# Project BW Final Iteration

Worcester Polytechnic Institute

Computer Science Department

CS3733 Software Engineering

## Team D: Dragonfly Dragons

| Team Members | Roles |
|---|---|
| Michael Abadjiev | Product Owner |
| Shreeja Bhattacharjee | Project Manager |
| Yufei Gao | Documentation Analyst |
| Mingquan Liu | Co-Lead Software Engineer |
| Treksh Marwaha | Software Engineer |
| Benjamin Newmark | Assistant Lead Software Engineer |
| Kenneth Quartuccio | Test Engineer |
| David Swenarton | Co-Lead Software Engineer |
| Daniel Wivagg | Software Engineer |
| | |
| Wilson Wong | Professor |
| Chris Meyers | Team Coach |

# Table of Content

## Introduction

Hospitals are complex, dynamic places where visitors and patients can get lost and staff sometimes struggle to manage the vast influx of visitors and guests, in addition to the tasks that come up each day. Technological solutions are good ways to mitigate these problems, but have not been widely implemented so far. In order to help visitors to the hospital get around and aid staff with management, this report proposes a kiosk system for the Brigham and Women's Hospital in Boston, Massachusetts.

The primary purpose of the kiosk is to provide maps and directions for people who are looking to get around the hospital. Visitors and patients have access to other features too, such as the ability to find nearby restrooms, vending machines, and waiting rooms. Additionally, the kiosk offers features that only authorized users can access, such as requests for spill or hazard cleanups, maintenance, patient transportation, or language interpreters in a given location. Administrators can edit maps, manage employees and service requests using the kiosk.

## Technologies

Technologies, software frameworks:
Java SDK 8, IntelliJ IDeA 2017.1.4, Java FX SceneBuilder 8.4.1, Git, Google Drive, Google Calendar, Google Slides, Google Survey, Google Sheets, Google Docs, Photoshop, SketchBookPro, When2Meet, Slack, Slackbot (for reminders), LucidChart

External software libraries:
Junit 4.11
Org.apache.derby 10.14.1.0
Com.jfoenix 1.4.0
Javax.mail 1.4
Javax.activation 1.1.1
Org.controlsfx 8.40.14
Jnativehook 2.1.0
Org.hamcrest 1.3
HealthcareAPI 3.2
Team-G-Food-Request

