horizontal line

Software Development Plan

**March 6, 2018**

Version 1.5

**Presented To:**

CSC 354-020

**Submitted By:**

Daniel Carman, Nina Schnyder, Dana Fidler, Roksana Ferkh

# 

# **REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Author** | **Distributed to** | **Version** |
| 02/19/18 | All | All | 1.0 |
| 02/20/18 | All | All | 1.1 |
| 02/22/18 | All | All | 1.2 |
| 03/08/18 | All | All | 1.3 |
| 3/21/18 | All | All | 1.4 |
| 4/5/18 | All | All | 1.5 |

# 

[**REVISION HISTORY**](#_pgcy09v2wx37) **1**

[**PRODUCT DESCRIPTION**](#_7mtis1co57cx) **3**

[**2. TEAM DESCRIPTION**](#_ymbf41qnkz2h) **3**

[**3. SOFTWARE PROCESS MODEL DESCRIPTION**](#_c5rpsdy8g2ak) **4**

[**4. PRODUCT DEFINITION**](#_fg33x9l1ghtq) **5**

[Context Diagram](#_sor3mv2va6si) 5

[Use Cases (click for individual descriptions)](#_x4mwbpu4h3yp) 6

[Matrix of Responsibilities](#_76f2encz891w) 12

[**6. PERT Chart**](#_3gb230qwgi5x) **12**

[Test Strategy](#_nd902swyx7ay) 13

[Wireframes](#_opjyn7jz7xz5) 13

[**8. RISK ASSESSMENT**](#_g0haqwiz9b7r) **14**

[Risk Identification](#_ktvmtr616f2z) 14

[Risk Prioritization](#_rdxymbv0ypb8) 14

[Risk Mitigation](#_nquf0c8tiwan) 14

[**9. CONFIGURATION AND VERSION CONTROL**](#_jhw9lddartcc) **15**

[**10. TOOLS**](#_7ls0gdjth5rn) **15**

[**11. ARCHITECTURE**](#_dpjl10ilxtj2) **15**

[**\*\*\* Check hyperlink for larger image**](#_93wwh388czsf) **1**

# **PRODUCT DESCRIPTION**

Finding record sellers in a local area can sometimes be a challenge, especially considering some places that sell them are not specifically record stores. On top of that, knowing what these stores have in stock is largely impossible unless you follow every single social media post they make. Our goal is to make an application that bonds record store collectors to record store owners to make it easier to find certain albums users are interested in purchasing.

The potential audience for this app would be anyone who has a passion for records and collecting and wants to have easier access to record shops and their location. Our audience can create an account depending if they are a record store owner or just a collector. If the user wishes to create a store owner account, they can upload records from their shop and connect with collectors who are interested.

If a user wishes to create a collector account they can connect with other users and create a wishlist on the app. They then can be matched to local records stores that have similar inventory as the user has on their wishlist. The collector can also browse pictures, updates, and any deals/sales posted by record stores.

# **2. TEAM DESCRIPTION**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Team Members  Concepts | Nina Schnyder | Dana Fidler | Roksana Ferkh | Daniel Carman |
| Database Management | X | X |  | X |
| GPS Implementation | X |  |  |  |
| IOS: Swift/ Objective C/ C# |  |  |  |  |
| Android: Java/ Ruby | X |  | X | X |
| Web Scraping |  |  |  |  |
| Security(hash tables) | X |  |  |  |
| User Interface | X | X | X | X |
| User Management | X | X |  | X |
| Research | X | X | X | X |

The skills needed for this project are:

* Time Management
* Good Communication Skills
* Positive Attitude
* Interpersonal Skills
* Respect each others ideas
* Experience in programming
* Experience in database management
* Everyone is willing to learn

As a whole, our team is diverse in many concepts needed for this project. Dana, Nina and Roksana come from a software background, so the programming/design aspect of this project is under control. Dan has more experience in IT, so he have good grasp on database and user management.

There is no need for a Subject Matter Expert (SME) because of the knowledge Dan and Nina share about record stores and record hunting. They both share the perspective of the customer in what would be ideal as an app to search for specific records.

There is expertise missing in the app development and the specific programming languages used to create them. Also, there is little to no expertise in GPS, web scraping, and security.

