Bilkent University

Department of Computer Science

Senior Design Project

GrapeHealth

Final Report

İlhami Özer, İbrahim Berker Kırdök, Murat Süerdem, Osman Sefa İbiş, Göktuğ Özdoğan

Supervisor: Prof. Dr. Halil Altay Güvenir
Jury Members: Prof. Dr. İbrahim Körpeoğlu - Doç. Dr. Özcan Öztürk
Innovation Expert: Kerem Erikçi (tarla.io)

Website: https://github.com/brker/GrapeHealth

Final Report
May 9, 2019
This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table of Contents

Table of Contents **1**

List of Figures **2**

Introduction **3**

Final Architecture and Design **5**

Subsystem Decomposition 5

Hardware/Software Mapping 8

Persistent Data Management 9

Tools and Technologies **9**

React Native 9

Redux Library 9

Native-Base 10

Firebase Realtime Database 10

Apache Tomcat 9 Web Server 11

REST API 11

Google Compute Engine 11

Weather API by OpenWeather 11

Image Recognizer API by Imagga 11

Plant Identifier API by Plant.id 12

Deep Learning with CNN 12

Pycharm 13

Atom Editor 13

Android Studio 13

Github for Atom Editor 13

ImageView Library 13

React-Native(RNCamera) Library 14

React Native Maps Library 14

React Native Image Crop Picker Library 14

React Native Dialog Input Library 14

React Native Firebase Library 14

Image Background Segmentation 15

Impact of Engineering Solutions **15**

Economical Impact 15

Environmental Impact 16

Societal Impact 17

Contemporary Issues **17**

Appendix A: User Manual & Installation Guide **19**

List of Figures

Figure 1. Subsystem Decomposition 6
Figure 2. Redux Architecture 7
Figure 3. Hardware/Software Mapping 8
Figure 4. Welcome Screen 19
Figure 5. Create Account Screen 20
Figure 6. Login Screen 21
Figure 7. Queries Screen 22
Figure 8. Farm Select Screen 23
Figure 9. Camera Screen 24
Figure 10. Photo Edit Screen 25
Figure 11. Query Detail Screen 26
Figure 12. Treatment Screen 27
Figure 13. Disease Detail Screen 28
Figure 14. Menu 29
Figure 15. Diagnosis Traverse Screen 30
Figure 16. Nearby Screen 32
Figure 17. Weather Forecast Screen 33
Figure 18. Help Screen 34
Figure 19. Settings Screen 35

1. Introduction

In the last few decades, the integration of agriculture and technology is becoming more substantial due to the economical and sustainability benefits for farmers. Agricultural technology is easing the workload of the farmers and enabling them to produce better quality for harvesting. Turkey is a country of agriculture, therefore, the technological improvements in this sector will advance our economical standing. One option to integrate agriculture and technology is detecting agricultural diseases. Detecting diseases is a challenging concept since there are multiple diseases even for one plant and detecting every one of them is computationally challenging. Therefore, our main focus is detecting the diseases related with grapes.

Grape is the most produced fruit in Turkey's market share. Turkey has excessively link field for grape production and has great potential for viticulture. Also, Spain, France, Italy and China are the world leaders of grape production. The importance of detecting grape diseases arises in thinking of the worldwide impact. The main purpose of our innovation is to be able to detect common grape diseases to change the viticulture experience of the farmers to easily diagnose their grape products.

Farmers solely depend on agriculturalists to diagnose and cure their products, with our mobile application farmers can detect if their grape has disease or not. The potential users are farmers, agricultural engineers, government's agricultural consultants, agricultural groups, grape merchants and drug companies or chemical stores. Our ambition is to help agricultural specialists and groups in being an additional resource to detect diseases. Grape merchants can decide whether or not the product they will buy has any of the common diseases. Farmers and drug companies can communicate more biologically to decide the accurate drug for the grapes. In some circumstances chemical stores and farmers may not have the information to detect the diseases. Integrating the potential users to diagnose diseases will help grape production sector to communicate better. Detecting diseases will improve efficiency and quality by adding value to fresh and healthy grapes.

GrapeHealth is a mobile application (Android) which is designed to serve farmers, agricultural engineers, government’s agricultural consultants, agricultural groups, grape merchants and drug companies or chemical stores. The innovative part of GrapeHealth is integrating data, image and agronomy to diagnose grape diseases. The main purpose of the application is recognizing the disease of the grape plants and informing the users about their grape condition by either making them to take photos of the leaf or using our diagnosis feature. By detecting diseases, the target audience will be able to notice the disease after a wider effect might occur and take precautions according to the feedback from our application. Grape is an essential part of our nutrition which has an important place in Turkey’s market share and providing an information service to the providers or relevant people will increase healthy grape stock.

In this final stage of reporting, we will first discuss the final architecture and design of our project. Then, we will talk about the new technologies and tools we integrated into our application. The following section will discuss the impact of our engineering solutions in economical, societal, environmental contexts. Then, we will discuss and go in depth about the contemporary issues in the field of agriculture and viticulture of grape sector in turkey and worldwide. Finally, we will provide a user manual for GrapeHealth application.

2. Final Architecture and Design

This section includes final architecture and design for the project.

2.1. Subsystem Decomposition

We decided to change our architecture into Redux. Since we are developing our application in React-Native, Redux architecture is a better suit than MVC architecture. Redux is an open-source JavaScript library for managing application state. It is most commonly used with libraries such as React or Angular for building user interfaces. Similar to (and inspired by) Facebook Flux architecture. Implementing a Redux model was a big challenge for us, however, it was a good practice for all of us.

Our system can be decomposed into four main subsystems; Client, Server, Database and WebSocket. We preferred Client/Server architectural style, since data in database is private and should not be seen by third party.

In Client side, we use redux architecture. The View component is used for all UI logic of the mobile application. User directly interact with UI and see necessary changes on screen.

In Server side, Communication between mobile application and server is provided by WebSocket. Network Controller manages location suggestions by using Database and send some suggestions to the users in Client side through WebSocket. Suggestions are based on location of the user. Firebase Realtime Database aimed to use for storing relevant data. Also, Network Controller manages query results; Network Controller take input data(photo) and give it to the Learning Model in order to take prediction which represents query result.

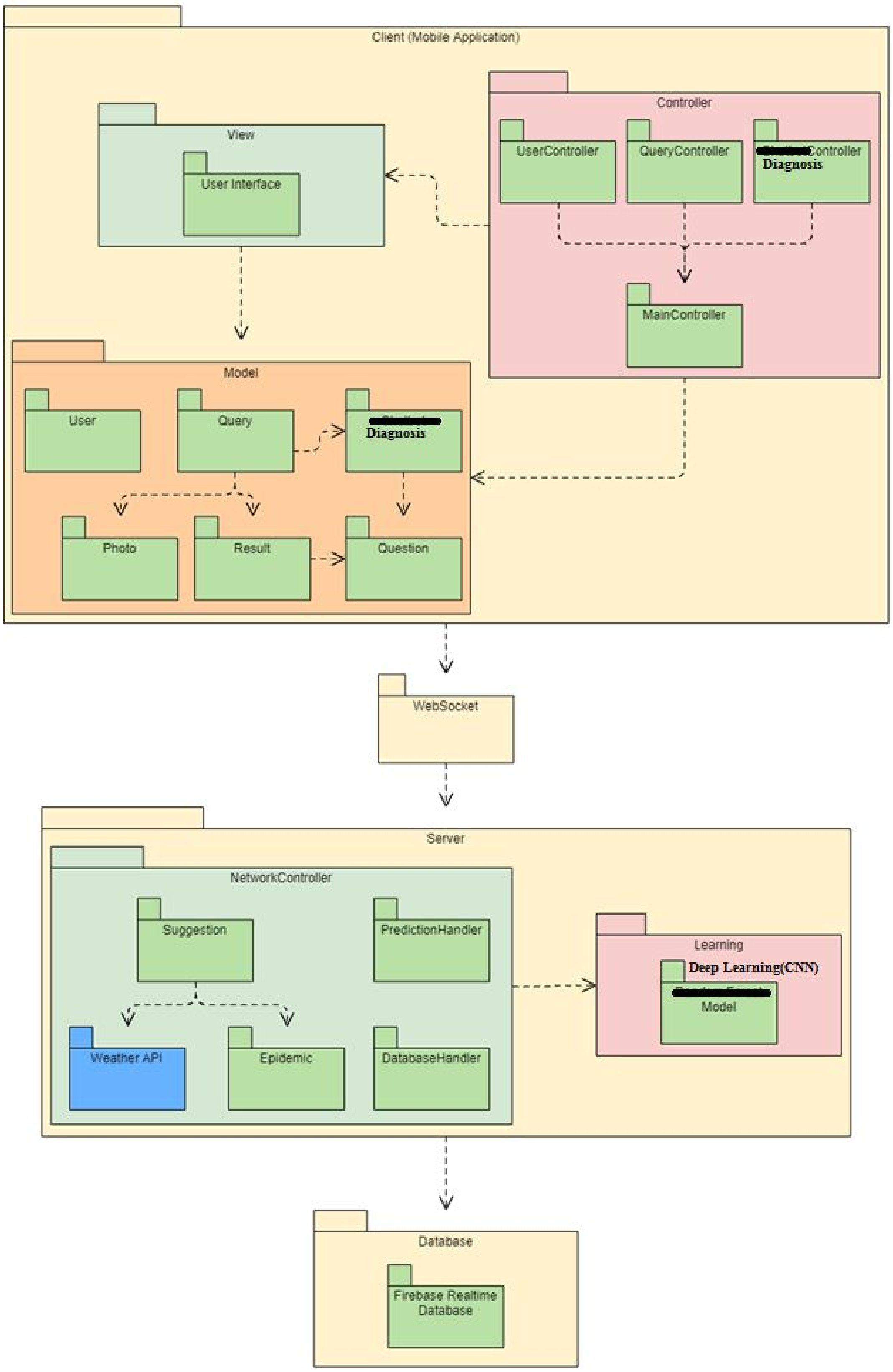


Figure 1. Subsystem Decomposition

2.1.1. Client Side

For the Client side of the application, we used Redux architecture. In order to do that we construct the application three parts: Views, Actions and Reducers. Figure below shows how the architecture works. When view part of a page takes an action, this action is sent to the action file of the page. This part is done what needs to be done (ie. *loginWithEmail()* function takes email and password and call firebase function to determine whether information are valid.) Then the reducer part compares previous and the changed state and if any change is found state is changed and sent to the view.

