**Software Engineering Project Report**

**For class Software Engineering – CSCE 606, Texas A&M University**

***Prepared by:***

**Anh Nguyen**

**Duc Nguyen**

**Dung Dam**

**Van Trinh**

***Tuesday, December 3, 2013***

***Version 1.0.0***

Table of Contents

[1. Introduction 4](#_Toc373854430)

[2. What We Did 4](#_Toc373854431)

[2.1. Preparation 5](#_Toc373854432)

[2.1.1. Team Forming 5](#_Toc373854433)

[2.1.2. Methodology Brainstorming 5](#_Toc373854434)

[2.2. Software Development Process 6](#_Toc373854435)

[2.2.1. Product Backlog 6](#_Toc373854436)

[2.2.2. Sprint Iteration 7](#_Toc373854437)

[3. Results 9](#_Toc373854438)

[3.1. Results and Discussion 9](#_Toc373854439)

[3.2. Key Learnings 11](#_Toc373854440)

[Appendix A – Background Research 12](#_Toc373854441)

[Scrum Process 12](#_Toc373854442)

[AI Search Algorithms 12](#_Toc373854443)

[Appendix B – Website Reference 12](#_Toc373854444)

[Website URL 12](#_Toc373854445)

[Screenshots 12](#_Toc373854446)

[Appendix C – Detailed Project Reports and Logs 13](#_Toc373854447)

[Scrum Task Board 13](#_Toc373854448)

[Sprint Schedule and Results 14](#_Toc373854449)

[Activity Logs 15](#_Toc373854450)

**List of Figures**

[Figure 1 - Main Search Page 12](#_Toc373854451)

[Figure 2 - Search Page with Results 13](#_Toc373854452)

[Figure 3 - Scrum task board on Trello 13](#_Toc373854453)

[Figure 4 - Detailed Information of a task 14](#_Toc373854454)

[Figure 5 - Project Schedule 14](#_Toc373854455)

[Figure 6 - Diagram: Total effort per team member 15](#_Toc373854456)

[Figure 7 - Diagram: Estimated & actual hours per sprint 15](#_Toc373854457)

[Figure 8 - Diagram: Commit by author 16](#_Toc373854458)

[Figure 9 - Diagram: Commits by sprint 16](#_Toc373854459)

[Figure 10 - Diagram: Average commits by day 17](#_Toc373854460)

**List of Tables**

[Table 1 - Team member competency 5](#_Toc373854461)

[Table 2 - Scrum Team Structure 6](#_Toc373854462)

[Table 3 - Product Backlog 6](#_Toc373854463)

[Table 4 – List of sprints 8](#_Toc373854464)

# Introduction

Public transportation route planning is always a substantial problem for any metro areas. Even in a small city such as College Station with only one bus system provided by the Texas A&M University Transportation Service, this remains a problem. Currently, the Transportation Service website only provides the daily schedule. To find out how to get from one point to another, one must know where they are currently located on the map to figure out which bus route they need to take. The process becomes even more complicated if they need to take more than one bus to get to the final destination. They have to manually retrieve different bus routes’ schedules to find out what their options are, what the total time is, and how long they have to walk between bus stops or wait at a bus stop between buses. This type of problem interests our group. Via this project, we provide a solution to eliminate this manual process and to also always guarantee to find the optimal results based on the users’ preferences.

Our product consists of three components: a crawler which nightly crawls the bus schedule from Transportation Service website, a route planning engine which takes users’ input from the user interface and combines with prior knowledge about the bus schedules to produce the results, and a web interface for user interactivities. These three components can be independently developed. Therefore, we tasked members of our team based on each person’s skills and interests. We used Python and PyCharm as the programming environment for our back-end tasks, PHP and PHPStorm for our web user interface tasks, and also Symfony2 and Flask to create the web service interface so that the Python back-end can communicate with the PHP front-end. Our product also offers an interactive way for the user to select their locations such a map to click on, the drag and drop ability to re-select, via the means of using Google Map API and Twitter Bootstrap.

In this project, since we only had 6 weeks from prototyping to developing and deploying our product, we decided to choose agile development methodology with scrum model. Our goal is to achieve a process that can support fast-paced development and easy checking on the progress. Each sprint consists of a number of tasks selected as a result of the collaboration between the product owner and the development team. These tasks need to be finished with complete testing so that at the end of each sprint, we can have a workable product with an increasing number of features over time. We decided that each sprint lasted in a period of two weeks with a number of 3 sprints in total.

