Deliverable 2

Project/ Group Name Pro TA

Nicolas Stencel, Keenan Jabri, Naumaan Hassan, & Manuel Vargas

1. Requirements Specifications

Functional Requirements

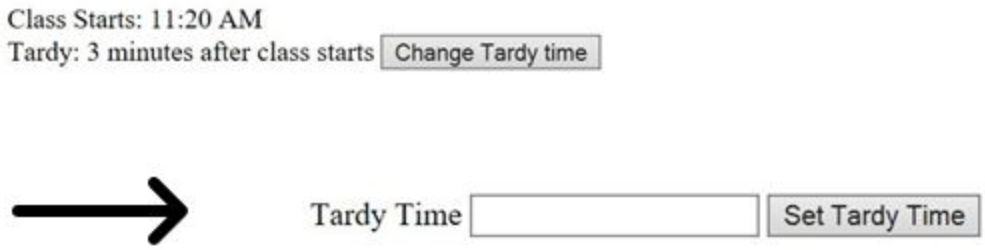
Requirement 1: Student Recognition

When a student swipes their UNT student ID card, the software should recognize them as part of the class that they're swiping into and mark them as present, tardy, or absent based on the time that they swiped in.

(The picture shows the general idea of the process that should occur. A student swipes their card through the card reader that's attached to the teacher's smartphone or tablet. Thereafter the student's attendance is marked)

Requirement 2: Attendance based on time of swipe

The software should know whether to mark a student as present, tardy, or absent based on the time that they swiped into the class. The software should also allow the teacher or whoever else is administering the class to change the tardy and absent time to whatever they please.



For example: The teacher changing the time of tardiness from 3 minutes after class starts to 5.

Requirement 3: Teacher going back to mark a student's absence as excused or unexcused

If a student doesn't show up to class, doesn't swipe in, and therefore is marked absent by the app software, the teacher should have the ability to mark it as an excused absence if it is later revealed that there was a legitimate reason for the student to miss class.

Vice-versa, the teacher should also be able to go back and mark an absence as unexcused if it turns out that the student's excuse was bogus for whatever reason.

Requirement 4: Software tracking entire semester of student attendance

At the end of a semester the teacher should be able to look at the student sheet for the class and see how many times a student was absent throughout the semester, and on which days. If a teacher decides a student's final grade based on the number of absences they have, the teacher should be able to do so. The software should also track if a student was tardy and by how much.

For example: If a student was late to class by a minute the software should read: "Tardy by: 1 minute"

Requirement 5: Add or delete students

Each semester there are students who are signed up for the class who do not show up on the official class roster. On the other side of that same coin there are also students that never show up to class and end up either failing or dropping it without ever setting foot in the room. Therefore, the teacher will have the ability to add or delete students from the app memory to make the process easier for everybody involved.

Non-Functional Requirements

Requirement 1: Performance

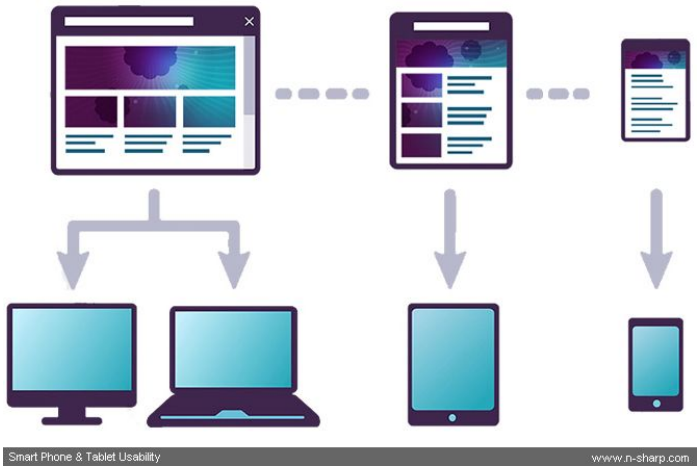
The performance of the application in terms of processing speed, from swiping the card to displaying that the user has been signed in is very important. Since we are trying to make this a fast process, if this takes a lot of time, it can create a large line and make it a distraction for other people in the class. The app should be able to work efficiently and quickly so that there's no hold up.