## User Stories

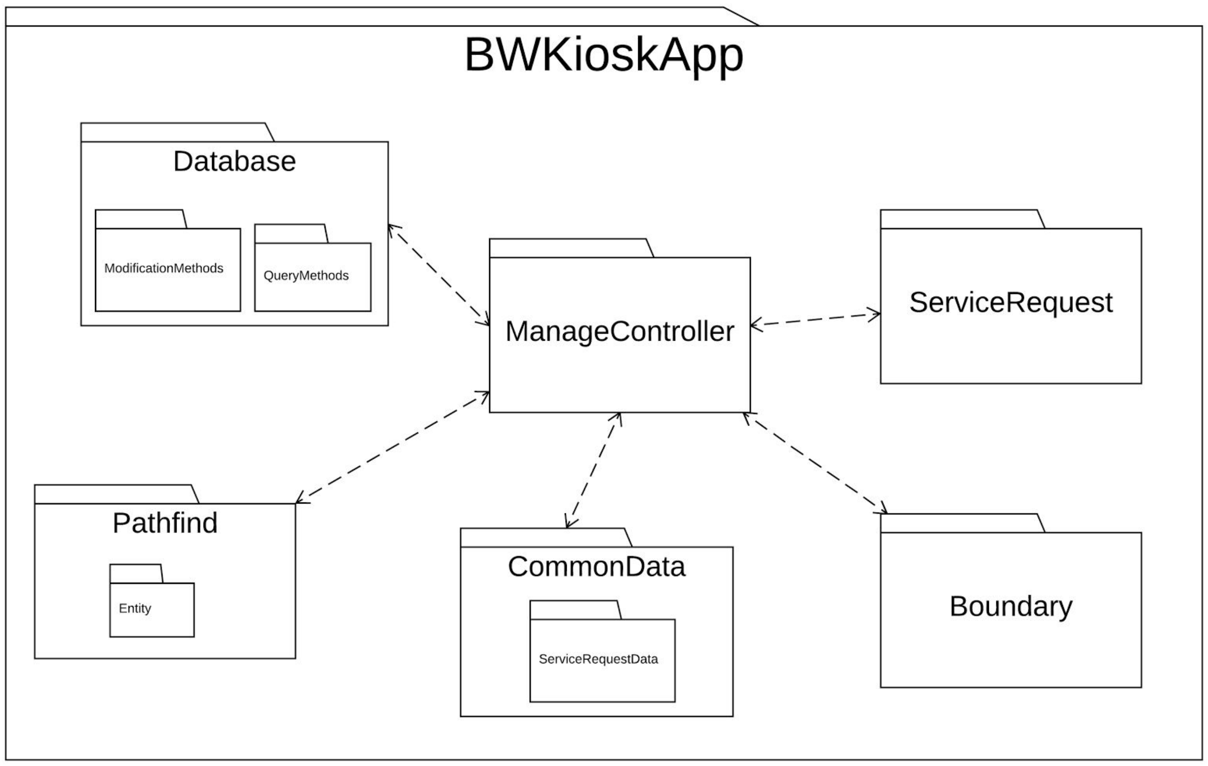
| User stories (implemented over the course of the entire term) | Relative Mass Ranking |
|---|---|
| As a hospital employee, I want to view a list of service requests that are not completed so I know where to direct my attention and staff. | 1 |
| As a patient/visitor, I want to view information about a room or location when I click on it so I can find information about where I'm going. | 1 |
| As a hospital employee, I want to close service requests that are completed so that I can keep track of what tasks need attention. | 1 |
| As a hospital employee, I want to request services, such as cleanups for random spills, so that the appropriate staff is made aware and is able to deal with it. | 1 |
| As a hospital employee, I want to request interpreters when any of the patients is going to need to communicate with employees so that the interpreter could arrive when needed at a specific place to provide help. | 1 |
| As a hospital employee, I want to be able to request transportation for a patient within the hospital so that we can efficiently move patients to new locations. | 2 |
| As an administrator, I want to be able to tag locations so that others can see important information about them. | 2 |
| As an administrator, I want the ability to add, modify, or remove nodes and edges from the map so that I can control the available paths a user can take. | 3 |
| As a patient/visitor, I want the kiosk to have an option to change the starting location, different from my current location, so I can get information for future use. | 3 |
| As a hospital guest, I want to view information about bathrooms, vending machines etc. based on specific map coordinates, so I can find what I'm looking for. | 4 |

| | |
|---|---|
| As a patient/visitor, I want to find paths that avoid stairs or obstacles so that I can follow a handicapped-accessible route. | 5 |
| As a patient/visitor, I want to drag a path on the map to force it to pass through a different point. | 5 |
| As a patient/visitor, I want the kiosk to be able to display text directions so I can find my way through the hospital. | 6 |
| As a patient/visitor, I want the kiosk to find the path from my location to my destination so I can find my way through the hospital. | 6 |
| As a hospital administrator, I want to change the algorithm used to find paths so that I can adjust the way kiosk users are directed through the hospital. | 3 |
| As a patient/visitor, I want the kiosk to email me with text directions so that I can follow the directions after I have left the kiosk. | 4 |
| As a patient/visitor, I want to be able to switch floors while using the kiosk so I can find paths between them. | 5 |
| As an administrator, I want to be able to view and manage hospital employees so that I can add or remove employees, and specify their skills | 3 |
| As an administrator, I want to be able to add new maps to buildings so that visitors can navigate to added floors | 3 |
| As an administrator, I want to be able to add new buildings with their main floor | 3 |
| As a hospital visitor, I want to view information about the people who created this amazing application so that I can hire them to work for me | 1 |
| As an administrator, I want to be able to click and drag nodes to easily adjust their location and make paths looks cleaner | 2 |
| As an administrator, I want to be able to click on the map when adding nodes to simplify the process of adding new locations | 1 |

| | |
|---|---|
| As a patient/visitor, I want to be given some hints of using the kiosk. | 1 |

