reporter, worker, or manager).

Exception Scenario:

1. The user starts the FixIt application to then be prompted to enter their login details.
2. The user enters their UCInetID and password.
3. FixIt is not able to verify the credentials (i.e. the credentials were incorrect).
4. The user either resets their forgotten password or exits the FixIt app.

Relationships to other use cases:

1. The use case is includes the **Authenticate reporter, employee, or manager** use case since logging in with one's UCInetID is a prerequisite to being authenticated as a reporter, Facilities employee, or Facilities manager.

Supplementary Information: Since the FixIt app enables WebAuth login, it may be logical for there to be a password reset tool implemented in case the user forgets their UCInetID password.

Open Issues: Since the FixIt app uses WebAuth, a valid UCInetID and password will be required, similarly to how it would be required when logging into EEE, Canvas, etc. In the case where the user forgets their UCInetID password, we are not yet sure if it would be ideal to have our own UCInetID password reset tool, or simply a link that redirects the user to OIT's password reset tool.

Authenticate reporter, employee, or manager

Author: Saif Kayani

Priority: High priority; essential for differentiating between users on FixIt.

Criticality: High criticality; without an authentication system, access rights management would not exist, resulting in there being no defined distinction between say, a student and a Facilities manager.

Source: *Ensure security* from the goal model

Goal(s):

- Enforce access rights among users so as to prevent issues/conflicts
- Create a hierarchical user system to expedite workflow processes

Primary Actor: End-user (either reporter, Facilities worker, or Facilities manager)

Secondary Actors: FixIt system

Preconditions: FixIt app must be opened and user must have a valid UCInetID and password.

Success End Condition: The user enters their correct UCInetID and password, thus allowing the FixIt system to determine their access rights (reporter, worker, or manager).

Failed End Condition: The user enters an incorrect UCInetID and/or password causing an unsuccessful login. Alternatively, the user may be a former student, causing the FixIt authentication to fail.

Trigger: After initiating the app, the user enters their correct UCInetID and password. Once logged in, the FixIt system correctly authenticates the user.

Main Success Scenario:

1. The user initiates the FixIt app, either on their mobile phone or computer browser.
2. The user enters their UCInetID and password when prompted.
3. WebAuth then verifies the login information is correct.
4. The FixIt system, after the WebAuth verification, authenticates the user by checking their associated department and status to determine if they should have reporter, worker, or manager access rights.

Alternative Scenario:

1. The user starts the FixIt application on their phone or browser.
2. The user forgets their UCInetID's associated password.
3. He/she then uses OIT's password reset tool to change the password.
4. The user enters their UCInetID and newly-set password.

5. The FixIt system, after the WebAuth verification, authenticates the user by checking their associated department and status to determine if they should have reporter, worker, or manager access rights.

Exception Scenario:

1. The user starts the FixIt app and is prompted for their login details.
2. The user enters their correct UCInetID and password.
3. WebAuth verifies that the user entered their credentials correctly.
4. The FixIt system fails to authenticate the user as a reporter, worker, or manager due to the user being a former student, with a fully intact UCInetID.

Relationships to other use cases:

1. The use case includes the **Autofills personal information** use case because the information that will be autofilled for reports will depend on how the user is authenticated (i.e. whether they are a reporter, employee, or manager).
2. This use case includes the **Browsing virtual map homepage** use case since a user must be authenticated as a Facilities manager to have access to the feature of viewing the virtual map.
3. This use case includes the **Obtain information on next repair job** use case since a user must be authenticated as an employee to view their next designated repair job.
4. Finally, this use case includes the **Worker editing assignment** use case due to the fact that only a worker can edit a report/issue. As a result, the worker, when logging in with their UCInetID, must be authenticated as such.

Supplementary Information: All UCI-affiliated members have UCInetIDs, but that alone is not enough to log in to the FixIt system. Upon WebAuth verification, FixIt will also try to authenticate the user depending on their current status and department.

Open Issues: We are not entirely sure whether former students and former faculty and staff should be allowed to use FixIt. Even though they may have a UCInetID, they may not have current student/staff status.

<u>Finding emergency issues on the map</u>

Author: Richard Lai

Priority: High priority; an essential feature for workers to identify urgent issues quickly on the virtual map.

Criticality: High criticality; this feature allows workers to prioritize repairs. Removing this feature will make the overall workflow less efficient.

Source: *Plan repairs* from the goal model

Short Description: This use case helps Facilities Management determine issues that are more urgent than other issues. This feature will be found on the virtual map in the application.

Goal(s):

1. To help Facilities Management sort out emergency issues quickly and efficiently.
2. Allow repair workers to plan the path for all repairs.

Primary Actor: Repair worker

Secondary Actors: Manager, Fixit System

Preconditions: Only the Facilities Management team can view emergency issues on the map. The user must login to the application/website using their UCInetID and be identified as a repair worker or manager.

Success End Condition: The worker is able to classify different types of issues according to their level of emergency on the virtual map.

Failed End Condition: The worker is unable to classify different types of issues according to their level of emergency on the virtual map.

Trigger: Open the homepage of the app to see the virtual map. All facilities' issues will be pinpointed on the map. Different categories of emergencies will have different colors on the map so workers can identify them easily.

Main Success Scenario:

1. A worker logs into the application/website using his/her UCInetID.
2. After the system identifies him/her as a repair worker or manager, he/she clicks on "View Virtual Map."
3. The application displays all the facilities issues on the map. Different categories of emergencies will have different colors shown on map.
4. The worker finds an emergency issue on the map (indicated as a red dot on the map) and takes action immediately.

Alternative Scenario:

1. A worker login the application/website using his/her UCInetID.
2. After the system identify he/she as a repair worker or manager, he/she see a list of repair work assigned to him/her.
3. The worker clicks on "Sort by: emergency level (High to Low)".
4. He/she selects the first issue on the list.
5. Some options and details of the repair are shown. Then, he/she clicks on "View the location on a virtual map"
6. The worker finds an emergency issue on the map and takes action immediately.

Exception Scenario:

1. A worker logs into the application/website using his/her UCInetID.
2. After the system identifies him/her as a repair worker or manager, he/she clicks on “View Virtual Map.”
3. The application displays all the facilities issues on the map. Different categories of emergencies are not displayed on the map. They are represented as the same color.
4. The worker is unable to sort out emergency issues on the map.

Relationships to other use cases:

1. The use case is extended by the **Prioritize reports** use case. Prioritizing reports can occasionally run into emergency issues, if there are any. This is a rare scenario though since FixIt is usually not designed for emergency issues.

Supplementary Information: The issue category is based on what is selected by the reporter. Different workers should have a different virtual map since some issues may not be assigned to some workers.

Open Issues: We are not sure how the emergency issue will be represented on a map. I assumed that different categories of emergencies will have different colors on the map so it can be easily identified. We are not sure if workers receive a push notification on their phone when an emergency issue is assigned to him/her.

Worker editing assignment

Author: Richard Lai

Priority: High priority; Allowing workers to edit assignments help those involved to quickly be updated of the issue.

Criticality: Medium criticality; this feature would make our system very responsive and efficient, but taking it out would not destroy our current system.

Source: *Plan repairs* from the goal model

Short Description: When a repair worker opens the Fixit application, there is a list of repairs assigned to him/her. For each repair, there is an option to add or edit notes. The worker can also add or edit notes on the virtual map.

Goal(s):

1. Increase efficiency during the repair process.
2. Allow every repair worker to plan for each repair.

Primary Actor: Repair worker

Secondary Actors: Reports, Fixit System

Preconditions: Repair workers can add or edit notes to each facilities issue that is assigned to him/her. They need to login in with their UCInetID to verify their identity as a worker and access their assignments.

Success End Condition: The repair worker successfully adds or edits notes to an assigned facilities issue.

Failed End Condition: The repair worker fails to add or edit notes to an assigned facilities issue.