Through this project, we expect to learn agile development process in action. Getting our hands on the real process with real tasks and real roles among the team members can give us a better picture on what it is and what we can benefit from it. Also, by tasking each member with things we have never done before, we want each to explore a different realm and learn different skill sets from each other. For example, a Python developer gets to learn about web service, a Java developer and a .NET developer get to learn about scripting language and rapid prototyping.

# What We Did

In this section, we will describe in detailed the things that we did, the process that we followed and the tools that we used during the project, from the preparation to development and release.

## Preparation

### Team Forming

Our team consists of four members. Each member self-evaluated his/her programming language competency as follow:

Table - Team member competency

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **C/C++ - CGI** | **Java - JSP** | **C# - .NET** | **PHP** | **Python** | **Ruby** |
| **Dung Dam** | 1 | 2 | 2 | 5 | 4 | 3 |
| **Anh Nguyen** | 3 | 1 | 0 | 0 | 3 | 0 |
| **Duc Nguyen** | 0.5 | 3.5 | 2.5 | 1 | 1.5 | 0 |
| **Van Trinh** | 3 | 1 | 4 | 0 | 3 | 0 |

The programming language competency increased from level 0 to level 5 with level 0 denoted that he/she never used that programming language before and level 5 denoted his highest competence in that language.

This evaluation table helped us to decide which technologies and platforms should be used for this project.

### Methodology Brainstorming

#### Scrum Process

In this project, we chose Scrum process as our development methodology. Scrum is an iterative, incremental process for developing any product or managing any work. Scrum produces a potentially shippable product at the end of every iteration. In Scrum, you break a task into a series of iterations, called Sprints, which get you ever closer to the goal. The team acts in a self-empowered and coordinated manner to ensure investment of a common goal and commitment to that goal.

There are several reasons why we chose Scrum. First, we all would like to learn more about Scrum with hands-on experiences. Second, our team and our project have characteristics matching with ones required by Scrum.

* Small teams: Scrum flourishes with small, face-to-face teams of 12 or fewer members. In our team, there are only four members.
* Small project: Large projects may not fit the Scrum methodology, which are targeted to smaller teams.
* Iterative fashion: Scrum works in an iterative fashion, in successive cycles, without an overall and upfront, must-do plan. In this project, we wanted to build our system with a few features first and increasingly added new features. We did not need to consider all aspects or plan all features of our project from beginning to end before the project even started.
* Collocation: In Scrum, this term indicates a preference that everyone is in the same location. This is also applicable to our team.
* Key personal quality: In addition to programming skills, we self-assessed that we possessed key qualities required for a Scrum team such as being experienced, motivated, committed, responsible, and independent programmers, as well as being team players.

#### ****Scrum Team****

A Scrum Team consists of three components: the Scrum Master, the Development Team, and the Product Owner. We assigned roles as below:

Table - Scrum Team Structure

|  |  |  |  |
| --- | --- | --- | --- |
| **Role** | **Description** | **Members** | **Note** |
| Scrum Master | A person who is responsible for implementing Scrum methods, values, and practices. He keeps the project on track and moving forward. | Dung Dam | He has experiences with Scrum process when working in his previous company |
| Development Team | The Development Team consists of professionals who actually implement tasks. | All members |  |
| Product Owner | The Product Owner represents for the customer. He knows what the customer wants for the product under development to do. | Anh Nguyen,  Duc Nguyen,  Van Trinh |  |

Though Scrum prefers separated people hold different roles, in this project we decided that all team members would hold the roles of the Development Team as well as the Product Owner. As the Product Owner, all members would be able to discuss and contribute their ideas about the features of the system.

## Software Development Process

### Product Backlog

At the first step, we, in the roles of the Product Owner, wrote down all requirements we can think of for our product. These consist of functional and non-functional requirements. They go into a Product Backlog in Scrum process. The items in Product Backlog are ranked in order based on both business values and development effort. Tying the Product Backlog and the business value of each listed item is the responsibility of the Product Owner. The estimated effort to complete each backlog item is, however, determined by the Development Team. This Product Backlog is almost always more than enough for a first sprint. It then can grow and be changed as more is learned about the product and its customers. Top-ordered Product Backlog items drive immediate development activities. The table below shows our product backlog items with 1 being the highest priority.