Requirement 2: Reliability

Since this application replaces a core function of class(taking attendance), it's imperative that it works reliably so that there isn't ever some sort of issue with the app. The teacher should feel secure in her choice of using the app by knowing that it won't ever just crash one day and stop working, or start working very slowly.

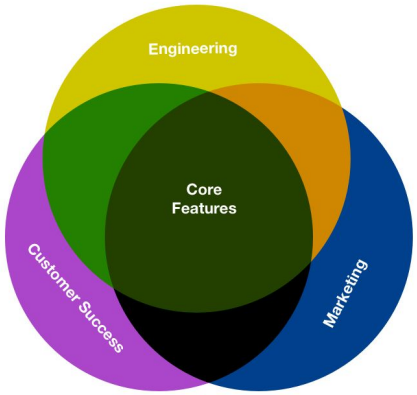
Requirement 3: Usability

As previously mentioned, the application should work reliably and at a fairly quick rate, it should also be easy for the users to use such that even the average person could understand fairly easily. Since we are writing an android application, we should see how the app looks like on different android devices and make sure those features all show up regardless of the device.



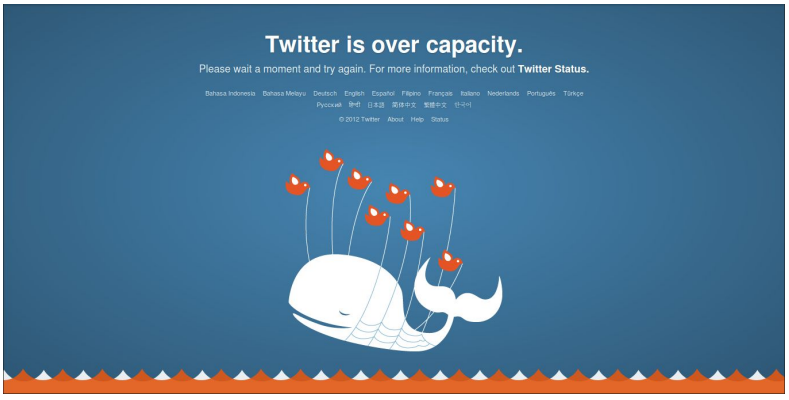
Requirement 4: Scalability

One of the most important things for our application is that it contains the core features at first, and then from there we branch of by adding extra things. The core features are a overlap of engineering, marketing, and customer success, so once those three areas are met, we can then focus on adding features that deal with only of the three sections and has minimal overlap.



Requirement 5: Capacity

One important aspect of our project is making sure that it can handle multiple swipes from students and not buffer or pause. If it can't handle that capacity, then we will run into usability, performance, and reliability issues. It's important that we make sure that it can handle the load of information that it will receive.

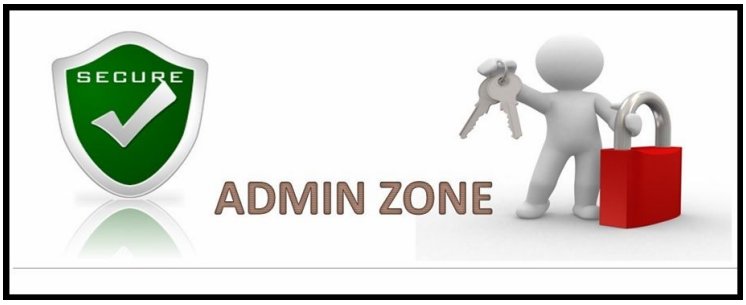


Interfaces

User:

In terms of the logical characteristic in each interface between the product(composed of both the software part and the card slider part) and the users, it is GUI based. We will have an interactive app that will display the data and have user interaction with it.

