6-9-2016

FixTheCity: Local Government Complaint Management System

Amanpreet Dhoor
Santa Clara University

Kevin Ta
Santa Clara University

Melissa Portillo
Santa Clara University

SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

Date: June 10, 2016

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Amanpreet Dhoor
Kevin Ta
Melissa Portillo

ENTITLED

FixTheCity: Local Government Complaint Management System

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREES OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING
BACHELOR OF SCIENCE IN WEB DESIGN AND ENGINEERING

Thesis Advisor

Department Chair

FixTheCity: Local Government Complaint Management System

by

Amanpreet Dhoor
Kevin Ta
Melissa Portillo

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 9, 2016

FixTheCity: Local Government Complaint Management System

Amanpreet Dhoor
Kevin Ta
Melissa Portillo

Department of Computer Engineering
Santa Clara University
June 9, 2016

ABSTRACT

For a large city, such as San Jose, it can be difficult to keep track of all of the city property that needs fixing. One way to remedy this would be to increase citizen involvement by creating an easy way for citizens to report problems. The current method for doing this is either to call the problem in or to navigate through department webpages to fill out a form. The city also has an app called San Jose Clean, through which people can report graffiti by taking a photo with their phone, but graffiti is the only problem it deals with and the app is managed by the graffiti removal company so the city has limited access. In response, we created an app citizens can use to report a variety of issues with city property and which the city itself can directly control. By creating a database for all of the information submitted through the app, our solution can help the city sort through the reports and prioritize based on that data, allowing them to address these issues as efficiently as possible. Information on these issues can be available to citizens so that they can volunteer to help fix problems as well. By having people participate in the maintaining of their city and giving the city a way to collect information, the app has the potential to increase collaboration between the city and its people.

Table of Contents

List of Figures

0.1 Introduction

0.1.1 The Problem with Problems

It is a common complaint that the government does not allocate resources properly. Particularly in a city as large as San Jose, it can be difficult to coordinate the needs of the people and allocate resources accordingly. There is always something that needs repairing, and these issues can inconvenience many people who use these resources and systems daily.

Many of these problems are relatively minor and include potholes, splintered park benches, and broken street lamps. Something like a traffic light would be a more immediate concern and would likely be reported to city hall sooner than more minor issues. This means that these more minor issues are often ignored for a long time due to the inconvenience of reporting a problem that does not completely disrupt daily activities. In the long term, these issues can be problematic in their overall disruption of a clean and smooth-running city.

0.1.2 Current Solutions

Currently, the process to report an issue to City Hall is fairly involved.

San Jose provides online forms where a citizen may report an issue, but trying to find these forms can be a nuisance. Navigating through webpages can be painstaking for some users, and then the appropriate form might be mixed in with other forms and links. In addition, forms often prompt users to give their contact information, which might deter some people.

Another option would be to go online and search for the right phone number to call. The information overload on most government pages can make it difficult to locate the appropriate number.

One option that the city has to ease the online hassle is the mobile app San Jose Clean. Citizens can use the app to take a photo of graffiti and submit it so that the city's graffiti removal company can clean it up. While graffiti can be reported in this manner, the city does not have access to this information because the app is handled by the graffiti removal company.

0.1.3 Making It Easier To Complain

We planned to remedy the situation by providing another option for the city and the people to communicate with each other to fix these problems. To do this, we proposed an application that would allow citizens to use their mobile devices to submit a report and that would provide a database for the city to manage these reports.

Submitting a report consisting of a photo and a GPS coordinate would be quick and anonymous for citizens, and having a database to display these reports would allow the city to prioritize accordingly.

0.2 Literature Review

0.2.1 System Goals

Our team hoped to create an application that will increase civic engagement within the city of San Jose. Giving citizens an easy way to report issues throughout the city, can foster an environment in which people care about the state and well-being of their city. We also hope that being invested in their city will mean that people might want to come together to help fix the various parts of their city as well. In addition to reporting when city property requires fixing, perhaps the application can be used for people wanting to volunteer to fix these problems especially since the issues being reported on the application would be fairly minor fixes. Finally, we hope our application is inviting, easy to use, and customized around the particular needs of the people of San Jose.

0.2.2 Creating Unique Project

Researching articles relevant to our senior design topic has been really useful in terms of discovering what kind of projects regarding civic engagement are already out there. The first source on our list caught my attention because the title sounded exactly like our project. That can be very helpful, but at the same time it was somewhat disconcerting because it is almost too similar. Part of senior design is making a product that is unique. Our advisor suggested this project so we know that it is acceptable for senior design, but our own personal goal is to expand on the technologies that are already out there. That source was from a different part of the country, which is ideal because then we can focus on customizing our application based on more local needs. San Jose currently does not have something exactly like this in place. It does have an application for cleaning up graffiti, but since the functions of that app are very specific, that gave us room to create something more dynamic. Even though our research has shown us that the technology we used is not necessarily new, we made it unique by focusing on the needs San Jose.

0.2.3 Community Aspect

One way we wanted to make our application unique is by including some sort of community aspect. There may be mobile applications out there that give citizens the ability to report problems with pictures on their smart phone, but that is the extent of their involvement. It would be great if there was a way for their involvement to continue even after they have reported a problem. We wanted to consider this especially because when we met with our client at San Jose City Hall, he was particularly interested in bringing local communities together. As mentioned, one idea that we had was to include a way for citizens to volunteer to fix

problems. When we were doing our research, that was an aspect that we focused on. We found many articles about the relationship between mobile or web technologies and civic engagement. Those were fairly useful when we considered the user base and what might be the best way to engage citizens with their government.

0.2.4 Admin Interface

The web application in this system is targeted towards providing an interface for an admin. This will allow filtering through the data in an attempt to search, process, or analyze the results in the database. The target audience in this case would be someone a bit more technically savy than the user of the mobile application. However, navigating must remain straightforward so that there is no need for the user to pause in order to consult resources.

0.2.5 Mobile Application

A mobile application is impractical if it is not being used. In this case in particular, there is a dire need for users because, without any inputs from the mobile application, the resulting database would be empty, thus rendering the whole system incomplete. Drawing in users can be an issue when considering our system, which is meant for interaction between the government and the public. Privacy concerns, distrust of the government and general lack of civil engagement (Americans' Mixed Messages) were a hurdle that needed to be overcome. In order to be able to create an application that would realistically be used for government, the mobile application specifically targeted the audience in mind. In order to get a thorough list of results, the target audience would be anyone with a cell phone—ranging from teenagers to senior citizens. The application needed to be simple to use for those who may have trouble with technology while providing good functionality and a clean interface. HTML 5, Javascript and certain Javascript frameworks are flexible for creating an application that suits the needs of the target audience (Moore).

0.2.6 Catered for San Jose

While the work is based on existing technologies, what it is doing is benefiting Santa Clara County by giving citizens a way to easily report city maintenance issues. Similar mobile applications exist in some form, but our project gives more options and convenience than what City Hall is currently using. Our project is about creating something to benefit the city.

0.3 Requirements

This is the list of requirements that the product must meet to satisfy the users. The list consists of functional requirements that define what the application must do and nonfunctional requirements that define how the functional requirements should be met. Each requirement is ranked under one of three tiers of importance: critical, recommended, or suggested. Any design constraints, which limit the way the design can be implemented, are listed as well.

0.3.1 Functional

Critical:

1. Submit photo with GPS and inputted information
2. Option to submit a photo from phone gallery
3. Database retrieves photo, GPS, and any other inputted information
4. Filter/sort options for database
5. Search through database
6. Eliminate repeats
7. Delete items that have been fixed

Recommended:

1. Show how many people have reported the same problem
2. Filter spam
3. Export data from database

Suggested:

1. Provide numbers to call
2. Notify the person who submitted the problem when it is fixed
3. Provide a log in for administrative use
4. Create a way for citizens to volunteer to fix problems

0.3.2 Non-functional

Critical:

1. Able to run on different web browsers
2. User-friendly
3. Aesthetically pleasing
4. Adaptable

Recommended:

1. Configurable

Suggested:

1. Having content available in different languagesHaving content available in different languages

0.3.3 Design Constraints

1. Available for both Android and iOS
2. Usable on laptops, tablets, and mobile devices

0.4 Use Cases

The mobile application is designed to be used by one main group of users, the general public. The public can populate the database with issues in their community. We planned for the web application to have both an admin user and a general user. Though both users would be able to query the database, only the admin user would be able to make changes to it, as shown in figure 1.

