Πανεπιστήμιο Θεσσαλίας

Πανεπιστήμιο Θεσσαλίας, 2016-2017

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών

Μεταπτυχιακή εργασία

Θέμα:

«Σχεδιασμός και υλοποίηση ολοκληρωμένης πλατφόρμας συλλογής, επεξεργασίας και απεικόνισης αγρομετεωρολογικών δεδομένων από δίκτυα ασύρματων αισθητήρων.»

«Design and implementation of an integrated agrometeorological data collection, management and visualization platform from Wireless Sensor Networks."»

Ευσταθίου Ελισάβετ - Αποστολία

UNIVERSITY OF THESSALY

Πανεπιστήμιο Θεσσαλιάς

Επιβλέπων καθηγητής:

Κοράκης Αθανάσιος (Επίκουρος Καθηγητής)

Συνεπιβλέπων καθηγητές:

Αργυρίου Αντώνιος (Επίκουρος Καθηγητής)

Κατσαρός Δημήτριος (Επίκουρος Καθηγητής)

Βόλος, Οκτώβριος 2017

Πανεπιστήμιο Θεσσαλιάς

CONTENTS

Πανεπιστήμιο Θεσσαλιάς

ΠΕΡΙΛΗΨΗ

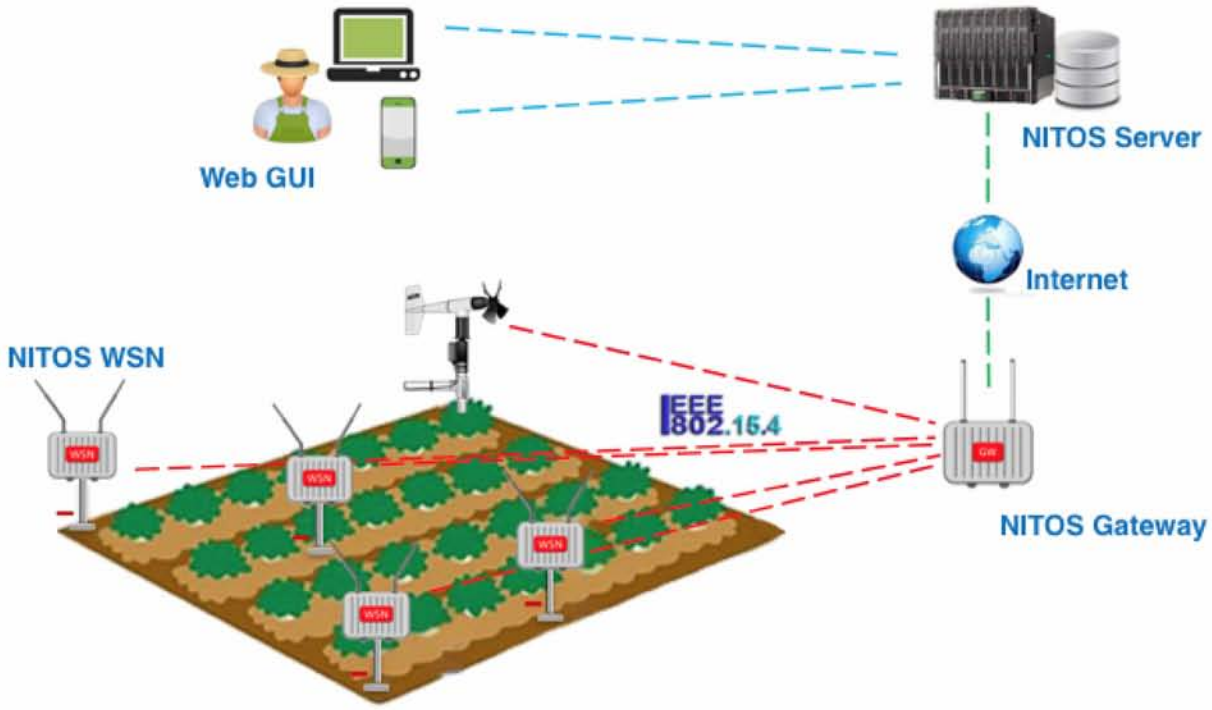
Η τεχνολογική ανάπτυξη δεν ακολουθεί φυσικά ή αόρατα όρια και επεκτείνει τις ρίζες της προς όλες τις κατευθύνσεις. Ακολουθώντας τα ίδια κριτήρια, ο κλάδος του " Internet of Things", έχει τις εφαρμογές του στον τομέα της Βιομηχανίας, της οικιακής ασφάλειας ,στα έξυπνα σπίτια και τις έξυπνες πόλεις και σε πολλούς άλλους τομείς. Ακόμη και ο γεωργικός τομέας δεν παρέμεινε άθικτος και έτσι δημιουργήθηκε ο νέος κλάδος του "Agriculture Internet of Things". Ο κλάδος του AIoT συμβάλλει στην αύξηση της παραγωγικότητας των καλλιεργειών μέσω της διαχείρισης και του ελέγχου των γεωργικών δραστηριοτήτων. Επιπλέον συνδέεται άρρηκτα με την ταχεία εξέλιξη των ασύρματων δικτύων αισθητήρων (Wireless Sensor Networks - WSNs) που βοηθούν στη δημιουργία αυτόνομων και ελεγχόμενων συστημάτων μέσω της παρακολούθησης, της συλλογής και της επεξεργασίας δεδομένων που προέρχονται από διάφορες μονάδες ανίχνευσης. Στα πλαίσια της συγκεκριμένης μεταπτυχιακής εργασίας σχεδιάστηκε και αναπτύχθηκε μια ολοκληρωμένη πλατφόρμα συλλογής, διαχείρισης και απεικόνισης γεωμετεωρολογικών δεδομένων. Αυτή η πλατφόρμα συγκεντρώνει δεδομένα από ένα δίκτυο αισθητήρων, επικοινωνεί με ένα διακομιστή 'backend' για την κατάλληλη επεξεργασία τους και τέλος παρουσιάζει τις χρήσιμες γεωργικές πληροφορίες στον τελικό χρήστη μέσω ενός γραφικού περιβάλλοντος "frontend" για κινητές συσκευές και για υπολογιστές.

Πανεπιστήμιο Θεσσαλιάς

ABSTRACT

Technological Development follows no physical or invisible boundaries and expands its roots in all directions. Working on same criteria Internet of Things [25] has its applications in field of Industry (as Industrial Internet of Things), home security and smart cities and so on. Even the agricultural sector isn't left untouched from the technological development of Internet of Things leading to development of Agriculture Internet of Things. AIoT helps in increasing crop productivity by way of managing and controlling the agricultural activities. It is inextricably linked with the rapid evolution of Wireless Sensor Networks (WSNs) that assists the formation of autonomous and controllable frameworks. In the context of this master thesis an integrated agrometeorological data collection, management and visualization platform has been designed and developed. This platform gathers data from a sensors network consist of several sensing modules , communicates with a backend server for appropriate processing and finally illustrates the useful agricultural information to a frontend web and mobile application.

IoT Agriculture system



Πανεπιστήμιο Θεσσαλιάς

1. INTRODUCTION

Smart environments represent the next evolutionary development step in building, utilities, industrial, home, shipboard, and transportation systems automation. Like any sentient organism, the smart environment relies first and foremost on sensory data from the real world. Sensory data comes from multiple sensors of different modalities in distributed locations. In this thesis we present a framework which based on a Wireless Sensor Networks (WSN) consists of agricultural sensory data. In the second chapter we give an introduction on WSNs and their existing MAC and Network Protocols. In the third chapter we present the modules, platforms, architectures and standards used to build the sensors network. In the fourth chapter we describe the topology of the WSN. In the fifth chapter we give a small explanation of precision agriculture. In the sixth and seventh chapters we analyze how crop water management is feasible through Agriculture Internet of Things with integration of Sensor Observation Service. Irrigation monitoring and processing provide a solution to manage water requirement or water supply for crop irrigation. Finally in the last chapters we present and analyse the frontend software that implemented in the context of this thesis ,web and mobile, for visualize all data measurements and information to the farmers.

