Software Project Management Plan

Shuttle Tracker and Ride Request Service (STARRS)

8 Oct 2017

Team Members

Will Gross
Zhuofan Zhang
Zilin Chen

Document Control

Change History

Revision	Change Date	Description of changes
V1.0	10/08/2017	Initial release
V1.1	12/10/2017	Modified the document to follow the comments.

Document Storage

This document is stored in the project's SVN repository at:
https://github.com/WillGross/STARRS/wiki/PMP

Table of Contents

List of Tables

1 Overview

1.1 Purpose and Scope

The purpose of this project is to create more efficient system for managing college provided transportation that is user friendly for both students, drivers, and dispatch. In its current state, Colby transportation, particularly concerning the shuttle and jitney, is manually managed. For the shuttle, this creates three particular inefficiencies. The first is that the shuttle driver is responsible for recording the trip data in addition to driving. The second is that the data collected has to be manually transferred from hard copy to digital to be analyzed. The third is that ridership relies on knowledge of and trust in the schedule by the riders, which results in reduced use when riders don't know the schedule and can't verify the status of the shuttle. The Jitney, while sharing the problems of driver recording and manual data entry, has many more problems of its own. The main issues are user knowledge of operation, unknown wait times, and an interface that is limited to calling in requests.

The scope of this project includes multiple iterative releases of this software with expanding functionality. The final deliverables will be an interface for students to know the locations, destinations, and arrival times of the shuttle and Jitney, as well as requesting the Jitney, a similar interface for dispatch with the added functionality of being able to remove requests, an interface for Jitney drivers to allow them to view and accept queued requests, and a database with ride information collected from gps data and ride requests.

1.2 Goals and Objectives

Project goals:

1. Reduce work done by drivers to log data
2. Increase shuttle ridership
3. Reduce data entry for analysis of ride data
4. Reduce transportation related calls to security
5. Improve rider experience by keeping them informed

Project objectives:

1. Create a student accessible web page with a real time map of the Jitney and shuttle's location, links to the Jitney schedule and the ride request form, instructions for canceling a request, and a list of requests currently in queue.
2. Create a security accessible web page with a real time map of the Jitney and shuttle's location, a link to the ride request form, a list of requests currently in queue, and a option to remove queued requests.
3. Create a driver accessible interface (web page, then app) that uploads driver GPS location to the real time map, presents the driver with a list of pending requests, and allows the driver to accept requests.

4. Create a database to log when the shuttle departs from and arrives at its stops.
5. Create a database to log when Jitney requests are made, the origin and destination of the request, the number of people the ride is requested for, the time of pickup, and the time of drop-off.

1.3 Project Deliverables

The following items will be delivered to the customer on or before 11/15/17:	
Version 1.0	A web page with a map displaying the shuttle route, stops, and real time location data of the shuttle.
Version 1.1	The addition of a link from the web page to a live updated schedule of Jitney operating hours.
Version 1.2	The addition of arrival time estimation to the shuttle map.
The following items will be delivered to the customer on or before 12/18/17:	
Version 2.0	This will deliver a ride request form accessible from the map web page, a database to keep track of rides and requests, a dispatch version of the page that will also allow for the removal of requests, and a web page for jitney drivers that broadcasts their gps location and allows them to view and select from a list of pending requests.
The following items will be delivered to the customer on or before 3/1/18:	
Version 2.1	This update will add a list of pending ride requests to the map webpage, as well as the Jitney's current destination and estimated time to arrival to the map.
The following items will be delivered to the customer on or before 5/10/18:	
Version 3.0	The driver webpage interface will be replaced with an app interface. All of the same functionality will transfer over, allowing the driver to keep the app open and running in the background instead of having to keep the webpage open.
Version 3.1	This will be a complete overhaul to the driver interface. Instead of only being a list of active rides and rides waiting, the app will present a graphical navigation interface with destinations selected from pending requests. This will allow

	drivers to stay in the app and not having to coordinate between two apps for ride requests and navigation.
Version 3.2	This will publish the route chosen by driver navigation to the map webpage.

