Binary Ninjaz

Harvest

Requirements and Design Documentation

Letanyan Arumugam	14228123
Sizo Duma	15245579
Teboho Mokoena	14415888
John Ojo	15096794
Kevin Reid	15008739
Shaun Yates	16007493

Stakeholders

SAMAC: Barry Christie

Contents

List of Figures

1 System Overview

1.1 Purpose

The need for this system is due to the fact current yield tracking and measuring worker performance is done manually and on paper. This allows for a more efficient way of carrying out that process. This more efficient process will help farmers better track and manage produce collection. Finding cold spots and irregularities in orchards or by workers.

1.2 Project Scope

The goal of the project, Harvest, is a system to assist growers with yield data and optimise worker performance. The aim of the project is to produce a system that can efficiently measure the amount of work done by a worker, track the foremen on a farm, record information and data, and display the necessary information.

1.3 Definitions

- Farmer - The lead manager of a farm.
- Foreman - Overseer of work done by workers in the field.
- Worker - Labourer that collects produce from the farm lands.
- Farm - A collection of orchards with a similar crop.
- Orchard - An area allocated by the farmer for where produce may be collected.
- Collections - Produce collected by workers on a farm.
- Entity - Either a farm, orchard, worker or foreman.
- Session - The period when foremen will be tracking workers collecting produce.

1.4 UML Domain Model

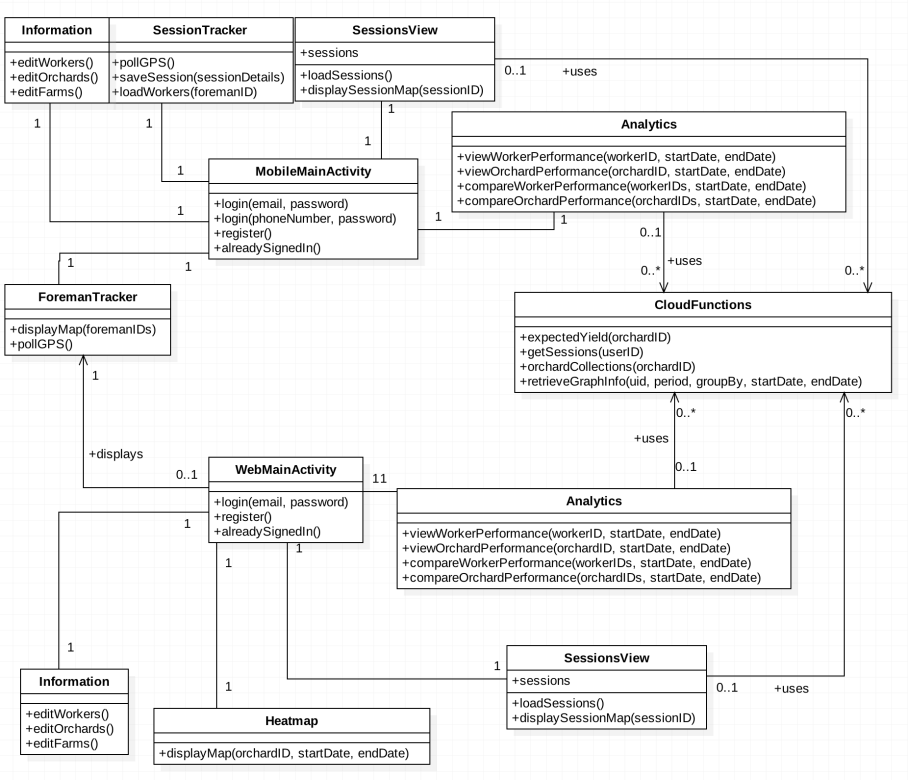

Figure 1: Domain Model

2 Functional Requirements:

2.1 Users

2.1.1 Farmers

Farmers will be the main beneficiary for using the system. Information about what data is collected and analysis on said collected data is provided. Managing administrative details is possible however not the main focus. It is rather for helping the system provide more specific insights into where and how collections are made. The system produces graphs, heatmaps and summaries of work done by entities. It is on the farmer to critque the data to find where performance can be improved or work may need fixing.