Table - Product Backlog

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Name | Description | Priority |
| 1 | Search shortest duration routes with multiple results | Given a user’s start time, starting point and ending point, the system will return multiple results which include bus routes that the user can take based on the shortest duration criteria. | 1 |
| 2 | Search earliest arrival time routes with multiple results | Given a user’s desired arrival time, starting point and ending point, the system will return multiple results which include bus routes that the user can take based on the earliest arrival time criteria. | 1 |
| 3 | Displays multiple results | Display multiple results in graphic user interface, display details when user clicks on a result. | 1 |
| 4 | Search place by name | User inputs a name that they want to find. The system will return all places that match that name. | 1 |
| 5 | Highlight bus stops on map | Highlight bus stops on map with marker. | 1 |
| 6 | Web service for communication between back end and front end | Provide a web service that allows the back end and the front end to communicate with each other. | 1 |
| 7 | Sort search results | Sort search results based on one criteria. | 2 |
| 8 | Responsive design | The system will change User graphic interface based on which devices are used to access the system, for example, displaying in different resolution depending on if the device is a smartphone or a tablet. | 2 |
| 9 | Auto detect user’s current position | Auto detect user’s current position based on GPS signal or network signal. | 2 |
| 10 | Highlight routes on map | Highlight segments of routes on map. | 3 |
| 11 | Store previous search results | Store previous search results for a user. | 3 |
| 12 | Compare 2 search results | Compare two search results for a user. | 3 |
| 13 | Intuitive search results | Intuitive search results. | 3 |
| 14 | Real-time tracking & re-planning | Real-time tracking & re-planning. | 3 |
| 15 | Suggesting nearby restaurants/bars/… | Suggesting nearby restaurants, bars, etc. along user’s bus route. | 4 |
| 16 | Real-time bus tracking | Track bus’s position in real-time manner. | 5 |

We decided that this list would be re-evaluated after each sprint to add new features or remove existed features if needed.

After the creation of Product Backlog, Scrum team will iterate through Sprints.

### Sprint Iteration

#### Sprint Planning

We started a Sprint with a Sprint Planning Meeting, in which we made a plan of the work to be performed in this Sprint. In this face-to-face meeting, the plan is usually created based on the collaborative work of the entire Scrum Team. The meeting includes two parts:

* Part One: determined what would be possibly done in this Sprint. In this part, we, in the role of *Product Owner*, focused on selecting the high-priority backlog items from the Product backlog, moving them to the Sprint backlog, and defining the Sprint goal(s).

Table – List of sprints

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Sprint** | **Duration** | **Sprint backlog** |
| 1 | Sprint 1 | 21/10 - 03/11 | Search earliest arrival time routes with multiple results  Display multiple results  Highlight bus stops on map  Search place by name |
| 2 | Sprint 2 | 11/04 – 11/17 | Search shortest duration routes with multiple results  Responsive design  Sort search results |
| 3 | Sprint 3 | 10/18 – 12/01 | Daily update bus route data  Data analysis and processing  Store previous search results |

* Part Two: estimated backlog items. First, each team member considered how many hours he or she would be able to contribute to this sprint. Then, we, in the role of *Development Team*, estimated how much time should be spent to complete a task, and who would be in charge of that task. We started with the highest-priority tasks first, estimated the time it would take, and then continued with the lower-priority tasks. In this part of Sprint planning, we also discussed about the solution for such tasks involving data structures, algorithms, technologies, platforms, and programming languages. From then until the next sprint, there would be no change in the sprint requirements.

To manage the tasks in team, we used Trello, a collaboration tool that organizes projects and tasks in a board, which is based on the idea of Kanban board. Tasks are divided into sections including: *Product Backlog*, *Known Issue*, *Defining Requirements*, *To-do*, *In-progress*, *Testing*, and *Done*. In this meeting, we updated each task with its requirement, time estimate, and persons in charge. All team members updated the status of the cards frequently so that at any point in time, we could easily sum up the total hours of effort spent, and estimate the total work remaining. (See Figure 5- Scrum task board on Trello in Appendix A).