1.4 Assumptions and Constraints

Assumptions:

1. The GPS tracker hardware for the shuttle will be compatible with the shuttle and will give us useful data.
2. Students and Security are willing to switch over to the new system.
3. Jitney drivers will always have GPS-enabled phones with browser web access.
4. Web pages can dynamically access and broadcast a device's GPS location.
5. A dynamic database can be created to serve as a background for the user interfaces and to log data history.
6. Web page to database, app to database, and database to web page and app communication can be achieved.

Constraints:

1. The different user interfaces must limit access to only its intended user.
2. No extra work shall be given to dispatchers.
3. Web interfaces must be accessible to all users regardless of platform, and app interfaces must be iOS- and android-compatible.
4. All work must be done by the end of the 2018 academic year.

1.5 Success Criteria

- Driver paperwork is eliminated with the exception of number of riders in the shuttle.
- Manual data entry is eliminated with the exception of number of riders in the shuttle.
- Calls to Security regarding the Jitney and the shuttle are reduced.
- Students are more aware of the Jitney operation schedule.
- Students have realistic expectations about Jitney and shuttle arrival times.
- Students recognize and begin to use this service to request the Jitney.

1.6 Evolution of the Project Plan

Before and after each version release, the project plan will be reviewed and updated to reflect any changes in timeline, changes in scope, changes in deliverables, and the actual effort and outcome of the previous release.

2 Project Organization

2.1 Interfaces

In the project, all the features facing the users are presented by either a webpage or a phone application, depending on possible project work time. To student users, the interface would contain a map interface to display useful information such as, but not limited to, vehicle location, vehicle route, vehicle movement, driver shift, etc. It should also display the current user queue and the estimated pickup time. To drivers, the interface should also include option to pick up the next user in queue. To the dispatcher, the interface should also include methods to manage the current queue, and access to driver work data.

The product would be produced by students in Group 3 of CS397, and later operated and administered by the Security Office when in use. Group 3 would undertake all coding, testing, management, and deployment processes, while maintaining interactions with the Security Office for requirement specifications, validations, and verifications.

2.2 Roles and Responsibilities

Some of the major implementation activities included in the project are building the interface (webpage or application), enabling real-time map through Google Maps API and on-board GPS device, and building the internal design to enable the queue and the control system.

In the project, Group 3 will fill in most of the documentations and coding work, including design, implementation, management, testing, and deployment. Colby Security Office can provide any resource if needed.

3 Startup Plan

3.1 Team Organization

Team Lead: Will Gross.

The team lead is the main point of contact for the customers. As such they will relay the customer requirements to the team, develop the project design and plan with input from the team members, and oversee the general progress.

Developers (3): Will Gross, Zilin Chen, Zhuofan Zhang.

The developers are in charge of technical research, coding, and testing. They will focus on delegated tasks and work toward the completion of one version at a time.

3.2 Tools

- Programming Language – (V1-V2) JavaScript, HTML, (V3) Corona SDK
- Version Control – Github repository at https://github.com/WillGross/STARRS

- Geolocation and distance/time calculations - Google Maps API
- Scheduling - Projectlibre