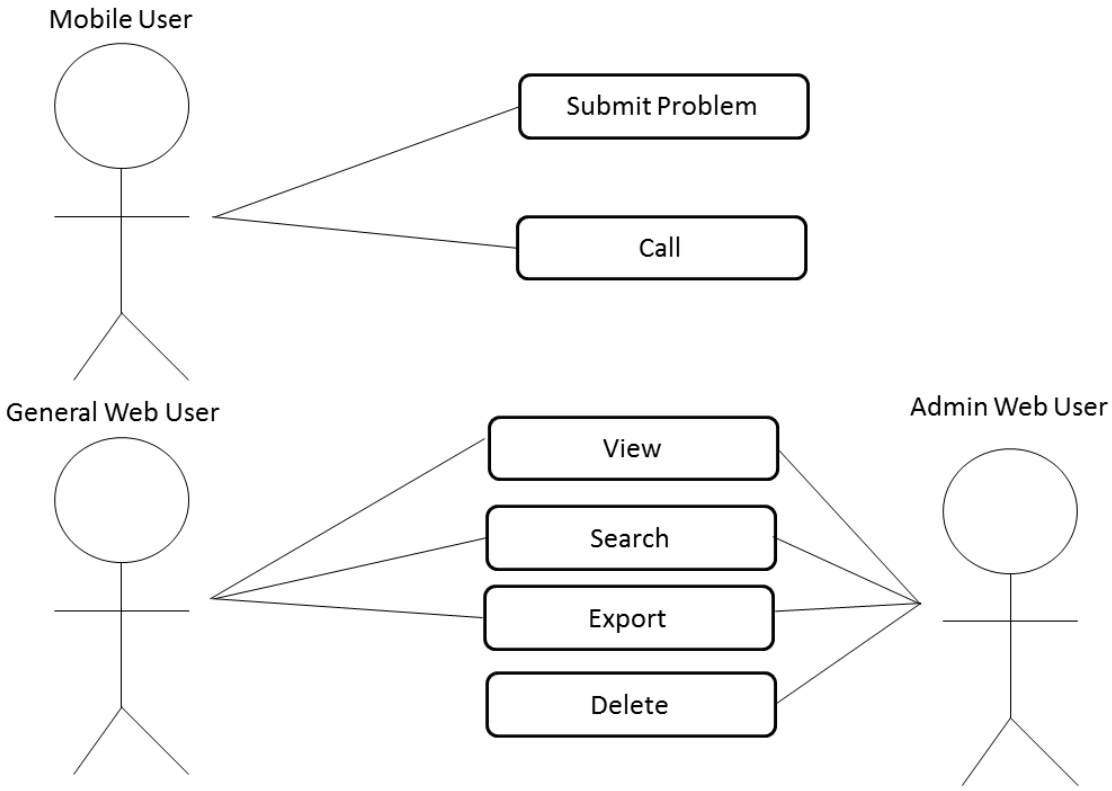


Figure 1: Diagram showing the use cases.

0.4.1 Mobile Application

Use Case 1: Submitting Problem

Actor:

- General mobile user

Goal:

- Add problem to database

Conditions:

- GPS enabled
- Data connection enabled

Steps:

- Take picture
- Allow GPS
- Select categories
- Add optional information
- Submit

Use Case 2: Calling in Problem

Actor:

- General mobile user

Goal:

- Report problem through phone call

Conditions:

- Mobile network connection

Steps:

- Select category
- Confirm number
- Dial

0.4.2 Web Application

Use Case 3: View Repository

Actor:

- General web user

Goal:

- View database

Steps:

- Select view
- Select number of entries to view
- Sort by columns

Exceptions:

- Database is empty

Use Case 4: Search Repository

Actor:

- General web user

Goal:

- Query database

Steps:

- Enter search string
- Select applicable filters
- Submit

Exceptions:

- No results found

Use Case 5: Export Repository

Actor:

- General web user

Goal:

- Export database to CSV or PDF

Conditions:

- Searched database or view all

Steps:

- Click export
- Select PDF or CSV
- Save As

Use Case 6: Delete from Repository

Actor:

- Administrator

Goal:

- Delete problem from database

Conditions:

- Logged in

Steps:

- Enter unique key
- Confirm deletion
- Submit

0.5 Activity Diagram

Activity diagrams illustrate the work flow of the applications. Figure 2 follows the web application, and Figure 3 follows the mobile application.

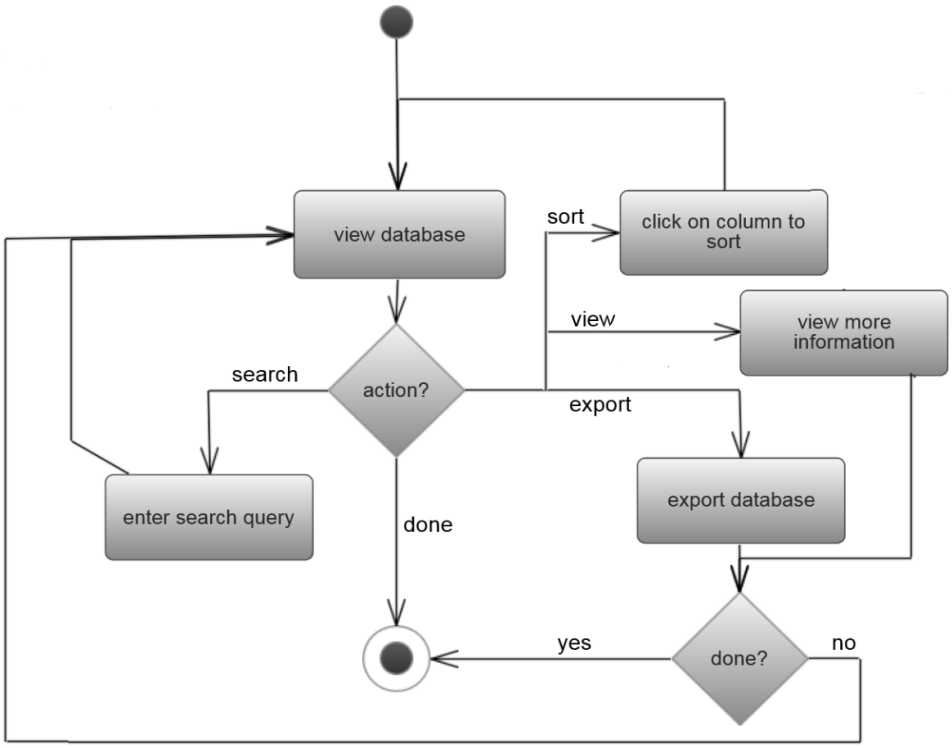


Figure 2: This diagram shows the workflow of the web application.

As seen in figure 2, when users view the database, they have a variety of paths they can take. They have the options of sorting by columns, searching through the reports, clicking on an entry to see more details, or exporting the data.