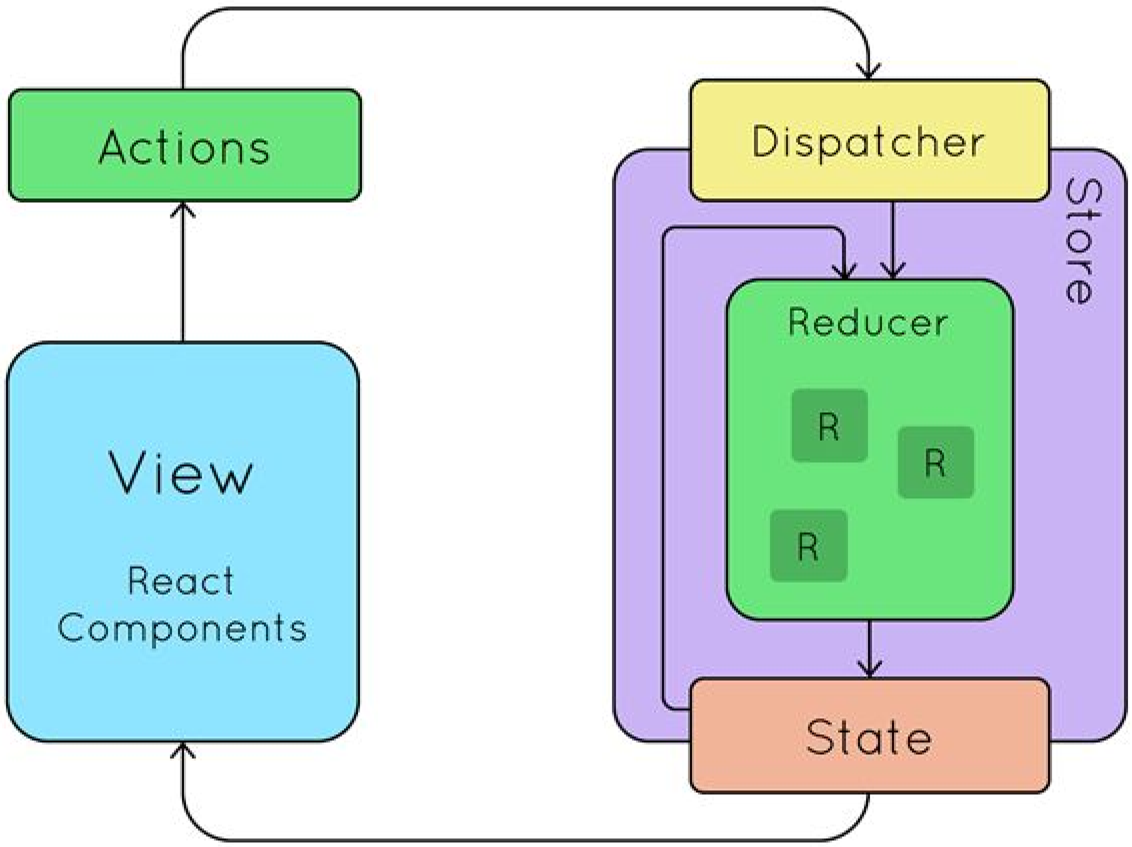


Figure 2. Redux Architecture

2.2. Hardware/Software Mapping

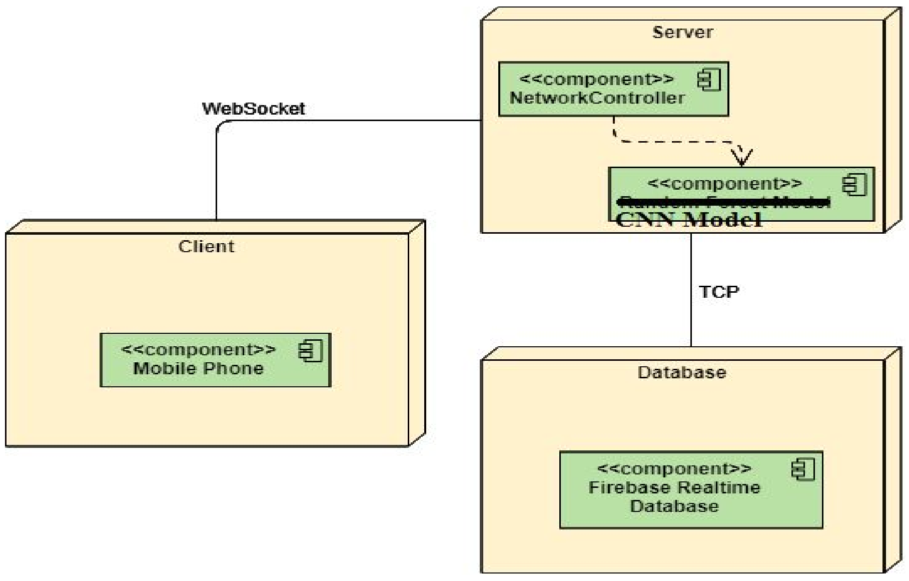


Figure 3. Hardware/Software Mapping

GrapeHealth runs on Android Phones. GrapeHealth mainly uses the mobile phones; camera, GPS, connectivity hardware and memory to detect the diseases of the grapes. On the mobile phones, GrapeHealth takes the photo from users and send it to the server. Deep learning model(Convolutional Neural Network) is applied on the server side. Therefore, Deep Learning Model do not use the user’s mobile phone to train and test. However, to able to take a picture and upload it to the program, mobile phone must have enough storage. React-native-camera library enables the program to use mobile phone’s camera. Communication between server and client are provided via WebSocket. WebSocket is a communications protocol, bi-directional, full duplex TCP connection from a user’s web browser to a server. WebSocket uses the handshake request from a browser’s HTTP connection and handshake request includes a 64-bit Sec-WebSocket-Key header. Database interaction is established via Google Firebase Realtime Database. Firebase is management tool for android developers. Creation of databases has been made on Firebase.

2.3. Persistent Data Management

The GrapeHealth application need to have a database to store queries (created by user to learn disease of specified grape by uploading photo of grape’s leaf or answering questions in diagnosis) and user identities. The quickness is very important for an application and user need swift access to his or her old queries with its multiple instances like result, photo, flowchart of diagnosis answered by specified user and treatment. Therefore, we used Firebase Realtime Database to store and interact with relevant data quickly. Data is stored as JSON and concurred in real-time to every connected client in Firebase Realtime Database, and it save a tremendous amount of custom development time.

3. Tools and Technologies

3.1. React Native

The front-end and back-end of GrapeHealth is developed with react-native. React Native is a JavaScript framework for writing real, natively rendering mobile applications. It’s based on React, Facebook’s JavaScript library for building user interfaces, but instead of targeting the browser, it targets mobile platforms.

3.2. Redux Library

Redux is a predictable state container for JavaScript apps. Redux is a state management library, and is often used with React Native to simplify data flow within an app. It helps you write applications that behave consistently, run in different environments. This means the entire data flow of the app is handle within a single container while persisting previous state. Redux can be broken down into few sections while building the application which are actions, reducers, components. In our project we used, redux architecture to control our data flow with state changes with every rendering. It allowed us to have control over the application.

3.3. Native-Base

We used native-base library for UI design(front-end) of our application. We used a free version provided by native-base which is “kitchen-sink”. NativeBase is a mobile application development framework which enables developers to use React Native to build native mobile applications running on the major mobile platforms. The applications stack of components is built using native UI components and because of that, there are no compromises with the User Experience of the applications. NativeBase is targeted specially on the look and feel, and UI interplay of your app. NativeBase without a doubt fits in well with mobile applications which cut downs one huge part of your app The Front end.

3.4. Firebase Realtime Database

We used firebase to store user related data (farm locations, credentials, query information, nearby diseased farm coordinates and disease information, diagnosis questions and answers). The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

3.4.1. Firebase Cloud Storage

We used firebase storage to store the images uploaded by the users.Cloud Storage is built for app developers who need to store and serve user-generated content, such as photos or videos. Cloud Storage for Firebase is a powerful, simple, and cost-effective object storage service built for Google scale.

3.5. Apache Tomcat 9 Web Server

The server side of our application was done with apache tomcat 9. Our Deep learning model code was here waiting for a request. Also plant identifier API is here.

3.6. REST API

We used REST API to handle request. Sending requests in JSON format and expecting requests in server side.

3.7. Google Compute Engine

We located our server side inside Google Compute Engine because it allows our application to run workloads on Google's physical hardware.

3.8. Weather API by OpenWeather

We used Weather API to notify the users about the real time weather details for their farms. Location based, the API gives informations like temperature, wind speed, humidity and more.

3.9. Image Recognizer API by Imagga

We used Image Recognizer API to identify images uploaded by the users.The Imagga API is a set of image understanding and analysis technologies available as a web service that allows you to automate the process of analyzing, organizing and searching through large collections of unstructured images.

3.10. Plant Identifier API by Plant.id

For leaf shape detection we used Plant.id API to detect whether the leaf belongs to a grape or not. The API allows to classify images uploaded by the users whether or not they are grape leaf images. Access key is needed to use it.

3.11. Deep Learning with CNN

In order to predict diseases from grape leaves Convolutional Neural Network(CNN) was used. The important part of CNN model is providing dataset and the dataset was obtained from PlantVillage dataset on Github. CNN was trained with PlantVillage dataset and also images that taken by us to do image classification. CNN model was created on Python. Keras library was used to construct CNN model.

CNN model consists of three layers. First layer is input layer, the middle layer is hidden layer and the last layer is output layer. Hidden layer consists of 128 nodes and output layer is consists of 4 nodes. Before giving images to input layer Convolution operation was applied to images. Convolution function uses 32 filters which have size 3X3. After that, images were converted 2D RGB array. In order to reduce the size of images, pooling operation was applied to images. Using flatten function of Keras Library, 2D array was converted to 1D array. Organized and converted images transferred into the hidden layer. Hidden layer consists of 128 nodes and uses ReLu function as an activation function. Initially random weights were provided to neural network and using ADAM optimization algorithm network weights were updated. ADAM optimizer is different from classical stochastic gradient descent algorithm. Adaptive Gradient Algorithm and Root Mean Square Propagation was used in the ADAM optimizer algorithm. The math behind ADAM optimizer was provided from Keras Library. As a hyperparameter values 8000 is used as an iteration steps in per epoch and 8 is used for epochs size is used. With the provided test set, the accuracy rate of CNN is %98.

3.12. Pycharm

As an Integrated Development Environment(IDE) Pycharm was used to implementation of Python programs. CNN model was implemented on Pycharm. As the Pycharm is easy to use and also uploading necessary tools and Libraries, such as OpenCV, numpy, Keras is easy in Pycharm, in the development of CNN model Pycharm was prefered.

3.13. Atom Editor

We used atom text editor to code Javascript for React-Native. Packages and github integration allowed us to work easily while developing. Atom is a free and open-source text and source code editor for macOS, Linux, and Microsoft Windows with support for plugins written in Node.js, and embedded Git Control, developed by GitHub. Atom is a desktop application built using web technologies.

3.14. Android Studio

We used android studio since we were developing an android application. We used for the emulator to see how our app looked and works.

3.15. Github for Atom Editor

Git integration with atom editor helped us to work as a team. Version control of our code is done with this convenience.

3.16. ImageView Library

Provides custom Image view for React Native that allows to perform pinch-to-zoom on images.

3.17. React-Native(RNCamera) Library

RNCamera library allows us to have a Camera component for React Native. Since the user needs to use the camera to take pictures of the grape leafs we used this dedicated library. Also user can go to camera roll and open flash mode.

3.17.1. RN-Fetch-Blob Library

A project committed to making file access and data transfer easier and more efficient for React Native developers. We used this library to transfer a photo into a blob so we can put it into firebase storage.

3.18. React Native Maps Library

We used this library in the nearby and farm selection features. The users can select any location from the map and assign as their farm location. Map components allowed us to show map look inside the application.