# **3. SOFTWARE PROCESS MODEL DESCRIPTION**

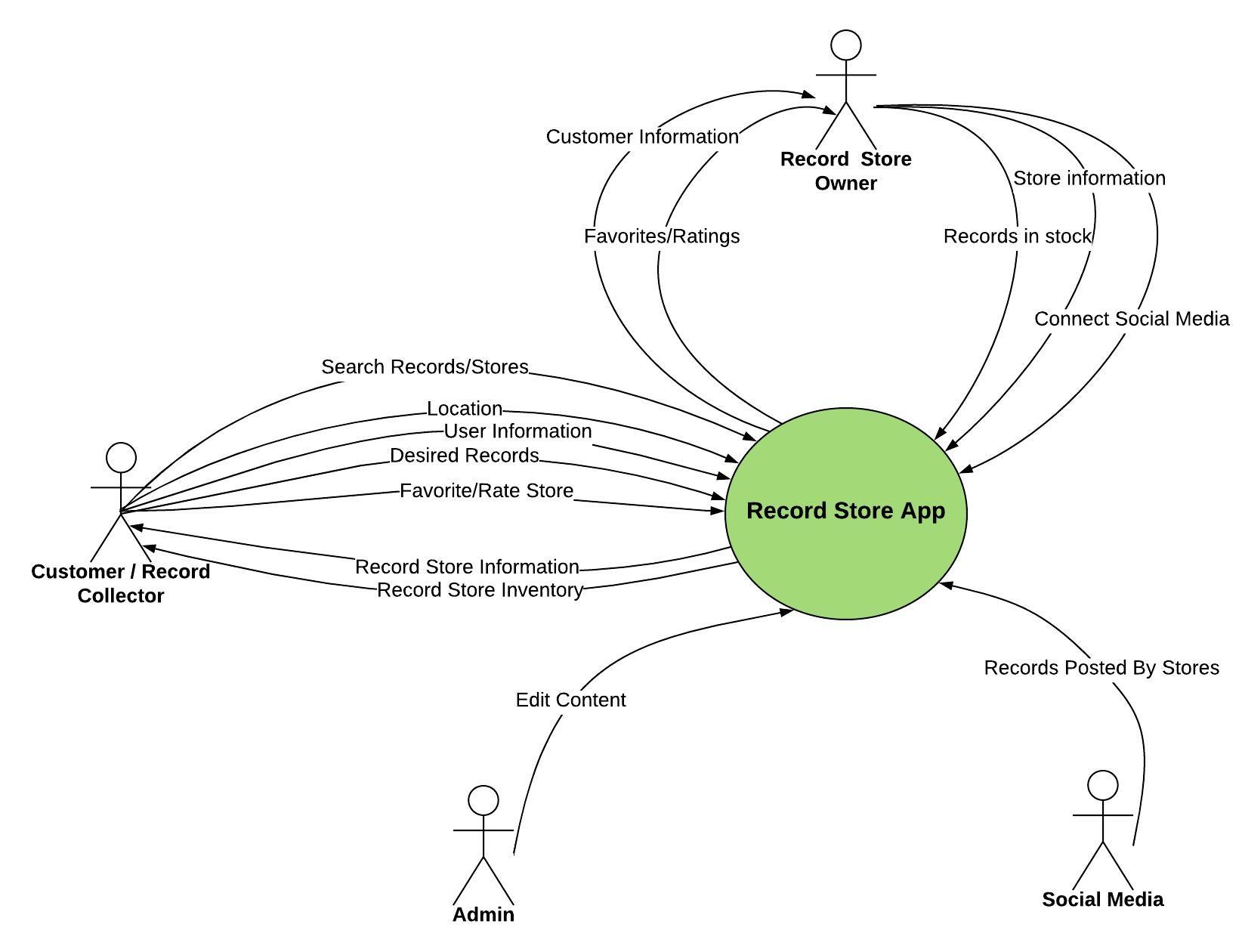
We are going to use the Agile method of app development for multiple reasons. The largest reason is that because the nature of the app is not mission critical, so the tedious steps in waterfall used to find and correct any errors is not entirely necessary. We also have about 10 months to complete the project, so speed of development is critical.

# 

# 

# **4. PRODUCT DEFINITION**

## **Context Diagram**



**Store Owners:**

The store owners in this case are people who own record stores. These owners can create a special account that have the information about their store such as hours, address, telephone number, location, name, etc. Other users/collectors can view their page and gather information on where they are and hopefully go visit. They can also have posts that share their sales, stock and other fun things going on in the store.