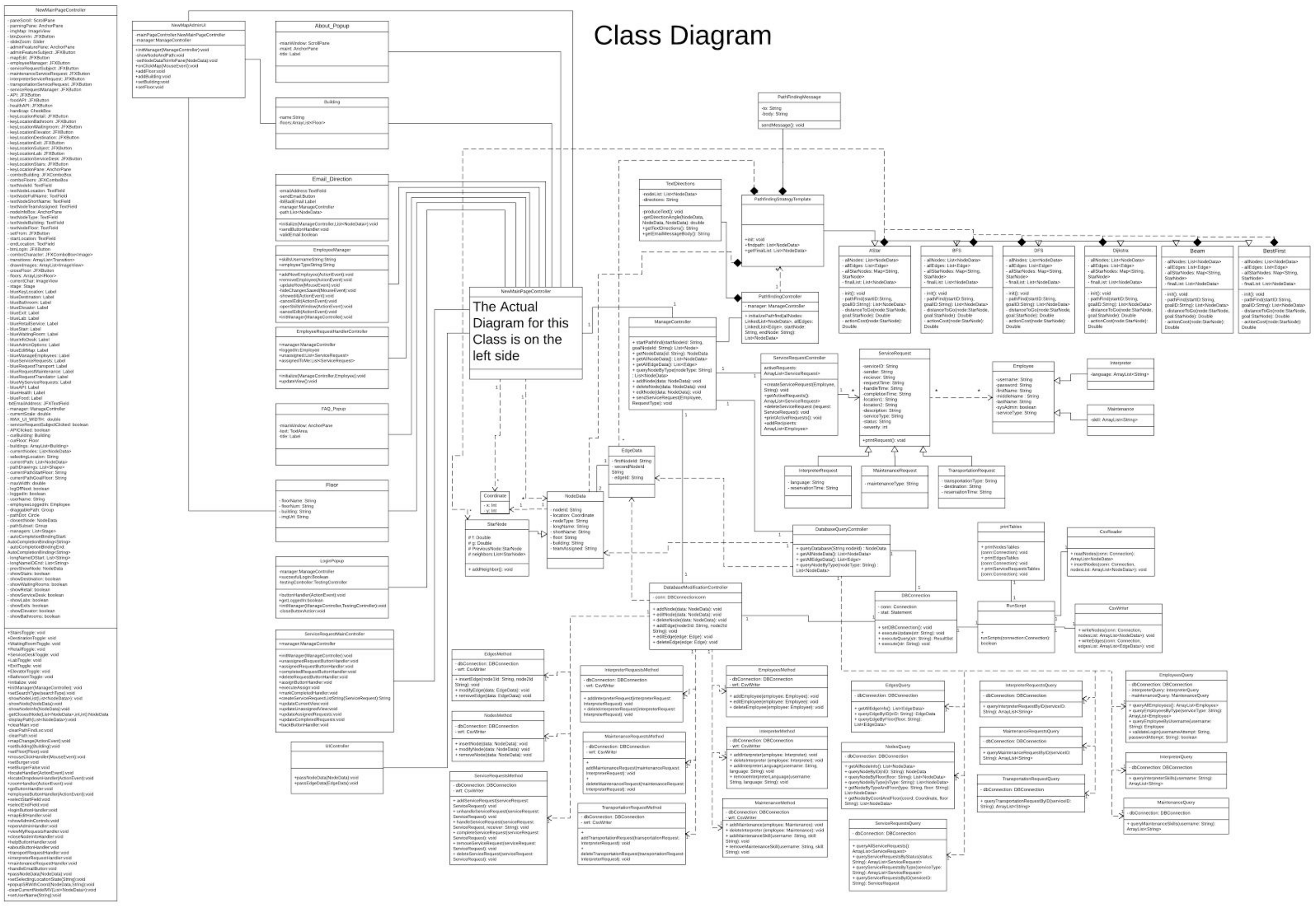
# Diagrams

## Package Diagram

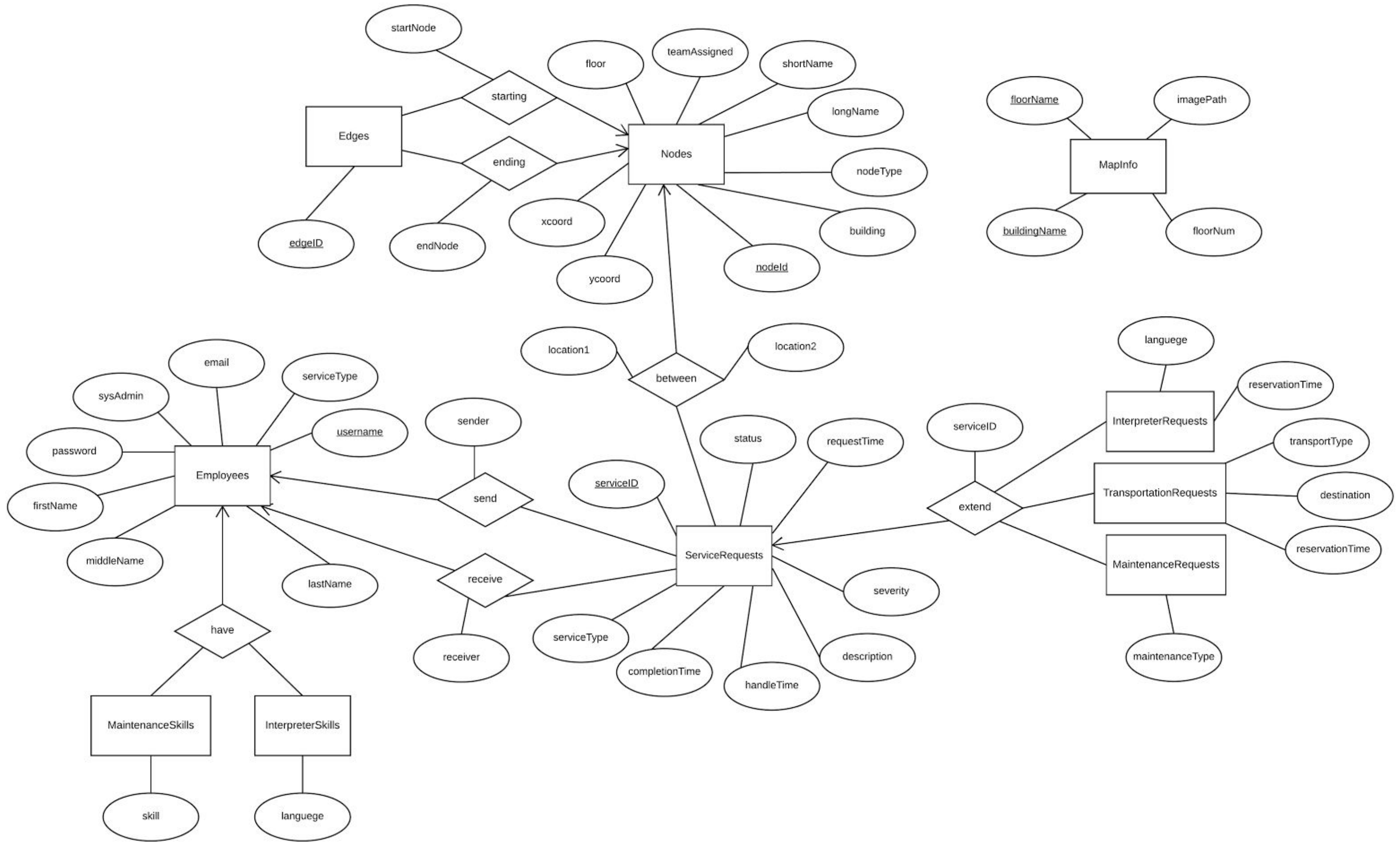


## Class Diagram

## Class Diagram



## Data Store Design (ERD)



## Design Patterns

### Strategy + Template Pattern

Class Names: PathfindingTemplate, Search Algorithm Classes

### Simpleton Pattern

Class Name: AppSettings

## Memento Pattern

Class Name: Main, GlobalKeyListener, GlobalMouseListener, NewMainPageController

## Observer Pattern

Class Name: NodeData, MapAdminController

## Code Related Responsibilities

| Iteration | Task | Person/People |
|---|---|---|
| 1 | Basic pathfinding | Ben, Trek |
| 1 | A star algorithm | Dan, Ken |
| 1 | Integrating embedded DB | Haiau* |
| 1 | Basic UI | Michael, Shreeja |
| 1 | Basic service request | Yufei, Shreeja |
| 1 | Database Classes | Yufei |
| 1 | Integration for iteration 1 | David, Mingquan |
| 2 | Implement the Strategy Pattern for accessible paths | Trek, Ben |
| 2 | Create sorting method for non-accessible paths (i.e. stairway edges) | Treksh, Dan |
| 2 | Implement the Transportation Request UI | Michael, Shreeja |
|  | Implement the Interpreter Request UI | Michael, Shreeja |
| 2 | Implement overall service request UI that allows employees to create, handle, and remove requests | David, Shreeja, Treksh |
| 2 | Implement Interpreter Request database fields | Haiau, Mingquan, Yufei |
| 2 | Finish service request controller with required functionality | Yufei |

| | | |
|---|---|---|
| 2 | Create database for tracking employees and their permissions | Haiau*, Yufei, Mingquan |