4 Work Plan

4.1 Activities and Tasks

1		**STARRS**	154 days	**9/25/17 8:00 AM**	**4/26/18 5:00 PM**	
2		**Version 1: Shuttle Tracker**	94 days	**9/25/17 8:00 AM**	**2/1/18 5:00 PM**	
3		**Version 1.0: Basic Tracking**	94 days	**9/25/17 8:00 AM**	**2/1/18 5:00 PM**	
4	✔	Get the google maps API	9 days	9/25/17 8:00 AM	10/5/17 5:00 PM	
5	✔	Create Github Repository	4 days	10/2/17 8:00 AM	10/5/17 5:00 PM	
6	✔	Determine gps system for shuttle	9 days	9/25/17 8:00 AM	10/5/17 5:00 PM	
7	✔	Create HTML document	7 days	10/6/17 8:00 AM	10/16/17 5:00 PM	5
8	✔	Initialize map	3.875 days	10/17/17 9:00 AM	10/20/17 5:00 PM	4;7
9	✔	Add Stops to map	3 days	10/23/17 8:00 AM	10/25/17 5:00 PM	8
10	✔	Overlay route onto the map	17 days	10/23/17 8:00 AM	11/14/17 5:00 PM	8
11	✔	Create animated marker for Shuttle	18 days	10/23/17 8:00 AM	11/15/17 5:00 PM	8
12		Match shuttle animation to shuttle schedule	19 days	11/16/17 8:00 AM	12/12/17 5:00 PM	11
13	✔	Add link to live jitney schedule, shuttle schedule, and transportatior	22 days	11/16/17 8:00 AM	12/15/17 5:00 PM	7
14	✔	Add a time till arriaval function for the shuttle	10.875 days	11/27/17 9:00 AM	12/11/17 5:00 PM	7
15		Secure website space	17.875 days	11/16/17 9:00 AM	12/11/17 5:00 PM	
16		Secure server space	17.875 days	11/16/17 9:00 AM	12/11/17 5:00 PM	
17		Create a database on the server	6 days	1/6/18 8:00 AM	1/15/18 5:00 PM	16
18		Create a location reporter for a remote device to log gps informatic	2 days	1/16/18 8:00 AM	1/17/18 5:00 PM	17
19		Interface the main page to detect and display new gps data as shu	6 days	1/18/18 8:00 AM	1/25/18 5:00 PM	18
20		Incorperate Colby themes and formatting through wordpress	6 days	1/6/18 8:00 AM	1/15/18 5:00 PM	15
21		Get final approval from communications, ITS, and Security	4 days	1/26/18 8:00 AM	1/31/18 5:00 PM	20;19;14;13
22		Publish V1.0	1 day	2/1/18 8:00 AM	2/1/18 5:00 PM	15;21
23		**Version 2: Jitney Request**	19 days	**2/2/18 8:00 AM**	**2/28/18 5:00 PM**	**2**
24		**Version 2.0: Basic Requesting**	13 days	**2/2/18 8:00 AM**	**2/20/18 5:00 PM**	
25		Create requestor site shell	4 days	2/2/18 8:00 AM	2/7/18 5:00 PM	
26		Create dispatch site shell	4 days	2/2/18 8:00 AM	2/7/18 5:00 PM	
27		Control access to dispatch	3 days	2/8/18 8:00 AM	2/12/18 5:00 PM	26
28		Create driver site shell	4 days	2/2/18 8:00 AM	2/7/18 5:00 PM	
29		Get phone gps data through site for Jitney	5 days	2/8/18 8:00 AM	2/14/18 5:00 PM	28
30		control access to drivers	3 days	2/8/18 8:00 AM	2/12/18 5:00 PM	28
31		Find way to store ride information from jitney and shuttle	3 days	2/2/18 8:00 AM	2/6/18 5:00 PM	