Trigger: The repair worker can edit notes on the virtual map. Each facilities issue is pinpointed on the virtual map. When the worker clicks on it, there will be an option to change the status of a report or edit and add notes.

Main Success Scenario:

1. A repair worker opens the Fixit application and types in his/her UCInetID and password.
2. After the system identifies him/her as a worker, he/she opens the virtual map on the
   application.
3. The repair worker clicks on a facilities issue and clicks on add/edit notes.
4. The repair worker types a description and clicks the “Save” button.
5. A message pops up notifying the user that the notes have saved successfully.

Alternative Scenario:

1. A repair worker opens the Fixit application and types in his/her UCInetID and password.
2. After the system identifies him/her as a worker, he/she finds a list of facilities issues assigned in order.
3. The repair worker clicks on one of the issues and chooses “add/edit notes”.
4. The repair worker types a description and clicks the “Save” button.
5. A message pops up notifying the user that the notes have saved successfully.

Exception Scenarios:

1. A repair worker opens the Fixit application and types in his/her UCInetID and password.
2. After the system identifies him/her as a worker, he/she finds a list of assignments to do.
3. The repair worker clicks on one of the assignments.
4. The repair worker discovers that he or she has already finished this assignment three days ago, and it has been reassigned again.
5. He changes the status of the report to “complete” which sends a notification to all parties involved.

Relationships to other use cases:

1. The use case is included by the **Authenticate reporter, employee, or manager** use case since a user must be authenticated as a worker to view/add/edit notes.
2. The use case extends the the **Send in-app notification of change in repair status** use case because the app does that when it is the repair status which the worker edits.
3. The use case extends the **Declare repair as complete use case** since the software will send a special notification of completion if the repair status is changed to "complete".

Supplementary Information: The workers can check and edit their assignments after logging into the app. They do not have access to the manager's virtual map.

Open Issues: We are not sure whether a repair worker can see other users' written notes. Also, we are not sure if a repair worker can edit other people's assignments.

Send notification of change in repair status

Author: Cesar Diaz

Priority: High priority, required for app functionality.

Criticality: High criticality, listed as a main feature of the application.

Source: *Inform users of repair status* from the goal model

Short Description: This occurs after a report has its repair status changed or when a comment/note is added to an existing report. It will create a brief summary of the change in status and send it to the user's email, send them a text, and display an in-app notification.

Goal(s):

1. Find a low-cost service for sending sms alerts to users.
2. Create succinct summaries of changes in repair status

Primary Actor: FixIt app

Secondary Actors: Outside sms sending service

Preconditions: User must have in-app/text/email notification preferences set to ON.

Success End Condition: Notification is sent through platforms that were opted into and can be viewed by the user.

Failed End Condition: The user does not receive the notifications.

Trigger: If a user updates a status of a report and then saves it. This can also be triggered by adding comments to the report.

Main Success Scenario:

1. An employee finishes a repair, logs into the application, and navigates to the report.
2. The employee sets the status of the repair as “complete”.
3. The application summarizes the report and sends a notification of completion to the reporter.
4. The reporter receives an in-app notification of the change in repair status.

Alternative Scenario:

1. The user who submitted the report has opted out of all notification types.
2. An employee sets the status of repair as complete on the user’s submission.
3. An in-app notification is not sent to the user.
4. The user does not view a notification of the change in status.

Exception Scenarios:

1. The user who submitted a report has opted into email notifications.
2. An employee sets the status of the repair as complete on the user’s submission.
3. An in-app notification is attempted to be sent to the user’s email that was used to register.
4. The user does not receive the email because it was sent to an invalid address.

Relationship to other use cases:

1. This use case is extended by the **Workers editing assignment** use case. This is because the worker must specifically change the status of the assignment to activate this use case.
2. This use case extends the **View in-app Notification of Repair Status** use case. This is because they must receive a notification of the change in repair status before they can open and view it.

Supplementary Information: This functionality is heavily dependent on users submitting accurate data about their contact details. It is assumed that users will manually enter their phone number and email correctly.

Open Issues: How do we create a good summary of the changes to a report if they’re more complex than just a change in status? How do we attempt to slow down notifications if a certain employee may be over-using the comments section of the report?

<u>Send notification of assignment</u>

Author: Cesar Diaz

Priority: High priority; Essential to employees being assigned tasks.

Criticality: High criticality; Without this, employees will only be able to see assignments if they are viewing the application

Source: *Inform users of repair status* from the goal model

Short Description: After the manager assigns an assignment to an employee, the employee will receive notification of the task that has been assigned. This will be done through all available channels, including text, email, and phone notification.

Goal(s):

1. Keep employees updated on what they've been assigned.
2. Prevent tasks from getting neglected if someone does not manually check the status.

Primary Actor: Employees

Secondary Actors: FixIt Application

Preconditions: A report must be created before it can be assigned.

Success End Condition: Employee is able to view that they have been assigned the task by management.

Failed End Condition: The employee was not able to receive the notification of assignment.

Trigger: Manager assigns an issue to a worker and changes the status to 'Assigned'. The worker's notification settings are used to send the message of assignment.

Main Success Scenario:

1. Manager logs in to the application.
2. Manager assigns report to an employee.
3. The assigned employee receives the notification through email, text and phone notification.
4. The employee clicks on the notification to view the details of their assigned task.

Alternative Scenario:

1. Management logs in to the application.
2. Management sees no report that needs assignment.
3. Management views the list of all reports and who has been assigned to them.
4. Management confirms that these assignments are correct and exits the application.
5. No employees receive a notification.

Exception Scenario:

1. Manager logs into the application.
2. Manager assigns an employee to a report.

3. The employee registered with an incorrect phone number and the notification is sent incorrectly.
4. The employee does not receive a text message but is notified by email and a phone notification.

Relationships to other use cases:

1. This use case is extended by the **Assign reports** use case because if an employee has the app working on his/her phone, he/she will receive a notification that the issue has been assigned to him/her.

Supplementary Information: The notification is not the only way to view an employee's assignments. They can view all of their assignments normally within the application.

Open Issues: We do not know if employees may choose to stop receiving notifications of assignments. There is no guarantee that the email and phone number an employee entered is correct, so it cannot be assumed that they received notifications via these channels.

Declare Repair as Complete

Author: Rachel Weber

Priority: Low-priority; The reporters are probably not too concerned about the status of issues they reported, but is nice for them to know that the issues have been handled.

Criticality: Medium-criticality; A failure to mark an issue as complete can result in the system continuing to try to repair the issue by assigning it to people. That should not become a significant issue as long as the repair is marked as complete fairly soon after being handled.

Source: *Inform users of repair status* objective from the goal model

Short Description: After a repair is done, the repair worker lets the involved parties know that the issue is handled. He/she does this through a notification feature in the app.

Goal(s):

1. Notify reporters that the work is done
2. Mark the work as completed in the database

Primary Actor: Repair worker

Secondary Actors: Repair database, reporters

Preconditions: Repair is completed and the repair worker is logged into the app.

Success End Condition: Users hear that the repair is complete and the repair gets removed from the list of ongoing issues.

Failed End Condition: The repair does not get removed from the list of ongoing issues.

Trigger: The repair worker completes the repair. He/she then taps an icon to mark the repair as complete.

Main Success Scenario:

1. The repair worker finishes repairing the issue.
2. The repair worker opens the app. He/she taps the icon to mark the repair as complete.
3. The app sends the database a query updating the entry for the repair such that the repair is handled.
4. The app sends the reporting users an in-app message saying that the repair is handled.
5. If the users' settings allow it, the app sends an email saying that the repair is handled.