| 2 | Implement employee management UI linked to the employee database | Ben, Ken |
| 2 | Create report generator for service requests (similar to text directions) | Yufei |
| 2 | Implement email feature that notifies employees of service requests via email | David, Yufei |
| 2 | Implement alternate pathfinding algorithms: BFS, DFS, Dijkstra’s Algorithm | Dan, Ken |
| 2 | Implement Facade Pattern for accessing different paths and update pathfinding controller | Dan |
| 2 | Add UI feature for changing which algorithm is used | Ben, Ken, Shreeja |
| 2 | Refactor the node database to align with new nodes from other teams | Haiau*, Yufei |
| 2 | Create a class to generate text directions from a given list of nodes | Dan, Treksh |
| 2 | Implement email feature that sends text directions to kiosk users via email | David |
| 2 | Implement zoom/pan feature for map UI | Michael, Shreeja |
| 2 | Add Proxy Pattern for switching between images | Michael |
| 2 | Implement floor switching UI components | Michael |
| 2 | Investigate/implement JFoenix | Michael, Ben |
| 2 | Redesign map builder UI | David, Ben, Ken, Shreeja |
| 2 | Add Help page | Yufei, Ben |
| 2 | Integration for iteration 2 | David, Mingquan |
| 3 | Gradle setup | David |
| 3 | Update text directions to include distances | Daniel |
| 3 | Fix login errors | Treksh, Michael |
| 3 | Update employee manager for skills | Daniel |

| | | |
|---|---|---|
| 3 | Created UIs for adding new buildings and new floors | Ben |
