10/22/2019

Marc Minnick, Galen Shirey, Joshua Moran

CSC-354-030

Project Plan

Encrypted Private Messaging Application

Contents

[1. Project Scope **Error! Bookmark not defined.**](#_Toc22589109)

[2. Project Schedule 5](#_Toc22589110)

[3. Feasibility Study 6](#_Toc22589111)

[4. Work Breakdown Structure (WBS) 8](#_Toc22589112)

[5. Process Model 9](#_Toc22589113)

Revision History

[1. Project Scope **Error! Bookmark not defined.**](#_Toc22589109)

Overview of Organization modified

[2. Project Schedule 5](#_Toc22589110)

[3. Feasibility Study 6](#_Toc22589111)

Fixed cost values

Made project sound more feasible.

[4. Work Breakdown Structure (WBS) 8](#_Toc22589112)

Increased Image Size

Removed Background Colors

[5. Process Model 9](#_Toc22589113)

Added more detail to our process model include more information about what Scrum is and why it is better than other outdated methodologies

6. Resource Allocation……………………………………………………………………………………………………………………… 10

Re-did format to be a task -> resource relationship

1. Project Scope

**Private Messaging Application**

1. **Project Overview**
   1. Overview of the organization
      1. Our organization consist of three Kutztown University students pursuing an undergraduate degree in computer science. We are looking to create something that can be used by anyone and everyone that is able to run the program. We believe that our combined experience in cryptography, information technology, and network programming can bring a desirable experience to the market.
   2. Current Situation and Problem Statement
      1. Currently, the market for messaging applications is saturated, however, the applications with encryption methods are far and few between. In today’s world with consumers’ data being a high priority for many criminals, encryption and protection of data is crucial. With these issues in mind, we are trying to make security our main priority by including encryption in the text messaging functionalities of this application.
2. **Statement of Purpose**
   1. The purpose of this project is to provide customers with an encrypted private and group messaging application with both audio and video calling capabilities. This includes having end-to-end encryption for both private and group messaging. Since security is a top priority with this project, we would need to ensure that security is being upheld to the highest possible standard that we can attain.
3. **Project Objectives**
   1. Provide the user with a safe method of communication that has not been compromised by outside threats for at least 6 months after release.
   2. The product passes a security test that finds any vulnerabilities that are listed in “OWASP Top 10 – 2017 The Most Critical Web Application Security Risk.”
   3. The product can pass at least one experienced penetration testers.
4. **Project requirements**
   1. User Requirements
      1. iPhone running iOS version 13.0
      2. Email for identity purposes
   2. Equipment
      1. At least one iPhone running iOS version 13.0
      2. Computers with internet access
   3. Skills
      1. Experience with iOS development
      2. Experience with cryptography and encryption
      3. Experience with program security
      4. Experience with AWS for authentication database
      5. Graphics design experience for user interface
   4. Tools
      1. AWS
      2. Xamarin
      3. Visual Studio
      4. Xcode
      5. Adobe After Effects
      6. Adobe XD
      7. Internet Browser
5. **Project Specifications** 
   1. High-level user requirements.
      1. A messaging application will be developed for Apple’s iOS.
      2. Users will be able to create a unique account using their email.
      3. Users will not be able to receive messages from anyone that is not a mutual contact.
      4. Users will be able to send “mutual contact” requests to other users. They will also be able to accept and deny incoming request.
      5. The application will allow a user to send and receive encrypted messages from other users of the application. Messages will be in the form of a picture, video, or text.
      6. The application will allow the users to decide the maximum number of messages that will be stored on the device.
      7. Any message that is sent with the application will only be stored in two places: the device of the sender and the device of the receiver.
6. **Critical Assumptions** 
   1. Availability of resources, scope, tech, specs.
   2. The development team is able to learn how to use the tools in tandem with development and within schedule.
7. **Project constraints** 
   1. The development team has no prior experience working with Amazon Web Services.
   2. The project must be completed by the end of April 2019.
   3. The application must be developed for the iOS operating system.
8. **Project Acceptance Criteria**
   1. For the project to be accepted, it must allow users to do the following:
      1. Create an account
      2. Add users by account ID
      3. Select users by account ID to create a message
      4. Users must also be able to create group messages.
      5. Successfully send/receive messages to/from other user(s)
      6. Fully encrypt user messages
      7. Audio and video calling between clients
9. **Project Success Criteria**
   1. A security breach does not occur for at least 6 months after release.
   2. We can maintain more than 5 active users 6 months after release.
10. **Project Deliverables**
    1. Users will receive our product as well as an in-app tutorial on how to use it. This will serve the purpose of a user guide.
    2. This will also include all consecutive updates of our product
11. Project Schedule
    1. GANTT Chart



