Department of Computer Science and Engineering
the University of Texas at Arlington

Padawan Programmers

Mavs Assistant Reporting System

Team Members:
Calvin Hovsepian
Bernae Tull
Minglu Wang
Bruce Derou
Thang Le

Table of Contents

Table of Contents

Document Revision History

Revision Number	Revision Date	Description	Rationale
1.0	9/23/2015	SRS First Draft	
2.0	11/12/2015	SRS Second Draft	Making changes on the updates
3.0	12/9/2015	SRS Semester Draft	Added Feasibility Assessment
4.0	5/3/2016	SRS Final	

List of Figures

1. Product Vision

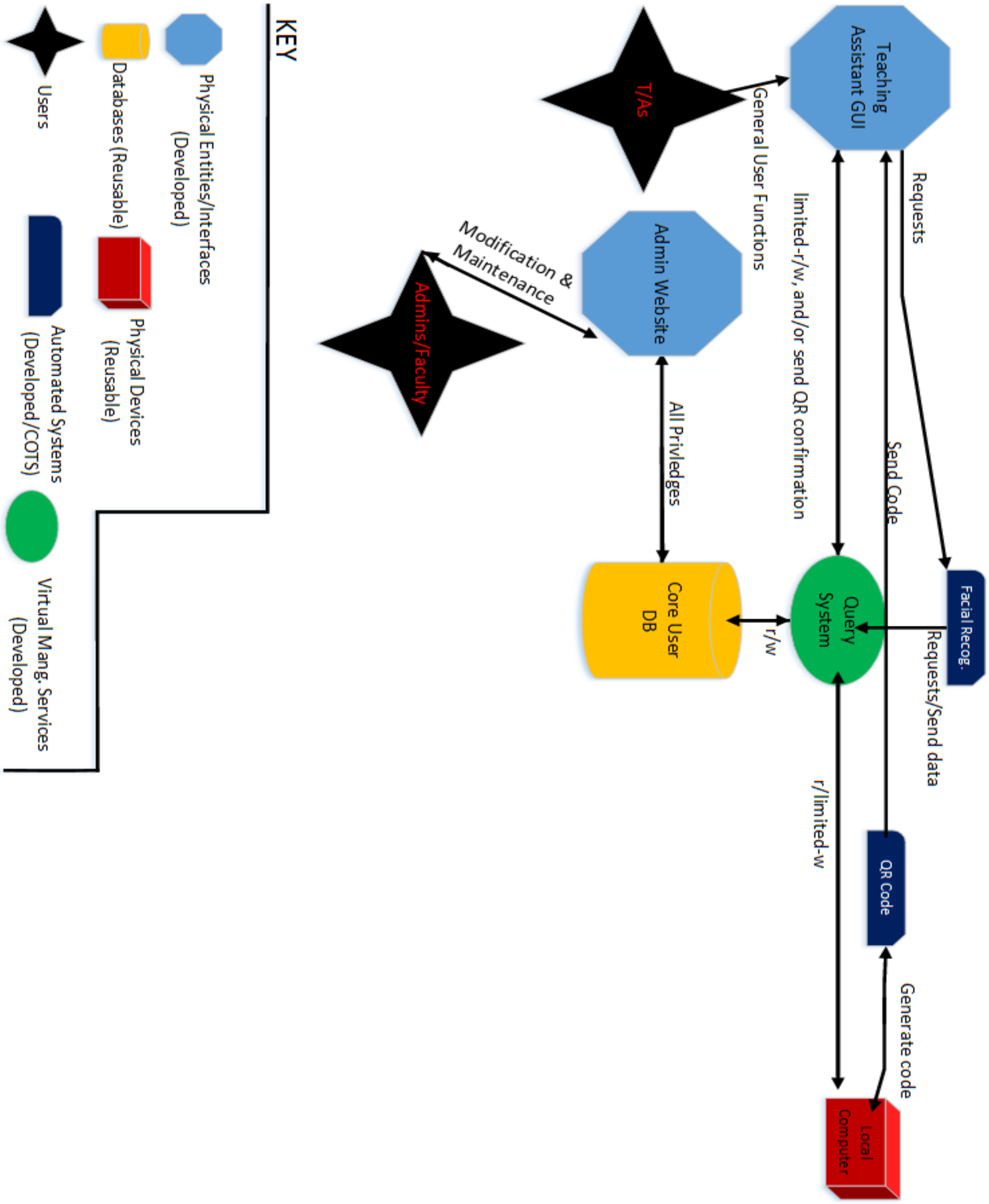
1.1 Purpose and Use

M.A.R.S. (Mavs Assistant Reporting System) will automate the tracking of hours, etc. of UTA College of Engineering undergraduate class assistants. Assistants will able to login via a mobile app to clock in/out. The clock in/out process will be verified with facial recognition. Assistant's clock in/out location and time will automatically be tracked using a QR code, generated