| 3 | Set up controllers to find and upload images files to preset folder | Ben |
| 3 | Integrated having multiple new buildings and floors into home screen; set up formatting guidelines for buildings/floors | Ben, Michael |
| 3 | Integrated adding new buildings and screens from the admin screen | Ben, Shreeja |
| 3 | add floor object | Michael |
| 3 | Fix various minor display bugs on the main screen such as the hamburger bug, node info layering etc. | Michael |
| 3 | Add floor switching notification to path | David |
| 3 | Add more fields to the database of employees and service requests | Yufei |
| 3 | Add classes of different types of service requests | Yufei |
| 3 | Add database query and modification functions for new tables | Yufei |
| 3 | Add employee manager UI | Daniel |
| 3 | Add more locations to the selection of key locations | Shreeja |
| 3 | Add minor functions to main UI to tidy it up, like a box for text directions, a checkbox for handicap path etc. | Michael, Shreeja,Treksh |
| 3 | Fix bugs in pathfinding like getting stuck in an elevator loop, and tidy up text directions | Ken, Daniel |
| 3 | Make nodes draggable to adjust their location in the map editor UI | David, Michael, Mingquan |
| 3 | Change nodeData to have an image relating to it that can be used and changed when displaying nodes, major overhaul of how nodes are displayed and interact with code | Mingquan,Treksh |
| 3 | Implement the singleton design pattern for selected values | Shreeja, David |
| 3 | Implement template and strategy design patterns in | Daniel, Ken |