**Users/Collectors:**

The users are the everyday record collectors/appreciators. They can create an account with a username and password and they need to input an email. These users will create a running wishlist of specific records they are looking for or want to purchase. The user can also look up the record they want and see if any local shops have it in their system. If a user is looking for a specific record shop, they can also look up them too. The user can like the record shop to be notified when certain events happen.

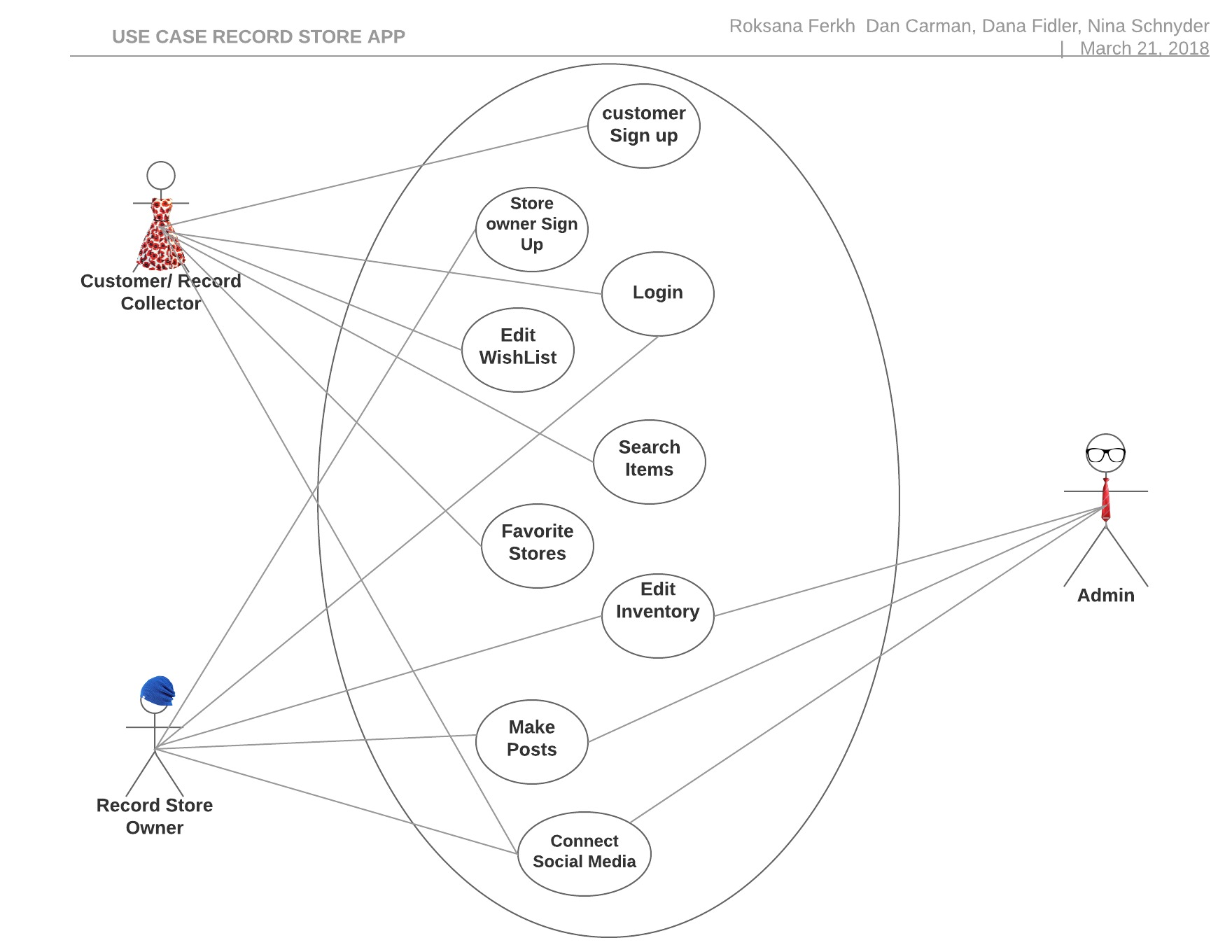
**Admin:**

As an admin to the app, we want to make sure no one is created fake record store accounts. This is to protect our users and store owners from spam and fake information.