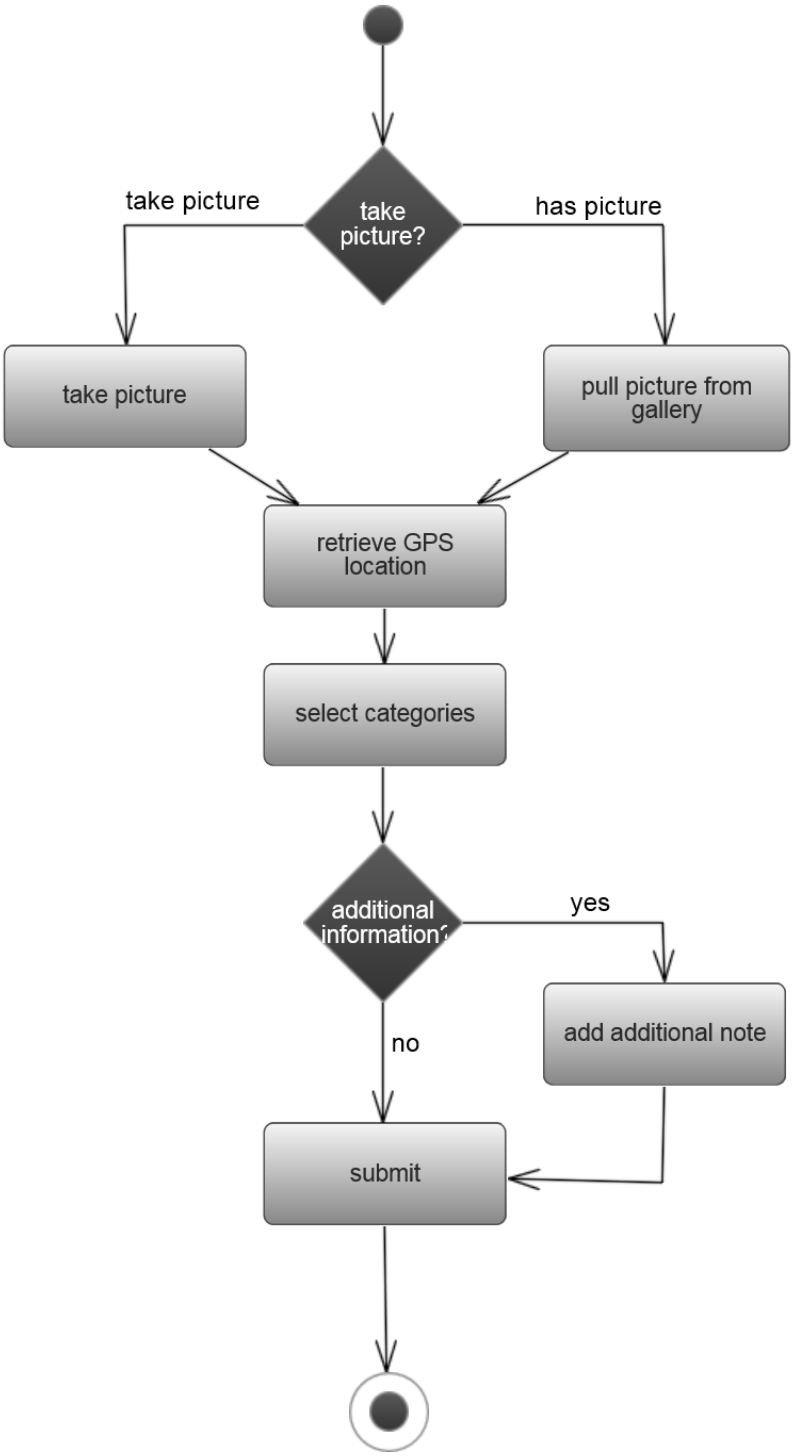


Figure 3: This diagram steps through submitting a problem via the mobile app.

As seen in figure 3, upon opening the app, the user must first take a photo. Then they must confirm their GPS location, identify the category of the problem, and decide whether to leave any additional comments before submitting the report.

0.6 Conceptual Model

In order to start visualizing our application, we created conceptual models to outline what would be featured in our application and what a basic layout would look like.

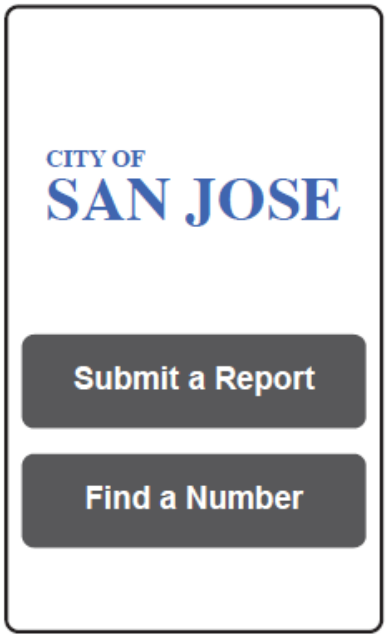


Figure 4: On the home screen the user can choose to submit a report or find a number for the problem.

Figure 5: After choosing to submit a report, there is the option to take a photo or pull from gallery.

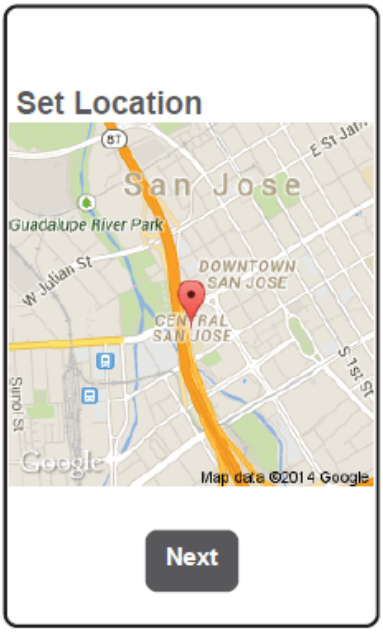


Figure 6: After a photo is taken, users will be asked to allow their GPS location to be confirmed.

Figure 7: The user selects a category for the problem.

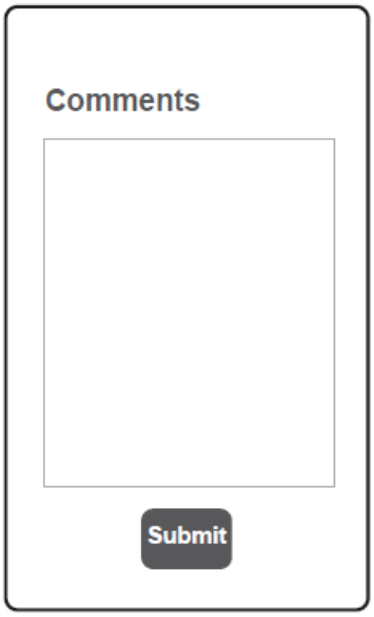


Figure 8: In the comments section the user has the option of including additional information.

Date Added	Issue	Location	Times Reported
11/09/2015 20:00	Pothole	1234 Main Street	10
11/09/2015 16:00	Graffiti	5678 Fifth Avenue	3
11/09/2015 12:00	Illegal Dumping	9900 North Drive	6
11/09/2015 8:00	Traffic Signal	6193 Tenth Street	21
11/08/2015 20:00	Sidewalk	3421 Story Road	1
11/08/2015 16:00	Graffiti	4579 Curtner Avenue	5
11/08/2015 12:00	Graffiti	500 El Camino Real	1
11/08/2015 8:00	Streetlight	5044 Santa Clara Street	4
11/07/2015 20:00	Pothole	9876 Chabrant Way	2
11/07/2015 16:00	Sidewalk	3874 South Avenue	1
11/07/2015 12:00	Park Concerns	4545 First Street	1
11/07/2015 8:00	Pothole	7777 Cherry Road	8

Figure 9: Problems are listed in the database and can be sorted depending on the user's choices.

Figure 10: Clicking on a problem will show a page with more details on reports for that problem.

0.7 Technologies Used

The technologies used are mainly web-based.

1. HTML - HyperText Markup Language, used to create the vast majority of webpages.
2. CSS – Cascading Style Sheets is used to apply visual styles to a web page.
3. Javascript - A dynamic scripting language that runs client side. Most of the time, it is used together with HTML and CSS.
4. Meteor – A platform for building webpages and mobile applications with JavaScript.

0.8 Architectural Design

The architectural design shows the relations between the technologies used, how the users interact with it, and how that leads to data being stored.