3.19. React Native Image Crop Picker Library

After user takes a photo, this library allows the user to crop and rotate the image. We used this library to make the users to crop the image which has only a grape leaf to make the deep learning prediction more stable.

3.20. React Native Dialog Input Library

We used this library to assign farm locations with user specific entered farm names to firebase realtime database.

3.21. React Native Firebase Library

We used this library to connect to our firebase account and manage the database accordingly. Firebase connection is done with this library. React Native Firebase is a collection of official React Native modules connecting you to Firebase services; each module is a light-weight javascript layer connecting you to the native Firebase SDKs for both iOS and Android.

3.22. Image Background Segmentation

To improve the deep learning training we outsourced leaf image segmentation code but finally we decided not to use it because the segmented leaf images were not as expected on the diseased leaf images. Therefore, we did not use it but we worked on it for our project [1].

4. Impact of Engineering Solutions

4.1. Economical Impact

The economical solution that GrapeHealth provides is substantial in our and in sector specialists view because grapes in case of a disease becomes unmarketable or affects the grapeyards income by affecting the harvest seasons return on investment. Grape is a standalone economical aspect of our daily live, it is consumed or used in different sectors and product. Therefore, grapes are an economical factor and with GrapeHealth we are tackling the diseases that affect the yields of each producer. Decreasing the disease rates and informing the users about the diseases with detailed information and suggesting treatment options can save a grapeyard from diseases that might affect them economically. Real Time image assessments inform users about their grapes diseases or help them to keep an eye on their farms by looking at the notifications. GrapeHealth is not only an disease detecting application, it is also an farm monitoring and a informer for nearby disease threats application. We implemented GrapeHealth in the mindset of the users, we thought of their needs in the application to economically benefit the users and the grape industry by making the users control, learn and monitor their grape fields.

GrapeHealth is a free to use application, if the application get popularized in the grape producers or other target users, the application has a potential to create a community of users which has the potential to be informative and decrease producer failures. So GrapeHealth will definitely make an economical impact if the application becomes viral.

Grape is a delicate fruit, after each disease passes the the yield quality and quantity decreases. Therefore, with GrapeHealth the producers will have a chance to increase their yields.

4.2. Environmental Impact

Disease is already an environmental factor affecting the healthiness of the grapes. Controlling diseases will decrease the environmental harm because by suggesting the producer to use organic control methods or precaution ways, the users will be able to learn other treatment methods than chemical methods. Some diseases require chemical treatment after a certain stage but for initial or before widespread the disease can be controlled with organic methods. Environmentally saving the mother nature by not suggesting chemicals in the first hand.

Informed producers will attain a better yield for their grapeyard. By increasing their return on grapes the producers will be able to invest in environmental grape production techniques. Diseases are very detrimental to grapes and some of the diseases can spread easily by wind affecting other plant types or grapeyard. Nearby feature of our application foresees if any nearby threat occur to the producers and consequently the producers can take action by applying precaution techniques. Also, producers can monitor their farms weather conditions and take precautions if necessary, because our application warns user specific warnings about their farms.

4.3. Societal Impact

Societal improvements that GrapeHealth provides relies in the economical and environmental impacts because if the users can be informed about better production and monitoring of their grapes, the better the producers conditions will become. A better production understanding, greater knowledge of diseases, monitoring nearby disease threats, monitoring weather conditions and learning organic and chemical treatments will benefit the producer community and eventually will make the society to utilise with better agricultural practices. Society will be able to reach more healthy and fresh grapes with greater quantities. Therefore, GrapeHealth impacts the society in healthy concerns.

5. Contemporary Issues

There are no applications in the market share which diagnoses and informs agricultural diseases which can both show notification nearby and weather locally. In the market there are some applications which could find out type of plants or diagnose general diseases. PlantNet is an example of them it does not diagnose diseases and none of these applications warn the user about treatment of the disease or local weather. These applications do not show treatments or recommend any date about treatment to the user. GrapeHealth resolve these concern and issues of the grape producers.

The area of our project needs engineering attention to prevent disease and improve the quality of grapes. Turkey is the fifth grape producer world wide and grape is the most produced fruit in Turkey’s market share. Turkey has excessively link field for grape production and has great potential for viticulture. The situation in Turkey is worse than european and american viticulture in the sense of knowledge and agricultural application techniques. GrapeHealth is an informative application for its users to tackle the knowledge aspect. By decreasing the diseases the producers can attain better economical standards and apply better production techniques, which eventually results in better grape production to reach european standards. Generally, downy mildew and powdery mildew are the dominating diseases in Turkey and they are are treatable but due to wrong applications or ignorance the producers cannot cure these basic diseases.

The pricing of the grapes are determined by authoritative institutions and the pricing is not done with an methodology because the general yield of Turkey is unknown and this causes pricing failures which affect the producers economically. If the consultants uses GrapeHealth they could predict the yield or at least foresee the yield by investigating the disease rate listed in each farm.

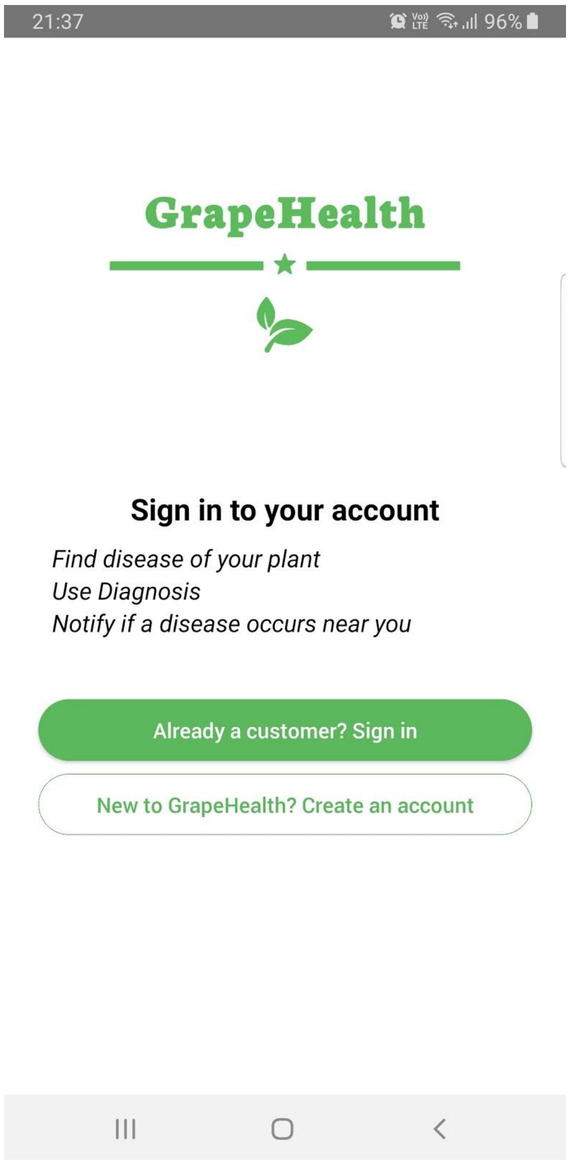
Grape specialists generally communicate with producers about what to use on the grapeyard if there is an disease. They become a intermediary between drug stores and grape producer, the producers rely on the specialists for treatments options but GrapeHealth offers to be an alternative solution for this relation. Grape Merchants who buy grapes from producers have this same relationship with specialists, therefore GrapeHealth can also benefit this issue. Merchants can assess the grapes either taking photos or using diagnosis questionnaire to learn about if there is a disease on the grape.

Appendix A: User Manual & Installation Guide

The report is to be accompanied by the software/hardware system itself along with a User's Manual including installation instructions. Application can be installed all android phones by provided .apk file.

1. Welcome Screen

This screen welcomes users and forward them to “Sign In” or “Sign Up” panels.



Sign in to your account

Find disease of your plant
Use Diagnosis
Notify if a disease occurs near you

Already a customer? Sign in
New to GrapeHealth? Create an account

Figure 4. Welcome Screen

2. Create Account

Users could sign up the application from this screen. This panel requires from user to enter Name, Surname, E-mail information and create a password for his/her account.

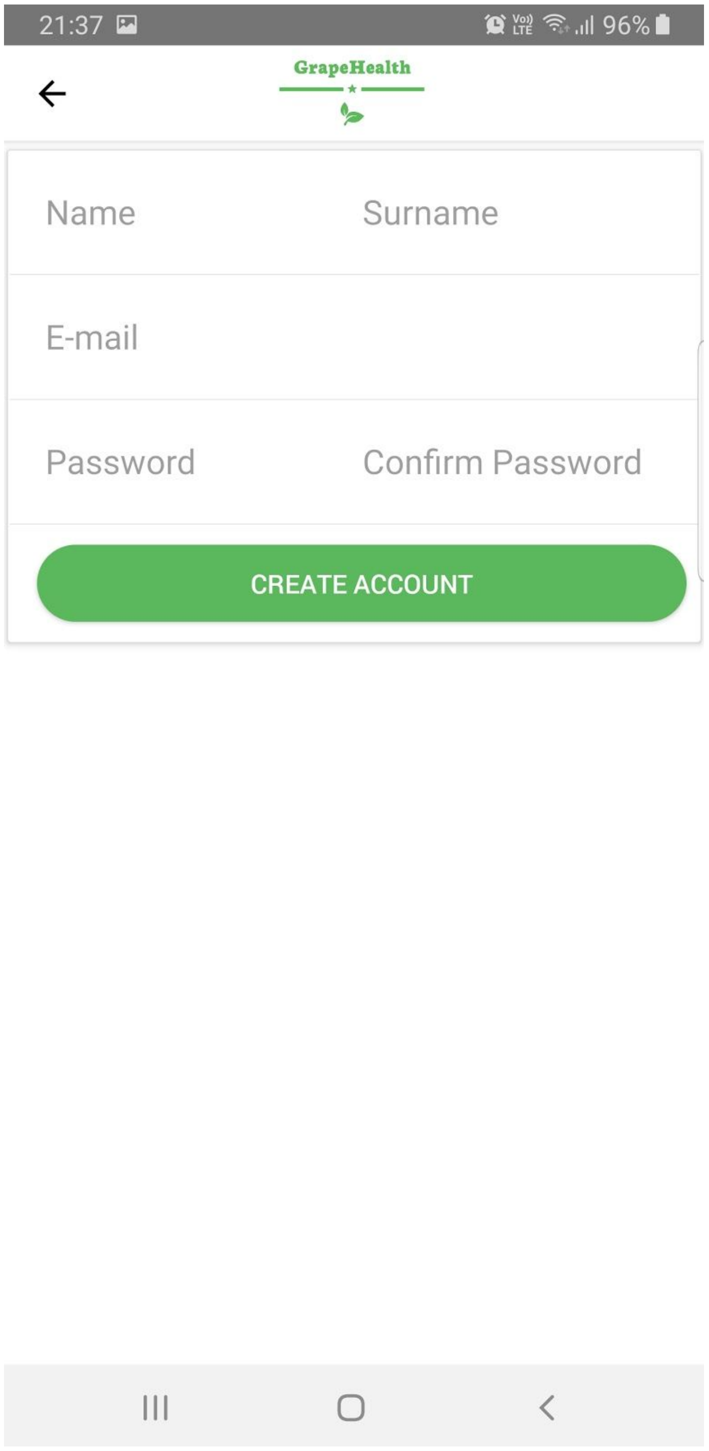


Figure 5. Create Account Screen

3. Login

A user could sign in to his/her account by using this panel. The user could sign in by using email/password information.