#### Daily Scrums

We decided that we would work together twice per week, on Wednesday (from 17:00 to 19:00) and Saturday (from 13:00 to 17:00), at Evans library. At the beginning of each day, we spent 15 minutes on reporting the current progress. Each team member reported on what we had done in the previous days, what we would be working on until the next meeting, and what roadblocks we were facing. After that, it was the time for development.

#### Development

##### CODING

* For revision control, we used GitHub, a web-based hosting service for software development projects that uses the Git revision control system. This tool helped us easily manage code changes of individuals and the whole team as well as effectively merge code from different branches. Before working on a specific task, each member pulled the latest source code from GiHub. Then, after completing a task and getting code review, we committed and pushed the new source code back to GitHub.
* Pair Programming: In this technique, two team members usually work together at one workstation. One, the driver, writes code while the other, the observer or navigator, reviews each line of code as it is typed in. The two members switch roles frequently. We used this technique in complex features such as building the bus route search algorithm.
* Team members logged actual spent hours into a shared Google spreadsheet (see Figure 5 - Project Schedule in Appendix B). That information was used for assessment and time estimate for future tasks.

##### COMMUNICATION

Communication is a crucial part of Scrum process. For internal communication, we used Facebook chat and Google hangout, which is free and easy to use for online discussion and remote working. To reach out to the external source, we chose email as the main communication tool. For example, to obtain the bus schedules, geo data of all bus stops, and so on, we sent emails to the Transportation Services, kindly requested for those information.

##### TESTING

Each member was responsible for unit-testing each functions he or she ever developed, including writing and running the tests. Then, we performed an integration test when integrating the back-end to the front-end via a web service. Finally, we carried out a performance test. It started with small data and simple cases, then with heavy data and complex cases.

##### DEPLOYMENT

We made our system run on Amazon Web Services, in which the frontend website and backend were deployed to Amazon Elastic Compute Cloud (Amazon EC2), and the database was put in the Amazon Relational Database Service (RDS). The system can be accessed at http://aggiehack.dngdiary.com

#### End-Of-Sprint Review

Sprint Review: concerned with “what was done” in a Sprint. Each task is enumerated, along with its acceptance criteria, to see if those criteria have been met. We ran a demo of the tasks that had been chosen to work on and determined if each was done or not. Those were not done would be pushed back to the Product Backlog.

Sprint Retrospective: captured on “how we did” in a Sprint. All team members discussed about the good and the bad of the last Sprint and suggested our solutions for improvements.

# Results

## Results and Discussion

Our goal for this project is to get a workable product satisfying all specified requirements, and a fast-paced development process based on agile methodology with scrum model. Following are the results of our team’s effort in 6 week period and a discussion of them.

Our workable product is a website which offers an interactive way for users to input their start and end locations, their expected leaving time, and get back the optimal routes based on the earliest arrival time criteria (see Figure 2 - Search Page with Results in Appendix A). The users can click on the map to select the starting and ending points. Also, they can drag and drop the icons to re-select different points on the map. After searching through all of the possible routes one can take, the search engine will return a result of top 5 routes. Each route can be a combination of different trips in such a way that in each trip the user can take a different means of transportation. For example, to get from A to D, the user has to walk from A to B, then take bus 12 from B to C, then walk to C, and finally take bus 8 from C to D. We limit our results to five possible best routes. For each possible route, we aggregate to a total time it would take, and also display the total waiting time and walking time so that the user can have a better idea of how their options look like.

The users can choose to save their search results to be able to retrieve them at a later point in time. These results are saved as long as the browser is open.

To support that, all of the back-end tasks also needed to be done ahead of time such as importing all geo-data of all bus locations from the data provided by Transportation Service and scheduling the nightly routine to extract the bus schedule from the Transportation Service website. Our web page also has mobile-friendly features with responsive design on three types of device such as PC, tablet, and mobile.

The only thing we did not really achieve in this phase of our project is the performance of the system. We implemented A-star search algorithm with Yen’s approach to find k-shortest paths. However, with a few hundred nodes and thousands of bus times during the day, our algorithm requires a wide range from a split of a second to over 15 seconds to run, depending on the hardness of the routes involved. The performance can definitely be improved with different techniques which we have already had ideas on how to implement. We saved these items to our product backlog for a later phase.

As for the process, we all agree that scrum model has worked well for a rapid prototyping and developing a product that evolves through time. Our scrum consists of 3 sprints, managed by a Scrum Master. Each cycle in the iterative process involves backlog refinement, sprint planning, developing and testing with daily scrum meeting incorporated in, and finally a sprint review and retrospective meeting.