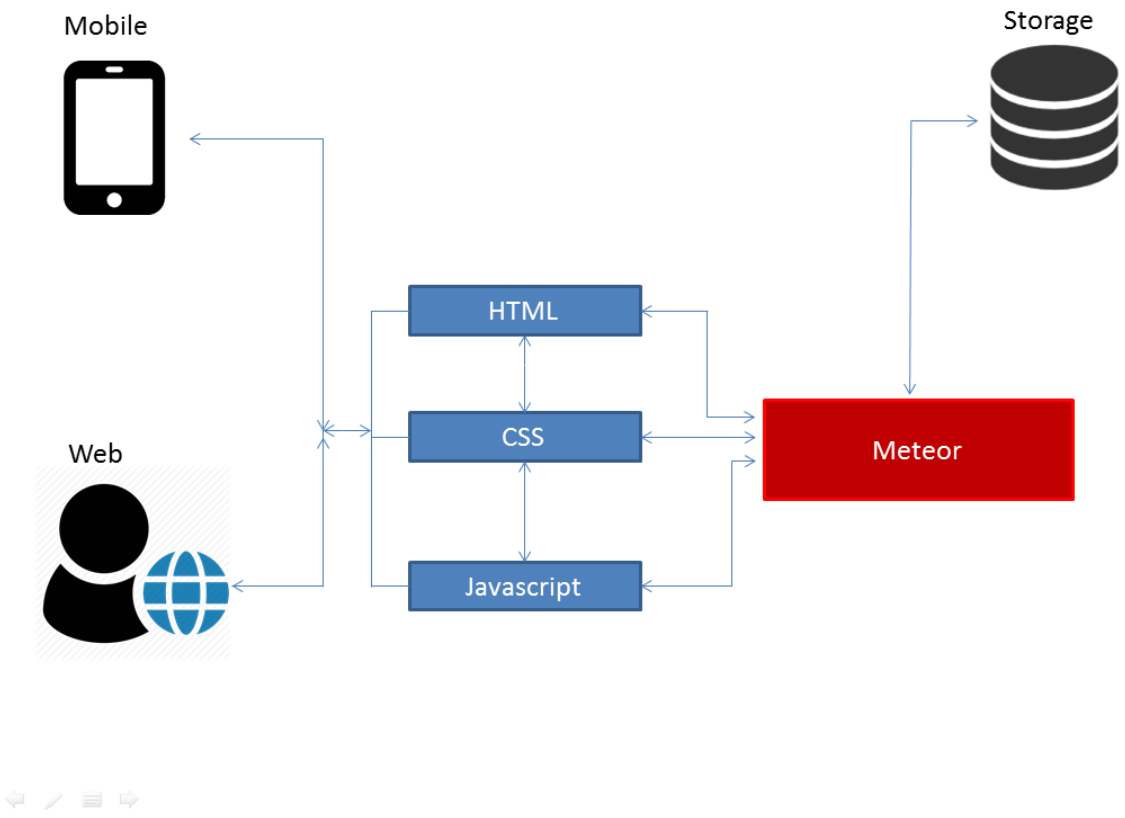


Figure 11: Diagram showing the relationship between technologies.

1. As shown in figure 11, the user at the bottom left interacts with the front-end via the mobile application or website. Both make use of HTML and CSS to design a viewable page, both structurally and aesthetically.

2. Javascript is used to write the functions in the site and application.

3. Meteor allows the user to write HTML, CSS, and Javascript within it and port it into a mobile application. It also has its own database functions, for storing and accessing the data from both the web and mobile side of the application.

0.9 Design Rationale

Our design rationale is based on ease of use and clear communication for the user. The planned design walks the user through the process of reporting public maintenance issues to the client. There are many categories for whether the issue is a traffic, pothole, graffiti, etc., problem, to make the data easier to sort through for the client. Drop down menus and autocomplete options were added for ease of use. On the mobile end, the option to retake a picture from the gallery is offered so that the user can insert the best picture of the problem possible.

Meteor, a Javascript framework, was chosen because it allowed us to port HTML, CSS, and JavaScript into a mobile application. Because we built it on both web and a mobile applications, the complete system is in one place, and because every Meteor project has a corresponding database, we used their MongoDB database rather than having to include one separately.

0.10 Testing Plan

Both black box and white box testing are used to validate and verify the system and its results.

0.10.1 White Box

White box testing is done by the developers with knowledge of the code base. In terms of the inner workings, we test for the following:

1. The image from the camera being sent and received by the client.

2. The image being properly sorted into categories displayed.

Test codes, scripts, and databases were used to simulate the client side. Afterwards, the real client side database was tested and adjustments were made accordingly.

0.10.2 Black Box

The purpose of blackbox testing is to check the end-to-end functionality of the system without knowledge of its internal workings. From a user standpoint, we test for:

1. The application starting and running.
2. The user is able to see categories to select from to report an issue.
3. The application opens the camera and the user is shown a "send" option.

This was done by using the app in a way that an average user would.

0.11 Problems Encountered

In order to manage the project, we anticipated the risks and how we planned to handle them. This helped us manage the issues as we ran into them so that we could complete the project in a timely manner.

Table 1: Risk Table

Risk	Consequences	Mitigation Strategy
Other Obligations	Inability to meet deadlines	Started schoolwork early or finished project requirements early. Notified team of absence one week in advance.
Framework Update	Changes to app structure	Saved all versions of the application. Ensured that compatibility is maintained.
Bugs	Inability to meet deadlines	Tested application while in development stages. Documented bugs as they appear.
Lack of Technical Ability	Delay in meeting deadlines	Divided work according to team's strengths. Consulted resources when there was a problem.
Deployment Changes	Inability to continue hosting application for free	Deployed on local machine. Uploaded most recent versions to Github.

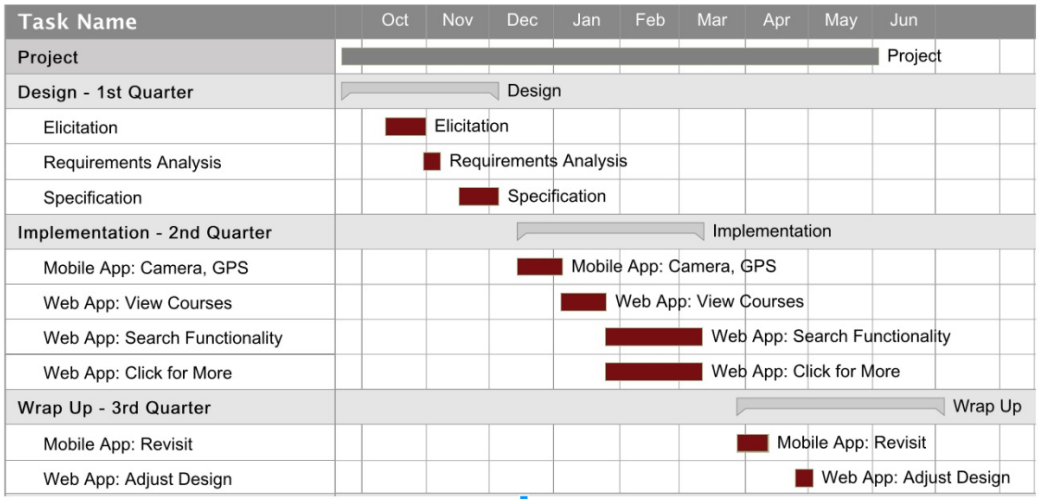
Table 1 outlines the risks our team encountered during our project. Each mitigation strategy contains two plans of action for each risk. The main problem we encountered was that Meteor is still a developing framework, which meant that there were sometimes bugs. Also, Meteor went through a major update during the final stages of our project. This involved changes to the framework itself as well as Meteor's deployment

services. In the past, apps could be deployed to their servers, but that option was replaced with a paid hosting service. Dealing with these unexpected changes meant communicating and planning accordingly.ng and planning accordingly.

0.12 Development Timeline

The Gantt chart in Table 2 details the deadlines and tasks for our project.