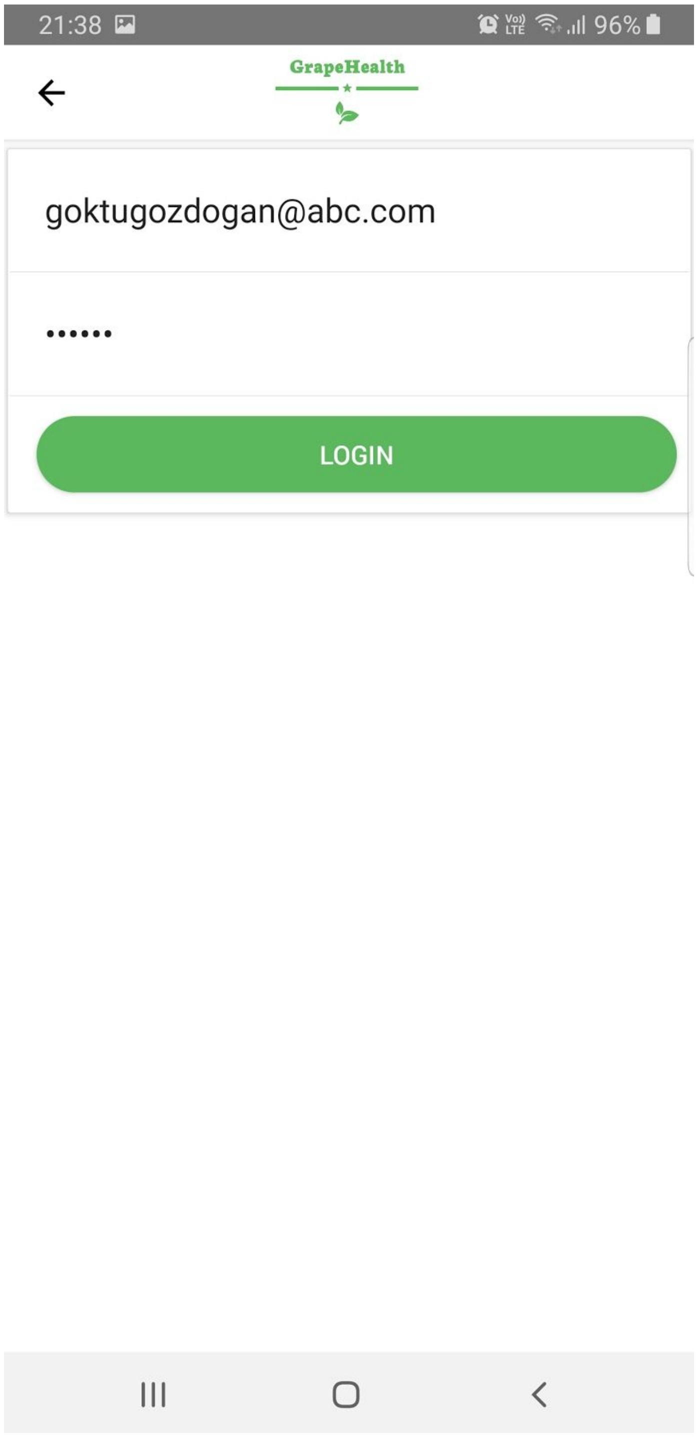
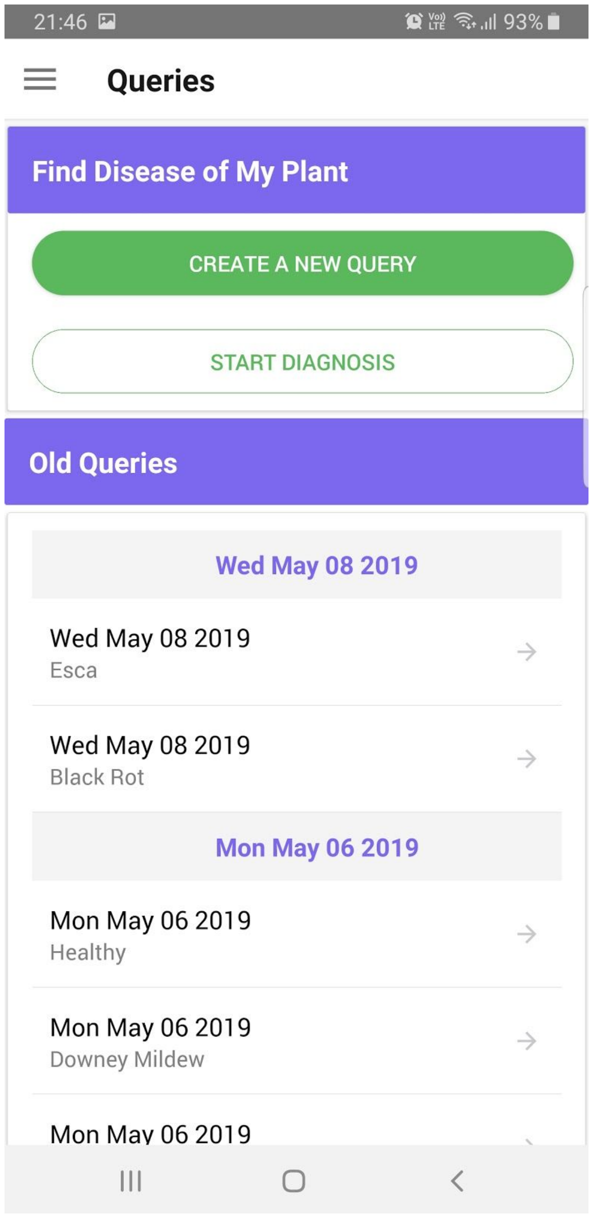


Figure 6. Login Screen

4. Queries

In this screen, old queries are listed. The user could see old queries date and briefly information about them. For detailed information the user could click on a query and program forwards you to “Query Details” screen. Moreover, the user could send a new query by clicking “Create a New Query” button. This button forwards him/her to “Farm Select” screen, then user can take photo. There is a button for “Diagnosis”. The user is forwarded to “Diagnosis” screen by clicking this button.



Wed May 08 2019
Esca

Wed May 08 2019
Black Rot

Mon May 06 2019
Healthy

Mon May 06 2019
Downey Mildew

Mon May 06 2019

Figure 7. Queries Screen

5. Farm Select

In order to take a photo, user has to select a farm. The user can do that either selecting an existing farm of his or by adding a new farm.

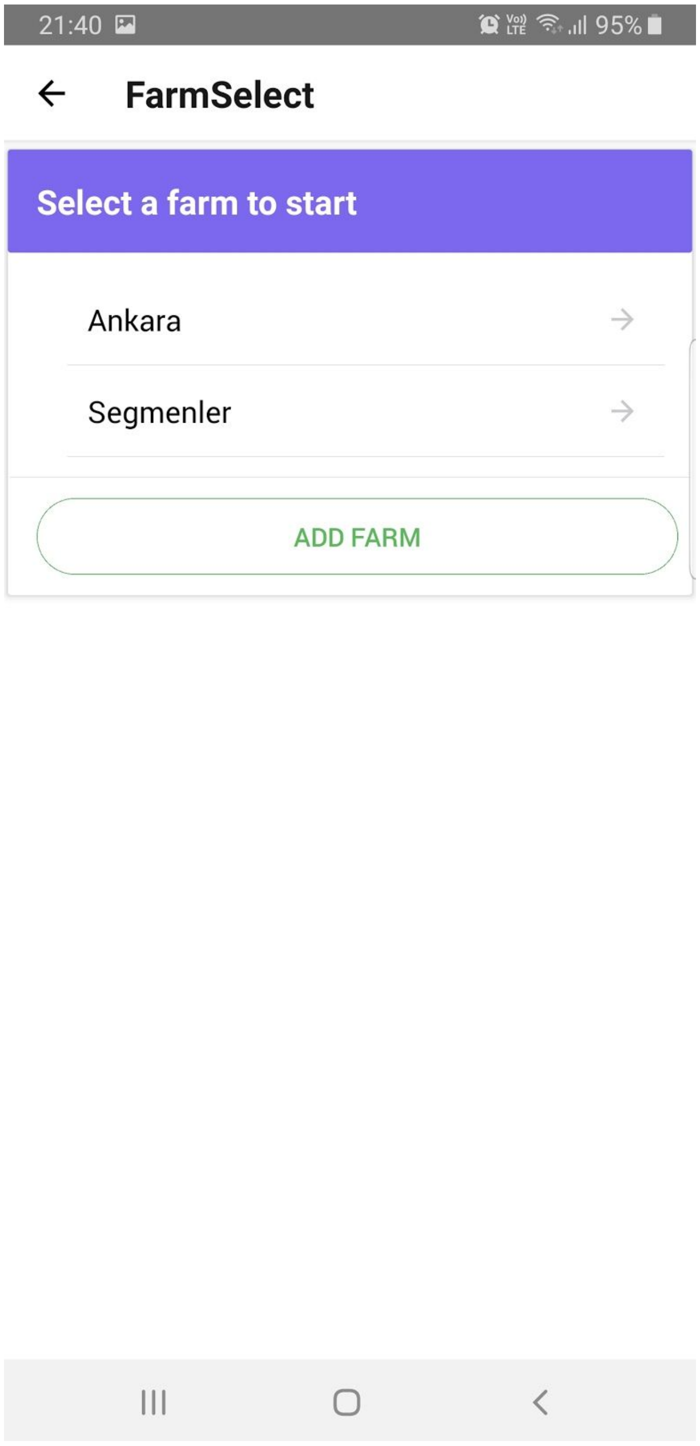


Figure 8. Farm Select Screen

6. Camera

In this screen, user has to give permission to Use Camera and Manage Files in order to continue. By clicking camera icon on the middle usr can take a photo. User can select an existing photo by clicking files icon on the left bottom corner. User can select flash options on the right bottom corner. There is a warning to user that how he should take the photo.