by a locally installed program on certain computers, to provide a seamless way for assistants to generate their timesheet. Additionally, this system will provide functionalities for assistants and administrators to view and manage various informations.

1.2 Intended Audience

The product is intended for Mav Class Assistants and Administrator.

Figure S-1: Overview


Teaching Assistant GUI
T/As
General User Functions
Requests
limited-r/w, and/or send QR confirmation
Send Code
Admin Website
Modification & Maintenance
Admins/Faculty
All Privledges
Facial Recog.
Requests/Send data
Query System
r/w
Core User DB
r/limited-w
QR Code
Generate code
Local Computer
KEY
Physical Entities/Interfaces (Developed)
Databases (Reusable)
Users
Physical Devices (Reusable)
Automated Systems (Developed/COTS)
Virtual Mang. Services (Developed)

2. Requirements, Features and Functions

2.1 Teaching Assistant Digital Timesheets

2.1.1 Story:

As an assistant I want to be sent a timesheet before the 2 week pay period is over.

2.1.2 Acceptance Criteria:

Criteria 1: Verify timesheet is sent before pay period is over.

Criteria 2: Verify clock in and clock out hours are accurate for that 2 weeks.

Criteria 3: Verify total number of hours is under maximum hours.

Criteria 4: Verify pay period date is in the format mm/dd/yy - mm/dd/yy.

Criteria 5: Verify personal information is correct.

2.1.3 Priority:

Critical

2.1.4 Effort:

Medium

2.2 Teaching Assistant Database

2.2.1 Story:

As an assistant, I want to have my hours saved in a database.

2.2.2 Acceptance Criteria:

Criteria 1: Verify correct information during registration saved.

Criteria 2: Verify admin teaching assistant/grader assignments are correctly saved.

Criteria 3: Verify clock in hours are saved.

> Criteria 3.1: Verify clock out hours are saved to correct clock in hours.

Criteria 4: Verify location is saved with correct hours.

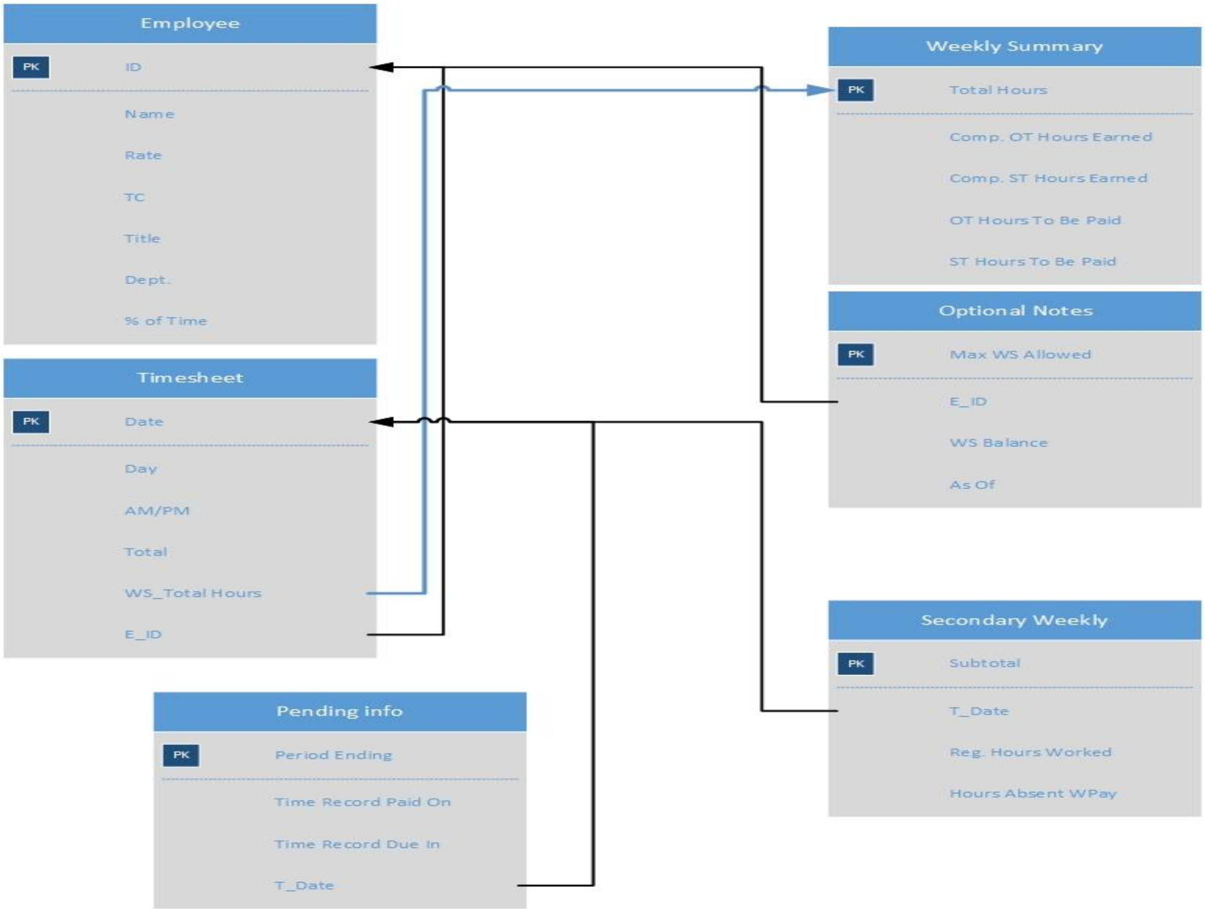
2.2.3 Priority:

Critical

2.2.4 Effort:

Medium

Figure S-2: Database Relational Diagram



2.3 Server

2.3.1 Story:

As an administrator, I need a server that will handle the database and request.

2.3.2 Acceptance Criteria:

Criteria 1: Verify server setup.

2.3.3 Priority:

Critical

2.3.4 Effort:

High

2.4 Clock in/out with Facial Recognition

2.4.1 Story:

As an assistant, I need to be verified through facial recognition before i am able to clock in/out.

2.4.2 Acceptance Criteria:

Criteria 1: Verify system verifies a user with at least a 65% match.

2.4.3 Priority:

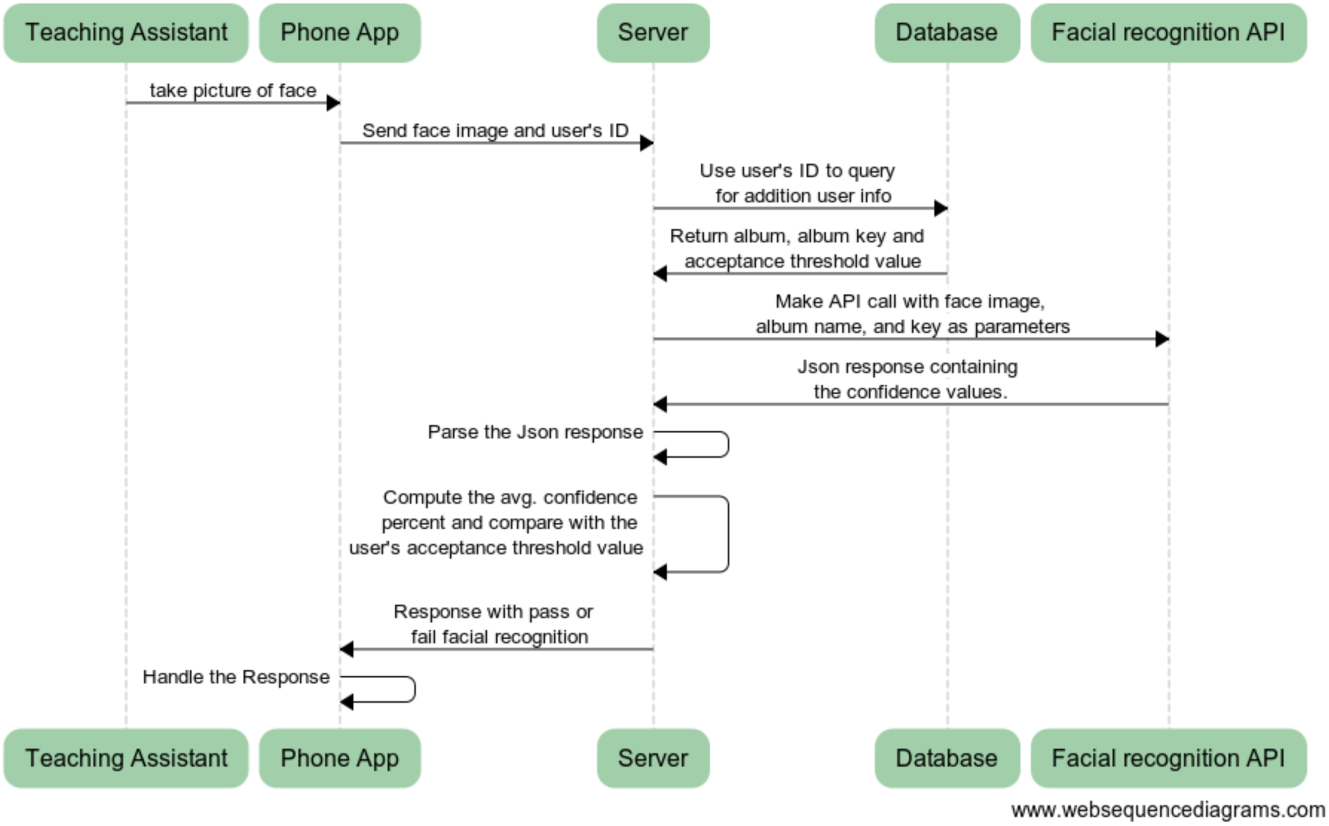
Critical

2.4.4 Effort:

High

Figure S-3: Facial Recognition Sequence Diagram

Facial Recognition Seq. (Lambda Labs API) Diagram


Teaching Assistant
Phone App
Server
Database
Facial recognition API
take picture of face
Send face image and user's ID
Use user's ID to query for addition user info
Return album, album key and acceptance threshold value
Make API call with face image, album name, and key as parameters
Json response containing the confidence values.
Parse the Json response
Compute the avg. confidence percent and compare with the user's acceptance threshold value
Response with pass or fail facial recognition
Handle the Response
Teaching Assistant
Phone App
Server
Database
Facial recognition API
www.websequencediagrams.com

2.5 Location Tracking

2.5.1 Story:

As an admin, I want my class assistants to scan a QR code in order to save their location into the database.

2.5.2 Acceptance Criteria:

Criteria 1: Verify system prompts user to scan QR code.

Criteria 2: Verify system can scan a QR code.

Criteria 3: Verify system saves date, time, and location into database.