Table 2: Gantt chart detailing development timeline



0.13 Societal Issues

Throughout the design of our application, something that we wanted to consider were the ethical implications of our project and other effects that it could potentially have. The following are the types of issues we wanted to take into account:

0.13.1 Ethical

Ethical considerations influenced both the way we worked together and the project itself. We tried to split the work evenly amongst ourselves and to treat each other with respect throughout this process. If a teammate was having trouble with part of the project, another teammate would help out. The project itself also meets the ethical requirements we had. As a web application, it does not do any significant harm. The main potential harm it could cause would have to do with privacy, but that would not be too much of a concern considering users do not have to identify themselves to submit reports. If this application can assist a community in taking caring of issues so that it is a more pleasant pleasant place to live, then it would be for the common good.

0.13.2 Social

In terms of social context, our goal is to foster a sense of community. For example, someone who sees a broken streetlamp may not directly benefit from reporting it, but reporting it may help a fellow citizen who could be negatively impacted by that broken streetlamp in the future. Having this sort of communication available to people allows them to do their part for their community.

0.13.3 Political

Because this project involves providing a means of communication with city government, there are plenty of political implications. We hope that this application gives people the opportunity to be more interactive with their city government and to realize that their input is valuable to the government.

0.13.4 Economic

Economic considerations came into play in the implementation of this project. One of the reasons why we chose Meteor is because it is a free framework, and at the time of our design phase, it was free to deploy with Meteor as well. Later on Meteor switched from its free hosting service to a paid option with Galaxy that starts at $0.035/hour.

0.13.5 Health and Safety

This application would not affect health, but safety could potentially be an issue. Since the app allows users to report what may be sensitive information, the person submitting the report can remain anonymous. Users can submit contact information if they wish, so privacy could be an issue if this was abused somehow.

0.13.6 Manufacturability

Since FixTheCity is a web-based application, manufacturability should not be an issue.

0.13.7 Sustainability and Environmental Impact

The application should not directly affect sustainability or the environment. However, if citizens use the mobile app to report certain issues, such as sewer spills, then perhaps the app can help with the cleanliness and maintaining of the environment.

0.13.8 Usability

Usability is an important part of this project because the city government would be depending on people to submit reports through the mobile app, so in order for it to be effective, as many citizens as possible need to use it. Some of our decisions, such as making it cross-platform and giving it a minimalist design, were made to make the system more usable. However, there is still more to be done to increase usability. For example, having the application available in multiple languages would give the city government the wider audience that it wants.

0.13.9 Lifelong Learning

This project has promoted the importance of lifelong learning for us because it involved learning outside the classroom. Even though we had done some web programming prior to this project, we had never used the Meteor framework before. This project provided us with a hands-on learning experience that involved experimenting with something we were unfamiliar with. Because Meteor is still a developing framework, there were some unexpected updates to the framework while we were building the application. Such unexpected changes is something that will continue to occur throughout our careers, so this was a valuable lesson in learning to adapt.

0.13.10 Compassion

Compassion is a component in this project because the goal in creating this application was to help with solving some of the issues faced by a large community. The issues to be reported on the app may be minor individually, but when there are numerous problems left unaddressed, this can negatively affect citizens. Hopefully this app could improve a community's welfare by providing a way to help maintain their environment.

0.14 Test Cases

In order to understand how our system was developing, we tested frequently. The main test cases used are outlined in Table 3 and Table 4. Due to the nature of these applications, these test cases outline the expected functionality of the system. The user interface of the system was tested with usability testing.

0.14.1 Mobile Application Testing

Table 3: Table describes the tests, along with their inputs and expected results

Test	Objective	Input	Expected Result
Camera	Access phone's camera to take picture	Click 'take photo' upon opening application	Native camera appears, Access front and rear cameras
GPS	Access location services	Location ON when going to GPS confirmation	If location is on, display correct GPS on map for confirmation
GPS Error	Access location services -error handling	Location OFF when going to GPS confirmation	If location is off, display no GPS marker, direct user to turn on location settings
Insertion	Insert information to database	Correctly enter all fields and click submit	If fields entered correctly, all information is correctly stored in database
Insertion Error	Insert information to database -error handling	Leave out certain required fields Submit on two phones at the same time	If fields entered incorrectly, the user is given a flag to show what to add Incomplete forms are not sent All complete reports are stored in database

As seen in Table 3, the mobile application should be able to correctly submit a report to the database. The test cases for this application focused on making sure that the information is correctly accessed on the device and that any possible errors are handled.

0.14.2 Web Application Testing

Table 4: Table describes the tests, along with their inputs and expected results

Test	Objective	Input	Expected Result
Tables	All information from database is represented	New information into database Repeats Bad data	All legitimate information shows, Repeats are collapsed Incomplete data is not represented
Filter	Search database	Valid and invalid search terms	Corresponding information shows, If none, present message
Export	Export information	Click export, select file type	Create CSV file or PDF
Expand View	View more information on a report	Click on a report	Any repeats or extra information is presented on a new page

Table 4 describes the main tests for the web application. The test cases for this application were geared to access and understand the information that is in the database in a variety of ways.

0.14.3 User Interface

The success of the mobile application heavily relies on users, which is why user interface is tested through cognitive walkthroughs. Users are given two tasks. The first task to submit a report and presented with a phone. The second task is to find an report in the web application and delete all instances of it. With this, the user provide feedback on a variety of fields, most particularly the following:

1. Understanding the goal of the application and its subtasks
2. Assessing visibility of possible actions
3. Connecting the goal to the possible actions that are perceived
4. Interpreting feedback of the actions
5. Providing any extra information regarding UI and ease of use

0.15 Test Results

Testing the functionality of the system was done initially through unit testing. Each of the individual test cases were performed and then the bugs were fixed in order to yield the expected result. Upon integrating the components full system walkthroughs were performed along with usability testing.

0.15.1 Mobile Application

Table 5: Table describes the tests, along with their inputs and expected results.

Test	Expected Result	Actual Result	Edits made
Camera	Native camera appears Access front and rear cameras	Native camera appears Access front and rear cameras	No edits or optimization needed
GPS	If location is on, display correct GPS on map for confirmation	Didn't consistently retrieve GPS, Often took three attempts to retrieve coordinates	The function to find GPS was put in a loop that consistently ran, This would also take into account the user moving locations after having snapped the picture
GPS Error	If location is off, display no GPS marker, direct user to turn on location permissions	Unfinished	To be continued in future version
Insertion	If fields entered correctly, all information is correctly stored in database	If fields entered correctly, all information is correctly stored in database	No edits or optimization needed
Insertion Error	If fields entered incorrectly, the user is given a flag to show what to add Incomplete forms are not sent	If fields entered incorrectly, the user is given a flag to show what to add Incomplete forms are not sent	No edits or optimization needed

As seen in table 5, the mobile application passed most of the tests as initially expected. The GPS retrieval had the most bugs, which centered on the phone not being able to immediately retrieve a location. Initially, it tried to retrieve the GPS location only once, however it was then edited so that it would continuously attempt to retrieve a GPS location. This change allowed the application to able to consistently retrieve more accurate results.

0.15.2 Web Application

Test	Expected Result	Actual Result	Edits made
Tables	All legitimate information shows, Repeats are collapsed Incomplete data is not represented	Not all information was represented Repeats did not properly collapse Incomplete data is not represented	Inconsistencies were found between t objects being stored in the database a the objects being pulled from it. Thes were changed to match one another. Additional arrays used to filter data were added
Filter	Corresponding information shows, If none, present message	Corresponding information shows, If no results, no matching results message is presented	There became no need to show such a message due to the interactive way in which the search function filters through the data
Export	Create CSV file or PDF file	Unfinished	To be continued in future version
Expand View	Any repeats or extra information is presented on a new page	Any repeats or extra information is presented on a new page	No edits or optimization needed

Table 6: Table describes the test results, along with their solutions and analysis

Table 0.15.2 details the test results for the web side of the application. The application passed most of the tests. Due to time constraints and a need for prioritization, the export functionality, having been a recommended requirement, was not added to the web application.