Daily meetings are to ensure everyone is on the same page before continuing on our development process. It also allows us to quickly and effectively communicate with each other about difficulties. If a change in direction is needed, quick meeting allows this to happen without any delay in the process. This is proved to be an advantage of agile development when our team had to change our take on the algorithm implementation several times to finally come up with a working one with reasonable performance.

Besides daily meeting, pair programming has been a great help in producing high quality code. With pair programming, we tended to consider more design alternatives and arrive at simpler, more maintainable designs. The code produced was more error-prone, in a faster rate. However, it took much more effort than the amount of effort one programmer could spend to solve the problem alone.

To manage our product backlog and sprint iterative development process, we used an online tool called Trello as mentioned above. This product supported us well in this project with its clean and easy-to-use interface. It allows us to create, edit cards, push them to the correct list, and display the lists in color code in such a way that at any given time, the developers and possibly the managers can obtain the current status of the project in a glance (see Figure 3 - Scrum task board on Trello in Appendix A). However, one disadvantage this tool has is that it does not provide a way to take a snapshot of the status of all tasks associated to this sprint. We need this feature so that at the end of each cycle, we can have a history of what have been done in the current sprint before starting to move the cards around in preparation for the next sprint.

The sprint review meeting and sprint retrospective meeting being combined into one worked well for us since we did not have a lot of time like in a real software development process. Another thing that worked for us is in this meeting, we were able to give suggested solutions for all of the reasons of failure in the previous sprint so that we can improve in these aspects in the next sprint. This too has been proven to work well.

What we did not do well in this process is to come up with a time estimate for some tasks in which under-estimation cost us the incompletion of them at the end of the sprint. Also, meeting is another time consuming factor in which we did not do a very good job in managing our meeting time with respect to the scope of this project.

## Key Learnings

Through this project we had a chance to practice agile development process, which is a quick and effective way to achieve a workable product where detailed requirements are defined on the go. We have learned how to do things in scrum-fashion. We have learned what makes a sprint failed, and how to correct our mistakes. We have testified that scrum model worked well for small team and also encouraged a very high work ethics. However, we need to do a better job in task estimation and meeting time management. If we were to develop another product that evolves through time, scrum model is the one to follow. As for the technical skill development, each of us has successfully stepped out of our comfort zones and tried to learn new things. We also played different roles in the development process such as developers, testers, and product owner. That allows us to acquire a variety of skills in different fields, adding valuable experiences to our skill sets.

Appendices

Appendix A – Background Research

Scrum Process

* Scrum Alliance: <http://www.scrumalliance.org>
* The Scrum Guide: [https://www.scrum.org/Portals/0/Documents/Scrum Guides/Scrum\_Guide.pdf](https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum_Guide.pdf)
* Kanban board: <http://en.wikipedia.org/wiki/Kanban_board>

AI Search Algorithms

* A-star: Artificial Intelligence: A Modern Approach (3rd Edition)
* Yen’s algorithm: <http://en.wikipedia.org/wiki/Yen's_algorithm>