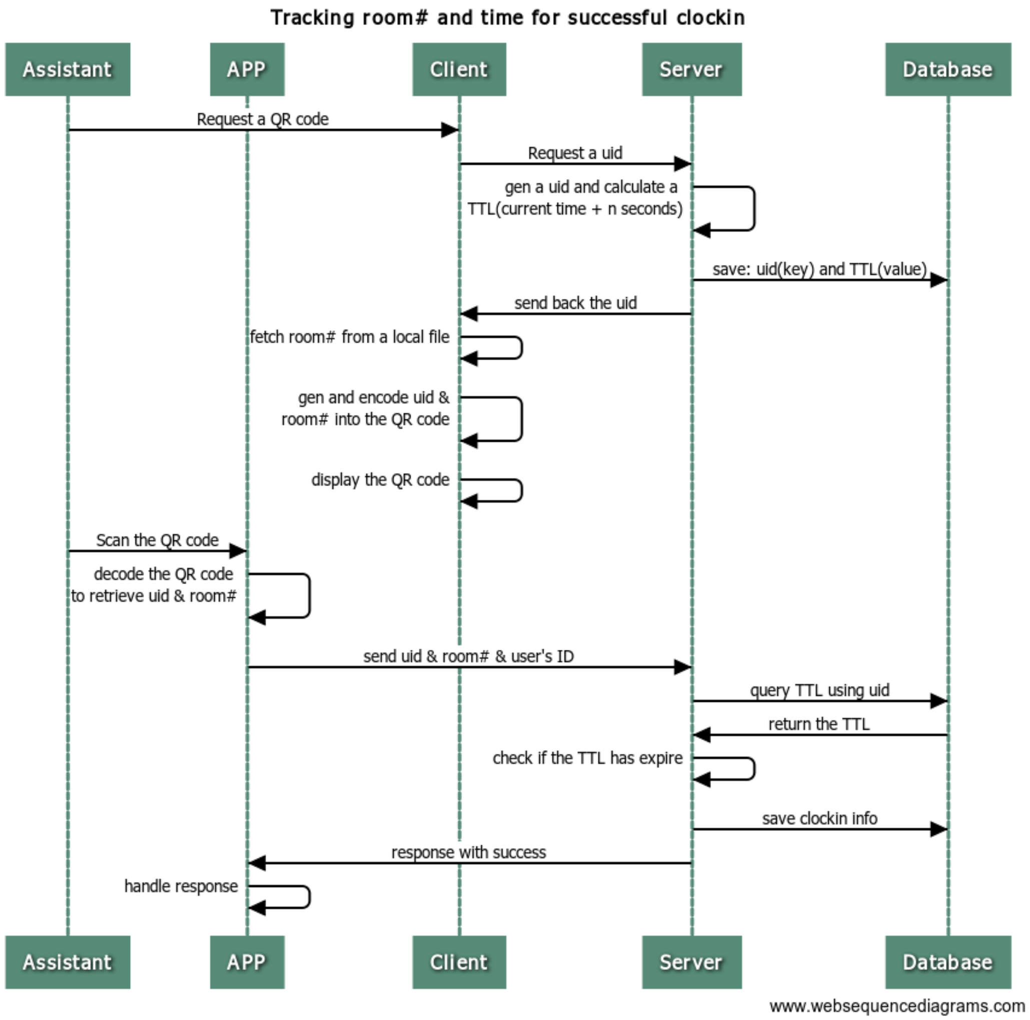
2.5.3 Priority:

Critical

2.5.4 Effort:

High

Figure S-4: Location Tracking Sequence Diagram


Tracking room# and time for successful clockin
Assistant
APP
Client
Server
Database
Request a QR code
Request a uid
gen a uid and calculate a TTL(current time + n seconds)
save: uid(key) and TTL(value)
send back the uid
fetch room# from a local file
gen and encode uid & room# into the QR code
display the QR code
Scan the QR code
decode the QR code to retrieve uid & room#
send uid & room# & user's ID
query TTL using uid
return the TTL
check if the TTL has expire
save clockin info
response with success
handle response
Assistant
APP
Client
Server
Database
www.websequencediagrams.com

2.6 QR Code

2.6.1 Story:

As an admin, I want a QR code generated via a locally installed program by me.

2.6.2 Acceptance Criteria:

Criteria 1: Verify program prompts for a code on installation in order to verify admin.

Criteria 2: Verify program prompts for name of computer which is used as location.

Criteria 3: Verify QR code generated with date, time, and location.

2.6.3 Priority:

Critical

2.6.4 Effort:

High

2.7 Grader Location

2.7.1 Story:

As a grader I do not need to verify my location.

2.7.2 Acceptance Criteria:

Criteria 1: Verify system does not ask for a QR code when a grader is clocking in.

2.7.3 Priority:

High

2.7.4 Effort:

Low

2.8 Capability on different systems

2.8.1 Story:

As an assistant, I want to login and logout on mobile device through different systems, such as Android and iOS.

2.8.2 Acceptance Criteria:

Criteria 1: Verify users can login from multiple platforms.

Criteria 2: Verify users data will be accessible from any device.

2.8.3 Priority:

High

2.8.4 Effort:

High

2.9 Administrator Assigning

2.9.1 Story:

As an administrator, I have to be able to assign users as teaching assistants, graders, or admins upon registration.

2.9.2 Acceptance Criteria:

Criteria 1: Verify that the request has the user with the title options.

Criteria 2: Verify user privilege/title is saved in database.

2.9.3 Priority:

High

2.9.4 Effort:

High

2.10 Registration Requests

2.10.1 Story:

As an admin, I would like the system to notify my when a new user registers so I can assign them privileges.

2.10.2 Acceptance Criteria:

Criteria 1: Verify request sent to admin once user registers.

2.10.3 Priority:

High

2.10.4 Effort:

Medium

2.11 Admin Detailed Sheets

2.11.1 Story:

As an admin, I would like to be able to generate detailed worksheets with names, dates, times and locations.