2. Wireless Sensor Networks and Existing MAC and Network Protocols

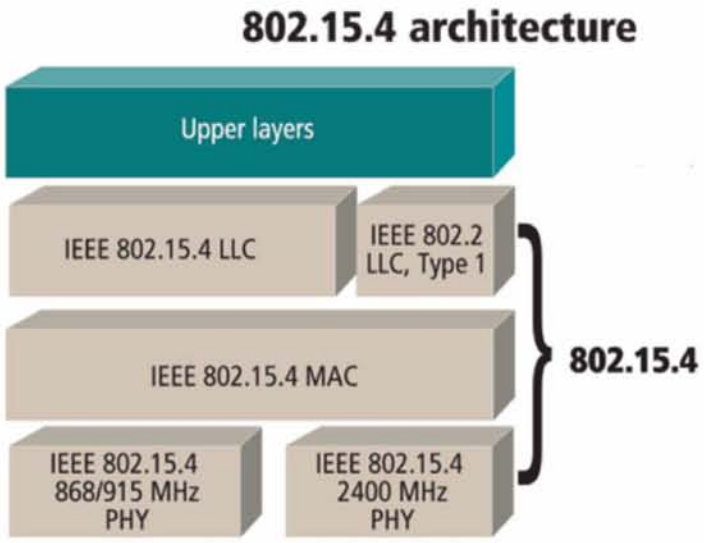
2.1. Wireless Sensor Networks

A wireless sensor network [1] is a collection of nodes organized into a cooperative network. Each node consists of processing capability (one or more microcontrollers, CPUs or DSP chips), may contain multiple types of memory (program, data and flash memories), has an RF transceiver (usually with a single omnidirectional antenna), has a power source (e.g. batteries and solar cells) and accommodates various sensors. A wireless sensor network (WSN) generally consists of a base station (or "gateway") that can communicate with a number of wireless sensors via a radio link. Data is collected at the wireless sensor node, compressed, and transmitted to the gateway directly or, if required, uses other wireless sensor nodes to forward data to the gateway. The transmitted data is then presented to the system by the gateway connection. Creating wireless networks can be done using a variety of RF protocols, the most widely used protocol is the 802.15.4 standard.

2.2. IEEE 802.15.4 standard

The IEEE [2] 802.15.4 is a standard which specifies the physical layer and media access control (MAC) for low-rate wireless personal area networks (WPANs). Devices are conceived to interact with each other over a conceptually simple wireless network. The definition of the network layers is based on the OSI (Open System Interconnection) model although only the lower layers

are defined in the standard, interaction with upper layers is intended, possibly using an IEEE 802.2 logical link control sublayer accessing the MAC through a convergence sublayer.



2.2.1. 802.15.4 Physical Layer

Physical layer is the initial layer in the OSI reference model used worldwide. The physical layer (PHY) ultimately provides the data transmission service, as well as the interface to the physical layer management entity, which offers access to every layer management function and maintains a database of information on related personal area networks. Thus, the PHY manages the physical RF transceiver and performs channel selection and energy and signal management functions. It operates on one of three possible unlicensed frequency bands:

- 868.0 - 868.6MHz: Europe - 1 channel

- 902.0-928.0MHz: North America - up to 10 channels

- 2400–2483.5 MHz: Worldwide – up to 16 channels

Supported Bit Rates:

- 868.0 - 868.6MHz: 20, 40, 100, 250 Kb/s
- 902.0-928.0MHz: 40,250 Kb/s
- 2400–2483.5 MHz: 250 Kb/s

2.2.2. 802.15.4 MAC Layer

This standard defines a communication layer at the 2nd level of the OSI model. The medium access control (MAC) enables the transmission of MAC frames through the use of the physical channel. Besides the data service, it offers a management interface and itself manages access to the physical channel and network beaconing. It also controls frame validation, guarantees time slots and handles node associations. Finally, it offers hook points for secure services.

2.2.3. 802.15.4 Network model

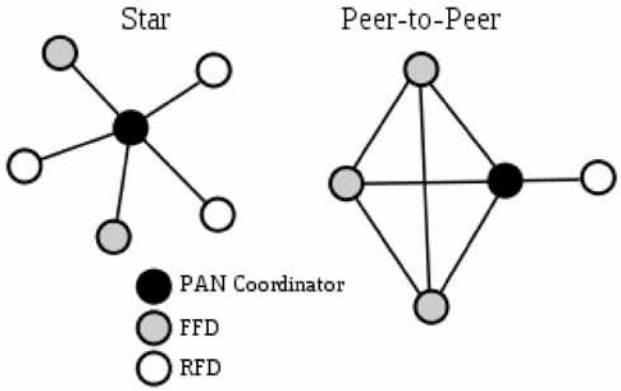
2.2.3.1. Node Types

The standard defines two types of network nodes. The first one is the full-function device (FFD). It can serve as the coordinator of a personal area network just as it may function as a common node. It implements a general model of communication which allows it to talk to any other device: it may also relay messages, in which case it is dubbed a coordinator (PAN coordinator

when it is in charge of the whole network). On the other hand there are reduced-function devices (RFD). These are meant to be extremely simple devices with very modest resource and communication requirements, due to this, they can only communicate with FFDs and can never act as coordinators.

2.2.3.2. Topologies

Networks can be built as either peer-to-peer or star networks. However, every network needs at least one FFD to work as the coordinator of the network. Networks are thus formed by groups of devices separated by suitable distances. Each device has a unique 64-bit identifier, and if some conditions are met short 16-bit identifiers can be used within a restricted environment. Namely, within each PAN domain, communications will probably use short identifiers.



Peer-to-peer (or point-to-point) networks can form arbitrary patterns of connections and their extension is only limited by the distance between each pair of nodes. They are meant to serve as the basis for ad hoc networks capable of performing self-management and organization. Since

the standard does not define a network layer, routing is not directly supported, but such an additional layer can add support for multi-hop communications. Further topological restrictions may be added; the standard mentions the cluster tree as a structure which exploits the fact that an RFD may only be associated with one FFD at a time to form a network where RFDs are exclusively leaves of a tree, and most of the nodes are FFDs. The structure can be extended as a generic mesh network whose nodes are cluster tree networks with a local coordinator for each cluster, in addition to the global coordinator.

A more structured star pattern is also supported, where the coordinator of the network will necessarily be the central node. Such a network can originate when an FFD decides to create its own PAN and declare itself its coordinator, after choosing a unique PAN identifier. After that, other devices can join the network, which is fully independent from all other star networks.

2.2.4. On top of IEEE 802.15.4

There are several protocols which use IEEE 802.15.4 [3] as its MAC layer. The most known is ZigBee [4], although the protocol is the basis for several other specifications such as:

- Wireless HART [5]: It is the wireless version of the HART protocol which is the most used in the automation and industrial applications which require real time. It uses Time Synchronized Mesh Protocol (TSMP). A "time coordinator" node is required in order to assign the time slot to all the notes.

- ISA - SP100 [6]: It also centers in the process and factory automation. It is being developed by the Systems and Automation Society (ISA) and tries to be an standard for this kind of projects.
- 6LoWPAN [7]: As the may point out it is the implementation of the IPv6 stack on top of 802.15.4 to let any device be accessible from and to the Internet.
- MiWi [8]: It is designed for low data transmission rates and short distance.
- Mesh protocols: A lot of different mesh networking protocols have been and are being implemented by companies over the 802.15.4 MAC layer. One in particular is Digi's [9] DigiMesh™. DigiMesh™ is a proprietary peer-to-peer networking topology for use in wireless end-point connectivity solutions. The nature of its peer-to-peer architecture allows DigiMesh™ to be both easy to use and equipped with advanced networking features, including support for sleeping routers and dense mesh networks.

3. Hardware devices used to implement WSN

In order to be able to implement the platform for the agrometeorological data collection of this thesis several hardware modules had to be programmed and used. These modules are presented below.

3.1. Arduino

The most common open-source electronics platform is Arduino [10]. Arduino is a single- board microcontroller, the hardware consists of an open-source hardware board designed around an 8-bit Atmel AVR microcontroller or a 32-bit Atmel ARM. The Arduino platform is low-cost and can be used in a variety of projects. An important aspect of the Arduino is the standard way that connectors are exposed, allowing the CPU board to be connected to a variety of interchangeable add-on modules known as shields. Most boards include a 5 volt linear regulator and a 16 MHz crystal oscillator. What differentiates the Arduino from other platforms is that an Arduino's microcontroller is pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory, compared with other devices that typically need an external programmer.The Arduino board can easily be programmed through a cross-platform Arduino Software IDE.