2.1.2 Foremen

Foremen will use the system to track how much each worker collects. The system will log where and when the user logs a collection. Foremen simply need to tap on button next to a workers name when said worker collects a bag of produce.

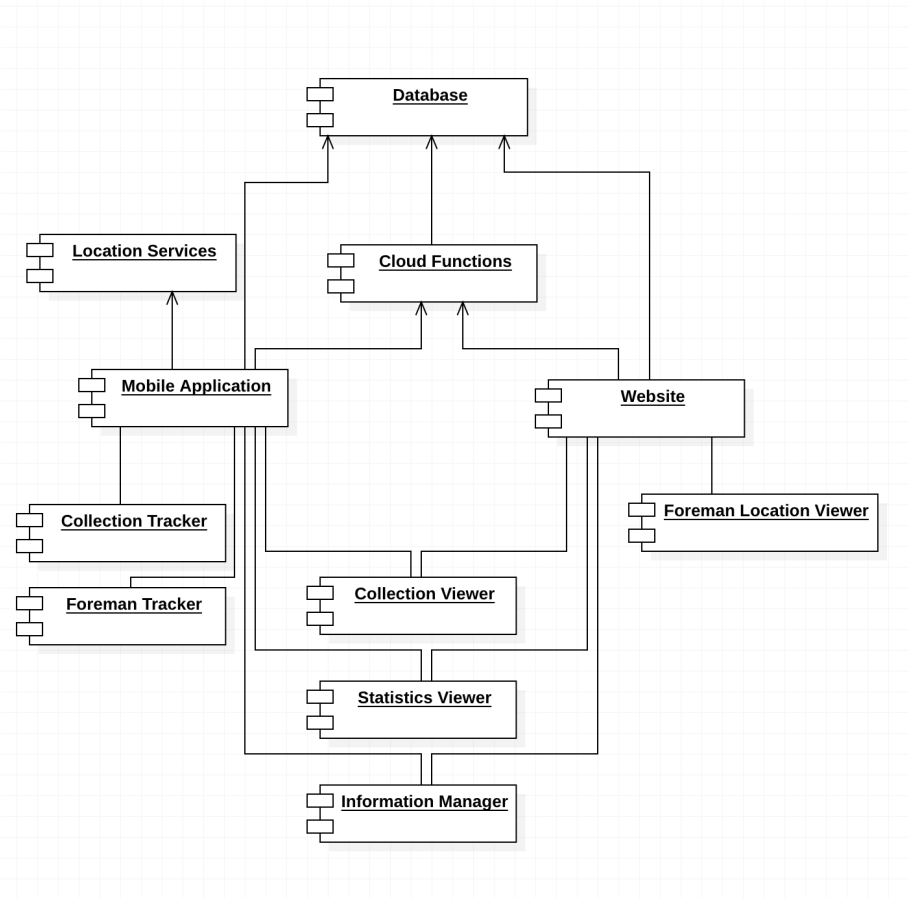

Figure 2: Component Diagram

2.2 Subsystems

- Database - Stores information such as entity details and collection data.
- Location Services - Manages location information regarding the foremen's mobile devices.
- Cloud Functions - A helper service that runs intensive computation on servers where the database is stored.
- Mobile Application - The mobile device application used in field.
- Website - A web based interface to interact with the system.

See 2 for diagram representation.

2.3 Specific Requirements

- On sign in Harvest shall verify credentials against the database before signing them in.