2.11.2 Acceptance Criteria:

Criteria 1: Verify generation of timesheet.

2.11.3 Priority:

High

2.11.4 Effort:

Medium

2.12 Deleting Users

2.12.1 Story:

As an administrator, I want to be able to delete users once they are no longer using the system.

2.12.2 Acceptance Criteria:

Criteria 1: Verify deleted users are denied users access to system.

Criteria 2: Verify data and hours is kept for tax purposes.

2.12.3 Priority:

Medium

2.12.4 Effort:

High

2.13 Early Generation of Digital Timesheet

2.13.1 Story:

As an assistant I want to be able to generate my timesheet early.

2.13.2 Acceptance Criteria:

Criteria 1: Verify clock in and clock out hours are accurate for that 2 weeks.

Criteria 2: Verify total number of hours is correct.

Criteria 3: Verify pay period date is in the format mm/dd/yy - mm/dd/yy.

Criteria 4: Verify personal information is correct.

2.13.3 Priority:

Medium

2.13.4 Effort:

Medium

2.14 Administrator Corrections

2.14.1 Story:

As an administrator, I want to be able to update a user’s clock out time just in case they forget to clock out.

2.14.2 Acceptance Criteria:

Criteria 1: Verify that the admin can find the user’s times.

Criteria 2: Verify the admin can update a clock out time.

2.14.3 Priority:

Medium

2.14.4 Effort:

High

2.15 Total Hours Notification

2.15.1 Story:

As an assistant and admin I want to be notified by email when my hours reach a total of 19.5.

2.15.2 Acceptance Criteria:

Criteria 1: Verify when a user has 19.5 hours an email is sent to the user with the running total hours.

Criteria 2: Verify when a user has 19.5 hours an email is sent to the admin with running the total hours.

2.15.3 Priority:

Medium

2.15.4 Effort:

Low

2.16 Back-up

2.16.1 Story:

As an admin, I need the data backed up.

2.16.2 Acceptance Criteria:

Criteria 1: Verify data will be backed up for 1 year.

2.16.3 Priority:

Medium

2.16.4 Effort:

Medium

3. Future Items

3.1 Alternative Login with Finger Prints

3.1.1 Story:

As a class assistant, I want to be able to login and logout my hours with finger prints.

3.1.2 Acceptance Criteria:

Criteria 1: Verify system verifies a user with at least a 80% match to a previously stored fingerprints.

3.1.3 Priority:

Medium

3.1.4 Effort:

High

4. Feasibility Assessment

4.1 Research Analysis

At this point in time, our focus in our research is how to implement facial recognition. In order to avoid researching more than one topic at once, we chose to start with Android and then move to iOS. Android is also a higher priority than iOS according to the product owner. In order to make the system more robust, we must research what threshold of acceptance to set our facial recognition at to have an acceptable amount of false negatives and false positives. We will have to research the standards of facial recognition systems to figure this out.

We will need to research iOS app development, once we get to that stage of the product. The team estimates this to be difficult based on the materials on hand to us. We not only need to research how to develop iOS, we have to research how to do it on Microsoft and Apple. We have to research both, because at the moment no one has access to an Apple computer. If this circumstance changes, we need to have the knowledge to start developing on either operating system.

We previously allocated time to research implementation of QR codes in Java. We applied our time effectively and completed the research and implementation within 1 month. We used the resources available to us and communicated with the product owner to get ideas of implementation, and then used our knowledge to correctly apply the Java library we found.

Due to the fact that we efficiently researched 1 component in a timely manner, we believe the 4-5 months that are left in this product is enough time to research and develop 3 more components.

4.2 Technical Analysis

The project does not involve hardware/mechanical devices, it will be completely software based, which will include a installed software on certain computers, an Android application, an iOS application and a Web page. Our team's diverse talents coupled with research will allow us to complete all the platforms in the given timeframe. We will start with the systems we are most familiar with and leave the more difficult tasks to later dates. This will allow us to complete more components in the being and then slow our velocity in the second half of the project.

There will be 2 teams working on a platform at a time. The teams will consist of 2 people that are familiar with their specific application with a member that floats around and helps out where needed. Each team member has individualized talents which will determine which platform they develop.