1. Feasibility Study
   1. ECONOMIC FEASIBILITY

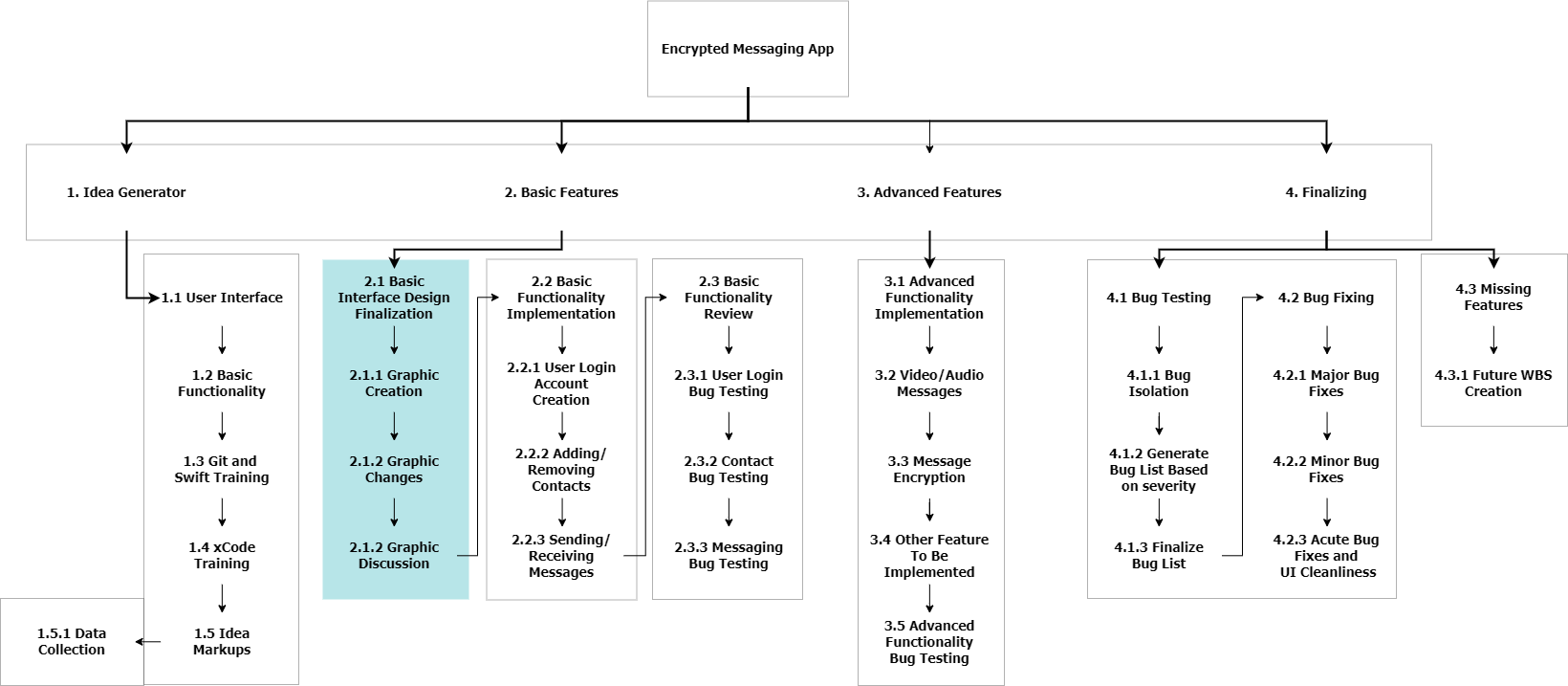
The following analysis assumes that we will can monetize this product.

|  |  |  |
| --- | --- | --- |
| **Consideration** | **Estimated Cost** | **Notes** |
| End-user computer hardware purchase cost | $1,000.00 | Users will use their iPhones |
| Software license purchase cost | N/A | We will use free and open source software |
| Hardware & Software deployment cost | $99 | App will not be published on the App store($99/year) |
| Hardware warranties and maintenance costs | $1,000.00 | Cost of replacing one workstation one time |
| Software license tracking cost | N/A |  |
| Cost of security breaches | $1,000,000.00 | Will destroy the reputation of secure app |
| Cost for electricity and cooling | $10,000 | Office space for three months |
| Server hardware and software cost | $100 | Max cost if pay-as-you-go model is chosen for AWS |
| Insurance cost | N/A |  |
| Testing cost | $1,000 | External tester cost for security |
| Cost to upgrade or scalability | $100 - $xxx | AWS pay-as-you-go prices vary based on usage |
| IT personnel cost | N/A | Our team will perform IT activities |
| Backup and recovery process cost | $5.00 | 16GB flash drive to store backup data. |
| Costs associated with failure or outage | $100,000 | No monetary cost |
| Diminished performance incidents (users having to wait) | N/A | No monetary cost |
| Tech training cost of users and IT staff | $100,000 | Users are familiar with messaging applications and intuitive design should mitigate cost. |
| Infrastructure | N/A | Our goal is to use outsource as much infrastructure |
| Migration cost | N/A |  |
| **Total Cost:** | $1,212,100.00 |  |