3.2. ICARUS Wireless IoT Sensor Mote

The prototype NITOS wireless sensor mote [11], is comprised of open-source and configurable modules. ICARUS mote features the ATmega32u4 [12] microcontroller running at 8MHz and operating at 3.3V. The aforementioned microcontroller is fully compatible with the Arduino platform that enables ease of software development and provides compatibility with several commercial sensing modules. Moreover, the platform is equipped with an XBee radio interface that enables communication with the respective gateway. The XBee module is a tiny device ideal for setting up mesh networks and has a defined rate of 250 kbps. This module uses the IEEE

802.15.4 stack which we analyzed above.Apart from the Xbee module, NITOS mote can also feature a WiFi wireless interface in order to communicate with WiFi gateways. The developed mote currently features specific sensing modules, an air temperature and humidity sensor, a light intensity sensor and a human presence sensor. Various types of sensing modules and actuators can be further integrated exploiting existing Arduino software that implements several existing communications protocols. The firmware can be easily uploaded through the on-board USB connection. The mote features:

- 8-bit AVR RISC-based microcontroller.
- Operates at 8MHz/3.3v.
- 32KB flash memory.
- 10-bit A/D-converter.
- USB 2.0 Interface.
- 4 x ADC pins.
- 12 x Digital I/Os.
- Arduino compatible.

ATmega32u4

In this thesis the IoT sensor mote that has been used for the deployment is based on the ATmega1284p micro-controller as we mentioned above. In brief the characteristics of the mote:

- Support for two wireless interfaces (XBee footprint see paragraph 3.3).
- Flash memory to support over-the-air programming.
- Programmable analog switches to turn off electronic blocks when not in use (wireless interfaces, sensors, other peripheral circuits) to minimize overall power consumption.
- DC-DC voltage converters (instead of LDO regulators) for higher power efficiency.
- Expansion ports for connecting external sensors.
- On-board digital temperature, humidity & ambient light sensors.
- IC for Battery charging up to 1A, over solar panel.
- Fuel Gauge Battery monitoring IC.
- Real-Time-Clock IC.

Icarus Wireless IoT Sensor Mote

3.3. XBee

The XBee radios are a set of RF modules designed to operate under the IEEE 802.15.4 networking protocol. They are developed by Digi International and provide low-cost wireless connectivity to WSN nodes. These modules provide high-throughput and low latency communication. Communication is achieved through a UART.

For the deployment the type of XBee module that have been used is:

XBee S8[13] and XBee 865/868 Low Power SMT embedded RF modules provide wireless connectivity to end-point devices in mesh networks. Utilizing the XBee-PRO Feature Set, these modules provide wireless communication based on the IEEE 802.15.4 standard and are interoperable with other devices, including devices from other vendors. With the XBee, users can have their network up-and-running in a matter of minutes without configuration or additional development. More specific ,the features of this xbee are:

- Indoor/Urban Range : up to 370 ft (112 m) w/2.1 dBi antenna, up to 46 ft (14 m) w/embedded antenna
- Outdoor RF line-of-sight Range : up to 5.2 miles (8.4 km) w/2.1dB dipole antenna, up to 0.4 mi (640 m) w/ embedded antenna.
- Transmit Power Output: 12 dBm (16 mW
- RF Data Rate (High) : 80 kbps
- RF Data Rate (Low : 10 kbps
- Serial Interface Data Rate (software selectable) : 9600-230400 baud
- Supply Voltage: 2.7 to 3.6 VDC

- Operating Frequency Band: 863 to 870 MHz
- Operating Temperature: -40º to 85º C (industrial)
- Antenna Options: Pad on module edge, U. FL RF connector, embedded antenna
- Digital I/O: 13 I/O lines, 5 dedicated to SPI that can be used as digital output
- ADC: 6 10-bit analog inputs.

XBee s8 868MHz

The XBee modules can be configured through Digi's XCTU program. The XCTU is a multi-platform application designed to enable developers to interact with the XBee RF module. This interaction includes firmware updates in order to change operating modes (e.g. from ZigBee Coordinator to ZigBee Router) and to manage and configure the RF devices.

XBees have two modes of operation:

- AT Command Mode (Transparent Mode): The XBees that operate in AT Command Mode can be configured through simple AT commands issued through the serial UART. The XBee module enters the command mode when the characters "+++" are sent to the

- XBee. Afterwards the XBee replies with "OK" and the XBee is in command mode. An example of an AT command is "ATID" which returns/sets the XBees PAN ID.

- API Mode: API (Application Programming Interface) mode is a frame-based method for sending and receiving data to and from a radio's serial UART. These frames are called ZigBee Device Objects [14]. The API is an alternative to the default transparent mode. The API allows the programmer the ability to:
    - Change parameters without entering command mode
    - View RSSI and source address on a packet by packet basis
    - Receive packet delivery confirmation on every transmitted packet

XBee comes with various connectors for the antenna connection or with a built-in wired antenna.

3.4. External Soil Sensors for WSN

Sensors are common and serve a variety of functions in industrial applications. The industrial category serves as a catch-all classification, encompassing diverse applications such as smart buildings, asset tracking, medical, energy & oil, and others. Within that category is smart agriculture, sometimes referred to as precision farming. What effect is IoT having on the smart agricultural sensors? If we define IoT as direct or indirect cloud connectivity, the answer is currently mixed. While sensors are driving a number of applications within agricultural

applications, IoT or cloud connectivity has to make its case within each sub-application of agriculture.

As part of the networking of sensors for agro-meteorological and agricultural needs in the wireless IoT Sensor Motes that we described above, certain types of sensors were added.

3.4.1. Soil Moisture

Measuring soil moisture for growing purposes for our platform the WATERMARK [15] sensors have been used which are a calibrated, indirect method of measuring soil water tension. Soil water tension (or matric potential) has to be overcome for the plant to move water into its root system. Different soil types will have different tensions even at the same volumetric measurement, making volumetric information relative to local conditions and often requiring site calibration for reading equipment. Because we use soil water tension, there is no site calibration required when using our sensors. The Watermark sensor is consisting of two electrodes highly resistant to corrosion embedded in a granular matrix below a gypsum wafer. The resistance value of the sensor is proportional to the soil water tension, a parameter dependent on moisture that reflects the pressure needed to extract the water from the ground. It is designed to be a permanent sensor, placed in the soil to be monitored and "read" as often as necessary with a portable or stationary device.

Soil Moisture (Watermark)

Sensor Features

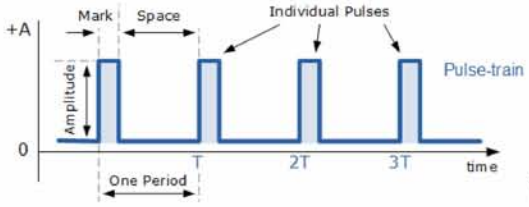
- Range of measurement from 0 to 239 cb (kPa)
- Fully solid-state
- Not affected by freezing temperatures
- Internally compensated for commonly found salinity levels
- NO maintenance required

Sensor programming for soil moisture measurements data for the deployment.

Sensor setup and frequency calculation.

```
#include "include/watermark.h"

void watermark_setup()
{
 pinMode(watermarkPin, INPUT);
}

float calculate_freq()
{
 float frequency, Htime, Ltime, Ttime = 0;

 for (int i =0; i<100; i++)
 {
  delay(5);
  Htime = pulseIn(watermarkPin, HIGH);
  delay(5);
  Ltime = pulseIn(watermarkPin, LOW);
  delay(5);
  Ttime += Htime + Ltime;
 }
 Ttime /= 100.0;
    frequency = (1000000.0 / Ttime);
    frequency /= (1000.0);

 return frequency;
}
```

Inside calculate_freq() function we use a loop of 100 iterations to take samples of the sensor's transmitted signal. Inside the loop and with the help of pulseIn() function, we find the duration of High (picks) and Low pulses of the signal in order to find out how often the sensor changes because of the soil moisture in the ground. After the end of iterations we have the total time (Ttime) of the samples and with a division we take the average time and convert it to frequency.