| | | |
|---|---|---|
| | pathfinding | |
| 3 | Add about page | Ken |
| 3 | Add UI for employees to handle their own service requests | David, Dan |
| 3 | Update CsvReader and CsvWriter to account for new subtables | David |
| 3 | Integration for iteration 3 | David, Mingquan |
| 3 | Jar file fixes | David |
| 4 | Wait and Load screen | Treksh |
| 4 | 2 scenic routes | Ken |
| 4 | Database integration for maps (buildings and floors) | Ben |
| 4 | Animated Path | Michael, Yufei |
| 4 | Redo About Page | Yufei |
| 4 | Redo FAQ Page | Shreeja,Yufei |
| 4 | Format text directions to be more user-friendly | Dan |
| 4 | Implement Memento Pattern | Dan |
| 4 | Zoom Slider | Michael |
| 4 | Fuzzy Wuzzy search | Shreeja, Yufei |
| 4 | CSS theme | David |
| 4 | Service Request UI overhaul | David |
| 4 | MainUI Overhaul | Mingquan |
| 4 | New/fully refactored controllers | Ben |
| 4 | Visual Enhancement | Shreeja, Yufei |
| 4 | Informative Keywords on hover | Shreeja |
| 4 | Integration for iteration 4 | Ben, David, Mingquan |

## Non-Code Related Responsibilities

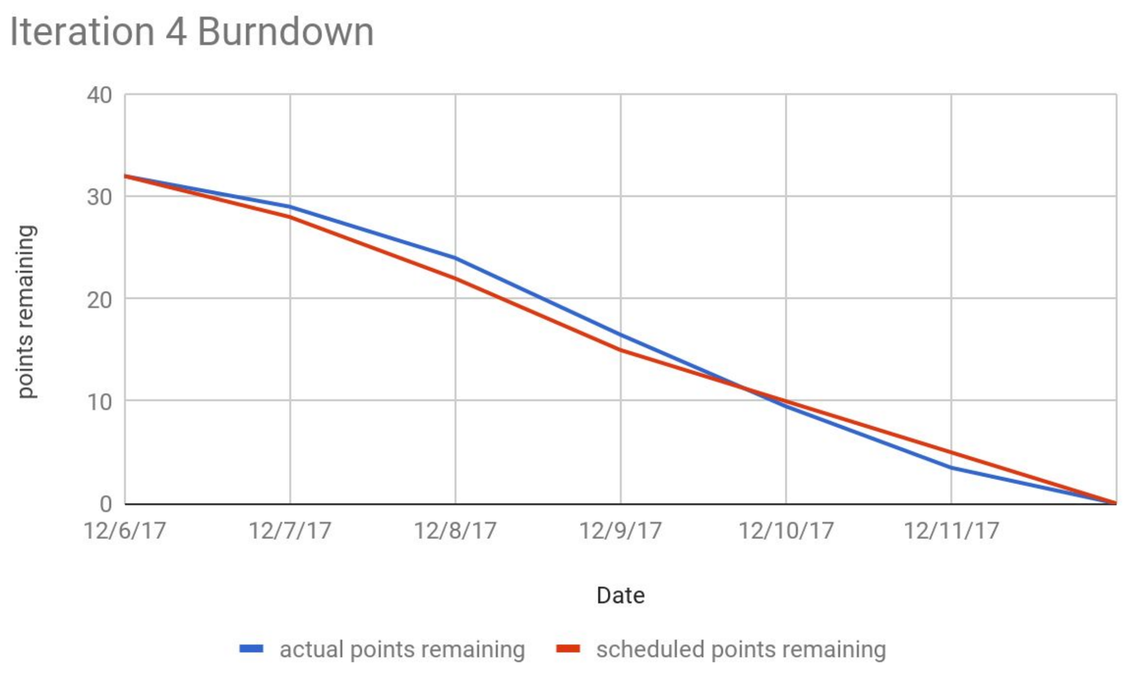
| Responsibilities | Person/People |
|---|---|
| Map Data Entry | Yufei |
| Reports for all iterations | Yufei, Shreeja, Michael, Daniel |
| Powerpoints for all iterations | Yufei, Shreeja, Michael, Daniel, Treksh, Ken |
| Class Diagrams for all iterations | Yufei, Shreeja, Treksh |
| Sequence Diagram for iteration 3 | Mingquan, Shreeja |
| Package Diagram for final iteration | Shreeja, Yufei |
| Entity Relation Diagram for all iterations | Yufei |
| Github Issues Assigning for all iterations | Daniel, Michael |
| Product Burndown and Velocity Chart | Daniel, Michael |
| Salary Survey for all iterations | Michael, Shreeja |
| User Manual for final iteration | Yufei, Michael, Treksh |
| All graphic other than scenebuilder(Node icons, wait and load screen, animation for stick figures etc) | Treksh |