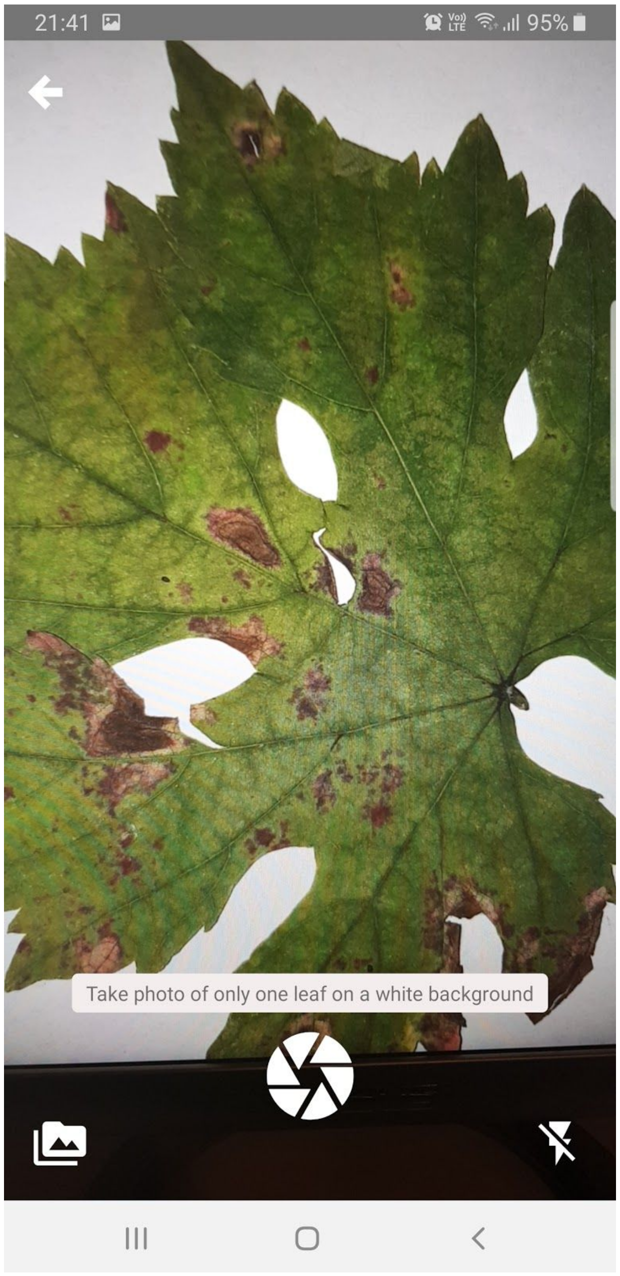


Figure 9. Camera Screen

7. Edit Photo

After taking a photo, user can edit it. User can zoom the photo or rotate it. If all the edges just inside of the highlighted square, user can processed to upload photo.

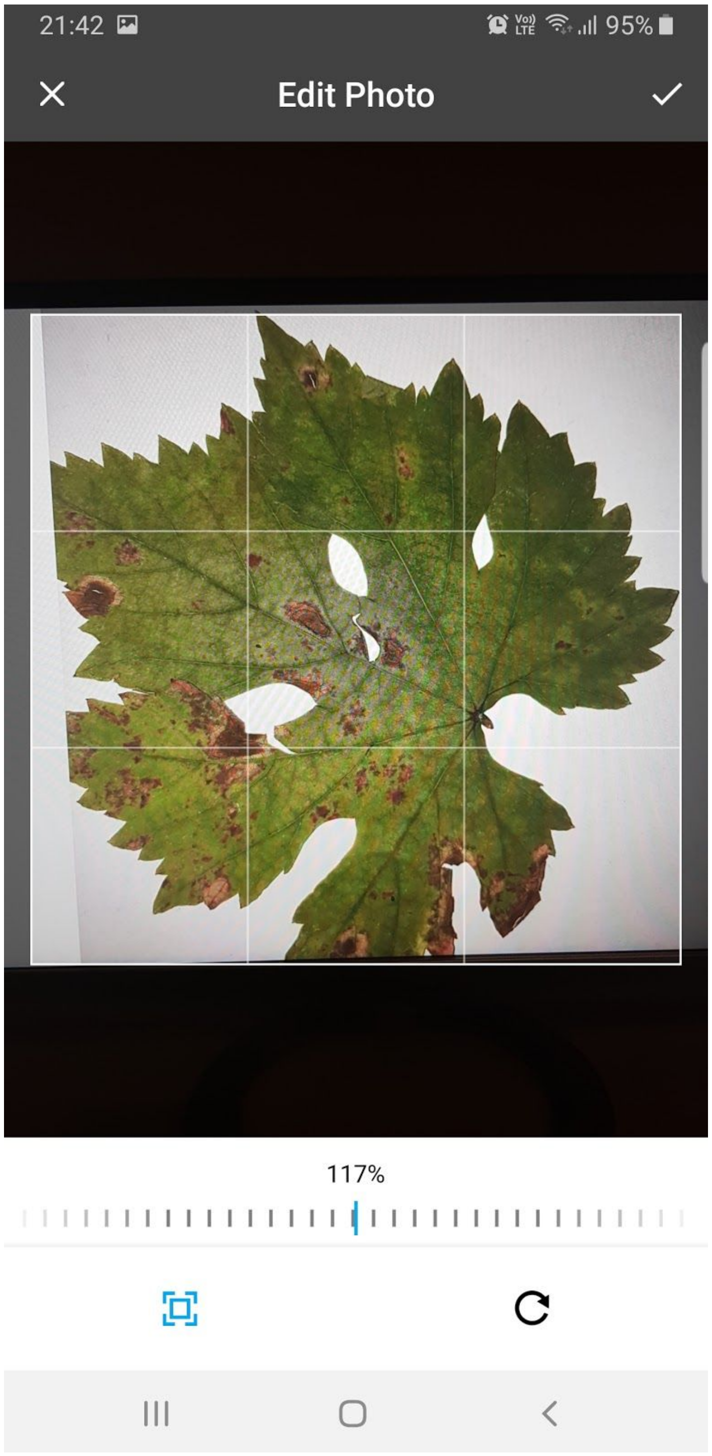
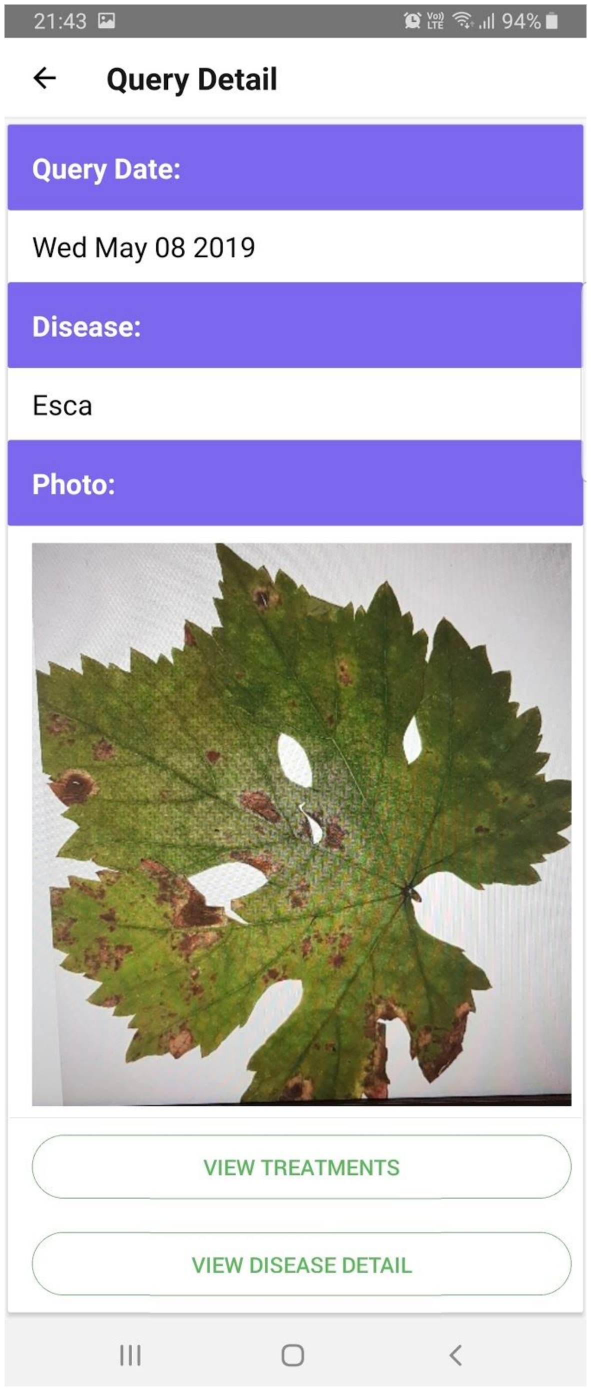


Figure 10. Photo Edit Screen

8. Query Detail

Users are directed to this screen after results are uploaded. USer can see date, disease and photo. By clicking “View Treatment” and “View Disease Detail”, user are directed to according screens.



Query Date:

Wed May 08 2019

Disease:

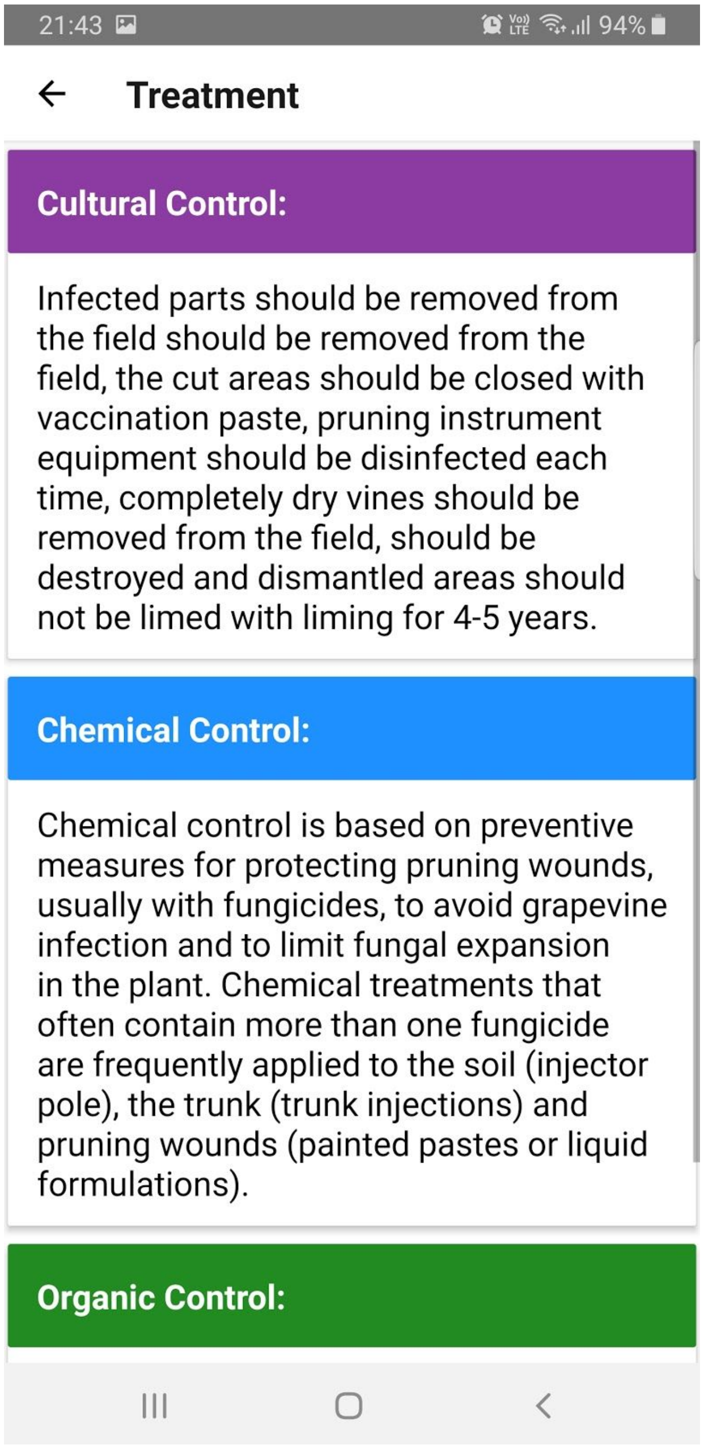
Esca

Photo:

Figure 11. Query Detail Screen

9. Query Treatment

User can see possible treatments of the grape from this screen. There are 3 different suggestions available for each disease; Cultural, Chemical and Organic.