0.15.3 User Interface

Cognitive evaluations were run to evaluate the usability of the two main tasks of the system. The main task of the mobile application is for a user to successfully submit a report about an issue that they have come across. The main task of the web application is to be able to search through the database and delete reports about issues that have been resolved. Each cognitive evaluation is assessed based on the following:

1. Understanding the goal of the application and its subtasks
2. Assessing visibility of possible actions
3. Connecting the goal to the possible actions that are perceived
4. Interpreting feedback of the actions
5. Providing any extra information regarding UI and ease of use

The main feedback and changes made are highlighted in the following sections.

Task One: Submitting a Report

1. Assessing visibility of possible actions: Due to the minimalistic style of the application, the few options are presented clearly.
2. Connecting the goal to the possible actions that are perceived: The report was initially one page, however it was then broken up into small subsections in order to give each step of the submission process a clear beginning and end.
3. Interpreting feedback of the actions: The required fields were not labeled as so. Changes were then made in order to make sure that requirements were met. There is no option to move onto the next step until the current one is completed. The exception to this is when there is an optional field.
4. Providing any extra information regarding UI and ease of use: Certain instruction details were not worded clearly and were then fixed. Labels were added to the form fields and examples were given in placeholders.

Task Two: Deleting an Entry

1. Assessing visibility of possible actions: The user is able to either search through each individual report or use the search bar at the upper right. The upper right is a common place for a user to go to for actions. The deletion icon is a trash can next to the report on the table.

2. Connecting the goal to the possible actions that are perceived: Search and delete actions are clearly labeled. The ability to view the in depth reports required more labeling because it was not clear to the user that you had to click on an initial report to go to a page where deletion is possible.

3. Interpreting feedback of the actions: The search function filters as one types so the user immediately sees feedback from their actions. Deletion is also immediate.

4. Providing any extra information regarding UI and ease of use: There is no confirmation message before deletion. This has not been added as of yet.

0.16 Suggested Changes

There are many ways this project could be expanded. During the summer, graduate students will be continuing this project, so they will be adding on these new features. The web database in particular could benefit from additional functions.

Multiple Levels of Access

At the moment, the application has only one level of access. In this case, anyone using the system is assumed to be an admin. Something that a city would probably want is multiple levels of access. Instead of having just one standard login for city workers, there would be administrators that could allow people access, but those people would not have the same level of access as an administrator. Additionally, this allows for scalability so that other functions can be given to users. At the moment, there is one administrator who can delete anything, however some tasks should be able to be delegated. In this way, the city can allow different levels of users to have the ability to access or manipulate certain data.

Location

Another improvement would be to refine the way location appears in the database. At the moment, when the location shows up on the web application, it appears as longitude and latitude coordinates. For readability purposes, it would be ideal if the location showed up as an address. It could also be helpful if the location linked to a map element showing where the coordinate is.

0.17 Lessons Learned

- Coordination is crucial to any group project
  - Considerations of some factors were done to aid in planning
    - Long term projects were not an area of expertise for group members
    - Members' schedules varied across the timeline
  - Planning around these considerations meant:
    - Establishing weekly meeting times
    - Communicating with advisors
    - Assigning member tasks at each of these meetings
- Knowledge needed to carry out plan
  - Time management needed to be done on both the individual and group level
    - Individual time management meant that each member needed to find time in a schedule where they were already busy in order to meet the deadlines set by the group.
    - Group level time management involved seeing how well the completed actions match up with the deadlines the group had set and replanning accordingly
  - Assuring technical compatibility across platforms
    - Each group member would have access to different technological platforms and resources while working separately; ensuring that the work each group member does can be carried across platforms helped prevent future problems.
    - One of the goals of the project is to be multiple-platform compatible, so the above also holds true for the design on the user side

0.18 Appendix

0.18.1 Install Scripts

The system is being run and tested on local machines. Both the mobile and the web sides can be run in a browser and, additionally, the mobile application can be ported over to run on a phone.

Due to a change in the Meteor platform's regulations midway through the project, the free web hosting was taken down. It has now been replaced by a paid program called Galaxy that can be used for deployment.

Install Meteor

Meteor is installed through the detailed instructions on the Meteor Install page. Meteor can be installed on OS X, Linux and Windows machines. https://www.meteor.com/install Installing on a Linux or OS X machine can be done through the terminal using: "curl https://install.meteor.com/ — sh"

On a Windows machine, one can download an installer from the link provided.

Running System on Browser

Each application is in a different folder and labeled accordingly. The web application must always be run first due to the fact that it stores that data. Then the mobile application while pointing to the database that the web application is running.

Once in the project's directory, running the web application is done through the following commands:

- move to the web directory using "cd web"
- run meteor using "meteor"

This causes the web application to run on localhost:3000.

Upon completing this, in another terminal, the mobile application is run from the project's home directory through the following commands:

- move to the web directory using "cd mobile"
- run meteor using "meteor"
- point this application's database to the one running on the web side using "export MONGO_URL=mongodb://127.0.0.1:30
- run meteor on another port using "meteor –port 3002"

Running on Mobile Device

Running a mobile device can, at the moment, only be done using Linux and OS X machines. The mobile application can easily be ported over to Android and iOS. The link on Meteor's website details the process of downloading the packages: The website https://www.meteor.com/tutorials/blaze/running-on-mobile

First the install scripts must be downloaded using the commands: "meteor install-sdk ios" "meteor install-sdk android". This runs the user through the setup for the ios and adroid SDKs.

Once the setup is done, running "meteor add-platform ios" and "meteor add-platform android" in the directory "./mobile" will add the functionality to the application.

To run on a mobile device, the mobile device must be connected to the computer through a USB cable. From here, one should go to the terminal, under the mobile directory, and type "meteor run android-device". The command to run an iOS device is "meteor run ios-device" however it requires an apple developer account.

0.18.2 User Manual

Mobile:

Submitting an issue:

1. Touch button labeled ‘Submit A Report’.
2. Touch button labeled ‘Take Photo; if photo displayed is not satisfactory, touch ‘Take New Photo’. Otherwise, touch ‘Use Photo’ button.
3. Touch ‘Next Step’ button.
4. Select category by touching small circle icon next to the category that most matches the issue.
5. Touch ‘Next Step’ Button.
6. Share location if prompted to.
7. Touch ‘Next Step’ button.
8. If you wish to provide extra details, enter them in the text boxes; if not, touch ‘Next Step’ button.
9. Touch ‘Submit’ button.
10. Submission Complete.

Web:

Searching issues:

1. Type key term that you wish search into textbox labeled ’filter’.
2. Entries that match should automatically display.

Viewing issues:

1. Click a row of the category that you wish to view.
2. All the entries of that category will be displayed.
3. Click ’Back’ button to return to previous page.

Deleting issues(Administrator only):

1. Click a row of the category that you wish to view.
2. All the issues of that category will be displayed.

3. Click trash can icon next to entry that you wish to remove.

4. Entry is removed.

5. Click ’Back’ button to return to previous page.

0.19 Annotated Bibliography

Foth, Marcus, Ronald Schroeter, and Irina Anastasiu. "Fixing the city one photo at a time: mobile logging of maintenance requests." Proceedings of the 23rd Australian Computer-Human Interaction Conference (2011): 126-129. ACM Digital Library. Web. 21 Nov. 2015.

This is an article based on the three years of research the authors did on exploring how citizens can have a say in urban planning through social media and mobile technology. The fact that they spent so much time researching makes this a valuable source. It is clear they did not just jump into making an application but really thought about what was needed first. The negative part of this is that they did not discuss their own project at length. Most of their article discussed prior research and their approach. There was less content on their implementation and results, but they did have some. However, it would have been interesting to learn even more about that. The project they did is very similar to what we are doing. They are using the camera and GPS functions of smart phones to take geo-tagged photos of things that need fixing, and that is what we plan to do as well. I found this article during the class that we had in the library when we were learning how to use the library databases to find sources for our project.