- Harvest shall create a profile on the database for the user through a registration form.
- Harvest shall measure yield through the input view of the mobile interface.
- The Harvest mobile interface shall make use of GPS data and based on weight and location assumptions, it must give approximate yield estimates not only for each orchard, but for each approximate location where the data was entered.
- The Harvest system shall make use of analytics to measure worker performance.
- The Harvest website shall display heat map data, showing the locations in which the most yield has been collected.
- The Web interface shall show detailed information about produce. (e.g. cultivar, year planted).
- The Harvest system shall do administrative tasks through the web interface.
- The Harvest website shall give the farmer/administrator a real time map view of the foreman's location.
- Both foremen and farmers shall have access to the mobile interface, with foremen unable to carry out any administrative changes or view analytical data on it.
- Only farmers shall make use of the website, as it is designed to track their workers, foremen and measure performance in and around their farms.

3 Non-functional Requirements

3.1 Conservative Battery Usage

3.1.1 Why?

Foremen will be expected to use the mobile application throughout a session, which may last multiple hours. If mobile devices were to be running out of battery before the end of session the services will not be able to produce accurate productivity measurements and statistics.

3.1.2 How?

Reducing location accuracy might not provide the most accurate results. The battery savings are more important than having perfect accuracy. Network transactions will also be limited to only a few calls only when necessary. Heavy usage of cache storage will be employed. Calls requested data changes will be made and requests to only download new data and changes will be download instead of full re-downloads of data.

3.2 Intuitive User Interface

3.2.1 Why?

The target audience of the system are people who care about doing their work and not about learning a new systems.

3.2.2 How?

We intend to use the design languages that are well established on the platforms that the user facing experience will be displayed on. Following design guidelines set by Apple [1] and by Google Android [2].

3.3 Smooth User Experience

3.3.1 Why?

Having an experience that requires a steep learning curve or has sharp edges will quickly lead to people reverting back to an old system. Even if the old system has the same sharp edges their known about and users are used to it. Hence to make sure users move away from their old system of completing their tasks, we must provide as smooth a transition as possible.

3.3.2 How?

The video game industry has faced this problem since their inception. We will look to game designer to see how they have solved their problem of introducing and teaching concepts to their players seamlessly. One such example is tutorial systems, where to introduce a new concept, the user is faced with a problem that has one and only one obvious solution. We will employ this technique in our implementation of the system.

4 System Architecture

4.1 Determining Design Objectives

- The design is aimed to provide accessibility, data integrity and optimize performance.
- The design is also aimed at producing a user friendly easy to use system.
- The design is aimed at easing facilitation and automation of work.

4.2 Interfaces

4.2.1 User Interfaces:

- Mobile Application Interface
  - Authentication Screen: Provides a way for farmers and foremen to sign into the system. Foremen will sign in using a mobile phone number sign in. Foreman sign in will produce a different view which will only show the yield collector. Farmer sign in will show all options available.
  - Yield Collector: Track the work done by each worker during a session.
  - Information: Manage information about entities.

[1] https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/
[2] https://material.io/design/

  - Sessions: View raw session data. Such as time started and ended, where and when pick ups were made.
  - Stats: View graphs about over periods of time regarding entity comparison differences.
  - Settings: Manage certain settings such as login details. View help about the system is also provided.

- Web Page Interface
  - Authentication Screen: Provides a way for farmers to sign into the system.
  - Foreman Tracker: Track where foremen currently are during a session.
  - Information: Manage information about entities.
  - Sessions: View raw session data. Such as time started and ended, where and when pick ups were made.
  - Stats: View graphs about over periods of time regarding entity comparison differences.
  - Settings: Manage certain settings such as login details. View help about the system is also provided.

4.2.2 Hardware Interfaces:

Most hardware interaction aside from basic usage will be with GPS systems found on mobile devices. The system will use GPS data provided by mobile devices to tag collection data and track foremen during sessions.

4.2.3 Software Interfaces:

- Firebase Realtime Database: The system interacts with the database using firebase API on Swift, Java and Javascript. This interaction is simple for storage and retrieval of JSON data.