Alternative Scenario:

1. The repair worker finishes repairing the issue.
2. The repair worker opens the app. He/she taps the icon to mark the repair as complete.
3. The app attempts to send the database a query to update the record for the repair. However, the wireless connection has broken. The query cannot be sent.
4. The app holds onto the query until it detects it has connectivity again. It then sends the query which updates the database.
5. The app sends the reporting users an in-app message saying the repair is handled.
6. If the users' settings allow it, the app sends an email saying the repair is handled.

Exception Scenarios:

1. The repair worker finishes repairing the issue.
2. The repair worker opens the app. He/she taps the icon to mark the repair as complete. The app attempts to send the database a query to update the record for the repair. However, the wireless connection has broken. The query cannot be sent.
3. The app holds onto the query. While doing so, the app crashes and the query gets lost.
4. The app never informs the database that the issue was handled.

Relationships to other use cases:

1. This use case is extended by the **Worker editing assignment** use case because after declaring a repair is complete, many parties are notified. This includes the people who reported the issue and the manager, so that the manager knows that he can assign more tasks to that worker.

Supplementary Information: The user should have an email associated with his/her account. That information is never missing.

Open Issues: Connectivity issues may sometimes delay reports of completion from going through. If the report of completion gets lost, the repair worker has to try to mark the problem as “complete” again and may forget to do so.

Obtain Information on Next Repair Job

Author: Rachel Weber

Priority: High priority; We prefer for the worker to use his/her time efficiently by immediately knowing where the next repair is.

Criticality: Low criticality; The back-end will continue to operate even if the worker cannot immediately see the information

Source: *Maximize efficiency for workers* from the goal model

Short Description(how is it made and where is it found feature): The information for the repairs come from a database. They are then organized into a list in the app. The first item on the list should be the next repair.

Goal(s):

1. Give the worker the next repair to do
2. Choose the next repair such that the worker’s travel between repair locations will be most efficient in the long run.

Primary Actor: Repair worker

Secondary Actors: Repair database

Preconditions: Users have reported repairs which are now stored in the repair database.

Success End Condition: The workers knows the next issue to repair.

Failed End Condition: The worker has no idea what to do next and so spends a precious 30 minutes walking across the campus back to the repair office to obtain further orders.

Trigger: The repair worker logs into or reopens the app. The repair worker taps the icon to access the list of repairs.

Main Success Scenario:

1. The repair worker logs into or opens the app.
2. The repairer taps the icon to access the list of repairs.
3. At the top of the list, the worker finds the next repair.
4. The worker taps that repair issue for more information.

Alternative Scenario:

1. The repair worker logs into or opens the app.
2. The repairer taps the icon to access the list of repairs.
3. The list is empty. The worker hits refresh a few times. It turns out the phone did not have connectivity. The phone finally displays repairs.
4. The worker taps the repair issue at the top.

Exception Scenarios:

1. The repair worker logs into or opens the app.
2. The repairer taps the icon to access the list of repairs.
3. The list is empty. The worker does not realize that the connectivity is bad. He/she does not think of trying to refresh.
4. The worker starts walking back to the manager's office to clock out, thinking that there is no work to be done.

Relationships to other use cases:

1. This use case includes **Authenticate reporter, employee, or manager** use case, because the repair worker must be logged in before he/she can view the next repair.

Supplementary Information: The system has already gone through the repairs and ordered them such that the worker will handle them in the most efficient order. The repair worker knows to choose the first item (and not yet bother with the rest). The app may show a message telling the repairer to do the repairs in order, starting with the first one.

Open Issues: Connectivity issues can prevent the app from updating the list of repairs. This may cause a problem for getting work done when the user is too dependent on the app for information.

View in-app notification of change in repair status

Author: Aditya Gupta

Priority: High priority; essential to keep a track on progress of multiple reports

Criticality: High criticality; Without this, users will not know as the report progress through various stages.

Source: *Allow workers to view and edit assignments* from the goal model

Short Description: After the manager or a worker changes the status of a issue using the **Change status feature**, the application sends a notification to everyone involved with the report. All the recipients can then view the new status attached to the report.

Goal(s):

1. Keep everyone updated about the progress in resolving a issue.
2. Prevent tasks from getting neglected if someone does not manually check the status.

Primary Actor: All users

Secondary Actors: FixIt Application

Preconditions: User must have used the **Change status** feature to change the status of a report to trigger the notification.

Success End Condition: Everyone is able to receive and view the notification when the status is changed.

Failed End Condition: A change in status does not lead to some applicable users receiving the notification.

Trigger: Manager assigns an issue to a worker and changes the status to 'Assigned'. A worker starts working on an issue and changes the status to 'In Progress'. A worker finishes working on an issue, successfully resolves the problem and changes the status to 'Resolved'.

Main Success Scenario:

1. Manager logs into the application.
2. Manager will see an unassigned report with status 'Reported'.
3. Manager will decide which worker will be assigned to the issue and updates the report.
4. The status of the report changes to 'Assigned' and the user (reporter), the assigned worker and others receive notification of the status change.
5. They view the triggered notification.

Alternative Scenario:

1. Worker logs into the application.
2. Worker will see an assigned report with status 'Assigned'.
3. Worker will decide to work on the assigned issue and updates the report.
4. The status of the report changes to 'In Progress' and the user (reporter), the assigned worker and others receive notification of the status change.
5. They view the triggered notification.

Exception Scenario:

1. Worker logs in to the application.
2. Worker will see an assigned report with status 'Assigned'.
3. Worker decides not to work on the assigned issue right now and moves on to another report.
4. The status of the report does not change, and no one receives any notifications.

Relationships to other use cases:

1. This use case is extends by the **Send in-app notification of change in repair status** use case because the application will need to be able to send the notification back to the other users so they can view it.

Supplementary Information: The notification is not the only way to view the current status of the report. All users can, at any time, open the report in their application to view the current status of the application.

Open Issues: There is no feature for a user to stop receiving updates about a particular issue. If they find these notifications spamming, they need to be able to stop receiving them. There is no guarantee that when a notification is send to a user, he/she will view it or act accordingly. They might accidently or voluntarily choose to ignore it.

Browsing virtual map homepage

Author: Edward Chen

Priority: High priority; Very important for our campus navigation feature.

Criticality: High criticality; without this feature, navigating lists of reports would be very taxing.

Source: *Visualize issues on campus* from the goal model

Short Description: This use case is generated when the manager logs into the app. Once he logs in, the virtual map homepage automatically pops up, and , and they can decide to search for reports, browse a comprehensive list of reports, or browse the virtual map from building to building. It is found in our FixIt **virtual map feature.**

Goal(s):

1. Reduce the time searching for a report
2. Make managing reports more efficient
3. Achieve an efficient virtual map feature

Primary Actor: Manager

Secondary Actors: Virtual Map

Preconditions: Manager must be logged into the UCI Facilities Management system to have access to the virtual map feature.

Success End Condition: The manager locates the report quickly by determining its general location.

Failed End Condition: The manager fails to locate the report, or takes too long to locate the report.

Trigger: The manager opens the app and enters his credentials. He is logged into a special interface where he has additional privileges over other users. He selects a button called navigate reports, and a virtual map expands to view on his screen where he can enter the ID or description of the report.

Main Success Scenario:

1. The user opens the map and enters the ID of the report
2. The report is automatically brought up
3. The user clicks the report and a description of the report pops up
4. The user confirms that that the report is the one he or she is looking for.

Alternative Scenario:

1. The user enters a brief description of the report such as location into a search query on the top of the virtual map.
2. A list of all queries that fit the description pop up.
3. The user clicks through each report one by one until he finds the right one.
4. The user confirms that the report on the third of the list is the one he is looking for.

Exception Scenarios:

1. The user enters an ID from a report.
2. No reports pop up, so he decides to enter the location of the report.
3. He searches through the list of the reports, but is unable to find the report.
4. He moves on to the next report because the search information was inaccurate.