Cultural Control:

Infected parts should be removed from the field should be removed from the field, the cut areas should be closed with vaccination paste, pruning instrument equipment should be disinfected each time, completely dry vines should be removed from the field, should be destroyed and dismantled areas should not be limed with liming for 4-5 years.

Chemical Control:

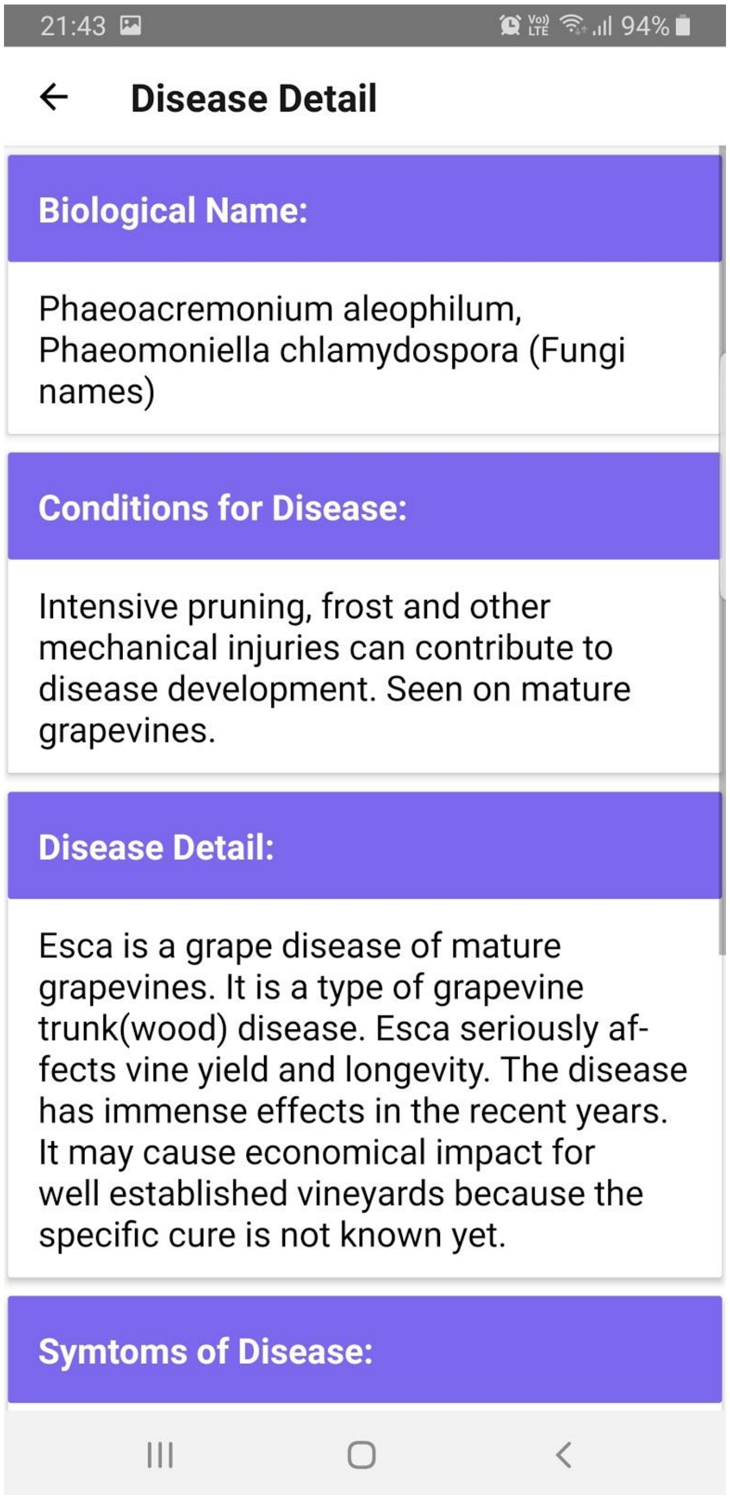
Chemical control is based on preventive measures for protecting pruning wounds, usually with fungicides, to avoid grapevine infection and to limit fungal expansion in the plant. Chemical treatments that often contain more than one fungicide are frequently applied to the soil (injector pole), the trunk (trunk injections) and pruning wounds (painted pastes or liquid formulations).

Organic Control:

Figure 12. Treatment Screen

10. Disease Detail

In this screen, user can see detailed disease description.



21:43 94%

Disease Detail

Biological Name:

Phaeoacremonium aleophilum, Phaeomoniella chlamydospora (Fungi names)

Conditions for Disease:

Intensive pruning, frost and other mechanical injuries can contribute to disease development. Seen on mature grapevines.

Disease Detail:

Esca is a grape disease of mature grapevines. It is a type of grapevine trunk(wood) disease. Esca seriously affects vine yield and longevity. The disease has immense effects in the recent years. It may cause economical impact for well established vineyards because the specific cure is not known yet.

Symtoms of Disease:

Figure 13. Disease Detail Screen

11. Menu

User can access Menu by clicking menu icon on the left top corner.This button opens a menu component. User can traverse between desired screen from Menu component.

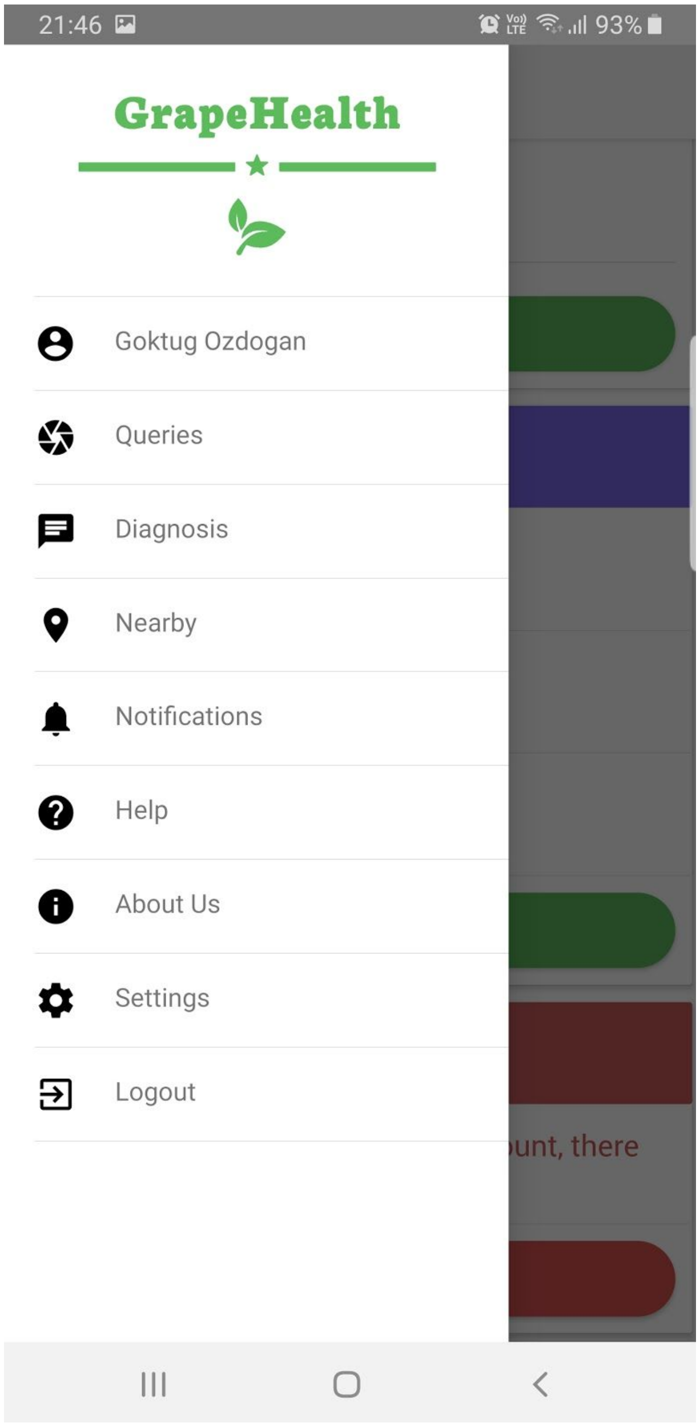
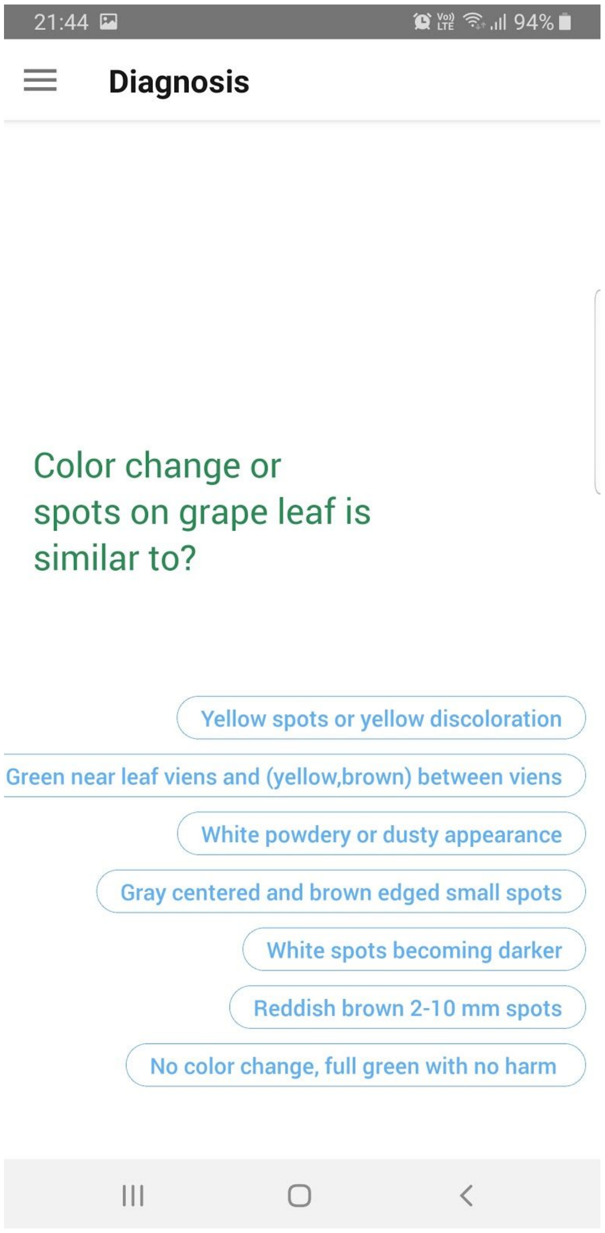