**Data from Instagram:**

We want to make it as easy as possible for record store owners to upload records to our app. We found that many of these stores are posting regularly to a instagram with updates on their stock and sales. We want to create a system using #hashtags to get information pulled from instagram directly to our application.

## **[Use Cases](https://docs.google.com/document/d/12XgfNrVNvVZUDHk7BYBtv-j7iTASqcnU_EvzfFDVBv4/edit) (click for individual descriptions)**



**Use Case #1.1:**

**Name:** sign up

**Participating Actor(s):** Teacher/Instructor

**Entry Conditions:**

- The teacher accesses the signup page of the course platform.

- The teacher has an email address and other necessary information ready for signup.

**Exit Conditions:** The teacher's account is successfully created on the course platform, and they gain access to the instructor dashboard.

**Flow of Events:**

1. Navigate to Signup Page: a. The teacher navigates to the signup page of the course platform.

b. The signup page provides fields for entering personal information and creating an account.

1. Enter Personal Information:

a. The teacher enters their full name, email address, and other required personal details into the signup form.

b. Optionally, the teacher may provide additional information such as educational background or teaching experience.

3. Choose Username and Password:

a. The teacher selects a unique username for their account.

b. The teacher chooses a secure password and confirms it for account security.

4. Verify Email:

a. After completing the signup form, the platform sends a verification email to the provided email address.

b. The teacher checks their email inbox and clicks on the verification link to confirm their email address.

5.Complete Signup Process:

a. Upon email verification, the teacher is redirected back to the course platform.

b. The teacher's account is successfully created, and they gain access to the instructor dashboard.

**Special Requirements:**

- The signup form should include fields for essential information such as name, email address, username, and password.

- The platform should implement email verification to ensure the validity of the teacher's email address.

- Teachers may be required to agree to the platform's terms of service and privacy policy during the signup process.

- User-friendly error messages should be displayed if any signup form fields are incorrectly filled or if email verification fails.

- The platform should prioritize the security of user accounts by enforcing strong password requirements and implementing measures to prevent unauthorized access.

**Use Case #1.2:**

**Name:** Signup

**Participating Actor(s):** Student

**Entry Conditions:**

- The student accesses the signup page of the course platform.

- The student has an email address and other necessary information ready for signup.

**Exit Conditions:** The student's account is successfully created on the course platform, and they gain access to the student dashboard.

**Flow of Events:**

1. Navigate to Signup Page:

a. The student navigates to the signup page of the course platform.

b. The signup page provides fields for entering personal information and creating an account.

1. Enter Personal Information:

a. The student enters their full name, email address, and other required personal details into the signup form.

b. Optionally, the student may provide additional information such as educational background or interests.

3. Choose Username and Password:

a. The student selects a unique username for their account.

b. The student chooses a secure password and confirms it for account security.

4. Verify Email:

a. After completing the signup form, the platform sends a verification email to the provided email address.

b. The student checks their email inbox and clicks on the verification link to confirm their email address.

5. Complete Signup Process:

a. Upon email verification, the student is redirected back to the course platform.

b. The student's account is successfully created, and they gain access to the student dashboard.

**Special Requirements:**

**-** The signup form should include fields for essential information such as name, email address, username, and password.

- The platform should implement email verification to ensure the validity of the student's email address.

- Students may be required to agree to the platform's terms of service and privacy policy during the signup process.

- User-friendly error messages should be displayed if any signup form fields are incorrectly filled or if email verification fails.

- The platform should prioritize the security of user accounts by enforcing strong password requirements and implementing measures to prevent unauthorized access.

- Optionally, the platform may offer social media signup options or integration with existing accounts (e.g., Google, Facebook) for a smoother signup process.

**User Case #2:**

**Name:** Login

**Participating Actor(s):** Student

**Entry Conditions:** Having the email and password that were previously registered on the platform.

**Exit Conditions:** The platform has been successfully accessed and the student has permission to use the services provided.

**Flow of Events:**

1. Enter your email.
2. 2- Enter your password.
3. 3- Confirm the log in.

**Special Requirements:**

1. If you enter an incorrect email and password, you must review the email and password with which you registered for the platform.
2. 2- If the student has not registered before, he must register through the registration interface using the name, email, password, and attached photo (optional).