Relationships to other use cases:

1. This use case is included in **Convert report to assignment** use case because all the management is done in this virtual map interface.
2. This use case is included in the **Prioritize Reports** use case because the manager's must access the virtual map homepage in order to manage the reports.
3. This use case is included by the **Authenticate reporter, employee, or manager** use case because the system must verify the user is a manager if they want to access the virtual map homepage.

Supplementary Information: There are two search bars the user can utilize for locating a report: I.D. and general search. Typing in the exact ID will bring up that report, while searching for a general description is slower and brings up a couple of reports that match parts of the description. The comprehensive list of reports is a list of all submitted reports. It is useful if the manager wants to tackle reports from all locations. Double tapping a building on the virtual map brings up al list of reports from that location.

Open Issues: The general search query is subject to change. We are not sure if it should take in either a general description of the repair or a specific location.

Convert report to assignment

Author: Edward Chen

Priority: Medium priority; Somewhat important for increasing efficiency of handling reports.

Criticality: Medium criticality; important for managers to create clear assignments from reports to reduce the need for workers to decipher reports, but the system could run without it.

Source: *Visualize issues on campus* from the goal model

Short Description: This use case is generated when the manager brings up a list of reports and must validate the reports that are following protocol, and it can be found in the **edit reports feature.**

Goal(s):

1. Reduce time spent on incomplete or completed reports.
2. Create clear assignments that workers can understand.

Primary Actor: Manager

Secondary Actors: List of Reports

Preconditions: Manager must be logged in and using the virtual map feature before accessing the reports.

Success End Condition: The manager removes incomplete or completed reports and saves himself time. He can also edit reports if they are missing certain information

Failed End Condition: The manager wastes time on reports that don't have sufficient information.

Trigger: The manager logs in to the UCI Facilities Management system. The virtual map feature pops up, and he selects a button that opens a comprehensive list of report. After making sure the reports follow the correct protocol, he begins converting the reports to assignments.

Main Success Scenario:

1. The manager clicks a building on the virtual map and opens up a list of reports.
2. He sees reports that are marked as complete.
3. He takes the reports out and archives them in case they need to be re-opened.
4. He starts reviewing the reports one by one, and converts them into assignments by clicking the convert button.

Alternative Scenario:

1. The manager clicks a building on the virtual map and opens up a list of reports
2. He opens a report and sees that it is missing a description, and the photo taken of the issue is blurry.
3. He deletes the report after deciding it has insufficient information.
4. He opens up a report with no photo, but its description states that the issue is located in ICS Building 364.
5. He changes the GPS coordinates to match that location. Adds a note to check the GPS coordinates, then converts it to an assignment by clicking the convert button.

Exception Scenarios:

1. The manager clicks a building on the virtual map and opens up a list of reports.
2. He sees two reports with the same photo of the issue but a similar description.
3. He is unsure whether to toss the report or not.
4. He decides to merge the two reports because there is no time to think about whether they are actually different.

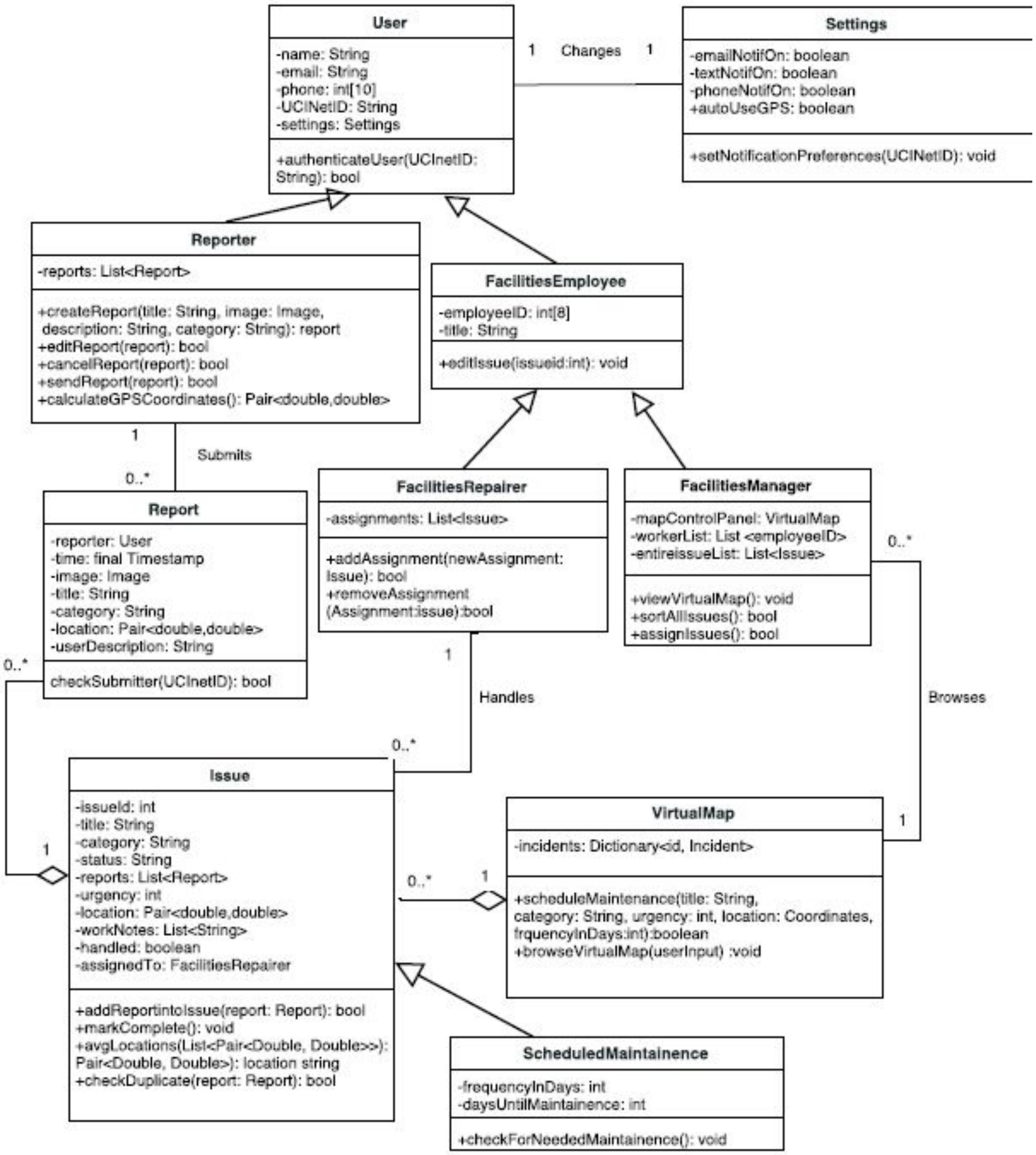
Relationships to other use cases:

1. This use case is included by the **Browsing virtual map homepage** use case because the manager must go there to find the report before he/she can convert it to an assignment
2. This use case extends the the **Merge duplicate report** use case because the manager must occasionally merge duplicate reports before converting them to an assignment

Supplementary Information: The convert button is what turns a report into an assignment once the manager is satisfied with its contents. The manager can clearly see if a report is marked as complete by checking if the color of its status is green. He can also see if a report is incomplete by checking what percentage of the report was filled out.

Open Issues: We are not sure if relying on the manager's judgment is the best way to manage reports. His judgment may falter from time to time.