Hardware:



In terms of the physical and logical interfaces between the software and the hardware, we are planning on having an Android tablet as the device type, and plan to have control interactions by limiting what the user can access in terms of the app as a student. Only the administrator or the teacher can see overhead data, or analytics, students would just be able to sign in.

Software:

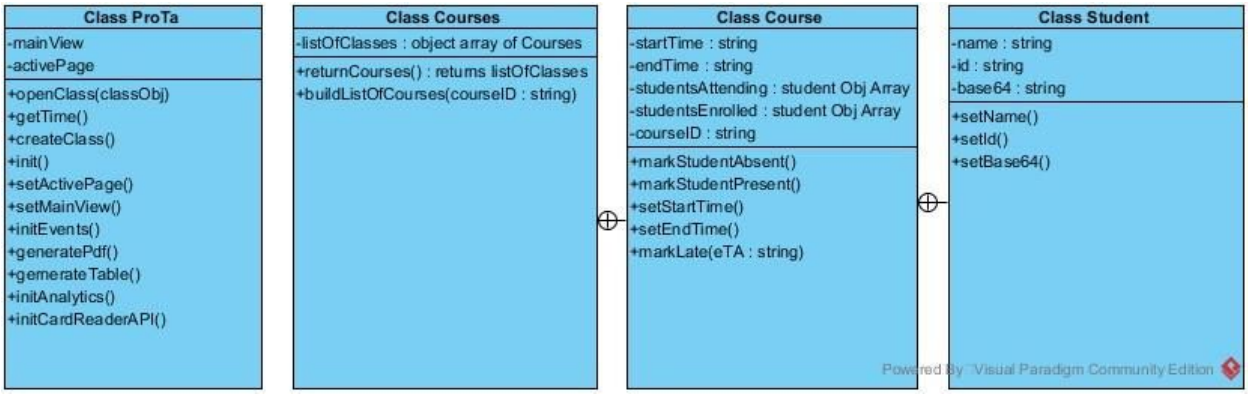
In terms of connections between the system, and/or other specific software components, we are looking to have the card reader pass information to the application, to do so we need to have a proper software implementation to make sure that the card reader information is properly extracted and passed to the app.

Communication:

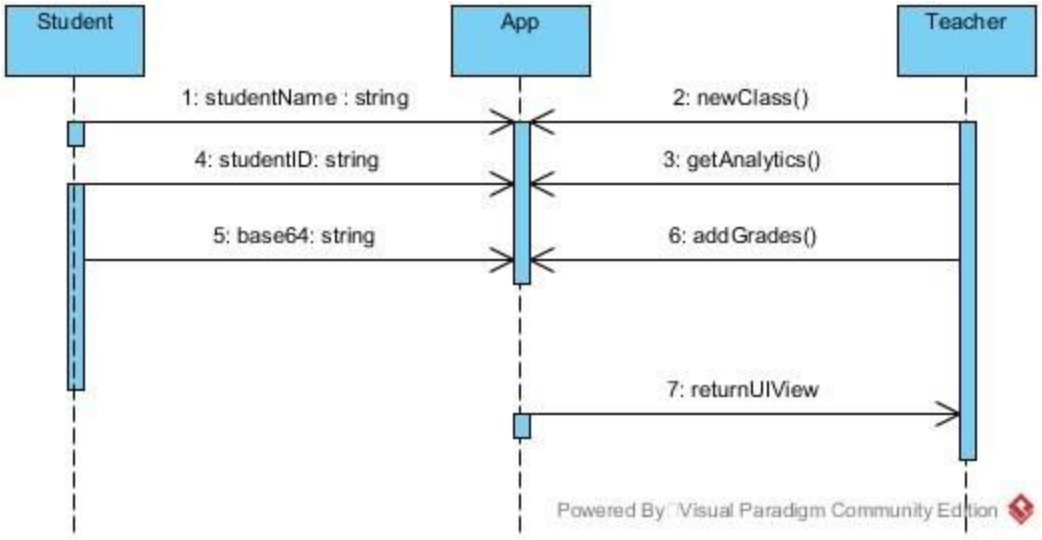
In regards to communication functions required by the system, for the core features, we simply need the application to be able to communicate with the card reader in terms of passing the information that's been taken from the card after having it swiped, and then sent to the application. After that, one the app, it will do the checking of if that student is enrolled in the class, and if that returns true, it will log the time that the person swiped in at.