32		Link database to the interface pages	4 days	2/8/18 8:00 AM	2/13/18 5:00 PM	25;26;28;31
33		Add instructions for canceling ride to master page	3 days	2/8/18 8:00 AM	2/12/18 5:00 PM	26
34		Add link to main site to the requestor page	1 day	2/8/18 8:00 AM	2/8/18 5:00 PM	25
35		Populate the main map and dispatch with the Jitney GPS location	3 days	2/15/18 8:00 AM	2/19/18 5:00 PM	29
36		create easy acess to database information for ride analysis by secu	5 days	2/7/18 8:00 AM	2/13/18 5:00 PM	31
37		Publish V2.0	1 day	2/20/18 8:00 AM	2/20/18 5:00 PM	32;35;36
38		**Version 2.1: Add jitney information to map**	6 days	**2/21/18 8:00 AM**	**2/28/18 5:00 PM**	**24**
39		Add requests in queue to main page	5 days	2/21/18 8:00 AM	2/27/18 5:00 PM	
40		Add current ride destination to map	4 days	2/21/18 8:00 AM	2/26/18 5:00 PM	
41		Add time to current destination to map	3 days	2/21/18 8:00 AM	2/23/18 5:00 PM	
42		Publish V2.1	1 day	2/28/18 8:00 AM	2/28/18 5:00 PM	39;40;41
43		**Version 3: App based driver interface**	41 days	**3/1/18 8:00 AM**	**4/26/18 5:00 PM**	**23**
44		**Version 3.0: moving current driver functionality to an app**	18 days	**3/1/18 8:00 AM**	**3/26/18 5:00 PM**	
45		Create app shell	10 days	3/1/18 8:00 AM	3/14/18 5:00 PM	
46		Limit access to app to drivers	4 days	3/15/18 8:00 AM	3/20/18 5:00 PM	45
47		Relay gps location through app to map page and dispatch page	7 days	3/15/18 8:00 AM	3/23/18 5:00 PM	45
48		Comunicate to database through app	7 days	3/15/18 8:00 AM	3/23/18 5:00 PM	45
49		publish v3.0	1 day	3/26/18 8:00 AM	3/26/18 5:00 PM	47;46;48
50		**Version 3.1: adding navigation to the app**	19 days	**3/27/18 8:00 AM**	**4/20/18 5:00 PM**	**44**
51		Add map to app	8 days	3/27/18 8:00 AM	4/5/18 5:00 PM	
52		add gps logation to map	4 days	4/6/18 8:00 AM	4/11/18 5:00 PM	51
53		add direction finding capability	6 days	4/6/18 8:00 AM	4/13/18 5:00 PM	51
54		add time estimation	4 days	4/16/18 8:00 AM	4/19/18 5:00 PM	53
55		interface with accepted rides	6 days	3/27/18 8:00 AM	4/3/18 5:00 PM	
56		create interface to view and select from pending requests.	4 days	4/4/18 8:00 AM	4/9/18 5:00 PM	55
57		Publish V3.1	1 day	4/20/18 8:00 AM	4/20/18 5:00 PM	54;56
58		**Version 3.2**	4 days	**4/23/18 8:00 AM**	**4/26/18 5:00 PM**	**50**
59		Populate main map with information from driver navigation	3 days	4/23/18 8:00 AM	4/25/18 5:00 PM	53;54
60		Populate Dispatch map with driver navigation info	3 days	4/23/18 8:00 AM	4/25/18 5:00 PM	53;54
61		Publish V3.2	1 day	4/26/18 8:00 AM	4/26/18 5:00 PM	59;60