A.2.4 Class Diagram model


User
-name: String
-email: String
-phone: int[10]
-UCINetID: String
-settings: Settings
+authenticateUser(UCInetID: String): bool
1
Changes
1
Settings
-emailNotifOn: boolean
-textNotifOn: boolean
-phoneNotifOn: boolean
+autoUseGPS: boolean
+setNotificationPreferences(UCINetID): void
Reporter
-reports: List<Report>
+createReport(title: String, image: Image, description: String, category: String): report
+editReport(report): bool
+cancelReport(report): bool
+sendReport(report): bool
+calculateGPSCoordinates(): Pair<double,double>
FacilitiesEmployee
-employeeID: int[8]
-title: String
+editIssue(issueid:int): void
1
Submits
0..*
Report
-reporter: User
-time: final Timestamp
-image: Image
-title: String
-category: String
-location: Pair<double,double>
-userDescription: String
checkSubmitter(UCInetID): bool
FacilitiesRepairer
-assignments: List<Issue>
+addAssignment(newAssignment: Issue): bool
+removeAssignment (Assignment:issue):bool
FacilitiesManager
-mapControlPanel: VirtualMap
-workerList: List <employeeID>
-entireissueList: List<Issue>
+viewVirtualMap(): void
+sortAllIssues(): bool
+assignIssues(): bool
0..*
1
Handles
Browses
0..*
0..*
Issue
-issueId: int
-title: String
-category: String
-status: String
-reports: List<Report>
-urgency: int
-location: Pair<double,double>
-workNotes: List<String>
-handled: boolean
-assignedTo: FacilitiesRepairer
+addReportintoIssue(report: Report): bool
+markComplete(): void
+avgLocations(List<Pair<Double, Double>>): Pair<Double, Double>): location string
+checkDuplicate(report: Report): bool
1
VirtualMap
-incidents: Dictionary<id, Incident>
+scheduleMaintenance(title: String, category: String, urgency: int, location: Coordinates, frquencyInDays:int):boolean
+browseVirtualMap(userInput) :void
1
0..*
1
ScheduledMaintainence
-frequencyInDays: int
-daysUntilMaintainence: int
+checkForNeededMaintainence(): void

Class Annotations

User - This is our primary class that is generalized to multiple subclasses. FixIt first determines who the user is by checking their UCInetID. The user's class determines what operations they can do that are specific to their class, such as a manager being able to assign reports. Derived from all the human actors sharing common attributes, and partially February 17th field notes #1

Settings -Settings is important because every user has different preferences and capabilities. Having settings means they can receive notifications related to reports or assignments. Derived from our February 17th field notes. #5

Reporter - A user that sends in a report that can be categorized into an issue. Derived from primary actor.

FacilitiesEmployee - Derived from the two human actors: Manager and Worker. The former prioritizes and assigns issues to workers, while the latter handles the actual issues themselves.

FacilitiesRepairer - Derived from the actor which does the actual repairing

FacilitiesManager - Derived from the human actor who prioritizes certain issues and assigns them to employees

Report -This class is important for the user because filling in all the required fields of a report makes handling it easier. Derived from February 12th field notes # 8 and Case Study A

Issue - This class is important to classify reports together and make it easier to handle multiple reports. Derived from February 12th field notes # 2 and our homework 4 use case: Convert report to assignment

ScheduledMaintainence - This class is important because regular maintenance is different and must be handled differently from other maintenance since it must be done regularly. Derived from our February 17th field notes # 6

VirtualMap - This class is important because it allows managers to efficiently assign and sort reports with a very smooth and effective virtual medium. Derived from our homework 4 use case: Browsing virtual map home page.

A.2.5 Elicitation Materials
Includes Field Notes from Homework 2, 3, 4, 5, 6, 7

Homework 2:

1.How does your system currently determine which reports are duplicates?
Two or more events that have the same location, category of emergency, time frame, and description of event.

2.What are the general steps a user must currently take to successfully submit a report?

Go on the application, fill in the description of event, ensure location is reported, choose category of event, and optional picture.

3.How should we notify employees about scheduled maintenance on campus?
Contact them via the application.

4.What maintenance and repair issues have people most frequently reported in the past year?
Broken pipes, ground is unstable, electric failing, clogged bathrooms, and technological failures.

5.Take the outside facilities organization UCI does the most business with. How often do
workers from that company come in to make repairs?
Outside facilities are only contacted if it’s a specialized issue that UCI affiliates are unable to handle such as fixing unstable ground.

6.If the site of a needed repair is in a building, how would you normally specify the location in the building such that a repair person knows exactly where to go?
There will be an option to call the UCI affiliates and the users will be able to specify which building/room need the repair in the description of the event.

7.How do you currently categorize the urgency of repairs?
Prioritize issues that are most life threatening/cause significant damage.

8.What repair classifications (i.e. plumbing, sidewalks and roads) exist in your current repair system?
Plumbing, sidewalks, roads, and, electricity.

9.What information must students provide when they create accounts?
UCI ID, name, and UCI email.

10.How should the system to verify the students’ information?
Cross referencing the student directory if the UCI ID is in the database and if the given name and email match the ID.

11.In the app, what do you define as the difference between faculty users and staff users?
The faculty users are the people that work for the school with the same privileges as students and staff users are the people that do the repairs.

12.Is there any information, besides a photo, classification, and GPS coordinates, that a user should provide on a reported issue?

There will be an option to further describe the reports.

13.Who can assign issues to employees? Managers, any employee, contractors, etc.?
The cleaning administration (managers) handles the assignment and picks people to deal with the issues.

14. When an FM employee sets the urgency of an issue, is it only visible to them? Or are urgency levels global to all employees?
It should be visible to all employees. It is important that every employee know how urgent an issue is to know which to solve first.

15.How should the issue reporter be notified when their issue has been updated (email, in-app notification, text)?
They should be notified by in-app notification through a text bubble pop-up.

16. If the original reporting affiliate cancels the issue, should it be permanently removed from the view of facilities management?
There should be an option to retrieve the data of the issue, so that if the user cancels the issue by accident they can submit again.

17. What platforms should the application support?
The application should support the most popular platforms such as Android, Iphone, Mac, PC, Linux.

18. If the site of a needed repair is in a building, how would you normally specify the location in the building such that a repair person knows exactly where to go?
It can be specified by the user-inputted data if applicable. Otherwise, the GPS will try to pinpoint the location as accurately as possible.

Homework 3:

Field Notes (7 pages)

1/17/18

Edward Chen, Kim Lang, Anthony Hei Tung Cheung... +4 moreIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

January 17, 2018 at 10:43am

- Reply
- _Message actions

Hello Team BBB,
Professor Navarro paired our 4 pm discussion group with your 5 pm because of lack of room.
Since some of us have class starting at 5, we were thinking we could compromise by having our meetings be from 4:30 to 5:30. Or if none of your members have scheduling issues, we can also go to the 4:00 one. Please let us know.
Thanks. Edward Chen (Team Lucky)

Anthony Hei Tung Cheung, Kim Lang, Edward Chen... +4 moreIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

January 17, 2018 at 1:57pm

- Reply
- _Message actions

Hello Edward,

Some of our members have class/work from during the time frame of 4-4:50, but we will be able to have some members come at 4:30 and have our meetings go on from there.

Sincerely,
Anthony

We then met with Team BBB on 1/17/18 where we had them answer our elicitation questions.
They in turn answered ours.
We submitted the questions they answered in homework 2.

1/31/18

Message 1:
Edward Chen, Kim Lang, Anthony Hei Tung Cheung, Kevin Chow, Andrew Q. Vo, Stanley K. Tang, Jackson Tsoi IN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
January 31, 2018 at 4:50pm

- Reply
- _Message actions

For homework 3, we are required to document field notes which detail our meetings together and ask any questions. Since our time schedules are inconvenient, do you guys mind sending questions online and answer them that way?
We can also meet up some other time if you really want to get this done.

Attached is a list of questions we have for you.
Please finish them by friday, and we will finish any questions you have by friday as well.

Reply 1:
Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
January 31, 2018 at 6:52pm

- Reply
- _Message actions

We have answered your questions

*He attached his field notes here

Edward Chen, Kim Lang, Anthony Hei Tung Cheung, Kevin Chow, Andrew Q. Vo, Stanley K. Tang, Jackson Tsoi IN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
January 31, 2018 at 6:30pm

- Reply
- _Message actions

By the way, usually Navarro gives the class free time towards the end of class on Thursday, so if you could send a couple of guys over and we can discuss the field notes that'd be great.
We have to document our meetings and emails. That's why I'm a bit rushed.