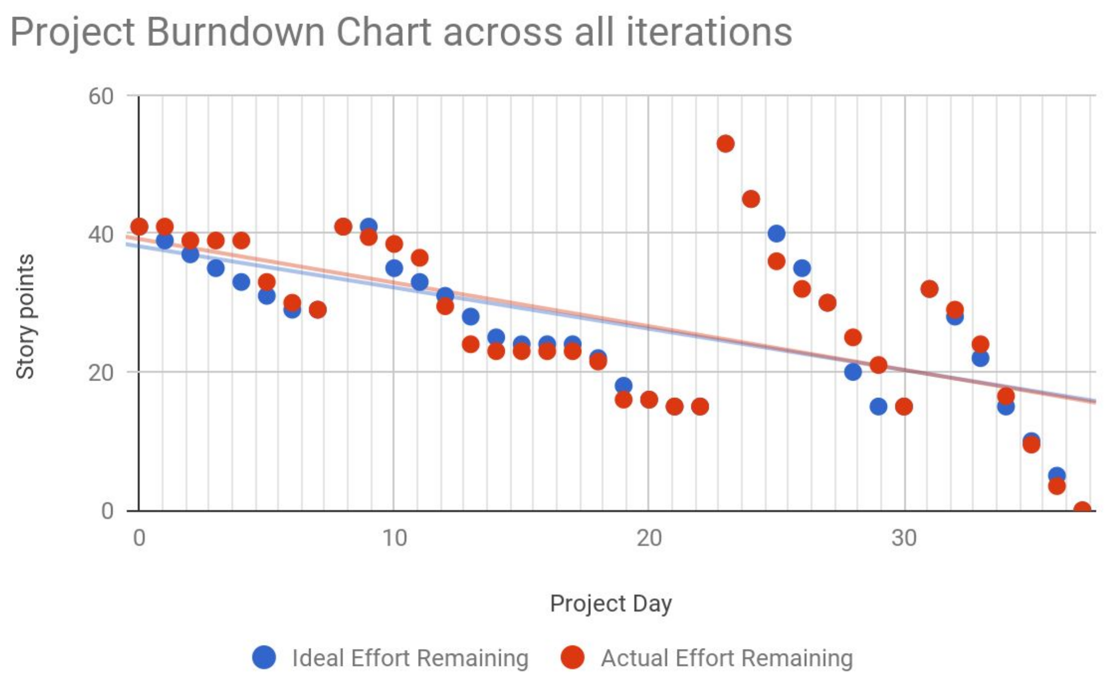
## Team Velocity & Product Burndown Chart

### Team Velocity

During the course of this project, the team accomplished a total of 110 user story points, spread across 4 iterations. This came out to an average of 27.5 user story points per iteration, however the product burndown chart seen below shows that there was a concentration of user story points accomplished in the third iteration. This was a result of poor team organization during the beginning of the project, but as we progressed through the term we learned a lot about how to manage the workload, using tools like github issues and a continuous integration technique.

## Product Burndown Chart





## Team’s Reflection

### What did the team learn about software designing through the iterations? Did your later designs better match the subsequent code written or not? Discuss why.