Transmitted signal

Read frequency and return values between 0 - 200 in centibars

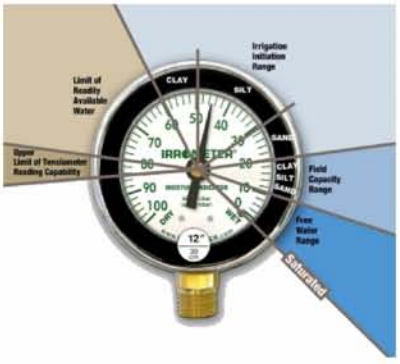
```
float read_soil_moisture ()
{
 float freq = calculate_freq();

 if (freq < 0.091)
 {
  return 200.0;
 }
 else if (freq < 0.331)
 {
  return (77.0 + ((freq - 0.331) * ((200.0 - 77.0)/(0.091 - 0.331))));
 }
 else if (freq < 0.383)
 {
  return (65.0 + ((freq - 0.383) * ((77.0 - 65.0)/(0.331 - 0.383))));
 }
 else if (freq < 0.453)
 {
  return (37.0 + ((freq - 0.453) * ((65.0 - 37.0)/(0.383 - 0.453))));
 }
 else if (freq < 0.495)
 {
  return (29.0 + ((freq - 0.495) * ((37.0 - 29.0)/(0.453 - 0.495))));
 }
 else if (freq < 0.540)
 {
  return (25.0 + ((freq - 0.540) * ((29.0 - 25.0)/(0.495 - 0.540))));
 }
 else if (freq < 0.595)
 {
  return (13.0 + ((freq - 0.595) * ((25.0 - 13.0)/(0.540 - 0.595))));
 }
 else if (freq < 0.786)
 {
  return (1.0 + ((freq - 0.786) * ((13.0 - 1.0)/(0.595 - 0.786))));
 }
 else
 {
  return 0.0;
 }

}
```

The read_soil_moisture() function uses the previous function calculate_freq() to take a frequency value as an input and separates 9 cases to extract the values of the results from 0 to 200 in centibars(measuring unit for pressure needed to extract the water from the ground),according to the soil moisture measured by the sensor. The calculation of the return

values is coming from a Linear Interpolation Equation $y_2 = \frac{(x_2 - x_1)(y_3 - y_1)}{(x_3 - x_1)} + y_1$. To interpolate the y_2 value: x_1, x_3, y_1 and y_3 need to be entered/copied from the table. x_2 defines the point to perform the interpolation. y_2 is the interpolated value and solution.

To explain the meaning of these interpolated values we use the following readings as a general guideline from the sensor's tutorial [16]:

- 0-10 Centibars = Saturated soil
- 10-30 Centibars = Soil is adequately wet (except coarse sands, which are beginning to lose water)
- 30-60 Centibars = Usual range for irrigation (most soils)
- 60-100 Centibars = Usual range for irrigation in heavy clay
- 100-200 Centibars = Soil is becoming dangerously dry for maximum production. Proceed with caution!



Guideline for centibars

3.4.2. Soil Temperature (dallas temperature)

The DS18B20[17] digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems.

Dallas Temperature Sensor

Program code for taking soil temperature in Celsius and Fahrenheit.

```
#include <OneWire.h>
#include <DallasTemperature.h>

// Sensor input pin
#define DATA_PIN 3
// How many bits to use for temperature values: 9, 10, 11 or 12
#define SENSOR_RESOLUTION 9
// Index of sensors connected to data pin, default: 0
#define SENSOR_INDEX 0

OneWire oneWire(DATA_PIN);
DallasTemperature sensors(&oneWire);
DeviceAddress sensorDeviceAddress;

void setup() {
  Serial.begin(9600);

  sensors.begin();
  sensors.getAddress(sensorDeviceAddress, 0);
  sensors.setResolution(sensorDeviceAddress, SENSOR_RESOLUTION);
}


void loop() {
  sensors.requestTemperatures();
  // Measurement may take up to 750ms

  float temperatureInCelsius = sensors.getTempCByIndex(SENSOR_INDEX);
  float temperatureInFahrenheit = sensors.getTempFByIndex(SENSOR_INDEX);

  Serial.print("Temperature: ");
  Serial.print(temperatureInCelsius, 4);
  Serial.print(" Celsius, ");
  Serial.print(temperatureInFahrenheit, 4);
  Serial.println(" Fahrenheit");
}
```

3.5. External Weather Sensors for WSN

3.5.1. Air Temperature & Air Humidity

The humidity and temperature sensors (SHT21) [18] integrate sensor elements plus signal processing on a tiny footprint and provide a fully calibrated digital output. A unique capacitive

sensor element is used for measuring relative humidity, while temperature is measured by a band-gap

sensor. Its CMOSens® technology guarantees excellent reliability and long-term stability. The humidity sensors are seamlessly coupled to a 14-bit-analog-to-digital converter and a serial interface circuit. This results in superior signal quality, a fast response time, and insensitivity to external disturbances .

Humidity and Temperature

Code for initialize sensor and read temperature and humidity.

```
#include "include/sht21.h"

void sht21_setup()
{
 Wire.begin();
}

float read_humidity()
{
 power_twi_enable();
 Wire.begin();
 return SHT2x.GetHumidity();
}

float read_temp ()
{
// power_twi_enable();
// Wire.begin();
 return SHT2x.GetTemperature();
}
```

3.5.2. Wind Speed & Direction

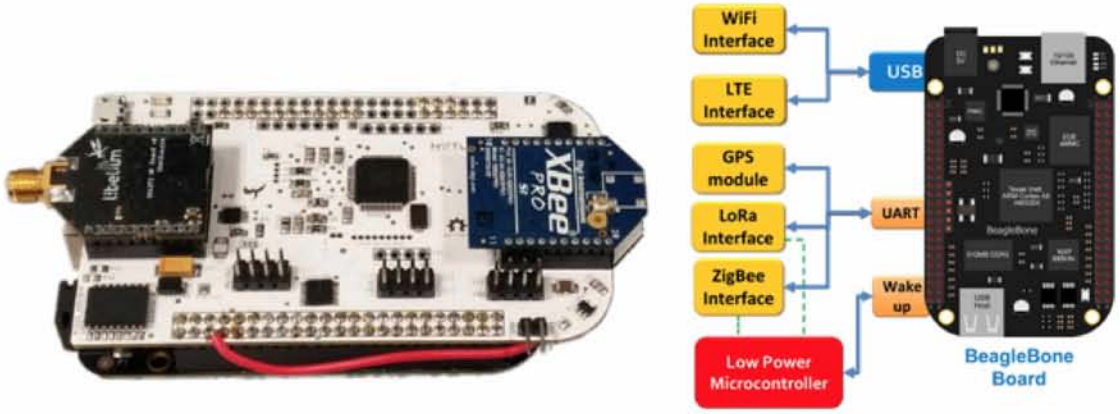
The 05103 Wind Monitor [19] is a lightweight, high-performance wind-speed and wind-direction sensor, a sturdy instrument for measuring wind speed and direction in harsh environments. Its simplicity and corrosion-resistant construction make it ideal for a wide range of wind measuring applications.

Wind sensor

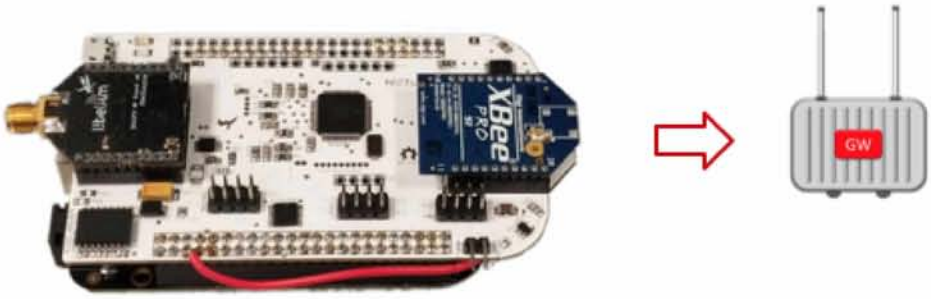
3.6. NITOS IoT Gateway

As mentioned above, all type of data are collected at the wireless sensor nodes,in our case ICARUS IoT nodes, compressed and transmitted to the gateway directly or, if required, uses other wireless sensor nodes to forward data to the gateway. In this paragraph there is a description of the NITOS gateway that used in the thesis.

This gateway developed on top of BeagleBone(a low-power open-source single-board computer) platform, through it, we can retrieve measurements from the WSNs as well as to configure parameters of each node.Actually it acts as a Web Server, implementing an interface between the user and the distributed sensors network.