2. UML Design Documents

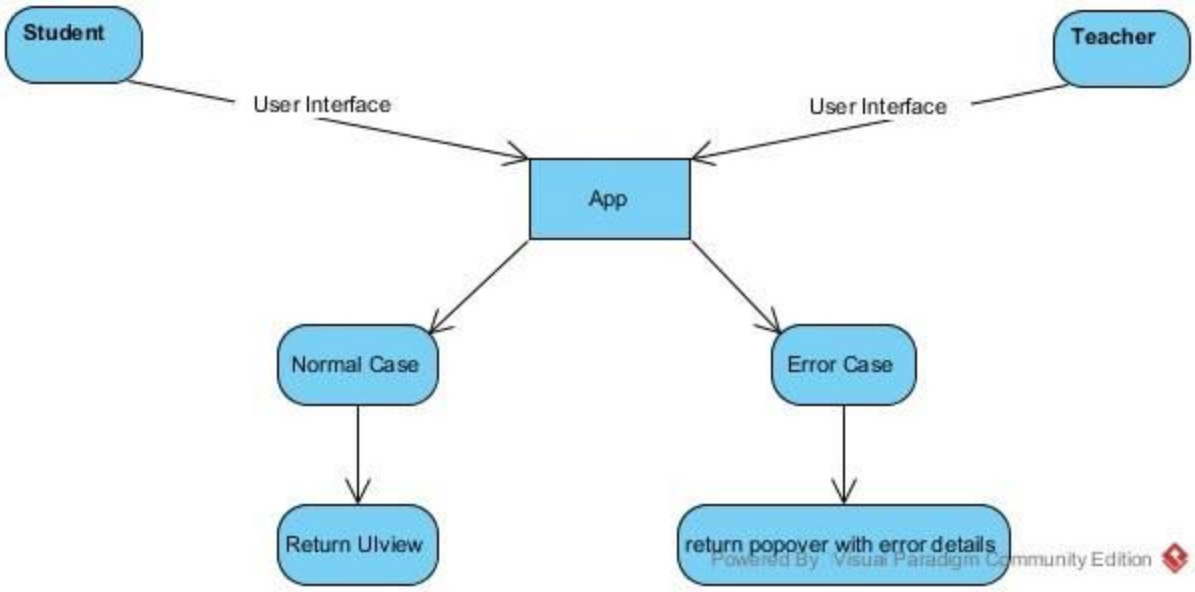
A. Class Diagram



B. Sequence Diagram



C. Activity Diagram



3. Initial Test Plan

Our test coverage has been defined above in the requirements and we will be referring to them by the numbers given in Step 1. Overall we have decided to create workable prototypes that include the functionality of each requirement. Each requirement will be considered complete once it has passed it's prototype tests and it's UI has been finalized.

Test Methods

Functional

1. Student Recognition: We will test this by using card.io to see what information is registered from the card. From then we can move on and create our software depending on what information is given. Worst case is it gives us consistently encrypted data which we can use as a unique ID for the student; we just need to prompt for a name or ask to login through blackboard to connect the student to the class/professor. This requirement will be considered complete when we can successfully read in any repeatable data from the student's cards.
2. Attendance based on time of swipe: We will test this by getting the current time when the student swipes along with creating preferences for professors to choose predefined definitions of tardiness vs. absense; after 40 minutes of class the student is considered absent. This will be considered complete when we are able to add these options to a settings menu/page.
3. Teacher going back to mark a student's absence as excused or unexcused: although we will have predefined definitions we will create an attendance report the teacher can adjust to their liking; adjusting what counts as tardy or absent. Will be considered complete once we can make the professor's report editable.
4. Software tracking entire semester of student attendance: Will be considered complete when we can store and retrieve an individual student's information from a few check-ins.
5. Add or delete students: Considered complete when Professor settings include addition or deletion of student from the current class database.

