VinePilot
Project Requirements Document

Team Name: Savvy-gnon
Project Name: VinePilot
Team: Amy Chen, Iris-Eleni Moridis (S), Richard Waltman (L), Chien Wang, Ariel Xin

Revision History

No revisions so far.

Intro

VinePilot is an application that will allow all of the data to be accessible to the farmer, giving them the ability to store information on particular blocks of their vineyards to facilitate the process of screening for ripeness, monitoring the spread of disease and viewing the amount of irrigation. Additionally, they will be able to add notes to any given block in order to further personalize the application and show them the effect of their proactive actions.

Glossary of Terms

VinePilot: Name of the application.

Server: A running instance of an application capable of accepting requests from the client and giving responses accordingly. Stores all user information and images.

CocoaPods: A manager for the objective-C language that provides standard formats for citing external libraries.

System architecture overview – High level picture

The system architecture is built around the applications' communication with the server to get near-infrared images of individual vineyards. Images will be stored on the UCSB Servers as will the user's notes, pins, and images uploaded to the app.

Both the web application and the iOS application will begin with a login screen. Once a correct username and password have been given, the application will automatically load the *most recent* picture of the farm from the server. There will be a menu function that allows the user to do the following:

- Drop pins
- Add notes (farm level, pin level to be implemented at a later time)

- View History **-** the user will be able to select an image from all the available dates from when the image was taken.

Requirements (functional and non-functional)

***Each user case is implemented twice, on both the Web Application and the iOS Application**

1. As a user, I want to log in so that my data is secure
    a. Acceptance Test:
        i. *Scenario:* The user opens the application
        ii. Given the application is open, when the user provides a correct username and password, then the application will forward the user.

2. As an application, I want to reject any user with incorrect login information
    a. Acceptance Test:
        i. Scenario: The user enters the wrong login information
        ii. Given the user enters the wrong combination of account name and password, when the user presses “Log In” button, then the application will reject the login and give an error message.

3. As a user, I want to see only my farm’s data
    a. Acceptance Test:
        i. *Scenario:* The user logs into the application
        ii. Given the application is opened for the first time, when the user enters the correct login information, then the user will be able to see only the specified farm’s information.

4. As a user, I want my password to be hidden while I type so that nobody can see my data but me
    a. Acceptance Test:
        i. *Scenario:* The user is entering their password into the login screen
        ii. Given the user is on the login screen, when the user starts typing in their password, then the password box will only display *’s.

5. As an application, after login I want to go to the menu page
    a. Acceptance Test:
        i. *Scenario:* The user successfully logs in with their username and password
        ii. When the login is successful, then the application will forward the user to the full application.

6. As a user, I want to be able to set notes for my farm
    a. Acceptance Test:

        i. *Scenario:* The user is logged in and selects the *notes* function (accessible via menu)
        ii. Given the user is logged in securely, when they click the *notes* menu option, then a window dialogue opens for them to write and save notes in.

7. As a user, I want to be able to drop pins
    a. Acceptance Test:
        i. *Scenario:* The user is logged in and selects the *pin* function (accessible via menu)
        ii. Given the user is logged in securely, when they click the *pin* menu option, then a pin marker will replace the mouse and drop on the map when clicked.

8. As a user, I want to be able to select images of my farm throughout the history of the service
    a. Acceptance Test:
        i. *Scenario:* The user is logged in and selects the *history* function (accessible via menu)
        ii. Given the user is logged in securely, when they click the *history* menu option, then a list of all available pictures will be given.

9. As an application, I need to load the correct picture when a different picture from *history* is selected by the user
    a. Acceptance Test:
        i. *Scenario:* The user is logged in and clicks a **different** picture than what is currently shown
        ii. Given the user is in the *history* menu and clicks an option **different** than what they currently see, then the application must *get* the correct image from the server and display it.

10. As a user, I want to zoom in and out of my farm
    a. Acceptance Test:
        i. *Scenario:* The user is logged in and pinches/scrolls out/in
        ii. Given the user is using the application and pinches/scrolls out/in, then the map should zoom in and out accordingly

Prototyping code and test cases (Github URL)
iOS: https://github.com/PARanOiA1120/VinePilot_iOS
Web: https://github.com/rtwaltman/VinePilot_Web

System models - N/A

Appendices

1. Software Tools
    a. iOS
        i. XCode - for Objective-C as well as the built in XCode simulator
        ii. Cocoa Pods - A manager for the objective-C language (iOS) that provides standard formats for citing external libraries.
    b. Web
        i. WebStorm - IDE for Javascript, HTML5, and CSS
        ii. QUnit - Javascript unit testing for TDD
        iii. JQuery - Javascript API for web development
        iv. Bootstrap - HTML/CSS framework for formatting webpages
2. Server
    a. Each application uses the same UCSB server, which will allow the users to have the same data on both.