NITOS IoT Gateway Architecture

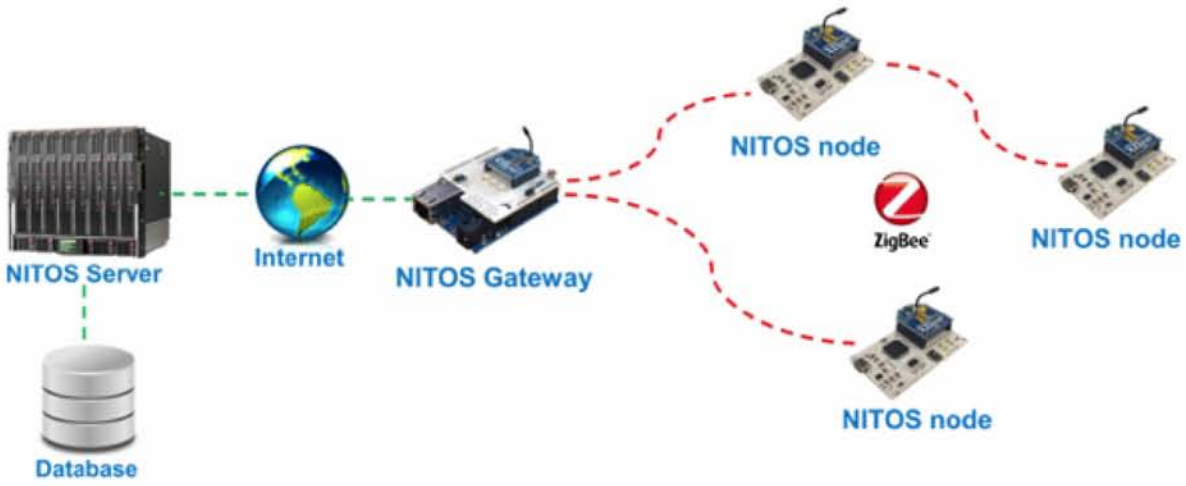


4. Topology Tool

The NITOS WSN Topology Tool is a monitoring tool designed to report and display the network topology of an existing WSN. The tool acquires the LQI Table (Link Quality Indicator) and the Routing Table of each node. The acquisition is accomplished by sending ZigBee Device Objects (ZDOs) through the XBee wireless interface of a dedicated Arduino- based device.

4.1. Existing WSN Topology

The tool was implemented based on the characteristics of the deployed NITOS Indoor WSN testbed. The NITOS Indoor WSN consists of several WSN motes and a WSN Gateway based on the Arduino platform. The gateway node is connected through an Ethernet network to the NITOS Server. Each NITOS WSN mote is equipped with a microcontroller, several sensors and an XBee wireless interface. The wireless interface is configured to operate in AT Command mode. In order to acquire the topology information of the WSN we must use two types of ZDOs (0x0031 & 0x0032), and to be able to send the ZDOs we need an XBee in API mode.



Πανεπιστήμιο Θεσσαλιάς

5. Precision Agriculture (PA)

Precision agriculture [20] (**PA**) or **satellite farming** or **site specific crop management** (**SSCM**) is a farming management concept based on observing, measuring and responding to inter and intra-field variability in crops. The goal of precision agriculture research is to define a decision support system (DSS) for whole farm management with the goal of optimizing returns on inputs while preserving resources.

PA focusing on:

- Efficient use of natural and manmade resources.
- Increase in production while reducing costs.
- "Looking for re-interpretation of the farming practices through use of data-centric technologies".

The deployment of thesis agrometeorological platform focused on two areas:

1. Irrigation
2. Pest management

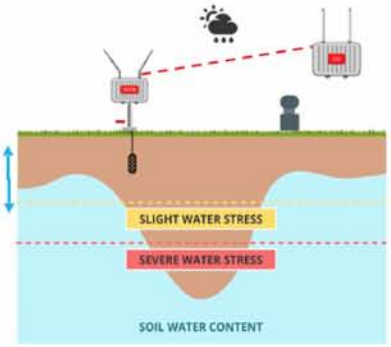
6. Irrigation Monitoring

Saving water is always important.There is no one-size-fits-all in agriculture, so there should not be a one-size-fits-all solution. This irrigation control system is easily configurable to meet the farmer's needs. Create an optimized water budget with irrigation schedule. Effortlessly create an

optimized schedule through the use of soil sensors and other essential hardware components that pointed out in previous chapters. Soil Sensors measure the field's water inputs and outputs:

- Inputs
    - Precipitation - The definition of precipitation is any form of water - liquid or solid - falling from the sky. It includes rain, sleet, snow, hail and drizzle plus a few less common occurrences such as ice pellets, diamond dust and freezing rain
    - Past irrigation
- Outputs
    - Evapotranspiration - the process by which water is transferred from the land to the atmosphere by evaporation from the soil and other surfaces and by transpiration from plants
    - Infiltration
    - Runoff - the draining away of water (or substances carried in it) from the surface of an area of land

WSN with soil moisture sensor communicates through 802.15.4 protocol with gateway



7. Irrigation Processing

In the framework of this postgraduate thesis we extend an existing backend service to communicate through MQTT protocol with NITOS Gateway. MQTT is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol, designed as an extremely lightweight publish/subscribe messaging transport. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium. After the data retrieval data analysis was used to calculate the amount of irrigation required. In a more abstract way the equation is like this:

Irrigation Quantity = Water Outputs – Inputs ± Δθ

For establishing connection between backend with GW(http://10.64.92.26) these commands were used.

e3atmisodiapnoi.js | parameters.js | sensor.js | aggregator.js | logic.js

```
var mqtt = require('mqtt')
var client = mqtt.connect('http://10.64.92.26');

var sensor_hash = [];
var sensor_array = [];

client.on('connect', function() {
        client.subscribe("beaglebone/control/#");
        console.log("Beaglebone IoT Gateway connected");
});
client.on('close', function() {
        console.log("Beaglebone IoT Gateway disconnected");
});
```

Πανεπιστήμιο Θεσσαλιάς

7.1. Estimate Evapotranspiration

Evapotranspiration [21] is the sum of evaporation and plant transpiration from the Earth's land and ocean surface to the atmosphere. Evaporation accounts for the movement of water to the air from sources such as the soil, canopy interception, and waterbodies. Transpiration accounts for the movement of water within a plant and the subsequent loss of water as vapor through stomata in its leaves. Evapotranspiration is an important part of the water cycle. When all measurements from the WSN gather together in GW and send to the backend service in the cloud, must go through a new process, the process of evapotranspiration function. This process is important for irrigation suggestions to the farmers and crop irrigation. Adequate water supply is an essence for agriculture and the crops can be damaged in either of situation of excess of water supply or in shortage of water supply. For the thesis integration we asked for the help of an agronomist to create an evapotranspiration estimation method and we translated this into a program code.