Non-Functional

1. Performance: We will test this by seeing how fast the app takes to load specific screens. Considered complete when we have written the program to minimize the amount of time the student has to wait for our app to complete tasks; they should be done in the background.
2. Reliability: We will complete this by securing the information in a database and test by retrieving information in a variety of mock situations.

3. Usability: We will test this by seeing how fast a student can complete check-in with the app. Considered complete when we have written the program to minimize the amount of time the student has to interact with the app and simplifying the UI.
4. Scalability: Scalability will be tested last because we need to have core functionality finished. We will test this by adding multiple class and professors so that we will accumulate a database full of data from a variety of class that use attendance as a grade marker.
5. Capacity: We will test this by running multiple cards one after the other and seeing what data the reader scans. Considered complete when our app knows how to handle all of the card inputs appropriately.

4. Updated Risk Management

Based on our first Risk Management of our top 5 risks, we've assessed and updated them as we've progressed with the more technical aspects of our project. Our new Risk Management is as follows:

Top 5 Risks:

1. Group Members not pulling their weight
    a. As we have assessed and began preliminary coding of the project, we have come to the conclusion that our project is very dense, and requires us all to be on the same page in terms of completing work. If one person is unable to complete their portion of the assignment, they hinder the rest of the group since everything is interconnected. To proceed with the app, we need to make sure the card scanner api is working, so without that, we only have limited amounts of work we can do until we are given that aspect.
    b. Since we all have other classes and other projects to do, it's important that we make sure that any issues we have in our own personal lives are clearly communicated to the group. For example, verbally committing to completing a portion of the code by a particular date, and then later on realizing you need more time to complete the assignment is a very common issue, and we are all aware of that and understanding. However, what we need is to be kept in the loop in regards to possible adjustments to the timeline.
    c. In that regard, our strategy for now is to not use our Slack days and have them there as a backup, and hope that we won't ever need to use them; however, if something does happen, we do have that safety net.
    d. The Risk Impact of this is quite simply not completing the project and possibly not even having a preliminary prototype ready.
    e. The Risk Probability of this has yet to be determined since so far, we have all been very transparent with working on the project and made sure to all do our due diligence in regards to completing the parts assigned to us by deadlines.

2. Being unable to use the Card Scanner we have
    a. As previously mentioned in our preliminary Risk Management, passing the information from the card reader to our application is extremely important, as this is essentially the backbone of our entire project. If we are unable to do this, we would need to look for another way to be able to pass information from some sort of equipment to the application.
    b. Being unable to use this card scanner would set us back significantly, in terms of financial liability(having to buy a new one), as well as in time since we would

have to spend time figuring out if we can use the new card scanner to acheive the desired outcome.
    c. The Risk Impact of this would be, as previously mentioned, needing to find a different card reader, spend money and time figuring how it works, and then incorporating it into our project.
    d. The Risk Probability of this occurring has yet to be fully determined; however, if we need to, we will simply look for an alternative card scanner and make sure it is a feasible option.

3. Taking out the Card Scanner aspect of our app due it not working
    a. In the event that we are simply not able to get our UNT Student IDs to pass any information to the card scanner, we would have to take our Android application and incorporate a method of signing in on the device.
    b. By doing so, we would still be able to capture the time students log in at.
    c. One method we've considered is by having the class roster in the system, and students select themselves from that, and if they decide to, can assign themselves an alias, and associate that alias with their name. By doing so, students who prefer to go by a different name, such as their middle name, can have that entered into the application and have those preferences incorporated.
    d. The Risk Impact of this would be, as previously mentioned, needing to find an alternative method to get the attendance list, most likely would have to be application-side, and then incorporating it into our project.
    e. The Risk Probability of this occurring has yet to be fully determined; however, if we do, we will simply look to create an alternate system that we can build on our application.