Pang, Carolyn, Carman Neustaedter, Jason Procyk, Daniel Hawkins, and Kate Hennessy. "Moving towards user-centered government: community information needs and practices of families." Proceedings of the 41st Graphics Interface Conference (2015): 155-162. ACM Digital Library. Web. 21 Nov. 2015.

This study was about how local governments can better serve families by providing them with information that is relevant to them. They discuss how other options to connect the city with families through technology are not always user-friendly. City websites often contain an overwhelming amount of information and can be difficult to navigate. Social media allows people to be aware of events in their city, but communication with citizens through that medium can only go so far due to privacy and security concerns. The researchers found that people wanted information related to the places they encountered daily. They concluded that civic engagement should be a personalized experience for the citizen. There was not a lot of technical information in this document, but it still provides information about how people might want to use an application about civic engagement. It focused a lot on user-centered design, which is something that my group should be focusing on as well. I found this document in the "Cited By" section of one of my other sources.

Sandoval-Almazan, Rodrigo, J. Ramon Gil-Garcia, Luis F. Luna-Reyes, Dolores E. Luna, and Yaneileth Rojas-Romero. "Open government 2.0: citizen empowerment through open data, web and mobile apps." Proceedings of the 6th International Conference on Theory and Practice of Electronic Governance (2012):

30-33. ACM Digital Library. Web. 29 Nov. 2015.

This article explains how open government can be facilitated through the use of open data and mobile apps. In order to conduct their research, the authors focused their study on the top countries according to a UN survey. They found that not all countries are implementing mobile apps for civic engagement but that there is still a variety of apps out there that potentially promote civic engagement. They also found that many of these apps are not being created by the government. A large portion of them are actually being created by private companies and even individual citizens. However, their sample size may have been somewhat small because only 7 of the 35 countries that they looked at had apps like these available. This is relevant to our project because we plan to make government data in terms of urban projects available to citizens through a mobile app. We are focusing on more local government than this article did, but the concept is similar. I found this by searching through the library's database ACM Digital Library.

Consoli, Sergio, Misael Mongiovic, Andrea G. Nuzzolese, Silvio Peroni, Valentina Presutti, Diego Reforgiato Recupero, and Daria Spampinato. "A Smart City Data Model based on Semantics Best Practice and Principles." Proceedings of the 24th International Conference on World Wide Web (2015): 1395-1400. ACM Digital Library. Web. 29 Nov. 2015.

This article was written by several authors from the National Research Council in Italy. Their focus is on how data for a smart city incorporates numerous aspects from transportation to waste collection. Because the authors are from Italy, their focus is also on a particular city in Italy. However, maybe this can still reveal information about city governments in general. They mention how communication across a city can be difficult and how technology and data can help manage that. They look at how this data can be collected in an efficient manner. The article is relevant to our senior design topic because a large part of our project involves presenting city data in a way that is effective and catered to the user. I found this article by searching through the library's database ACM Digital Library.

Camden, Raymond and Andy Matthews. jQuery Mobile Web Development Essentials Second Edition. Packt Publishing, 2013. Print.

We are planning on using javascript and possibily jQuery by extension in our project. The book details jQuery's use in mobile web development. There are explanations on how to add jQuery to HTML pages. There is a possibility that the project will involve at least a bit of jQuery usage, so knowing about its usage in mobile web pages gives us more tools to use in the project. This source was found by searching keywords: "meteor" and "js" in the Safari Books Online database.

Reyna, Marcelo. Meteor Design Patterns. Packt Publishing, 2015. Print. We are planning on using Meteor to develop our project. This book goes over Meteor programming patterns and is intended for people who have at least familiarized themselves with the basics of Meteor. The programming patterns here are meant to make the program more maintainable and scalable. This book could help with debugging down the line. This source was found by searching keywords: “meteor” and “js” in the Safari Books Online database.

“Americans’ Mixed Messages on Digital Government.” Federal Computer Week 28.3 (2014): 34. Applied Science and Technology Source. Web. 8 Oct. 2015. Our project deals with a mobile application that bridges the gap between government and the public, however, this would be completely useless if the public didn’t want to engage with their government in this manner. This article describes how Americans use US government online services and how they interact with the government through virtual resources. Overall, it evaluates whether there is a place for government in social media or mobile devices. It shows that we must be careful when creating our application in order to target it to a specific audience. We must then make it user-friendly and nonintrusive for all involved, otherwise it may get ignored. This source was found in the Applied Science and Technology Source Database by searching for articles containing information regarding the keywords mobile and government.

Moore, John. ”Mobile Usability: Targeting The Tool To The Audience.” Federal Computer Week 27.12 (2013): 27-29. Applied Science and Technology Source. Web. 4 Nov. 2015. Due to the nature of government and public relations, it’s important to make sure that the application we are creating is going to be used. A good way to do this is to make sure the application we are making suits the needs of the audience. This article goes into detail about web usability and how exactly web technology can be used to reach out. An emphasis is placed on the use of HTML 5 in order to build a cross-platform application. It also details the difficulties in creating manageable mobile applications. This source was found in the Applied Science and Technology Source Database by searching for articles containing information regarding the keywords usability and government.