e3atmisodiapnoi.js | parameters.js | sensor.js

```
var calculate_u2 = function(){
    var z = 200, uz = wind_speed;
    return uz*Math.pow(2/z, 0.2);
}
var calculate_G = function(){
    var Ti = Tmean_today, Ti1 = Tmean_yesterday;
    return 0.38*(Ti - Ti1);
}
var calculate_Rso = function(){
    var z = 200, Ra = calculate_Ra();
    return (0.75+2*Math.pow(10,-5)*z)*Ra;
}
var calculate_Rnl = function(){
    var T = Tmean_today;
    var Rs = calculate_Rs(), Rso = calculate_Rso(), ea = calculate_ea();
    var s = 4.903*Math.pow(10,-9), Tk = T + 273; //Kelvin
    return s * Math.pow(Tk, 4)*(0.34-0.14*Math.pow(ea,0.5))*(1.35*Rs/Rso-0.35);
}
var calculate_N = function(){
    var ws = calculate_ws();
    return (24/Math.PI)*ws;
}
var calculate_ws = function(){
    var f = 0.698; //lat
    var d = calculate_d();
    return Math.acos(-Math.tan(f)*Math.tan(d));
}
var calculate_d = function(){
    var J = calculate_J();
    return 0.4093*Math.sin(2*Math.PI*J/365-1.39);
}
var calculate_J = function(){
    var now = new Date();
    var start = new Date(now.getFullYear(), 0, 0);
    var diff = now - start;
    var oneDay = 1000 * 60 * 60 * 24;
    var day = Math.floor(diff / oneDay);
    return day;
}
var calculate_dr = function(){
    var J = calculate_J();
    return 1+0.033*Math.cos(2*Math.PI*J/365);
}
var calculate_Ra = function(){
    var GSc = 0.082, dr = calculate_dr(), ws = calculate_ws(),d = calculate_d();
    var f = 0.698;  //lat
    return 1440*GSc*dr*(ws*Math.sin(f)*Math.sin(d) + Math.cos(f)*Math.cos(d)*Math.sin(ws))/Math.PI;
}
var calculate_Rs = function(){ // total hours of exposure
    var n = total_n;
    var a = 0.18, b = 0.55;
    var N = calculate_N(), Ra = calculate_Ra();
    return (a+b*n/N)*Ra;
}
```

```
var calculate_Rns = function(){
    var Rs = calculate_Rs();
    return 0.77*Rs;
}
var calculate_Rn = function(){
    var Rns = calculate_Rns(), Rnl = calculate_Rnl();
    return Rns - Rnl;
}
var calculate_g = function(){
    var P = calculate_P();
    return 0.665+Math.pow(10,-3)*P;
}
var calculate_P = function() {
    var z = 200;
    return 101.3*Math.pow(((293-0.0065*z)/293),5.26);
}
var calculate_D = function(){
    var T = Tmean_today;
    var ea = calculate_ea();
    return 4099*(ea)/Math.pow((T + 237.3),2);
}
var calculate_ed = function(){
    var ea = calculate_ea();
    return (ea*RHmean)/100;
}
var calculate_ea = function(){
    var T = Tmean_today;
    return 0.611 * Math.exp((17.2*T)/(T + 237.3));
}
var calculate_ETr = function(){
    var T = Tmean_today;
    var D = calculate_D(), Rn = calculate_Rn(), G = calculate_G();
    var g = calculate_g(), u2 = calculate_u2(), ea =calculate_ea();
    var ed = calculate_ed();
    return (0.408*D*(Rn-G)+g*(900/(T+273))*u2*(ea - ed))/(D+g*(1+0.34*u2));
}
var calculate_ETc = function(){
    var ETr = calculate_ETr();
    var Kc = 0.611;
    return Kc*ETr;
}
var calculate_mean = function(results){
    sum = 0;
    results.forEach(function(element){
        sum += element.data;
    });
    return parseFloat((sum/results.length).toFixed(2));
}
```

```
// ************************* Main *************************
var calculate = function(){
    var async = require('async');
    var Sensor = require('../../models/sensor');
    var sensors_id = {
        temperature:    "58909eef611f71184757ddc4",
        wind_speed:     "5890a800611f71184757ddd0",
        humidity:       "58909f7c611f71184757ddc8",
        luminosity:     "5890a04b611f71184757ddcc"
    };
    async.parallel({
        temperature_yesterday: function(callback) {
            Sensor.findById(sensors_id.temperature).exec()
            .then(function(sensor){
                return sensor.getDayData(-1);
            })
            .then(function(events){
                callback(null,events);
            });
        },
        temperature_today: function(callback) {
            Sensor.findById(sensors_id.temperature).exec()
            .then(function(sensor){
                return sensor.getDayData(0);
            })
            .then(function(events){
                callback(null,events);
            });
        },
        wind_speed: function(callback) {
            Sensor.findById(sensors_id.wind_speed).exec()
            .then(function(sensor){
                return sensor.getDayData(0);
            })
            .then(function(events){
                callback(null,events);
            });
        },
        humidity: function(callback) {
            Sensor.findById(sensors_id.humidity).exec()
            .then(function(sensor){
                return sensor.getDayData(0);
            })
            .then(function(events){
                callback(null,events);
            });
        },
        luminosity: function(callback) {
            Sensor.findById(sensors_id.luminosity).exec()
            .then(function(sensor){
                return sensor.getDayData(0);
            })
            .then(function(events){
                callback(null,events);
```

```
            });
        }
    }, function(err, results) {
        if(err)
            console.log(err);

        // var Tmean_yesterday = calculate_mean(results.temperature_yesterday);
        Tmean_yesterday = calculate_mean(results.temperature_today); //TESTING ONLY
        Tmean_today     = calculate_mean(results.temperature_today);
        RHmean          = calculate_mean(results.humidity);
        wind_speed      = calculate_mean(results.wind_speed);
        total_n         = calculate_mean(results.luminosity);

        console.log(calculate_ETc());
    });
}
module.exports.calculate = calculate;
```

8. Visualization of measurements coming from WSN

In order to better serve the needs of a farmer, web and mobile interfaces were created for data visualization in this master thesis. Through these interfaces, the farmer can log in by register remotely from his computer or his mobile phone and then import his crops individually with their respective elements. Having completed this process he is now able to see his crops and by clicking on a crop to display information and notifications regarding growth, fertilization and possible pests for the selected type of crop. In addition, it can remotely monitor the state of its field through the sensors. In this implementation, the sensors that have been used , as described above, monitor air temperature, air humidity, wind speed and direction,as well as soil humidity, soil temperature and evapotranspiration.

For the software implementation of data visualization we used an implemented service as a backend service for the frontend applications. This service uses a rest api for sending data to the clients.

The Rest API documentation for the http requests is:

IP: http://194.177.207.12:8080/api

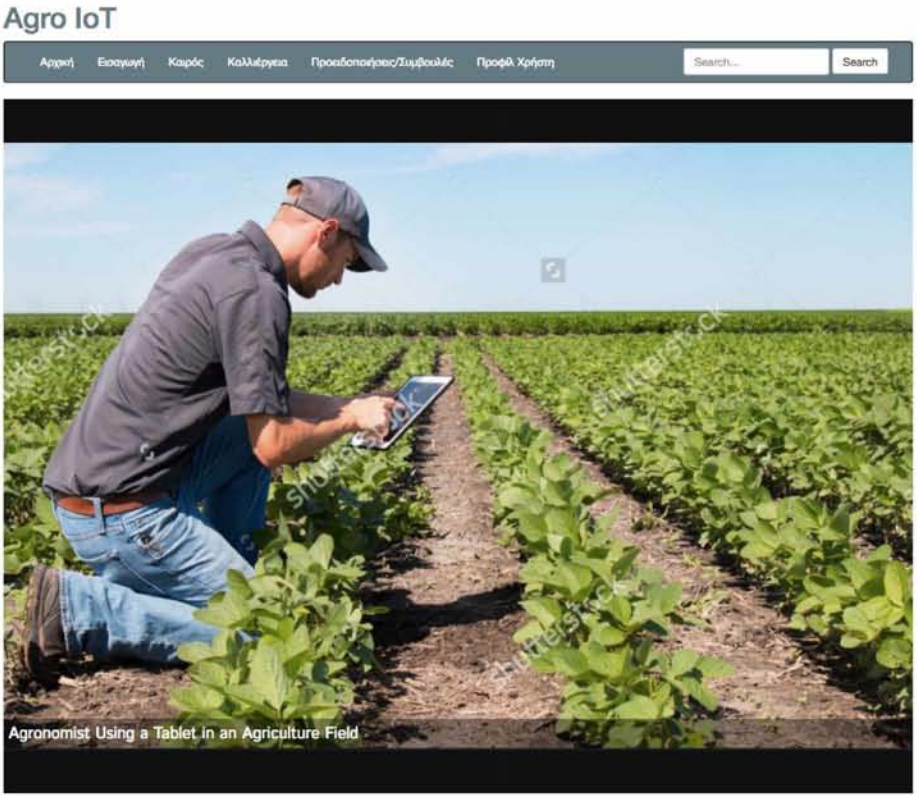
Routes Categories	
Login/Authorization/Authentication	/api
User Profile	/profile
Cultivations	/cultivations
Warnings	/warnings
Forecast	/forecast
Categories	/categories
Irrigations	/irrigation

Every route supports all methods POST,DELETE,GET,PUT.

8.1. Web Application

The programming languages used for the web implementation is Angular2JS for the functional part, Typescript for the descriptional part, HTML5 for the views and CSS - Bootstrap for the graphic part. The first image the user sees with the application opening is a menu with 6 options-tabs “Home”, “Forecast”, “Login”, “Cultivations”, “Warnings” , “Profile”. Below we will analyze the functions of each tab one by one.