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
January 31, 2018 at 6:54pm

- Reply
- _Message actions

The field notes are just about what took place during our interactions, but some of our members will be attending lecture tomorrow.

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
January 31, 2018 at 7:11pm

- Reply
- _Message actions

Alright cool, let me know if you have any questions too if you want to add more to your guys' field notes!
Thanks for the reply man!

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
January 31, 2018 at 8:04pm

- Reply
- _Message actions

Can you and your group finish answering the questions on this document? Thanks in advance!

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
January 31, 2018 at 8:20pm

- Reply
- _Message actions

I'll send you the questions tomorrow at 12. I have to review the answers with my group so that you can get some good info.

I went over the following questions to Anthony. Their group then finished answering them on discussion day:

Start of Elicitation 1/31/18
1.What is your estimated cost for this project?
$250,000

2.How are the report being prioritize, who is prioritizing them, or are there an algorithm?
The UCI facilities management services team will determine which reports to prioritize and they will prioritize the reports where lives are in danger, such as a dangerous gas leak, a potential flood, a building falling apart, etc. The managers of the management services will determine which reports to prioritize and the application will not do the prioritizing. There will also be an option for users to report whether a situation is dire and requires immediate assistance or not.

3.Should the reported issues have fulfillment schedules which will help decide if any tasks are getting neglected or need immediate action?
Yes, if a report has not been looked into within 24 hours, there will be a section with each report to make sure they are not forgotten.

4.When issues are completed, should the reports be archived or deleted?
The reports should be archived in order to keep information about the repairs needed on campus.

5.Should the Facilities Management employees get a notification or alert when a new issue is assigned to them?
Yes, they should get a notification when a new issue is assigned to them and they should be able to see their active assignments.

6.Should there be templates to make reporting of common issues easier?
There should be a list of possible problems and the user will be able to choose one, with the option of offering additional information.

7.Is there any penalty for reporting incorrect issues?
No users are connected by their UCI information and will be spoken with if they consistently report incorrect issues and do not cancel them.

Q.How do you currently categorize the urgency of repairs?
Prioritize issues that are most life threatening/cause significant damage.

8.For this answer, how would you determine if an event is considered life threatening or dangerous, and should it be based on user input?
It is up to the UCI facilities management service team, reports that are marked as dire should be prioritized and then it is up to what the team thinks of the ongoing reports.

Q.Who can assign issues to employees? Managers, any employee, contractors, etc.?
The cleaning administration (managers) handles the assignment and picks people to deal with the issues.

9.Can you clarify what you mean by cleaning administration; would that refer to cleaning duties or maintenance and repairs for the campus?
We are referring to the UCI facilities management services, the UCI affiliates that deal with these situations.

Q.What platforms should the application support?
The application should support the most popular platforms such as Android, Iphone, Mac, PC, Linux.

10.Some applications are unable to provide GPS but can still access the Internet; how would your application deal with this problem?
The application should be able to use maps with a collaboration with an outside company and providing GPS coordinates should be optional for the users. Users will be able to describe the location.

End of Elicitation 1/31/18

2/1/18

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
February 1, 2018 at 8:08am

- Reply
- _Message actions

Oh right. If we have time after class, my group is sitting close to the front on the left side.

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
February 1, 2018 at 9:16am

- Reply
- _Message actions

Anthony Cheung, Andrew Vo, Kevin Chow, Kim Lang, Jackson Tsoi, Stanley Tang

I went over the following questions to Anthony, and his group finished answering them that day:

Start of Elicitation 2/1/18

1.Will the users get a response when a report is submitted successfully?

There should be a popup that says a report has been submitted successfully.

2.How can you validate which user is the one using the application? (student, faculty, repair

worker)?

There should be buttons to make different accounts depending on whether the user is a student or

faculty member, or a repair worker. Students and faculty members will use their UCI IDs when

making an account while repair workers will be able to register with their worker IDs.

3.Will the users get a response when a report is closed by the worker?

The users should get a notification when a report has been deemed completed by the worker.

4.How can faculty pan regular maintenance using the app? Should the app assign work to

repair workers automatically?

The app should not assign work for them automatically, but the managers of the UCI facilities

management services should assign each. However, workers should be able to sign up for reports

if they are available.

5.What kind of information will the workers get when being assigned to a task?
They will get the information the user inputs in the report."

Our meeting with them lasted about an hour. We answered their questions as well that they had been saving.

Overall it was a very productive meeting.

End of Elicitation 2/1/18

2/2/18

We met online this time and chatted on Slack.
We sent them the questions, and they responded with:

Start of Elicitation 2/2/18

1."How would we categorize issues to determine what repair they need?
We would have several categories that the user can choose from. They can check all options that apply to the issue, such as: broken pipe, broken projector, broken light, clogged bathroom, and other technological failures.

2.How should users be notified that their report has been successfully submitted?
They should receive a message saying that they have successfully submitted a report

3.What should happen to a report once it has been resolved?
It should be archived in case it needs to be reopened again by the sender.

4.What should happen when a worker, whether it is the one assigned the issue or not, updates a ticket?
When the worker updates a ticket, all parties involved will receive a notification of the status.

5.Who can view unresolved reports?
Workers and those who submitted the report can view unresolved reports. Users who submitted the report can request it to be opened again.

6.How can FixIt optimize the workflow procedures for pending repairs?
It will generate a list from highest to lowest priority by using the FixIt sorting algorithm which sorts based on location and priority to increase efficiency. The worker can then save time by going to each of suggested locations on the list in order.

7.How will a visualization map aid the user in planning maintenance for reports?
They will have a better view of what issues are available, and they can also sort, edit, add notes, and design plans to address these issues.

8.Why is it important for workers to be able to update the status of an issue for others to see?
It is important for the worker to constantly update the issue, so that all parties affected can know the status of the issue

9."Why is it important that the upper management check what the issue is before assigning it to their workers?"
They should know what issue it is, so they can assign it to the correct worker who knows how to handle it.

10.How will the manager determine whether to use the provided GPS coordinates or the user description of the issue to determine where the repair needs to be done?
They will use their best judgment on the issue. If it appears that the user did not present the right GPS coordinates for the issue, then they will read the user data for the location and vice versa.

End of Elicitation 2/2/18

Overall, a very productive meeting. We ended contact with them after that.

Homework 4:

Field Notes

2/7/18

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

February 7, 2018 at 12:55pm

- Reply
- Message actions

Hey man,

Can you answer these questions for us?

Also can we meet tomorrow after class, so we have some form of field notes?

Or you can meet today in discussion at 4:00 as well if you want.

Tell me if you want someone to come during 5, but please be early if that happens.

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

February 7, 2018 at 1:05pm

- Reply
- Message actions

I'll ask my group right now. Also I don't think you attached any questions. Also can you answer these questions for us? Thanks man!

https://docs.google.com/document/d/1g08PNTFRE31rrmrlf5szNXwU3j5X1isvHZPOF_-NyhU/edit?usp=sharing

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

February 7, 2018 at 2:27pm

- Reply
- Message actions

Oh I sent you an email 3 days ago.

But here it is again.

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

February 7, 2018 at 2:44pm

- Reply
- Message actions

Can I ask you to write a little more detail in your responses as well? If you feel a question does not have sufficient detail in my case, feel free to ask me.

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

February 7, 2018 at 2:55pm

- Reply
- Message actions

Oh you're right. Here you go!

Start of Elicitation 2/7/18

1. Who is funding our project, and for how long will the funding last?
UCI is funding the project and the funding will last as long as the app exists.

2. Should we prioritize reports based on how frequently they are reported or how urgent the user says they are?
That is up to the UCI facilities management, but urgency should be prioritized.