4.3 Cost Analysis

According to the cost research that our team has done, it has been determined that the project can be successfully developed within the $800 budget that the team has been allocated. The project is composed of purely software, making it possibly to complete the product without using any of the budget, because the small scale of the product allows us to use a facial recognition API without a fee. The server will require a minimal fee a year from now, but that is out of the scope of this current team.

4.4 Resource Analysis

Team Padawan Programmers consist of 3 computer science students and 1 software engineering student. The team possesses a wide variety of skills and knowledge stemming from the various electives we have taken.

Every team member will be working on a software component, since MARS is software based. Members with experience with app development will be on the Android and iOS app and the remaining members will be focused on the website. The team has had the same curriculum throughout school, which will allow us to use each other as resources when we get stuck.

Android is coded in Java, which is a resource that everyone is familiar with. Java has a wide variety of libraries and API's that we can take advantage of when coding facial recognition and QR code. The language of the website is yet to be determined. It could be Javascript of PHP; either way there is plenty of online help available to us to use as reference.

The only resource we lack is an Apple computer which makes iOS development easy and friendly. However, we might use the resources at the school and work on a Mac in a lab. The only problem with that is, we do not know what development software is downloaded or if we have the privileges to download software. The other downside is we would have to work during times that the lab is open instead of whenever we are available.

Our resource constraints will not impede our progress on the product; we will be able to operate effectively and efficiently despite them.

4.5 Schedule Analysis

The first method of estimation that we implemented is the Jones First Order. The purpose of this method is to identify how big or complex the project is, in order for us to determine if we are able to successfully complete the project on the time frame available. Later, on we will estimate using a different method to compare accuracy.

First, we identified all the inputs, outputs, inquiries, logical internal files and external interfaces. After that, we classify them as low complexity, medium complexity or high complexity, as demonstrated on the table below, which we can then use to calculate the unadjusted function-point total.

Figure S-5 Function Point Table Breakdown

Program Characteristics	Low Complexity	Medium Complexity	High Complexity
Inputs	5 x 3	0 x 4	4 x 6
Outputs	0 x 4	1 x 5	3 x 7
Inquiries	14 x 3	1 x 4	0 x 6
Logical internal files	0 x 7	1 x 10	3 x 15
External interface files	0 x 5	0 x 7	1 x 10
Unadjusted function-point total			176

After calculating the unadjusted function point total, we assigned values to the influence multipliers, with the purpose of getting a more accurate estimation.

Figure S-6 Influence Multipliers

Adjustment Factor	Degree of Influence (0-5)
Data Communication	1
Distributed Data Processing	0
Performance	2
Heavily Used Configuration	0
Transaction Rate	5
On-line Data Entry	3
End-User Efficiency	5
On-line Update	3
Complex Processing	0
Reusability	5
Installation Ease	2
Multiple Sites	0
Facilitate Change	3
Sum	29

After getting an adjustment factor, we calculated the influence multiplier with the following formula:

Influence Multiplier = (29 * 0.01) + 0.65 = 0.94

Multiplying the Influence Multiplier by our Unadjusted Function Point total, we obtained our Adjusted

Function Point total. Adjusted Function Point Total = 0.94* 176 = 164.44

Lastly, we use the Jones First Order estimation procedure to calculate the time it would take to develop a project using the established influence multiplies and unadjusted function points. MARS falls under the "Systems" category, which is analyzed in Best Case, Average Case and Worst Case scenarios, the calculations are displayed in the following table.

Duration = 165.44^0.45 = 9.96 Months

Figure S-7 Jones First Order Estimation

	Best Case	Average Case	Worst Case
Adjusted Function Point	165.44^.43	176^.45	176^.48
Total	8.99	9.96	11.61

4.6 Scope Analysis

According to Figure 7 above our best case scenario is we finish in 9 months. While, the worse puts at 11 and a half months, which would mean we would not finish our project. However, this method does not take into account that that iOS development will go quickly after the Android app is complete.

We currently have 38 items on the product backlog with 20 of those being complete, meaning we have 18 more backlog items to complete within 5 months. That would be about 4 items that need to be completed each sprint. Our current velocity had us at completing 5 or more tasks each sprint. This semester was focused on completing the small backlog items such as design and research tasks, so that we could focus the next half on the big items. Our velocity will slow, but that works out perfectly since the items are getting more difficult.

Based on our estimated velocity and our Jones First Order best case estimation we feel confident that we will complete our product in the timeframe we have been allotted.