4. Running into issues coding the backbone of our Android Application
    a. Since we plan to have the majority of our project displayed on the Android tablet, suffice to say, it's extremely important that we insure that it is functioning.
    b. The GUI of the app should be robust enough that it can clearly display the needed information without looking too crowded or poorly displayed. By doing so, our project will be a lot easier to be understood by the customer or user.
    c. The Risk Impact of this would be immense, by running into issues coding our application, we would have to dedicate all our time and resources to finding a fix so that the application can work. The application is essential to this project, and as a result, if we run into any issues, we will have to allocate time towards solving them.
    d. The Risk Probability of this occurring is not that high, because after speaking to Keenan, who is the primary lead for our application development, he has confirmed a rough outline of how he expects the program to be and doesn't anticipate any issues. In the event that some of his calculations are incorrect, we will have to reassess at that point.

5. Being unable to complete a functioning prototype that at the very least can exhibit core features
    a. In the event that any core feature is not functional, we will have to make sure that the other features are presentable to try and make up for the absence of them. Those features can be seen as the following:
        i. Attendance checking, with a time log at the instance a card is scanned
        ii. Card scanning, passing information like student name, student ID, etc
        iii. Generating a list of students that attended class/didn't attend class(essentially some sort of analytics for the teacher)
    b. The Risk Impact of this would be needing to find some other way to demonstrate our product since the application right now is the only method we have for showcasing the product. The impact of not having the core features in our prototype at the very least can cause our product to look poorly designed.
    c. The Risk Probability of this occurring is not that high, because after speaking to Keenan, who is the primary lead for our application development, he has confirmed a rough outline of how he expects the program to be and doesn't anticipate any issues. In the event that some of his calculations are incorrect, we will have to reassess at that point.

5. Updated Project Plan

Deliverable 2

(Since Deliverable 1 is already done, there's no need to include it)
Start date: 9/13/2017, end date: 10/10/2017

- Consists of developing a set of specifications and a series of design documents, as well as a test plan. Should describe all the intended functionality, as well as take into account new updated risk management.

- Updates:
    - If needed, update the project plans and potential risks from Deliverable 1
        - In doing so, make sure to take into account new risks, as well as update accordingly since previous risks have most likely been taken into account
- Subtasks:
    - Written requirement specifications (due 9/22, completed)
    - UML Design Document (*The completion of this will be a milestone)
    - Test Plan (due 10/10)

- The rest of the report requirements are contained within this deliverable and have been discussed at length between group members after dividing the work.

Deliverable 3

(This will be essentially our entire project in a functioning prototype form)
Start date: 10/10/2017, end date: 11/13/2017

- Final Stage: Hand in working program based upon defined specifications as developed in the previous Deliverable, with an automated test suite for the program.
    - As mentioned in the updated Risk Management, we aim to have a functional core of our program with the three basic features, and then once that has been met, add additional features.

- We will also create a user manual that explains how to run and use the program, and contains a series of best practices.
    - The user manual will be a PDF that explains the app and are also considering recording a brief demo video as well.

- Subtasks:
    - Submit code for code inspection of the functionality of project (due 11/13

- Working Program code (*The completion of this will be a milestone)
    - Have started working towards this
- Report meeting specifications (due 11/21)
- Presentation creation (due on 11/27)

6. Meeting Minutes

Meeting 1: September 19th, 2017 – 30 Minutes

Members Present: Nicolas Stencel, Manuel Vargas, Naumaan Hassan, Keenan Jabri

In this meeting we went over our original planned out timeline, and looked to see if there were any changes needed based on the overall groups, and individual members, performance throughout the progress of the first deliverable. We determined that we had done well on delivering our parts for what was required and that we had stuck well to our original time table. Therefore, we decided to stick to our original timeline.