3. Should the app validate users with their UCInetID as both students and employees of UCI have UCInetIDs?
Yes, they should be required to use their UCInetID.

4. How will a visualization map aid the user in planning maintenance for reports?
They will be able to know the location of the situation.

5. Why is it important that the upper management check what the issue iions before assigning it to their workers?
Different issues require immediate attention as opposed to smaller issues, up to the discretion of the upper management and the urgency levels the reports are specified as.

6. How will the manager determine whether to use the provided GPS coordinates or the user description of the issue to determine where the repair needs to be done?
That is up to the manager to determine which is more specific.

7. How will the application determine where the issue's location is if the user is not present in the same location?
The user should include the location or else the report is unable to be solved.

8. Who notifies the authorities to take action against those who submit incorrect reports?
The managers will deal with false reports.

9. In one answer there was mention of workers being able to see active or new assignments, but does that mean workers cannot see their inactive or completed assignments?
Workers should be able to see their inactive, completed, active, and new assignments.

End of Elicitation 2/7/18

2/9/18

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

February 9, 2018 at 7:19pm

- Reply

- Message actions

How about these questions please?

https://docs.google.com/document/d/1t8kgiQuY0LTXE1nFWm1nWzm2yacTbCgFbfLWTmnEms8/edit?usp=drive_web&ouid=115033867655704919808

Towards the end

Start of Elicitation 2/9/18

1. Is it possible that a report will never be sent as an assignment because it is so low priority that other assignments finish before it? EC
All reports are assigned by the manager of the UCI facilities management department and he or she will determine who to assign to each report. If a report is deemed as not needed to fix, then the manager will not assign anyone and the report will be considered complete even if no work was needed.

2.How efficient do you think the virtual map feature be for navigating campus? EC
Very efficient where the location of a user will be known as long as their phone is on.

3. Will there be a maximum amount a person can submit before the previous one gets complete? EF
There is no maximum amount, assuming that an individual doesn't submit multiple reports of the same situation.

4. If something is not fixed properly, do the reporter sends in another report, or reopen the old one? EF
Once a report is deemed as complete, the worker or manager will classify the report as complete and its status will be unable to be changed.

5. Does management have the ability to ban users that have been deemed to be abusing the system? CD
Those users will be taken care of by the UCI police department, but they will not be banned from using the application in the case that an actual emergency is reported.

6. Can employees submit reports of their own on the same account or must they create a new User-level account? CD
Employees can also submit reports of their own.

7. Is there any specific color scheme that the app needs to follow? AG
It would preferably have some blue and gold, as those are the colors for UCI.

8. Does the application need to be built upon some standard design guidance, like Material Design or Flat UI? AG
Flat UI is ideal.

9. If an issue has multiple reporters and just one of them cancels it, should the report remain an issue since there are still others who reported the same thing? RW
Any report should still remain an issue until they are considered complete.

10. Should the user who cancelled the report stop receiving emails on the status while the other reporters continue receiving updates? RW
If a report is not active, the user who reported it will not be receiving emails or notifications.

11. Since we are enforcing that users login with their UCInetID, should there be a UCInetID password reset tool established within the FixIt app? SK
Yes, they are required to make an account based on their UCInetID and a password is required. If they forgot their password to their account for the app, they will be able to reset it within the app.

12. Should former students, who have a UCInetID but are not currently attending UCI, be allowed to use the FixIt app? SK
As long as their UCInetID is still in the database, they may use the app. However, there will be a guest feature too.

13. How different type of emergency issue will be represented on a map? RL
Emergencies will all be classified with a dot on the map, with users being able to click it and see a description of the event.

14. Will workers receive a push notification on their phone when an emergency issue is assigned to him/her? RL
Yes.

End of Elicitation 2/9/18

Homework 5

Field Notes

2/12/17

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
February 12, 2018 at 5:04pm

- Reply
- _Message actions

Hey can you fill this out for us please?
Thanks for filling out the other questions quickly btw.

- [Missing information.docx]()

2/13/18

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
February 13, 2018 at 11:45am

- Reply
- _Message actions

Answered. Can you answer these for me? Thanks!
https://docs.google.com/document/d/1g08PNTFRE31rrmrlf5szNXwU3j5X1isvHZPOF_-NyhU/edit

- [Missing information.docx]()

Start of Elicitation Questions 2/12/18-2/13/18

Questions to elicit said missing information (generate a question that can elicit said information gap OR clear up something)

1.How should managers correctly declare a report complete, and will it be based on their own judgment or by the set percentage of the report that is already filled out?
It will be based on their own judgment and they will be able to click a button to declare a report complete. They will also be able to offer information about how the repair went and other information they deem necessary.

2.If an issue has multiple reporters and just one of them cancels it, should the report remain an issue since there are still others who reported the same thing?
Any reports that are still incomplete should remain an issue.

3. Should the user who cancelled the report simply stop receiving emails on the status while the other reporters continue receiving updates?
Yes, the active reports will receive notifications for their situations.

4.How are we going to deal with report that have huge typos and errors. Do we call/message the reporter and clarify the mistakes, or do we just throw away those reports?
It should be possible to message the reporter and ask for clarification, but those reports are still important.

5.Should we add email and text validation to the system? This would include confirmation emails and texts.
Yes, they will receive notifications of updates through emails and/or text based on the user's preference.

6.How should repair workers or manager determine the emergency level of different issues on a virtual map?
The description for each event will be available when the issue is chosen on the map, including its emergency level.

7.Should former students, or any general visitor, have at least the ability to report an issue in FixIt as a guest?
Yes, there will be a guest feature where they can submit a report. They will need to submit their name, email, and number.

8.How closely should a user follow protocol for submitting a report?
They should fill in all the fields from the case study such as: picture, title, category, and description.

End of Elicitation Questions 2/12/18-2/13/18

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
February 13, 2018 at 12:19pm

- Reply
- _Message actions

We have answered your questions. Thank you for the response.
I think we will have one more set for you guys on Wednesday.

2/17/18

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)
February 17, 2018 at 3:28pm

- Reply
- _Message actions

We have answered your questions. Thank you for the response.
I think we will have one more set for you guys on Wednesday.

Start of Elicitation Questions 2/17/18

1. What kind of users will we have that will be most involved in our system?
We will have regular users like students. Repairers to handle issues, and managers to assign and sort issues.

2. What is your opinion on having reports be classified as the same issue; should they are still kept on file?
Yes, of course. Multiple reports signifies that the issue must be very important.

3. Can manager do everything a worker can do, or what are some feature that will be specific for workers?
Managers will be able to use the virtual map and assign and prioritize issues. Workers on the other hand will be able to handle their own assignments and add work notes or change status.

4. Will the current report be separate from the close report? How can we tell them apart other than the status of the incident?
Yes, they will be separate depending on distance. We will calculate the average location between those reports, and if they are more than 50 meters apart we will classify them as the safe.

5. Since FixIt relies on in-app notifications, should each user have their own individualized settings saved on the system?
Yes, each user should have different, specialized settings.

6. In the original case study, it was mentioned that there should be functionality for scheduling regular maintenance. Who should be able to schedule regular maintenance and how should the Facilities Management workers be notified of scheduled maintenance (via in-app notifications, shown through the virtual map, or both)?
Specifically the managers will be able to schedule regular maintenance.
Workers can be notified through a notification.

7. Do you think that a virtual map can properly model how managers can convert reports to assignments?
Yes, I think so. It will be easier to view.

8. How closely should a user follow protocol for submitting a report?
They should fill in all the fields from the case study

Homework 6:

2/20/18

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620) February 20, 2018 at 7:07am

- Reply

- Message actions

Can your group send the models of yours ASAP? My group wants to start working on Part 2 of the lab.

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620) February 20, 2018 at 11:40am

- Reply

- Message actions

Here you go!