Table 1: Activities and tasks management plan.

4.2 Release Plan

Release 1: 9/25/17 - 11/15/17 Version 1.0-1.2

Release 2: 11/15/17 - 12/18/17 Version 2.0

Release 3: 2/5/18 - 3/1/18 Version 2.1

Release 4: 3/1/18 - 5/10/18 Version 3.0-3.2

4.3 Iteration Plans

Iteration 1: V1.0
This version implements basic shuttle tracking: It tracks the location of the shuttle and displays this information on a webpage on a map with a route overlay.

Iteration 2: V1.1
This version adds a link to the web page that connects to a live schedule of Jitney hours of operation.

Iteration 3: V1.2
This version adds a time-to-stop function for the shuttle, which gives the user estimated time till arrival for up to three stops in advance.

Iteration 4: V2.0
This version adds functionality for students, and other different functionalities for two user groups, the Jitney drivers and the dispatcher (inside Security Office). For the students, this adds the ability to request a ride from the Jitney and see where the Jitney is. For the drivers, this allows them to see the queue of waiting requests, select requests to accept, and not having to manually track their routes on paper. For dispatcher, this will reduce the calls to request a ride and reduce the data entry for manually tracked ride data, reducing responsibilities to canceling requests.

Iteration 5: V2.1
This version adds Jitney destination to the map, adds estimated time to arrival to map, and adds a section to the site that lists the current requests in the queue.

Iteration 6: V3.0
This version tries to make the system more convenient for drivers by moving their interface from a web page to an app.

Iteration 7: V3.1
This version adds a navigation feature to the app to allow drivers to get directions while staying active on the app.

Iteration 8: V3.2
This version adds the navigation data from the driver's app to the webpage map, filling in the current route and estimated time to destination.

4.4 Budget Allocation

The time budgeted for this project is from now until the end of next semester. The projected time cost of the project is roughly 116 days of work. Taking into account the time draw of other classes and our 3-person team size, we estimate that this 116 days worth of work will place late april as the release date for version 3.2 of STARRS.

5 Control Plan

5.1 Monitoring and Control

This section includes plans and procedures for tracking progress and controlling performance.

Weekly – Team meeting.
Project participants report status, progress, difficulties, potential problems and possible improvements.

10/8/2017 – Project Management Plan.
Determine the scope and an overall schedule for each task.

10/18/2017 – Prototype.
We create a prototype of our website, and show the website to the security, the communication department at Colby, and the ITS staff. We will record customers' expectations and requirements on the website design and adopt changes. During the meeting, we make sure that we have the correct version of the bus schedule to display on our website, and we will report the money we have spent on the GPS.

10/22/2017 – System Requirements Specifications.

10/31/2017 – First working system.
We have our first working system for shuttle tracking, and we will deliver to both the Security and ITS. After making sure that the shuttle tracking system meets the expectation of the Security, we will send the URL of our website to the ITS staff to set a link from myColby to our website.

11/3/2017 – Team meeting.

Group members make sure that the additional links that are added to the original website meet the expectation of the team leader.

11/5/2017 – Software Design Descriptions.

11/4/2017 – Updated working system.

Update to include the live jitney schedule, then deliver to Security, ITS, and the communication department.

12/17/2017 – Updated working system.

We deliver our working system for jitney tracking and request to the Security, ITS, and the communication department at Colby. And we will update our website at myColby page for testing.

12/6/2017 – Group meeting.

Group members report any difficulties, and we shall test our website together. Then we meet with the Security to demonstrate our system. We adjust the requirements and potentially get feedback from drivers about their further expectations about the system.

1/8/2018 – Group reunion.

The group members all get back to campus for Jan Plan. We discuss potentials to improve our system for jitney tracking and request.

2/1/2018 – Meet with Security.

We deliver our improved version of the jitney tracking system to the Security, and we will update our website at myColby page.

2/28/2018 – Group meeting.

We test our system for estimating the arrival time for Jitney.

3/1/2018 – Meet with Security

We will deliver our working system to Security and update our website at myColby page. Then we will discuss the potential of creating an APP for the system with Security and possibly the Jitney drivers. We will record the new requirements and expectations towards the APP.

4/13/2018 – Team meeting.

We will test our APP within the group. We fix any bugs on our website if necessary.

4/15/2018 – Team meeting.

We will deliver the working App to the Security and update our website if necessary.

5/7/2018 – Team meeting.

Group members report difficulties and potential changes to the APP. We test our APP system for navigation. We fix any bugs on our website if necessary.

5/10/2018 – Final product.
We deliver our final product to the Security, the Communication department at Colby, and ITS. We update our final version website at myColby page.

5.2 Project Measurements

Product and process measures support project management and estimation by analogy. At the beginning of a project, estimates are made for product size, project cost and delivery dates. During a project, progress is tracked with measures of actual effort, integrated lines of code and actual expenditures. Keeping track of estimates and actuals during a project helps to calibrate whatever technique is being used to make estimates. Storing project performance data on completed projects provides a rich source of data for estimating future projects.