- Google Maps SDK: To provide map data we use the SDK provided by Google for Swift, Java and Javascript. The SDK is used for map image data. Map overlays provided by the SDK are also used. Overlays such as heat-maps, markers and polygons are used to provide a more complete user experience.

4.3 Architectural Styles

The architectural structural design of our the Harvest system is a Persistence Framework. The system has 3 main subsystems (Android Application, IOS Application and the website) all heavily reliant on the database system providing them with efficient object storage and retrieval without the need for implementation detail. It also hides the database details such as implementation from the subsystems, and therefore uses an abstract interface to manipulate data.

4.4 System Configuration

4.4.1 Equipment

- Any mobile device supporting Android 4.0 or above with GPS services.
- Any iOS device supporting iOS 9.0 or above.
- Any device with modern web browser support.

4.4.2 Development Technologies Used

- Firebase Realtime Database, Cloud Functions, Authentication
- Development Environments: XCode, Android Studio, Atom, WebStorm, Notepad++
- Development Language: Java, XML, HTML, Swift, Javascript

4.4.3 Deployment Diagram

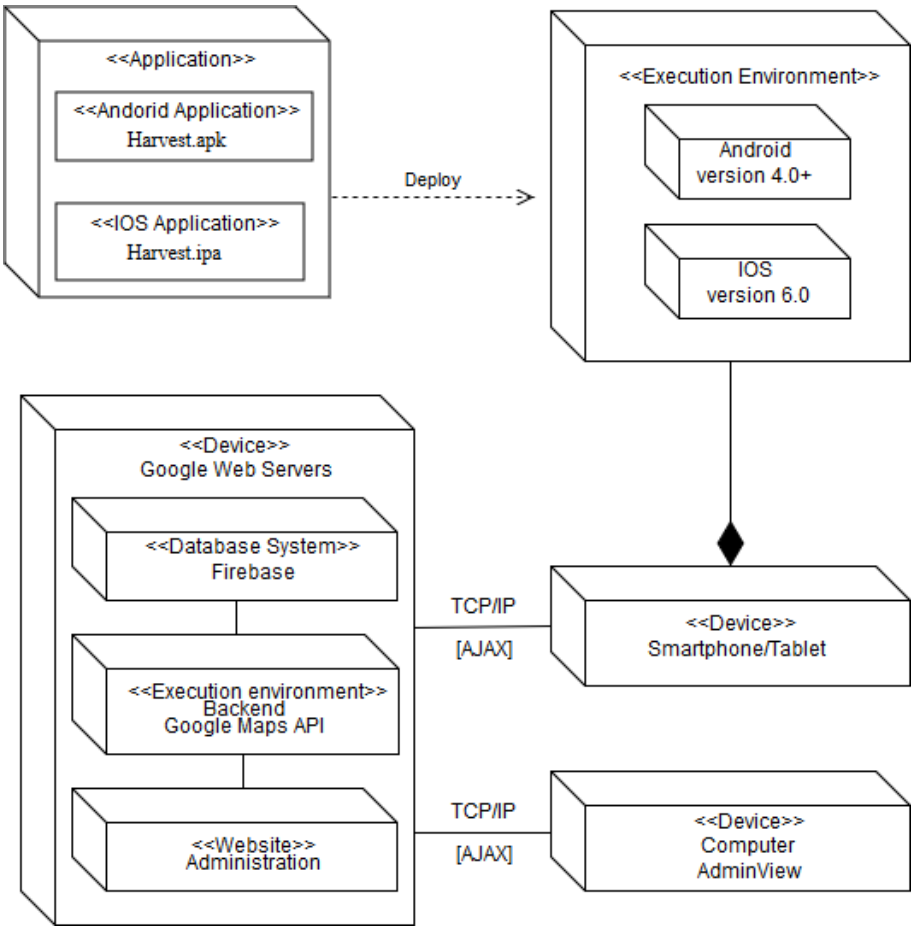

Figure 3: Deployment Diagram