We learned that integrating along the way (over the course of several days) is better than a ‘big bang’ approach, wherein all the integration is done the night before (and half your team doesn’t sleep). Over the course of several iterations, we got better about spacing our integration out. As a result, we had more robust and clean code during the later iterations. Our later designs also better match the subsequent code written. This might be because initially, we were extremely uncoordinated and inefficient about writing code. People would write code without discussing with the other members, and as a result, when it came time to integrate, we found that the separate pieces of code would not work together. Thus, in later iterations, as we began to discuss what we planned on doing, we created code faster and with fewer bugs.

### What did the team learn about software methodologies and working together as a team throughout the iterations? Did the team operate better or worse as time progressed? Why?

We learned how to better follow agile practices and implement scrum meetings. While we were aware of these, in the beginning, we didn’t fully implement it until the later stages. We found that agile practices really helped us. We kept assigning big tasks to everyone and kept trying to code for future iterations. Once we stopped doing this and focused on smaller goals and deadlines, as well coding for the current iteration, we started performing better. Furthermore, we became better at delegating tasks. This was partially due to the fact we figured out the different strengths of everyone. All of this together resulted in us operating better as a team as time progressed.

### What did your team do to improve team cohesion and foster a team culture?

As recommended by Professor Wong, we all went out for dinner (and for once didn’t discuss work). In addition, staying up till sunrise and sharing food strengthened team bonds. We also had daily scrums, which resulted in us meeting and talking every day. While our team faced many challenges (such as people dropping out), we all commiserated and succeeded in “adapting, improvising, and overcoming.”

We tried our best to foster a team culture where everyone is held accountable. This was mostly upheld by most members following through on the following guidelines. Members normally ask to leave if they cannot come to a meeting. If they cannot make it, they will still be assigned work to do. The use of iteration surveys and salaries also helped to keep team members accountable. Although we are not the best team, we are a very balanced team in that everyone is involved in the project.

## What would the team have done differently in hindsight? What did the team learn as a result of unexpected challenges? What advice would you give future teams taking CS3733 to improve their experience?

In hindsight, we would have done the following:

1. Followed Professor Wong's advice earlier on.
2. Put more people on UI earlier.
3. Assign smaller tasks, instead of in huge chunks.
4. Discussed what we envisioned our application to look like earlier.
5. Ask the Team Coach for help more.
6. Manage people differently depending on their personalities.
7. Earlier assessment of skill sets/strengths of group members.

We learned:

1. Everyone should be familiar with the UI. The UI is such a huge task it is completely unreasonable to assume one person can do it all. Not only does working on the UI help with writing code, everyone, in the end, contributes to the UI at some point.
2. The presenters should spend more time practicing before presenting to prevent mishaps, such as a code malfunctioning during the presentation.
3. Always be prepared for unexpected setbacks, such as member loss.

Our Advice:

1. Start working on the project as soon as you can.
2. Meet every day.
3. Set small goals rather than big
4. Change your leadership approach depending on the person
5. Integrate along the way (do not use big bang approach! Or at least do not do it the night before the project is due).
6. Carefully assess people's different skill sets.
7. The presenters should spend a sufficient amount of time practicing before presenting.
8. Distribute work reasonably.
9. Always be prepared for unexpected setbacks, such as member loss.
10. Try to make everyone involved.
11. Make sure your application is user-friendly, not just "bare-bones' functional.

## Salary Distribution

| Person | Salaries | Contribution |
|---|---|---|
| Ben Newmark | 24.63 | Database integration for maps (buildings and floors), Integration for iteration 4 |
| Treksh Marwaha | 9.70 | Wait and Load screen |
| David Swenarton | 28.36 | CSS theme, Service Request UI overhaul, Integration for iteration 4 |
| Daniel Wivagg | 23.88 | Format text directions to be more user-friendly, Implement Memento Pattern |
| Kenneth Quartuccio | 19.40 | 2 scenic routes |
| Mingquan Liu | 28.36 | MainUI Overhaul, Integration for iteration 4 |
| Shreeja Bhattacharjee | 23.13 | Redo FAQ Page, Fuzzy Search, Visual Enhancement, Documentation |
| Yufei Gao | 21.64 | Animated Path, Redo About Page, Redo FAQ Page, Fuzzy Search, Visual Enhancement, Documentation |
| Michael Abadjiev | 20.90 | Animated Path, Zoom Slider, product burndown |
| Summation | 200.00 | |