Phase	Measurement
Release Planning	Shuttle tracking: 9/25/2017 - 10/31 Add Jitney schedule: 11/1 - 11/4 Shuttle Time to Stop: 11/5 - 11/15 Jitney tracking and request: 11/15 - 12/18 Estimation for Jitney arrival time: 2/5 - 3/1 App development: 3/1 - 4/15 Add navigation: 4/15 -5/10
Iteration Planning	Version 1.0: 10/31/2017 Version 1.1: 11/7 Version 1.2: 11/15 Version 2.0: 12/18 Version 2.1: 3/1/2018 Version 3.0: 4/15 Version 3.1: 5/5 Version 3.2: 5/10 Update effort estimates for product features Update estimated dates in release plan

Iteration Closeout	Record actual effort for scheduled tasks Record actual effort for product features Record LOC count for modules written
System Test	Record the rate at which errors are found.
Project Closeout	Archive project performance data in process database. (See process database definition for a list of measures to record.)
Ongoing	Record defects found from integration testing through first year of release. Assign each defect to one of the following categories: blocker, critical, major, minor or trivial. Keep track of the state of each defect: open, assigned, fixed, closed.

Table 2: Project phase and management plan.

6 Technical Process Plan

6.1 Process Model

The project follows an incremental model. The first main task would be to implement the map tracking system with GPS, and then each following increment would integrate the previous product with certain additional features.

At the initiation of the project, the team will gather necessary techniques such as JavaScript skills and Google Maps API tutorials, obtain hardware such as a GPS tracker with data collection and transmission functionalities, construct webpage for the online tracking system, as well as find the suitable platform to build the product. Those acquisitions can be attempted or even finished in parallel while generating other necessary reports and designs, before real coding begins.

In the process of each iteration, the team shall first do a summary of the previous increment, including newly acquired skills, common traps and fails, performance review, etc. An analysis of the requirements and goals, along with necessary adjustments based on feedbacks, for the current increment is also needed. Next, the group shall briefly make an estimate for the increment, select features that take a proper amount of time, and develop a general list for schedule and goals. During the process, risks and obstacles should also be considered and tackled with.

After those activities in analysis, the group would follow the order of design/modeling, implementation/coding, testing/managing, and deployment for the current increment. During the process, sufficient communication and documentation is required. If time allows, the group can also perform reports once in one or two days, to keep track of the progress. Group members are

encouraged to look at each other's progress, and design extra test cases for each other, if time allows.

At the termination of each increment, the project should be able to pass a real-world situation test. The finished product would be published and distributed for broader use, and also to solicit feedbacks for the next increment.

The followings are the milestones of the project:

1. The map tracking interface is successfully built and run on a webpage. (Version 1.0, initial release)
2. The website is functioning in full, including driver shift and time-to-stop information. (Version 1.2)
3. Pickup request queue and management system is finished (Version 2.0).
4. The application is built (Version 3.0).
5. All the planned features are successfully implemented (Version 3.2).

At the termination of the project, the product should have been thoroughly tested for both functionality and compatibility. The phone application and the web server should be ready to use, and also robust enough to face unexpected extremal situations.

6.2 Methods, Tools, and Techniques

The product would be developed mainly under an incremental method, so that each version includes some new features, which shortens the interval between releases, as well as collects feedbacks for adjustments in parallel with further development.

Different parts of the project would be developed in different languages: The map module from Google Maps API would be developed in JavaScript, as well as the website; the phone application would be developed in Corona; other functionalities of the product, such as the queue, would most likely be developed in Java.

During the development process, each module would be created and tested separately to ensure functionality. Except for the main body, each module would later be tested with the previous version, to ensure compatibility. We would also maintain a log document to store the results and data from each test.