**Use Case #3:**

**Name:** Search

**Participating Actor(s):** Customers

**Entry Conditions:** User is logged in to a customer account

**Exit Conditions:** Search results are returned

**Flow of Events:**

1. Customer enters terms they wish to search for
   1. Store
   2. Album
   3. Artist
   4. Genre
   5. Label
   6. Year
2. List of results is returned

**Special Requirements:** No special conditions

**Use Case #4:**

**Name:**Course Feedback

**Participating Actor(s):** Student

**Entry Conditions:** The student has completed the selected course.

**Exit Conditions:** Feedback is recorded for the course.

**Flow of Events:**

1. Customer finds item they want added to list
2. Item is added to account’s wish list

**Special Requirements:** If the feedback form submission fails, display an error message and allow the user to retry.

**Use Case #5:**

**Name:** Teacher Adds Quizzes/Assignments

**Participating Actor(s):** Customers

**Entry Conditions:** The teacher is logged into the Language Learning Platform.

**Exit Conditions:** Quizzes and assignments are added to the course.

**Flow of Events:**

1. Log in to the platform as a teacher.

2- Accesses the course management dashboard.

1. Adds quizzes and assignments with relevant content and due dates.

**Special Requirements:**: If there are issues saving the quizzes/assignments, display an error message and allow the teacher to retry.

**Use Case #6:**

**Name:** Teacher Adds/Edit Course

**Participating Actor(s):** Teacher

**Entry Conditions:** The teacher is logged into the Language Learning Platform.

**Exit Conditions:** A new course is added, or an existing course is edited.

**Flow of Events:**

1. Log in to the platform as a teacher.
2. Accesses the course management dashboard.
3. Clicks on "Add New Course" or selects an existing course to edit
4. Fills in course details and content.

5- Saves the new course or updates the existing course.

**Special Requirements:** If there are issues saving the course, display an error message and allow the teacher to retry.

**Use Case #7:**

**Name:** Receive Course Certificate

**Participating Actor(s):** Student

**Entry Conditions:** The student has completed the selected course.

**Exit Conditions:** The student has completed the selected course.

**Flow of Events:**

1. Log in to the platform.

2 -Navigates to the completed course.

3- Downloads or views the course completion certificate.

**Special Requirements:** If there are issues generating the certificate, display a notification and suggest contacting support.