Figure 14. Menu

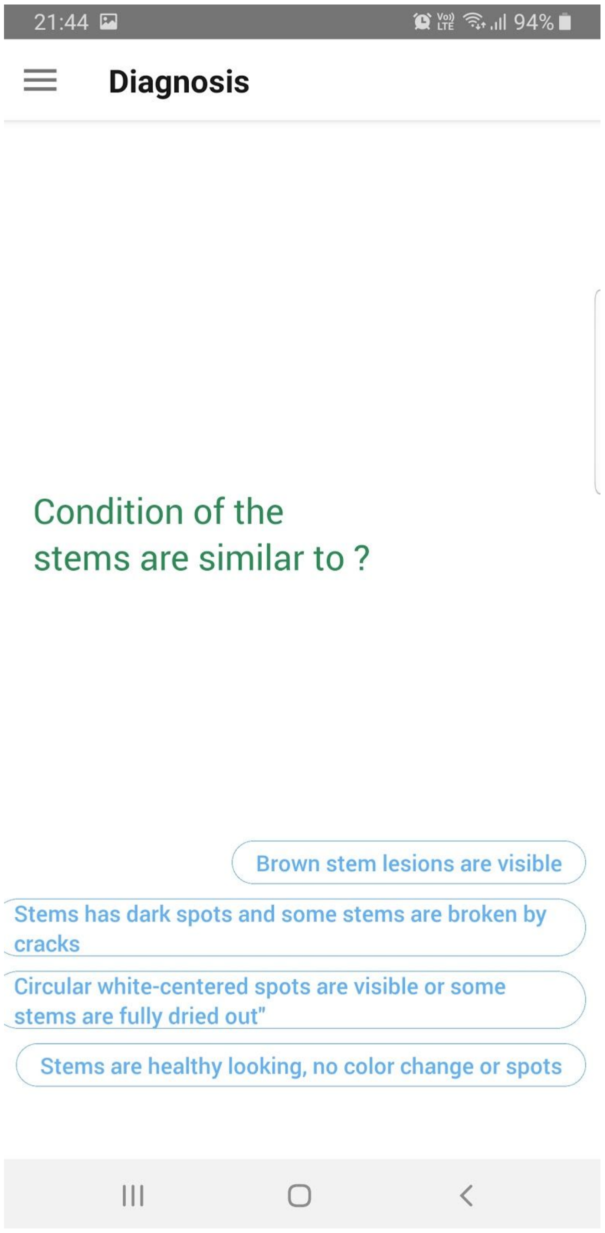
12. Diagnosis

In this screen, by answering contiguous questions that is prompted you will get a result. Answering every question carefully is advised otherwise the symptoms of the disease cannot be assessed correctly. After answering the questions, “Diagnosis” will either prompt the detected disease or prompt “No evidence”. From the resulting page you can view treatments or detailed information about the disease.



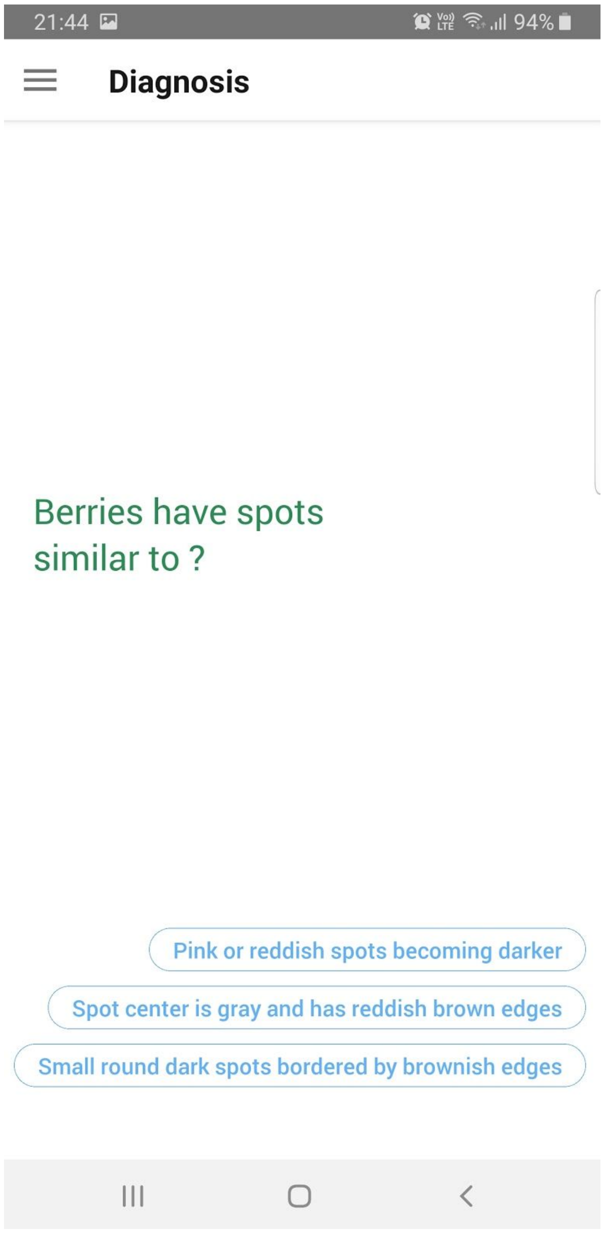
Color change or spots on grape leaf is similar to?

Yellow spots or yellow discoloration
Green near leaf viens and (yellow,brown) between viens
White powdery or dusty appearance
Gray centered and brown edged small spots
White spots becoming darker
Reddish brown 2-10 mm spots
No color change, full green with no harm



Condition of the stems are similar to ?

Brown stem lesions are visible
Stems has dark spots and some stems are broken by cracks
Circular white-centered spots are visible or some stems are fully dried out"
Stems are healthy looking, no color change or spots


21:44 94%

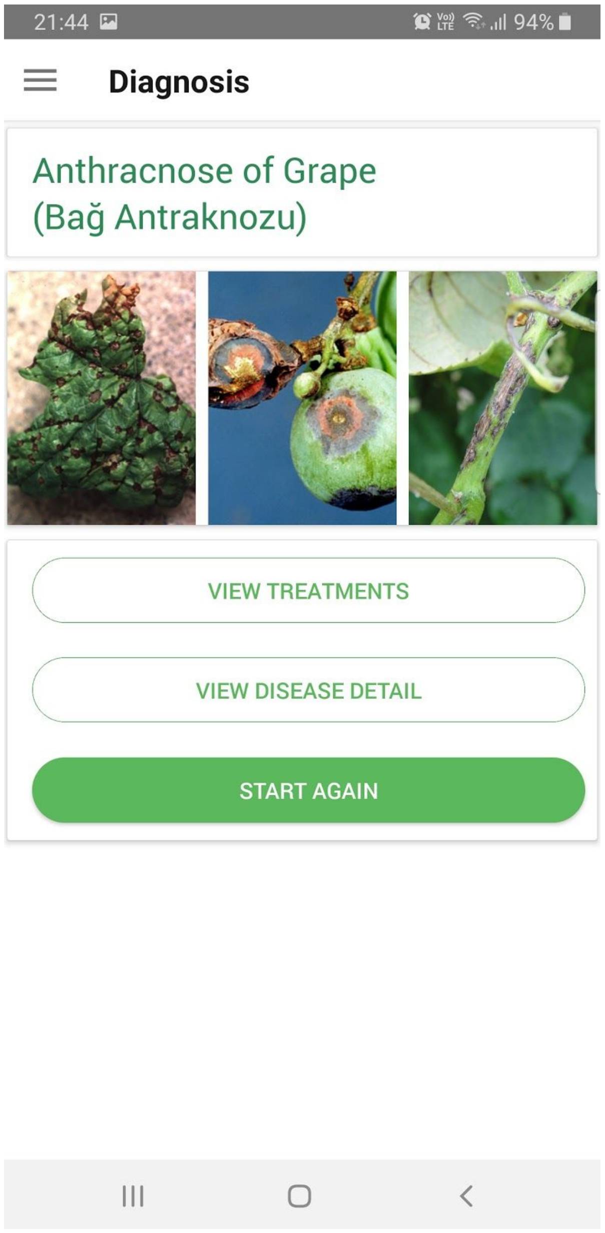
Diagnosis

Berries have spots similar to ?

Pink or reddish spots becoming darker

Spot center is gray and has reddish brown edges

Small round dark spots bordered by brownish edges


21:44 94%

Diagnosis

Anthracnose of Grape (Bağ Antraknozu)

Figure 15. Diagnosis Traverse Screen

13. Nearby

User can display his farms and scroll between them. Also, use can see other farms that contain diseases. Blue markers represent user's farms and red markers represent farms that contain disease(s). User can see the disease's names by clicking on the red markers.