The codes and documentations would be stored on online servers, notably GitHub, for proper version control. Any modification would also be recorded there. The final working product would also be web-based and rely on the Colby server for execution, operation, and data storage.

6.3 Infrastructure Plan

The project would mostly be developed in personal computers, as well as computers in labs of the Davis building, mostly under a mac OS system with suitable IDEs; the map function part would be developed online through Google Maps API, and a GPS device in hardware would be used to collect data, if needed. The project would mostly be using network service through Colby Connect, which is unlikely to fail for a time long enough to delay progress.

When actual testing is performed, the project would also need to access the shuttle and/or Jitney vehicle of Security Office.

7 Supporting Process Plans

7.1 Risk Management Plan

Considering the size of the project, and the technical experience and time restriction of the group members, we identify the following risks and rank them by priority:

- Estimation and Scheduling.

The workload from other courses tend to grow as the semester goes, as well as the pressure for finding a job/internship for all team members who are seniors, so any estimation being done currently is highly likely to fail. Plus, the team experience with designing software, building interface shells, and managing hardware and GPS gadgets is negligible, so it is hard to foresee potential technical difficulties that may arise and cause delay in the schedule.

Methods for managing risk includes setting more realistic goals for the project. For example, if adding route finder to the map is too complicated to be done in time, we can cut it from the project and focus on other core functions, so that the requirements can be met maximally. We can also use the incremental design methodology, and implement core functions before adding on extra features that the team are not too familiar with.

- Compromising on Designs

Because the time limit is tight, it would be more important to follow the planned designs and skip no necessary steps, so that no time is wasted on amending flaws. To reduce the risk of compromising, the team can schedule weekly meetings to go over what is done and what else is still needed. Team discretion helps finding omitted details, and making adjustment to current plans.

- Breakdown of Specification

Because team members are not very familiar with the technical aspect of the project, it is possible that some requirements conflict with each other in unpredicted ways. When facing this

kind of issue, we can choose to consult with experts in the field for suggestions and improvements, and also meet with the customer for requirement adjustments, if needed.

To monitor the above risks that can be foreseen, we can use weekly team meetings to exchange progress, choices, and opinions. Learning about the new technologies that are to be used before starting implementation can also increase our chance in spotting potential risks.

We have also identified some resources that we can use when facing risks. For example, we have contacted with the person who was responsible for developing the shuttle tracking website for Advance Transit, which operates shuttles around the Hanover, NH area; the person agreed to provide help if necessary.

7.2 Configuration Management Plan

1. All work products will be stored and managed through GitHub: https://github.com/WillGross/STARRS
2. Each file name should start with the name of the module that it belongs to, followed by some descriptive name of the file's function, e.g. map_locate, queue_functions.
3. When working on individual parts of the project, each member is expected to commit on their separate branch, so that management can be made easier. Team members are encouraged to propose test cases for each other, to spot potential bugs and edge cases. Merge should be granted only after a good amount of testing is done.
4. Sufficient comments should be included in each file, providing design choices and mechanisms for the file, and explanations on lines which might cause confusion.
5. Each version should occupy its own branch.

7.3 Verification and Validation Plan

Each increment should have its own validation method, mainly focusing on the functionality of its additional features and its compatibility with previous parts. Each test from previous increments should be saved and run again for double-check, while the new version also needs to pass its own tests. For versions where real-world interface is included, either dummy data or real-life testing should be fine for validation. For example, if the Google Maps API section is finished, then testing can be done either by entering dummy GPS data to the product, or by collecting location data with on-board GPS device. The final testing process can serve as demonstration.

Verification would be performed every time when the increments are enough for a new version publishment. Every time a significant new feature is added marks a milestone. The product will first be examined by the Security Office, and if they consider that the requirements are met

successfully, the product would be deployed for real-world using. User feedback can be used to adjust current requirements, and integrated into the next increment. If time commits, we can start the first verification with a prototype to gauge suggestions.