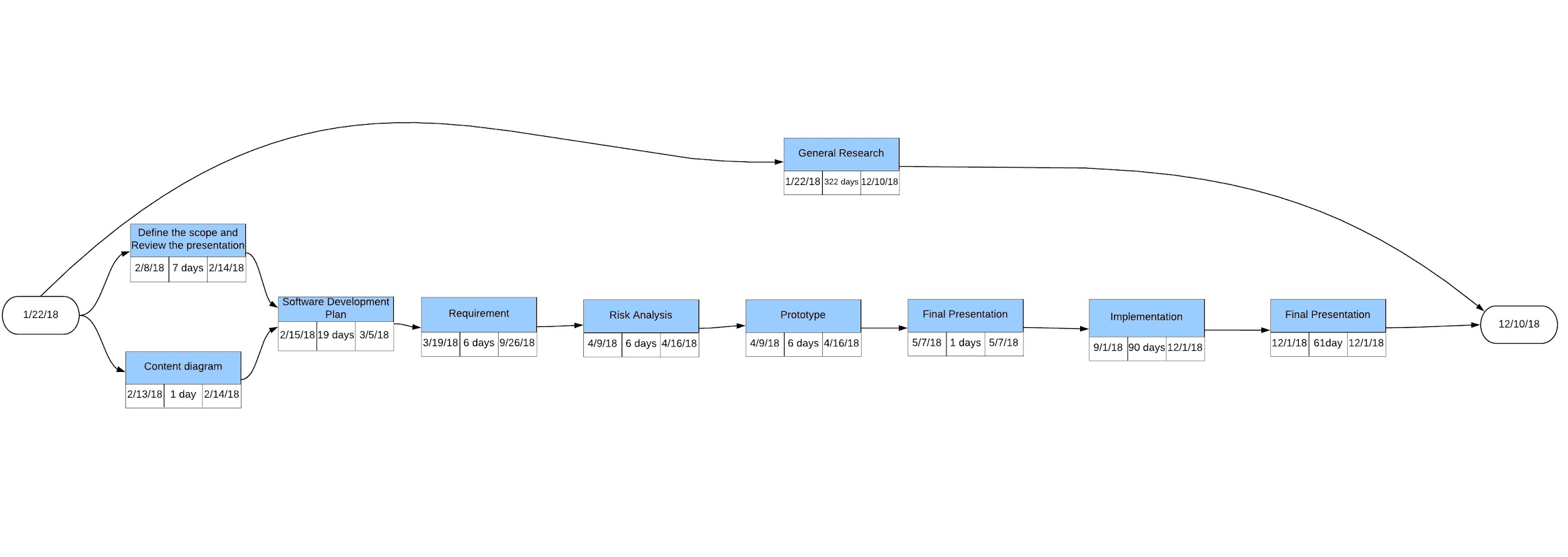
**5. PROJECT ORGANIZATION**

## **Matrix of Responsibilities**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Team Members  Concepts | Nina Schnyder | Dana Fidler | Roksana Ferkh | Daniel Carman |
| Database Management |  | X |  | X |
| GPS Implementation | X |  |  |  |
| IOS: Swift/ Objective C/ C# | X |  |  |  |
| Android: Java/ Ruby |  | X | X |  |
| Web Scraping | X | X |  | X |
| Security(hash tables) |  | X |  |  |
| User Interface | X |  | X | X |
| User Management |  |  |  | X |
| Research | X | X | X | X |

\*\*\*Underlined X represent leader on subject

# **6.** [**PERT Chart**](https://drive.google.com/open?id=1rj1XyvQ63gfzoeDCI-cp3nwjir7VcBzN)



**7. VALIDATION PLAN**

## **Test Strategy**

The definition of done can be described as completing a list of requirements to have the app ready to be used by the due date of the project, and have the customer find the record that they desire.

Success from our point of view is to have the app functioning properly and conveniently by having one person able to find a record they have been looking for. Be a reliable app that collects data from other social media making it useful as a central location for finding record store information.

Our main test plan will be as follows:

1. Create both a collector account and a store account
2. Link an Instagram account to the store account
3. Upload a few posts of records on the instagram
4. Search for those records using the collector account

Success will be if we can successfully find the uploaded record as a collector.

## **Wireframes**

## 

# **8. RISK ASSESSMENT**

## **Risk Identification**

As developing an mobile Application, our biggest risk is not getting the app to be noticed, so no one would use it. Another risk can be security. Insecure data storage and untrusted inputs are also big risks.

Our web scraping not effective enough to pick up on social media posts made by record stores- making the record app unreliable. A big factor in this app is it being able to be a central location for information; if we cannot supply that then it’s a risk.

We are largely new to mobile app development, so learning these languages on the fly in a relatively short period of time is a risk, because not being successful in this learning process would undermine the entire project

## **Risk Prioritization**

1. The application is not noticed and no one would use it.
2. Insecure data storage.
3. Unreliable collection of record stores social media posts.

## **Risk Mitigation**

Promoting and advertising the Record Store Application to show all the good features this app has, is a one way to get the other stores to use it for a small community then grows further to have more people use it. Security plays a big role in developing mobile applications. Data should always be stored within an encrypted data section and the app should be marked to disallow backup. Also, authorization of entered data from the other record stores is going to be considered to make sure we have a reliable data

The should also be a reliable collection of record store information so that this app can become a central location for people searching for specific records. We must make sure that our web scraper is collecting all the information possible.

# **9. CONFIGURATION AND VERSION CONTROL**

For the app: X.Y.Z

X = Major Releases

Y = Minor Feature Updates/Cosmetic Updates

Z = Bug Fixes

Example: First major release, third minor feature update, second bug fix :: Version 1.3.2

In order to keep documentation and different files in order with everyone each one will have a table similar that keeps information on all updates made after initialization and who made each change.

# **10. TOOLS**

* Database Management system.
* App Development software (android and ios)
  + Visual Studio
  + xCode
  + Unix
* Web Scraper to collect data from Instagram
  + Using hashtags
  + Collect Record Information
  + Collect Sale Information

# **11. ARCHITECTURE**

* Devices for testing
  + Android
  + iPhone
* Computers for programming