0.19.1 Source Code

Mobile:

```
List = new Mongo.Collection("list");//List Collection to hold entered data

if (Meteor.isClient) {
var MAP_ZOOM = 15;

/*functions to run at startup*/
Meteor.startup(function () {

/*Loads Google Maps and starts 'showMainMenu' session*/
  GoogleMaps.load();
  Session.set("showMainMenu", 1);
  Session.set("showPhoto", 0);
  Session.set("showCategories", 0);
  Session.set("showTopic", 0);
  Session.set("showDetails", 0);
  Session.set("showConfirmation", 0);

  sAlert.config({
    effect: '',
    position: 'top-right',
    timeout: 5000,
    html: false,
    onRouteClose: true,
    stack: true,
    offset: 0, // in px - will be added to first alert (bottom or top - depends of th
    beep: false,
    onClose: _.noop //
  });

});
```

```
/*Event handlers for 'MainMenu' session*/
Template.MainMenu.events({

  /*After clicking 'submit',active session is now 'showPhoto'*/
  'click .submitareport': function(event) {
    event.preventDefault();
    Session.set("showMainMenu", 0);
          Session.set("showPhoto", 1);
  }
});

/*Event handlers for 'PhotoPage' session*/
Template.PhotoPage.events({

 /*Clicking 'take photo' will cause the camera to take
 a photograph*/
 'click .takePhoto': function(event, template) {
    var cameraOptions = {
      width: 800,
      height: 600
    };
    MeteorCamera.getPicture(cameraOptions, function (error, data) {
      if (error) {
        var warining=sAlert.warning('error with picture', {timeout: 'none'});
      }
      else{
        $('.photo').show();
        $('.photo').attr('src', data);
        $('.takePhoto').hide();
        $('.retakePhoto').show();
        $('.next').show();
```

```
        $('label').hide();
      }
    });
    event.preventDefault();
  },
  /*clicking 'next' submits the photo, and starts the 'showCategories' session*/
  'click .next': function(event) {
    event.preventDefault();
    var PictureSrcSubmit= $( '.photo' ).attr('src') ;
    Session.set("picSubmit", PictureSrcSubmit);
    Session.set("showPhoto", 0);
    Session.set("showCategories", 1);
  }
});

/*Event handlers for Category Page*/
Template.CategoryPage.events({

  /*clicking 'next' will set variable 'category'
  to selected category, then start 'showGPS' session*/
  'click .next': function(event) {
    event.preventDefault();
    var elements = document.getElementsByName("category");//start of getting selectin

      for (var i = 0; i < elements.length; i++){
        if (elements[i].checked)
        {
          var category = elements[i].value;
        }
      }
    Session.set("categorySubmit", category);
    Session.set("topicSubmit", category);
```

```
    Session.set("showCategories", 0);
    Session.set("showGPS", 1);
  }
});


/*Event handlers for 'GPSPage'*/
Template.GPSPage.events({

  /*Clicking 'next' start 'showDetail' session*/
   'click .next': function(event) {
      event.preventDefault();

      Session.set("showGPS", 0);
      Session.set("showDetails", 1);
    }
});

/*Shows Google Map*/
Template.map.onCreated(function() {
    var self = this;

    GoogleMaps.ready('map', function(map) {
      var marker;

      // Create and move the marker when latLng changes.
      self.autorun(function() {

        var latLng = Geolocation.latLng();
        if (! latLng){

          return;
```

```
        }

        // If the marker doesn't yet exist, create it.
        if (! marker) {
          marker = new google.maps.Marker({
            position: new google.maps.LatLng(latLng.lat, latLng.lng),
            map: map.instance
          });
          Session.set("latitudeSubmit", latLng.lat);
          Session.set("longitudeSubmit", latLng.lng);
        }
        // The marker already exists, so we'll just change its position.
        else {
          marker.setPosition(latLng);
        }

        // Center and zoom the map view onto the current position.
        map.instance.setCenter(marker.getPosition());
        map.instance.setZoom(MAP_ZOOM);
      });
    });
  });

/*Helper functions for 'map*/
Template.map.helpers({

        /*Returns error message in case of an error*/
    geolocationError: function() {
      var error = Geolocation.error();
      return error && error.message;
    },
```

```
        /*Initializes map and centers in and zooms on location*/
    mapOptions: function() {
      var latLng = Geolocation.latLng();
      // Initialize the map once we have the latLng.
      if (GoogleMaps.loaded() && latLng) {
        return {
          center: new google.maps.LatLng(latLng.lat, latLng.lng),
          zoom: MAP_ZOOM
        };
      }
    }
  });
/*Helper functions for Confirmation page;
Each one returns the data entered from
each session*/
Template.ConfirmationPage.helpers({
  image: function(){
    return Session.get("picSubmit");
  },
  category: function(){
    return Session.get("categorySubmit");
  },
  topic: function(){
    return Session.get("topicSubmit");
  },
  latitude: function(){
    return Session.get("latitudeSubmit");
  },
  longitude: function(){
    return Session.get("longitudeSubmit");
  }
});
```

```
/*Event handlers for 'TopicPage'*/
Template.TopicPage.events({

        /*Clicking 'next' start 'showGPS' session*/
   'click .next': function(event) {
      event.preventDefault();
      var topic= event.delegateTarget.topic.value;
      Session.set("topicSubmit",topic);
      Session.set("showTopic", 0);
      Session.set("showGPS", 1);
    }
});

/*Event handlers for 'OptionalSetailsPage'*/
Template.OptionalDetailsPage.events({

    /*clicking 'next' starts 'showConfirmation' session*/
  'click .next': function(event) {
    event.preventDefault();
    var details= event.delegateTarget.details.value;
    var name= event.delegateTarget.username.value;
    Session.set("detailsSubmit",details);

    Session.set("nameSubmit",name);
    Session.set("showDetails", 0);
    Session.set("showConfirmation", 1);
  }
});

/*Event handlers for 'formhere'*/
Template.formHere.events({
```

```
/*assigns variables to data entered from sessions*/
"submit form": function (event) {
    // Prevent default browser form submit
  event.preventDefault();

    // Get value from form element
  var topic= Session.get("topicSubmit");
  var details = Session.get("detailsSubmit");
  var src=  Session.get("picSubmit");
  var category= Session.get("categorySubmit");
  var latitude= Session.get("latitudeSubmit");
  var longitude= Session.get("longitudeSubmit");
  var hidden="hidden";



  //end of selecting category
  var heading = category+longitude+latitude;//heading is combination of category a

  /*Insert variables into Collection*/
    List.insert({
      category: category,
      topic: topic,
      details: details,
      image: src,
      longitude: longitude,
      latitude: latitude,
      heading: heading,
      createdAt: new Date() // current time
    });
  Session.set("showConfirmation", 0);
  Session.set("showMainMenu", 1);
```

```
        }
});


/*Returns current session*/
Template.registerHelper('session', function( value ) {
  return Session.get(value);
});

/*must bind to `document.body` as element will be replaced during re-renders
add the namespace `.tplquestions` so all event handlers can be removed easily*/
Template.CategoryPage.created = function(){
  $(document.body).on('change.tplquestions', '.form', function(e){

     $('.next').show();
  });
};

/*remove all event handlers in the namespace `.tplquestions`*/
Template.CategoryPage.destroyed = function(){
  $(document.body).off('.tplquestions');
}



}
```

Web:

```
List = new Mongo.Collection("list"); //List Collection to hold all entries
Display = new Mongo.Collection(null); //Display Collection tol hold entries to be dis
var MAP_ZOOM = 15;
```

```
if (Meteor.isClient) {
  // This code only runs on the client
  var current; //Variable keeps track of clicked entry in reactive table

/*Helper functions for homepage*/
  Template.home.helpers({

      /*Scans List Collection for entries with distinct topics,
      counts number of entries for each distinct topic,
      then inserts this information into Display Collection*/
    list: function(){
    Display.remove({});
    var myArray = List.find({}, {sort:{createdAt: -1}}).fetch();
    var distinctHead = _.uniq(myArray, false, function(List) {return List.topic});
    var valuearray= new Array();
    var topics = function(myArray){return _.pluck(distinctHead, "topic")};
    var topicarray = topics(distinctHead);
    for(var i = 0; i<topicarray.length; i++)
    {
        valuearray[i]=0;
      for(var j = 0; j<myArray.length; j++)
      {
        if(topicarray[i]==myArray[j].topic)
        {
          valuearray[i]++;
        }
      }
    }
    for(var k =0;k<distinctHead.length;k++)
    {
      distinctHead[k].quantity=valuearray[k];
    }
```

```
    for(var m =0;m<distinctHead.length; m++){
    Display.insert(distinctHead[m]);
        }
    },

        /*Returns Display Collection with the
        fields: key for pulling the data from the collection and
        label for the corresponding header to be displayed in the
        reactive table in web.HTML*/
    settings: function () {
      return {
        collection: Display,
        fields: [
          {key: 'topic', label: 'Category'},
          {key: 'details', label: 'Description'},
          {key: 'createdAt', label: 'Date Added'},
          {key: 'quantity', label: 'Quantity'},
        ]
      };
    }
  });

/*Helper functions for the page that shows entries of clicked topics*/
  Template.show.helpers({

  /*Scans List Collection for entries that match topics with clicked entry
  in reactive table*/
    list: function(){
      var myArray = List.find({}, {sort:{createdAt: -1}}).fetch();
      var nlist= new Array();
      for(var i = 0; i< myArray.length;i++)
      {
```

```
      if(myArray[i].topic==current)
      {
        nlist.push(myArray[i]);
      }
    }
    return nlist;
  }

});

/*Event handlers the page that shows entries of clicked topics*/
Template.show.events({

  /*Routes to home page when user clicks back button with class 'btn-secondary' on
  "click .btn-secondary": function(){
    Router.go('/');
  },

      /*Deletes object from List Collection, when icon with class 'delete' is click
  "click .delete": function(){
    List.remove(this._id);
  }
  });

/*Event handlers for home page*/
Template.home.events({

/*Sets variable 'current' to topic of clicked entry in reactive table,'
then routes user to 'show' page*/
'click .reactive-table tbody tr': function () {
 current = this.topic;
 Router.go('/show');
```

```
    }
  });
}

/*Routes user to 'show' page, which displays all entries
of matching topics to clicked entry*/
Router.route('/show', {
  name:'show',
  template:'show'
});

/*Routes user to home page*/
Router.route('/', {
  name:'home',
  template:'home'
});
if (Meteor.isServer) {
  Meteor.startup(function () {
  });
}
```