Home tab view



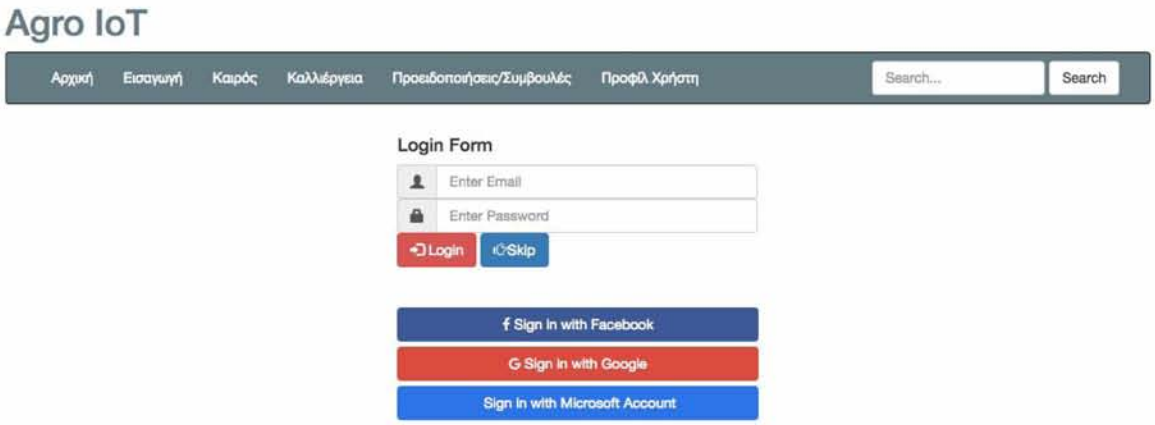
For the first time user has to enter an email and a password and a registration query is executed in the route /auth/register with parameters username and password. After success registration a login http post request is called with authentication through unique authentication token. The json reply from the server looks like : {message: ‘'User successfully created!' token:

‘JWT eyJhbGciOiJIUzI1NiIsInR ...' username : 'ellie' , password: ‘1234’ } . The authentication token is necessary because is included as http header in all http requests inside the app like this:

HTTP REQUEST HEADER : Authorization: “JWT nR5cCI6Ikp ...” , for security reasons. This token differentiates from user to user and is stored to local storage after the user’s login in order to retrieve it every time the user opens the application without re-login.

```
20
21      // Uses http.get() to load a single JSON file
22      login(data) {
23        var headers = new Headers();
24        headers.append('Content-Type', 'application/x-www-form-urlencoded');
25        // let urlSearchParams = new URLSearchParams();
26        // urlSearchParams.append('username', username);
27        // urlSearchParams.append('password', password);
28        // let body = urlSearchParams.toString()
29        var username = data.credentials.username;
30        var password = data.credentials.password;
31        var creds = "username=" + username + "&password=" + password;
32        console.log(creds);
33        return this.http.post('http://194.177.207.12:3000/api/signup', creds, {headers:headers})
34        .map((res:Response) =>
35             {
36
37                  console.log(res);
38                  var body = res.json();
39                  console.log(body);
40                  if (body.response){
41                      let user = res.json();
42                      if (user && user.token) {
43                          // store user details and jwt token in local storage to keep user logged in between page refreshes
44                          localStorage.setItem('currentUser', JSON.stringify(user));
45                      }
46                  }
47                  else{
48                      return body;
49                  }
50             });
51      }
52
53
54    logout() {
55      localStorage.removeItem("user");
56      this._router.navigate(['login']);
57    }
58
59  }
```

Login tab view.



From this screen is obvious that user has an option of social login, also known as social sign-in. This is a form of single sign-on using existing information from a social networking service such as Facebook, Microsoft or Google+, to sign into a third party website instead of creating a new login account specifically for that website. It is designed to simplify logins for end users as well as provide more and more reliable demographic information to web developers. Social Login is a simple process, with the following steps.

1. The user enters the application and selects the desired social network provider.
2. A login request is send to the social network provider.
3. Once the social network provider confirms the user's identity, a current user will get access to the application. A new user will be registered as a new user and then logged into the application.

Πανεπιστήμιο Θεσσαλιάς

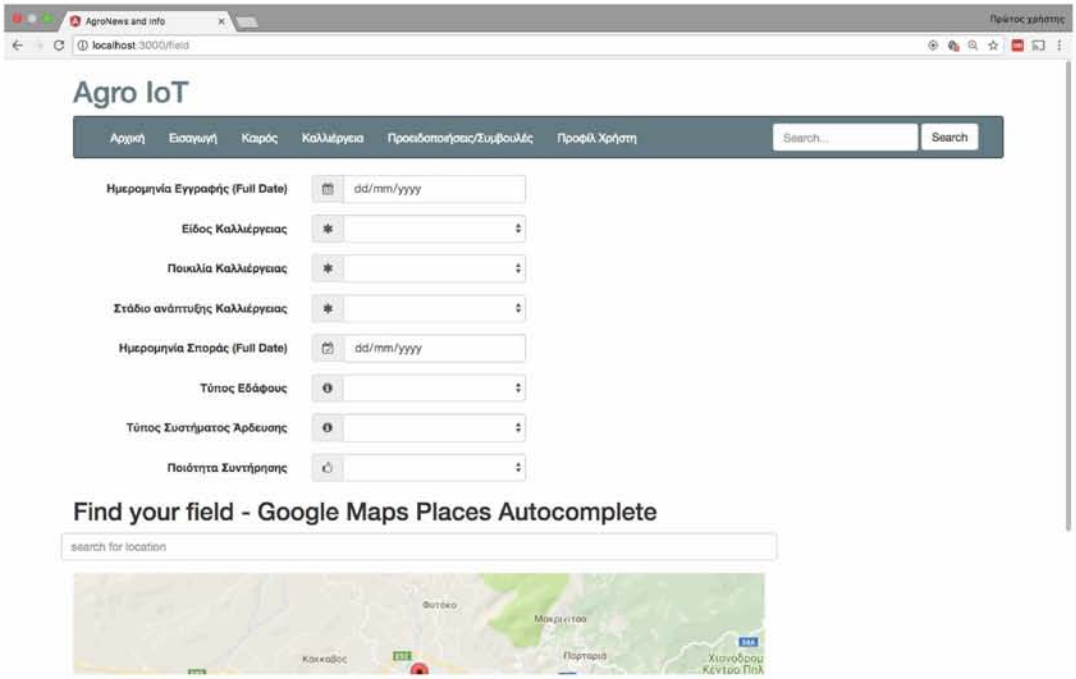
The code for authentication with Facebook provider.