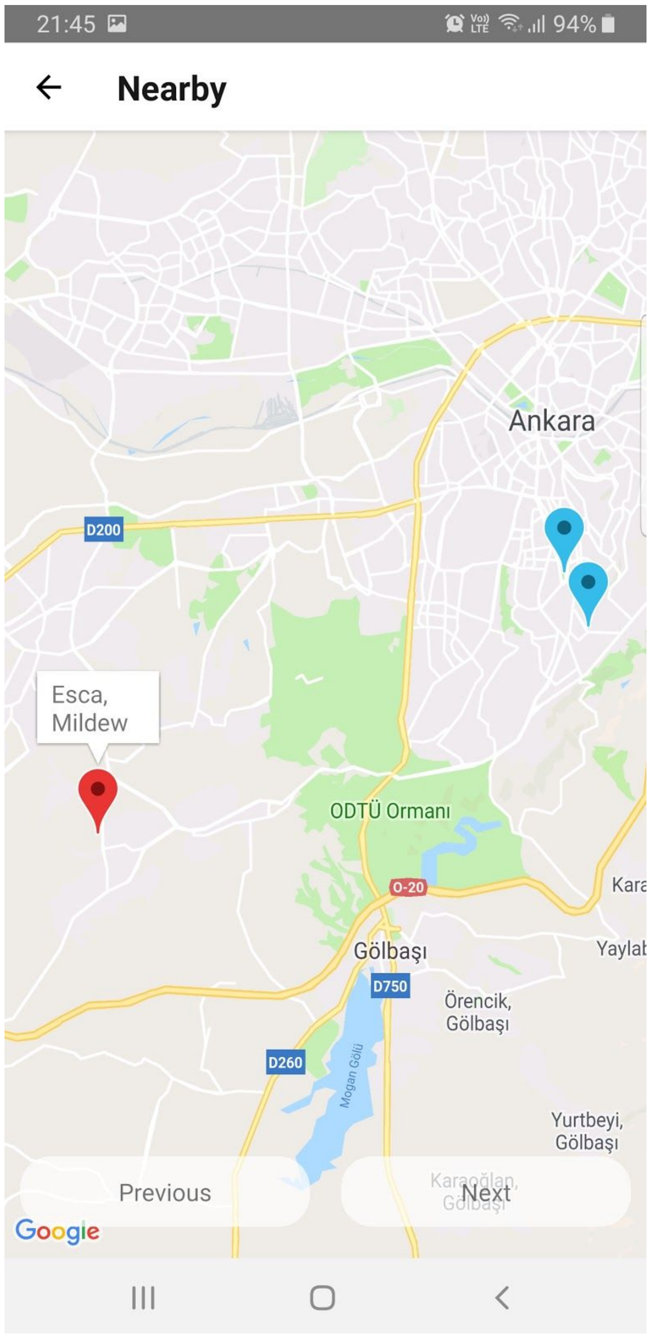
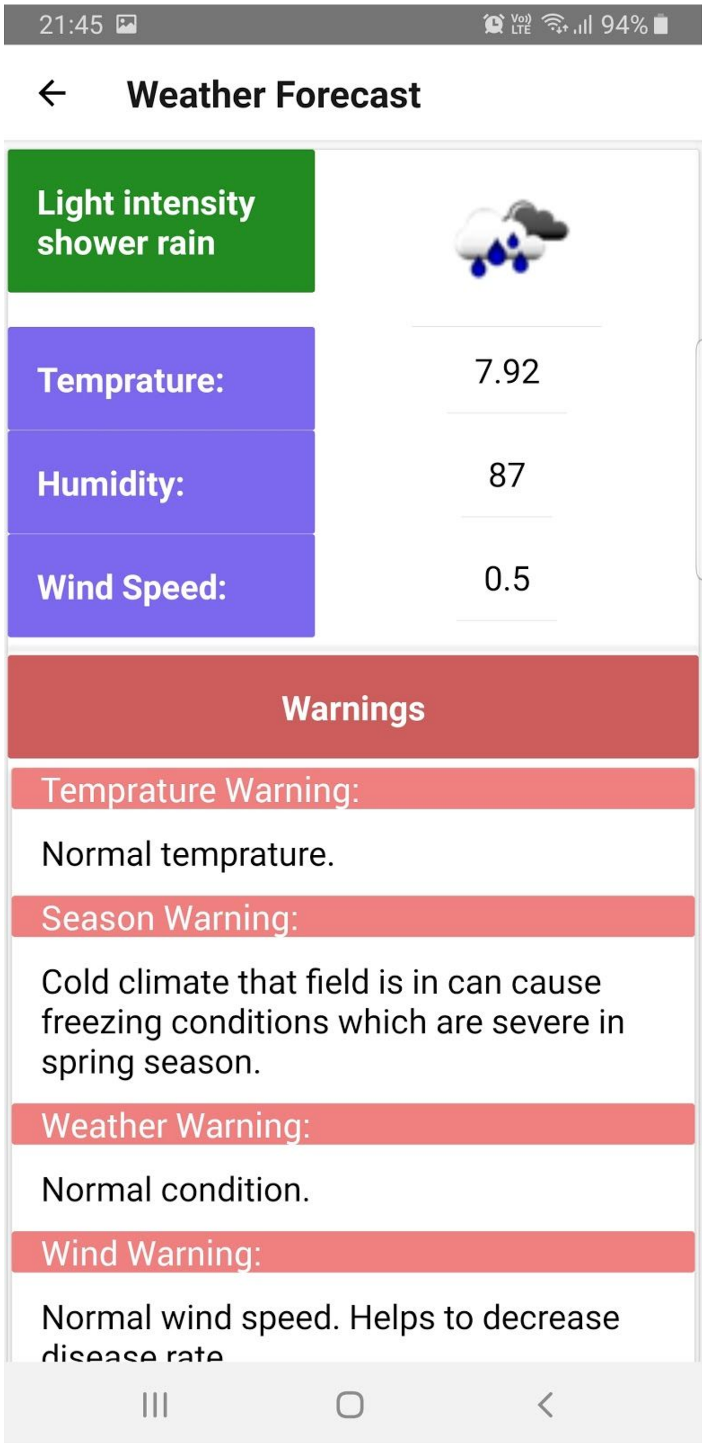


Figure 16. Nearby Screen

14. Weather Forecast

In this screen, use can see weather conditions about his farms. There are some warnings to related to how current weather can affect grapes.


21:45 94%

Weather Forecast

Light intensity shower rain

Temprature: 7.92

Humidity: 87

Wind Speed: 0.5

Warnings

Temprature Warning:

Normal temprature.

Season Warning:

Cold climate that field is in can cause freezing conditions which are severe in spring season.

Weather Warning:

Normal condition.

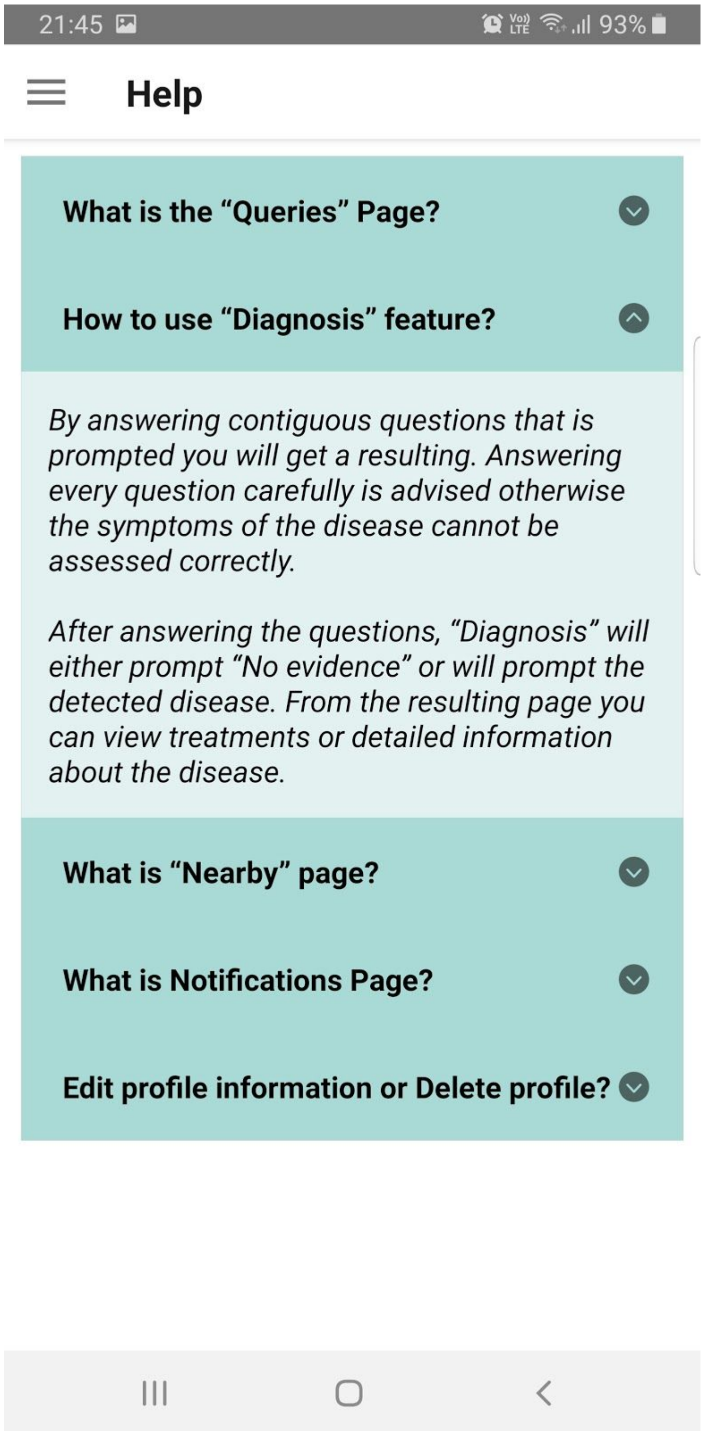
Wind Warning:

Normal wind speed. Helps to decrease disease rate

Figure 17. Weather Forecast Screen

15. Help

This screen helps user to get information about the application. It contains questions that frequently asked and its answers.



21:45 93%

Help

What is the "Queries" Page?

How to use "Diagnosis" feature?

By answering contiguous questions that is prompted you will get a resulting. Answering every question carefully is advised otherwise the symptoms of the disease cannot be assessed correctly.

After answering the questions, "Diagnosis" will either prompt "No evidence" or will prompt the detected disease. From the resulting page you can view treatments or detailed information about the disease.

What is "Nearby" page?

What is Notifications Page?

Edit profile information or Delete profile?

Figure 18. Help Screen

16. Setings

In this screen, user could change profile information and password if necessary. In this screen, user could also delete his/her account.

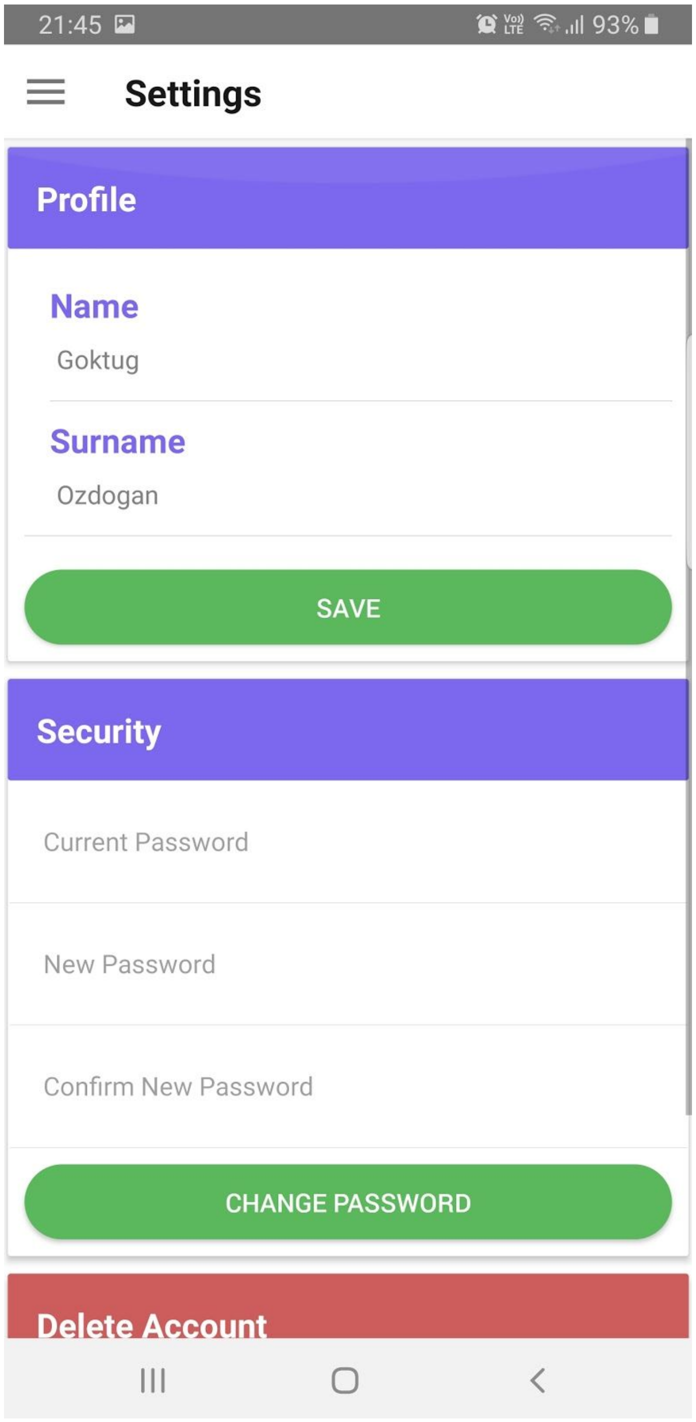


Figure 19. Settings Screen