Appendix B – Website Reference

****Website URL and Source Code****

**Website of the project is available at:** <http://aggiehack.dngdiary.com>

Source code is available at: <https://github.com/dungdt88/aggiehack>

****Screenshots****

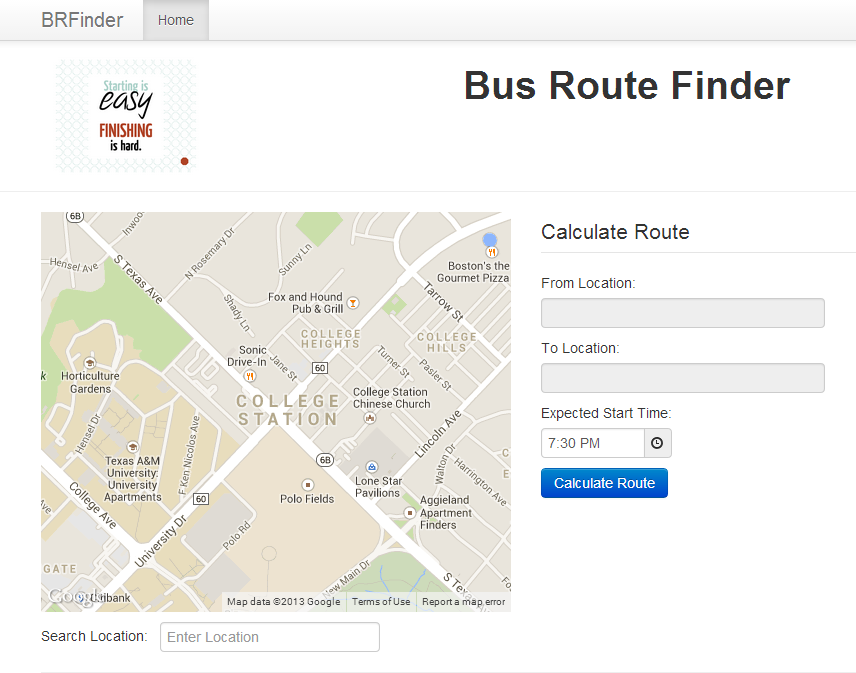


Figure - Main Search Page

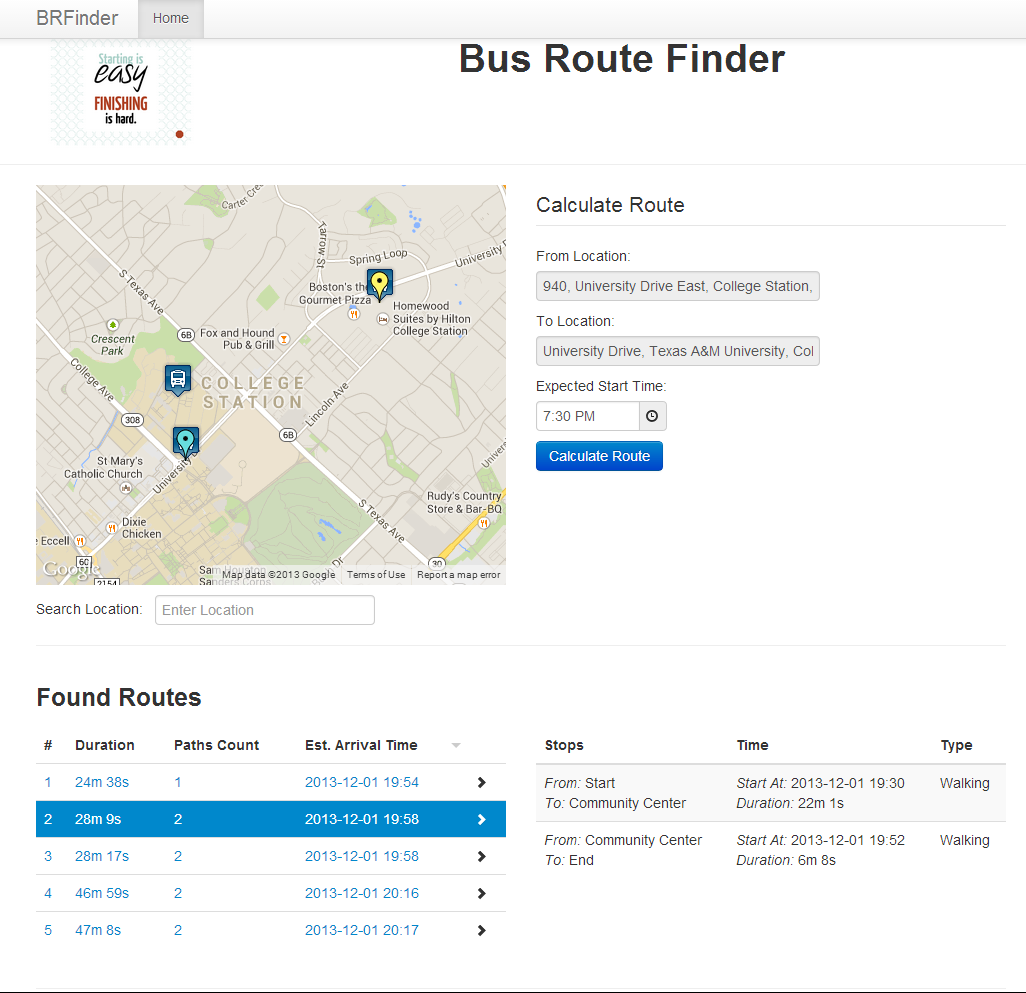


Figure - Search Page with Results

Appendix C – Detailed Project Reports and Logs

Scrum Task Board

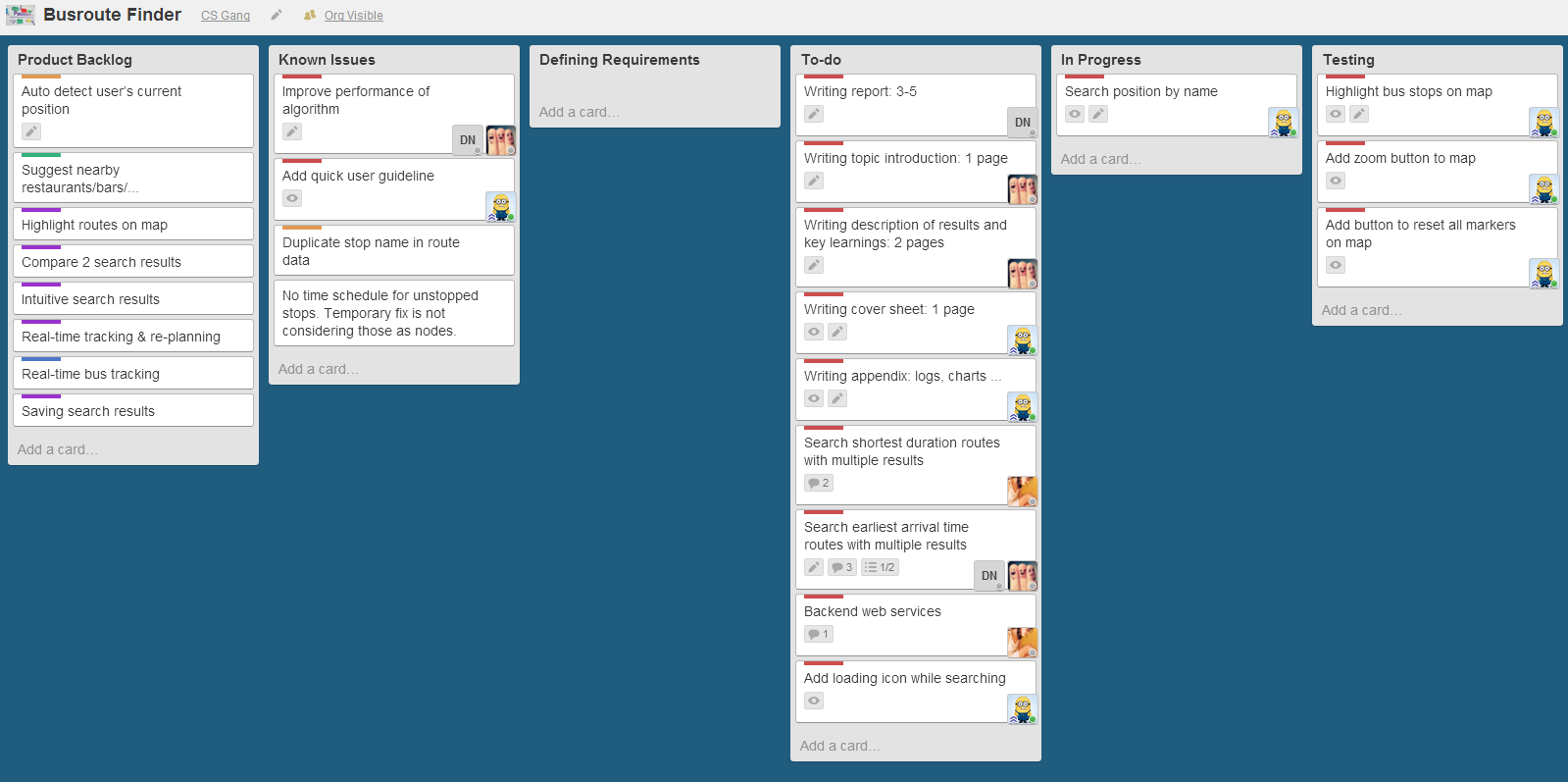


Figure - Scrum task board on Trello

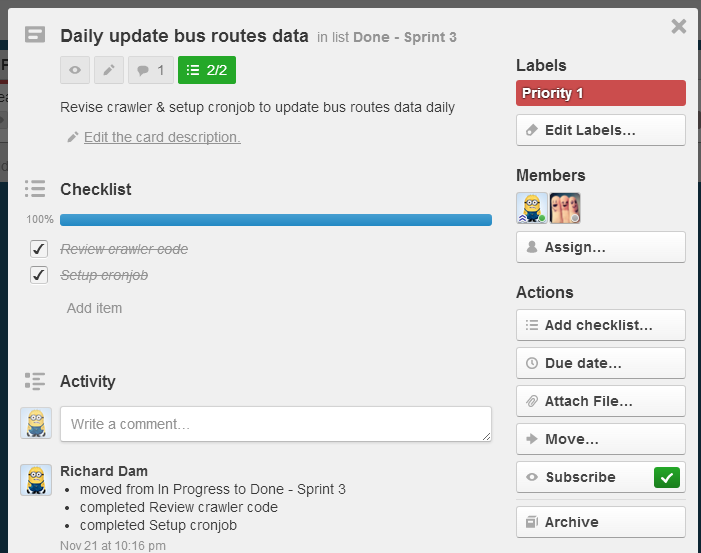