* 1. TECHNICAL FEASIBILITY
     1. As of the time of this report, we do not possess the expertise nor the experience to develop a product with the number of features listed in the specifications. Despite this impediment, the technology we need to implement already exist in many forms. This should substantially lower the risk involved with the project.
     2. Our group currently owns hardware that can be used as a server and a high-speed internet connection. Since scalability is not a concern at this moment, there is no need to worry about the limits of the server hardware.
  2. OPERATIONAL FEASIBILITY
     1. The application will provide the same functionalities as other messaging applications on the market, so while, it won’t necessarily stand out, it’s also not inferior. The product can find a market.
  3. SCHEDULING FEASIBILITY
     1. Regarding deadlines, the project schedule will be extremely contingent on the needs of our class as well as its schedule.
  4. SUMMARY
     1. While the project seems to be technically feasible, the project will result in a financial loss. If we are to only consider tangible benefits, then we would not proceed with this project. Despite the tangible loss, however, we would like to continue with this project due to the experience gained in the many technologies involved in creating such a product. The cost to create this product is minor in comparison to the benefits gained.

Work Breakdown Structure (WBS)

* 1. WBS Flowchart



4. Process Model

This project is going to involve several obstacles. The first is our inexperience in information security and creating a fully featured networking applications for iOS. Another complication is the introduction of uncertainty of clashes between the class schedule and our plan.

To address these problems, we plan on using the Scrum method. We believe that its focus on creating a fully featured product as quickly as possible will insure that even if we are not able to include every feature planned, we will at least have a product that works. Scrum is an agile methodology used throughout the modern-day software industry as a replacement to out of date methods including the Waterfall model, and V model. Contrasting the waterfall model to scrum is important when trying to define scrum. The Waterfall model is a methodology that requires a lot of backtracking during the stages of development. Software teams start with an initial plan based on the input of the customer. Planning can take anywhere from one to three months then is followed by the developmental process which can take around the same amount of time. After development, the team enters the testing phase, then review and deployment. If at any time during these stages a problem occurs, the model allows the team to be sent back to the previous stage or all the way back to the planning stage. As you may be able to tell, this can cause products to be deployed way later than expected and maybe even out of date based on the customer's needs. Scrum takes care of this problem by implementing a methodology that allows versions to be released within a few months’ time. Feedback is then taken by the customer and a new version starts its developmental process.

Scrum allows complex adaptive problems to be tackled while adhering to guidelines that create the highest value products. Its agile nature creates an environment where the entire development team has input while developing the iterations to deliver the product the most efficient way possible. Scrum and agile methodologies have taken the forefront of developmental methods in the past decade and are used by almost every software company. They allow continuous iterations and testing throughout the lifecycle of the software and promotes self-organization and helps teams in responding to the sometimes none predictable nature of problems.

Software development using the scrum methodology is much different than a standard managerial situation. A scrum master replaces the project lead, but to facilitate the agile principles of scrum instead of barking orders. They oversee managing the process for how information is exchanged throughout the team and customer. When a Scrum Master tries to take the role of a manager, the entire framework fails because development teams are not able to work on their own. Self-organization is key when using the Scrum method.

Scrum is a lightweight framework that allows the team to self-organize and deal with complex and unpredictable problems. Throughout the scrum sprint the customer first describes their needs to the product owner which is relayed to the scrum master. The scrum master then takes this information and inputs it into a backlog for the start of development. The product backlog is a list of all that is to be added to the project per the customer and contains user feedback throughout the entire project. Each phase in the figure can last between one to four weeks. Once the development team finishes their build, it is sent to the testing phase. During the testing phase, the product is tested for bugs and missing features. Finally, the build is then sent to the review stage to get ready for final deployment. Each time one of these cycles are complete, the customer gives their feedback which is sent back to the Scrum Master to be backlogged and start the process over again. This allows for an iterative development cycle with versioned releases. Each cycle a version is deployed to the customer and throughout the next cycle details can be added and removed.

|  |  |
| --- | --- |
| Graphic Design Plan | Galen |
| Graphical Coding | Galen |
| Coding for adding and removing contents | Galen |
| Message Encryption | Galen |
| Ensuring each phase meets standards | Marc |
| Database Programming | Marc |
| Video Calling Programming | Marc |
| System Design Plan | Josh |
| Coding for sending / receiving messages | Josh |
| Coding for audio messages | Josh |

# Resource Allocation