Meeting 2: September 21st, 2017 – 30 Minutes

Members Present: Nicolas Stencel, Manuel Vargas, Naumaan Hassan

We looked at the requirements and key points we needed to hit on the second deliverable and divided up the workload based on who would be contributing a lot to the code work later and previous work input. This means that group members that did not have much code experience would take more of the workload in this section to compensate for their inexperience in the later sections. Also those who had done a lot of work in the previous deliverable were given less of a workload on this one as a reward for their hard work.

Meeting 3: September 26th, 2017 – 45 Minutes

Members Present: Nicolas Stencel, Manuel Vargas, Naumaan Hassan, Keenan Jabri

In this meeting we talked about setting up a file directory for the actual code on GitHub and decided to install appropriate plug-ins so that we could test the app on our phones. We decided eventually on a plugin called Phone Gap. We also made our final decision as to the IDE and UI setup and went with Cordova for the backend, and Framework 7 for the user interface parts of the project.

Meeting 4: September 28th, 2017 – 30 Minutes

Members Present: Nicolas Stencel, Manuel Vargas, Naumaan Hassan, Keenan Jabri

During the 4th meeting we discussed how everybody was doing with their progress on the second deliverable thus far and decided to set our group deadline on Friday, October 6th, 2017. Although the actual deadline wasn't until October 10th we agreed it would be best to get it done earlier and get all the parts uploaded onto our own google drive so that we would be able to change and edit parts as needed.

Meeting 5: October 3rd, 2017 – 30 Minutes

Members Present: Nicolas Stencel, Manuel Vargas

Originally, we had planned for everybody to get their parts of the deliverable done by the night of October 6th, so that we would have time to change and edit anything should that be necessary. However, due to the tight work schedule and workload resulting from outside life and other classes we decided to move our deadline to Sunday, October 8, 2017. We also discussed the benefits of the PhoneGap app we would be using to test our own app, as well as other ways to integrate out phone app.

7. Progress Report

The progress of the project as of today is very good. Since the last report we have discussed more of the actual systems that we will need to use in order to create our application. Keenan has demonstrated the functionality of PhoneGap which we plan to use to build and test out our application with; PhoneGap is a mobile app development system that will let us build our application as you would a website and build to either iOS or Android. We have also attained a card scanner for mobile phones; we are currently running preliminary test to see how easy it will be to use and how accessible the information from the scanner can be used in our development environment. We found a Cordova plugin called card.io that we can leverage in our development environment.

As an update from the last report, we have assigned more specific roles for each of our teammates. Naumaan has decided to concentrate more on the written documentation that we must provide because he doesn't feel as comfortable with the development environment we have set up. We've all agreed this would be beneficial to the group because when testing we will already have an idea on what should be tested based on the test plan/documentation we have prepared in advance; the testers can then add to the progress of the documentation to improve scope, directions, and milestones. Manuel will continue to manage the group repository as well as turn in group assignments; he serves as the final check for what is delivered as well as the structure of the deliverables and assessing the team's progress in the progress report. Nicolas is continuing to take meeting minutes as well as joining Keenan and Manuel in the testing and building of the application.

8. Member Contribution Table

Member	Contribution Description	Overall Contribution	Note
Nicolas Stencel	Steps 1a, 6	25%	Attended every class and made sure to keep the group updated in regards to what work needed to be done, as well as share useful information that he found online in regards to our project in group discussions.
Keenan Jabri	Step 2	25%	Handled the most programming heavy portion of the assignment and essentially walked us through the pseudocode of the app as well as how he would like to proceed with it.
Manuel Vargas	Steps 3, 7	25%	Ensured that everyone stayed on task and knew what they were doing for the Deliverable, organized the deliverable file, as well as committed to pushing updates through GitHub, and kept us updated with the Progress Report as well as added in most references
Naumaan Hassan	Steps 1b, 1c, 4, 5 & 8	25%	Took into account updated risks and made sure that the group was aware of them, as well as tracked the rest of the members contribution, and offered a resource as well as some programming insight.

References

- PhoneGap
- Card.io
- Cordova
- Blackboard API
- PayPal Card Scanner