Figure - Detailed Information of a task

Sprint Schedule and Results

Due to the low availability of team members during the development, we cannot make the schedule with fixed hours and deadline for each task such as Gantt chart. Instead, we use a simple spreadsheet where team members can estimate the hours they can spend in each sprint, and make the schedule based on the availability of each member and estimate of each task. The screenshot below shows part of our schedule:

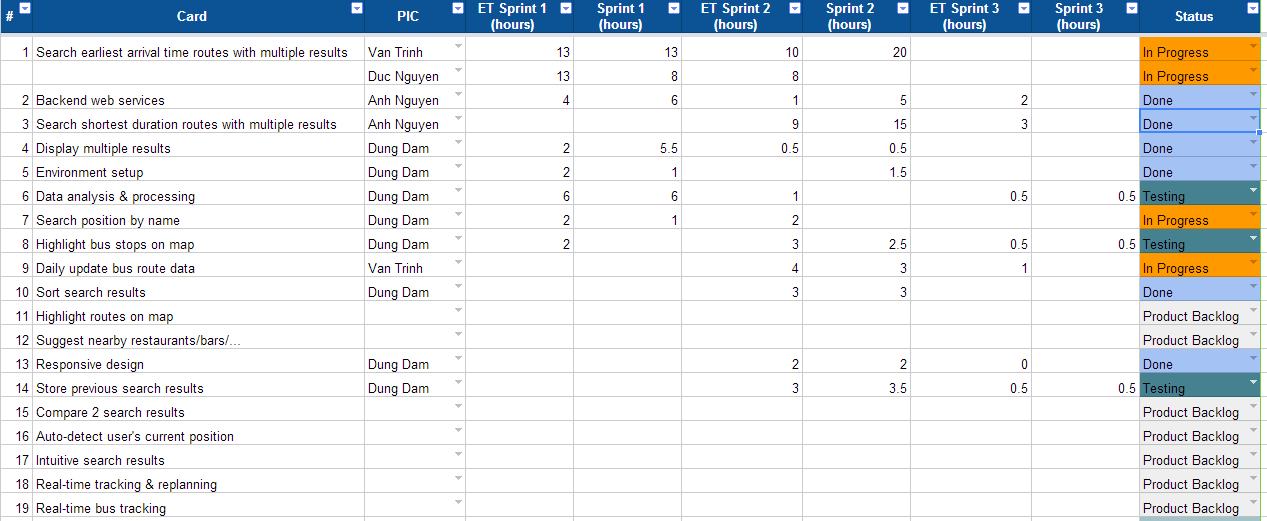


Figure - Project Schedule

The overall effort of the team can be built from the hours that each member log into the spreadsheet after they finished any task.

Figure - Diagram: Total effort per team member

Also, the estimated hours and the actually spent hours in each sprint can be extracted easily from the spreadsheet:

Figure - Diagram: Estimated & actual hours per sprint

Activity Logs

The diagrams below are extracted from Github, highlighting the overview of the coding activities during the development of project.

Figure - Diagram: Commit by author

Figure - Diagram: Commits by sprint

Figure - Diagram: Average commits by day