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620) February 20, 2018 at 6:43pm

- Reply

- Message actions

Sorry, it turns out we need the descriptions too. https://piazza.com/class/jbts2ja2ymn5je?cid=122 I've attached the models with descriptions below. Could you resend yours again? Thanks.

2/21/18

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620) February 21, 2018 at 12:12am

- Reply

- Message actions

Oh yeah here you go, sorry I was busy earlier!

2/23/18

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620) February 23, 2018 at 9:06pm

- Reply

- Message actions

Did you guys add the scenarios from hw 4? I don't seem to see them. Not the use case scenarios.

2/24/18

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620) February 24, 2018 at 10:38pm

• Reply

• Message actions

Added.

2/26/18

Start of Elicitation 2/26/18

1. Why are the UI/UX low priority?

We are assuming that users will be competent at navigating the app. Therefore, UI will be the least of our concerns in favor of functionality.

2. Why does "Ability to revert a status change of the report" and "Original reporters will be notified immediately when a report is marked resolved" have a conflict?

If someone updates a report to complete but changes their mind, then a user may get a notification that it is complete when it actually isn't.

3. In each stakeholder description, how do you determine their level of expertise? Do you think each description matches each stakeholder?

We determine expertise by their position in the UCI Facilities Management. We think each description matches properly.

4. Why are the priority of some goals higher than others? Do you think you should

explain more? We think that the priority of our goals can be based more by looking at our descriptions and the case study.

5. Is there a reason for the lack of systems as actors in the high level goal model for "CheckInAndCheckOnReports"?

We do think that should have been added in hindsight. Thanks for pointing that out. **6. Does the complete system use any outside resources or products to satisfy application needs?**

It uses the WebAuth system to validate log-in. Other than that, it is mostly different. **7. Non-UCI users and Reports do not have a relationship between them in the class**

diagram. Going off of the case study, shouldn’t non-UCI users be able to view the status of other users’ reports? That is true. We did not include that in our diagram, but it is implied in our descriptions that this can be the case. 8. How is the operational plan, a case study requirement, going to represented in the

class diagram? It would be represented by when UCI police or Non-UCI responders handle a report. Plans would be the action that falls under views in our class diagram.

9. How did you come up with the high level goals in HW3, and do you think you did them correctly?

We based our high-level goals on the case study, and although some may feel repetitive, we did do them correctly. 10. Why is it that for some parts of your document they mention the case study, but not

a specific part of it? An oversight on our part. We should have mentioned each specific part of the case study we were sourcing.

End of Elicitation 2/26/18

Homework 7:

3/11/18

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

March 11, 2018 at 4:34pm

- Reply
- Message actions

Can you answer these questions for us? Thanks!

Start of Elicitation 3/11/18

Anthony Hei Tung Cheung, Edward ChenIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

March 11, 2018 at 6:49pm

- **Reply**
- **Message actions**

What is the most important part of the user interface for FixIt?

The interface for submitting a report and the list of reports that UCI management sees.

What kind of interface will a user have to navigate FixIt?

There will be options to enter username and password and press login, after there will be a View button to see all the situations currently on campus and a report button to report situations. There will also be a user button to edit privacy settings.

On what hardware should FixIt be expected to run on?

Mobile applications and laptops/computers.

How would you describe a core functional requirement in FixIt?

Being able to submit reports where there should be no problems submitting them**.**

How would you describe a core non-functional requirement in FixIt?

Security where users should feel safe having their location shared.

What are some potential design constrains in FixIt?

How non-UCI users are able to use the application.

What are some performance requirements you might have in mind?

Making sure the reports are submitted and transferred quickly to the UCI campus police. There should be no clutter.

How will the system be able to retrieve data from the user?

Based on the data the user inputs and their privacy settings.

End of Elicitation 3/11/18

3/12/18

Edward Chen, Anthony Hei Tung CheungIN4MATX 113 LEC A: REQT ANALYSIS & ENG (36620)

March 12, 2018 at 11:12pm

- **Reply**
- **Message actions**

Thanks for answering!

Just a few more questions perhaps.

Start of Elicitation 3/13/18

What are the core users of FixIt?

UCI affiliates, UCI campus police, non-UCI responders, and guests.

How did you design your logical data model?

Wrote about the actors first and thought about what each should be able to do using the app and what information they need.

What elicitation materials did you prepare for your appendix?

All the models and all the questions/answers from your group.

Should managers have separate privileges from the user?

Yes, they are able to assign responders to reports and are able to view all the reports, no matter what the users' privacy settings are.

Should each user have their own customizable set of notification preferences?

Yes, they should be able to turn on notifications for anything they want, whether it be notifications of any event on campus or certain ones.

End of Elicitation 3/13/18

A.3 Traceability

ID: FR1

TITLE: Report Repair Problems

EVENT/USE CASE: Filling Out Report [A.2.3 Use Case Model]
SOURCE: A.2.5 Elicitation Materials and Initial Case Study Document
SUPPORTING MATERIAL: A.2.2.2 Goal Model Diagram 1
HISTORY: Initial Case Study Document

ID: FR2
TITLE: Communicate Tasks to Workers
EVENT/USE CASE: Worker Editing Assignment [A.2.3 Use Case Model]
SOURCE: A.2.5 Elicitation Materials
SUPPORTING MATERIAL: A.2.2.2 Goal Model Diagram 2
HISTORY: Initial Case Study Document

ID: FR3
TITLE: Communicate Task Completion
EVENT/USE CASE: Send Notification of Change in Repair Status [A.2.3 Use Case Model]
SOURCE: A.2.5 Elicitation Materials
SUPPORTING MATERIAL: A.2.2.2 Goal Model Diagram 2
HISTORY: Raised by team BBB, Elicitation document 2/17/18 [A.2.5]

ID: FR4
TITLE: Prioritize Reports
EVENT/USE CASE: Prioritize Reports [A.2.3 Use Case Model]
SOURCE: A.2.5 Elicitation Materials
SUPPORTING MATERIAL: A.2.2.2 Goal Model Diagram 1
HISTORY: Raised by team BBB, Elicitation document 2/2/18 [A.2.5]

ID: FR5
TITLE: Minimize Duplicate Reports
EVENT/USE CASE: Merge Duplicate Reports [A.2.3 Use Case Model]
SOURCE: A.2.5 Elicitation Materials
SUPPORTING MATERIAL: A.2.2.2 Goal Model Diagram 3
HISTORY: Raised by team BBB, Elicitation document 2/2/18 [A.2.5], Elicitation document 2/13/18 [A.2.5] and Initial Case Study Document

Missing Information
Team Lucky

Gaps in information

Unsure of if our application will be accessible to handicapped users.

We are unsure that what the system should do if the assigned repair worker is unreachable.

We are not sure on how are we going to manage changes/edits made by the original user.

We are unsure what we are going to do with the user that sends in fake report

One thing we missed was if locations should be used as criteria for determining the order of the list of repairs a worker is assigned. A worker's list of repairs can be done in an order that minimizes the total travel time between points. This algorithm would increase efficiency (while it alone would not account for the fact that some repairs are more urgent).

Assumptions

We assume that the application should meet UCI's standard guidelines for accessibility.

We assume that the FixIt system were able to handle large amount of reported issue without any break down, and we also assume that each repair worker will be reachable through their mobile device.

We keep the assignment as it was before the edit.

After a set amount of fraud reports, the user will be fined or banned from using the application

As of now, we assume the app is not required to try to order repairs such that travel time is minimized. Incorporating that extra feature would require extra work which we prefer to do only if the client wants it.

Questions to elicit said missing information

Are there certain guidelines we should be following to make the application accessible to handicapped users?

What the Fixit system should do if the application were unable to reach a repair worker regarding an emergency issue?

What actions are to be taken if a user makes major changes to a report?

How do you ensure the report is legit and can be trusted?

Should the app try to minimize the total travel distance a worker covers when he/she must address a list of repairs on different parts of the campus?