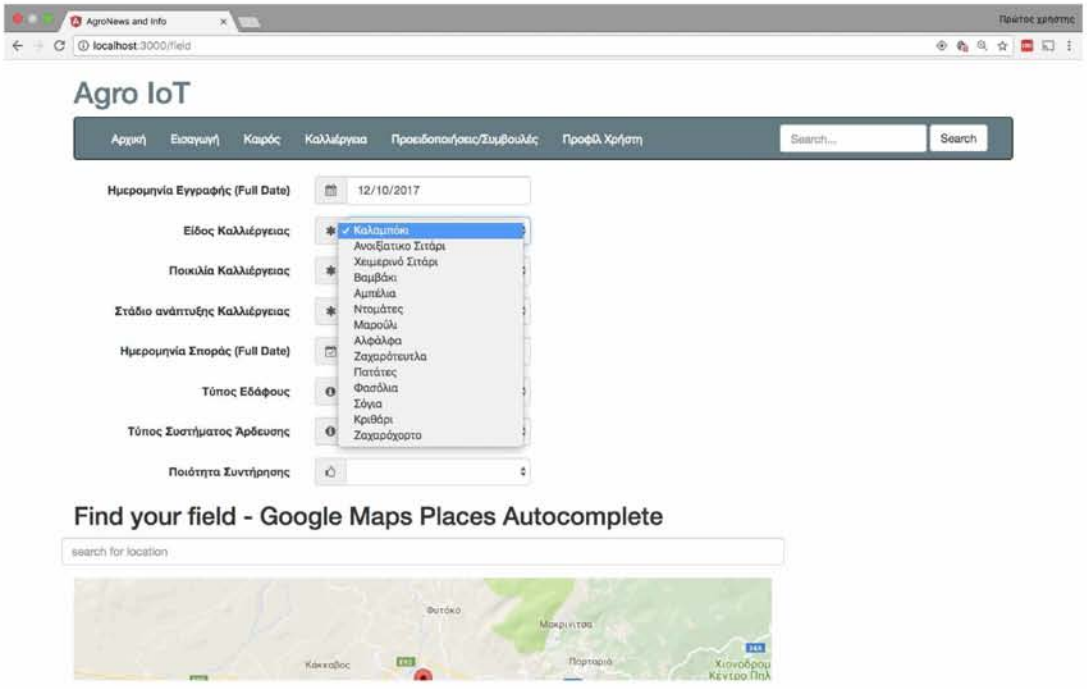
signService.js userService.js ×

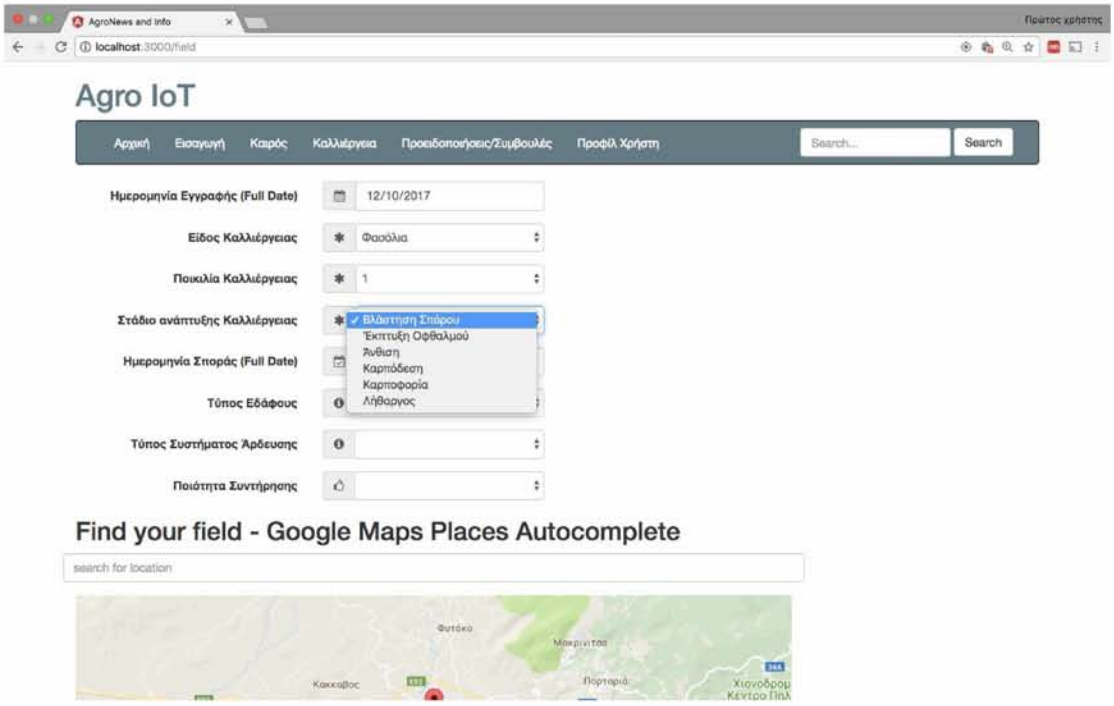
```
angular.module('starter.services')

.service('UserService', function() {
  // For the purpose of this example I will store user data on ionic local stora
  var setUser = function(user_data) {
    window.localStorage.starter_facebook_user = JSON.stringify(user_data);
  };

  var getUser = function(){
    return JSON.parse(window.localStorage.starter_facebook_user || '{}');
  };

  return {
    getUser: getUser,
    setUser: setUser
  };
});
```

In '/field' tab the user has to insert his cultivations with their characteristics.There is a drop down menu with cultivation types so the user just have to choose one option and depending from the selection appears other drop down menus with growth stage, cultivation soil, irrigation system and maintenance quality for this type.

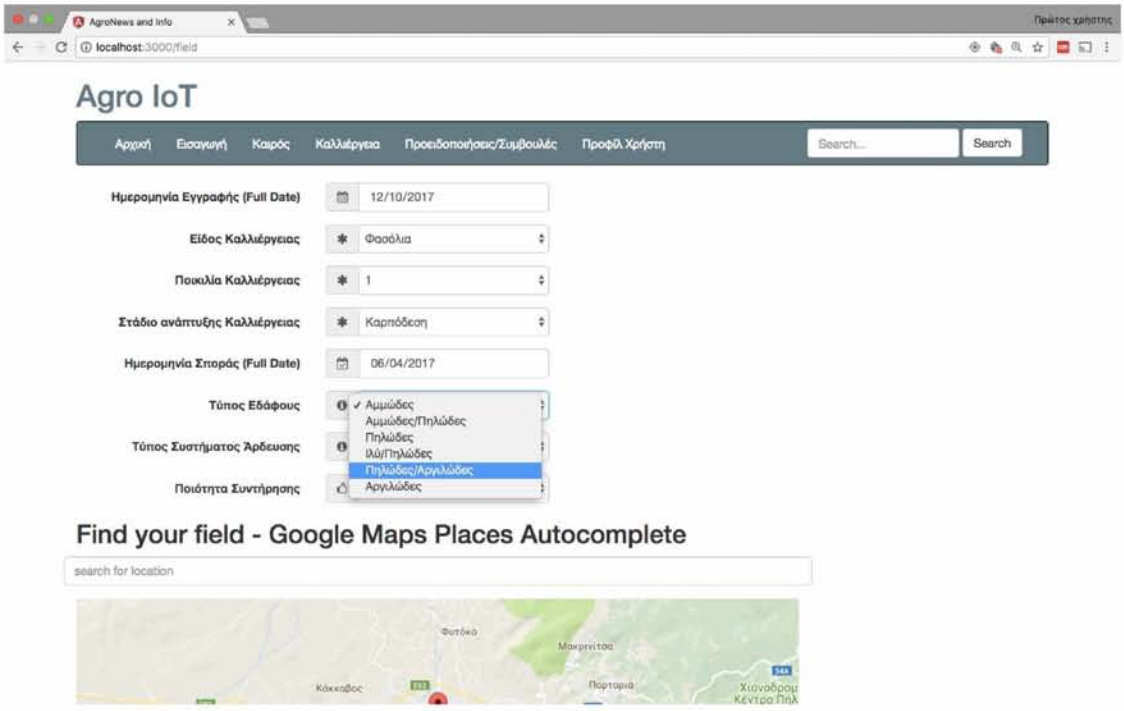
Cultivation tab views.

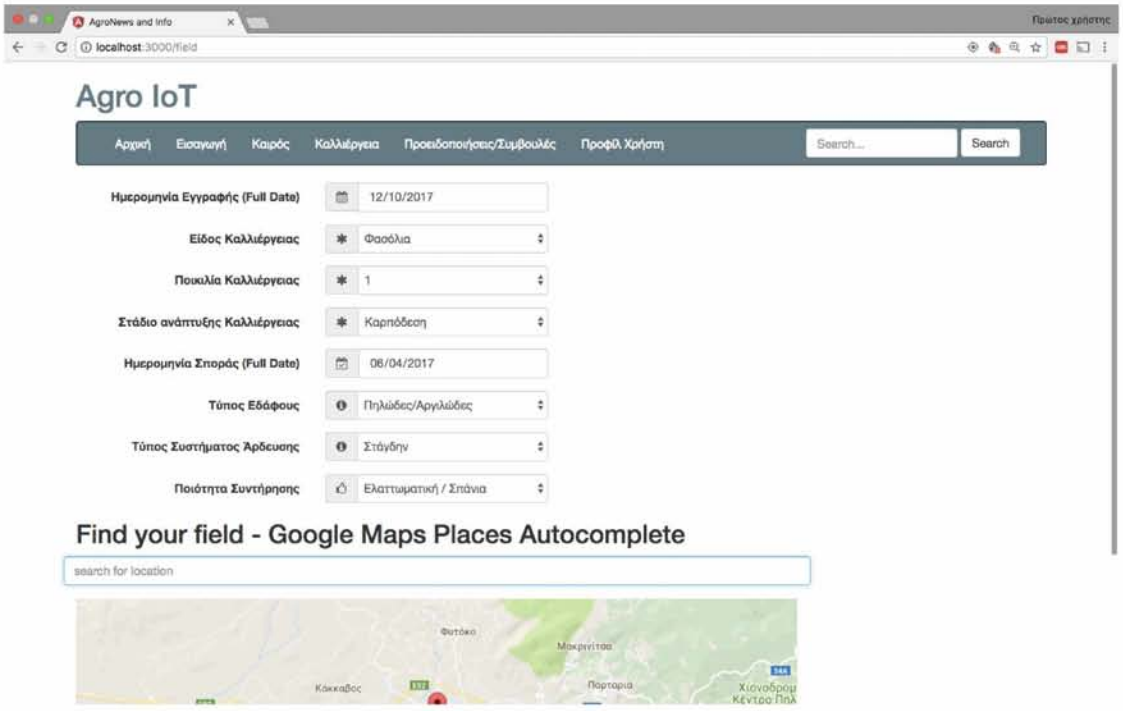




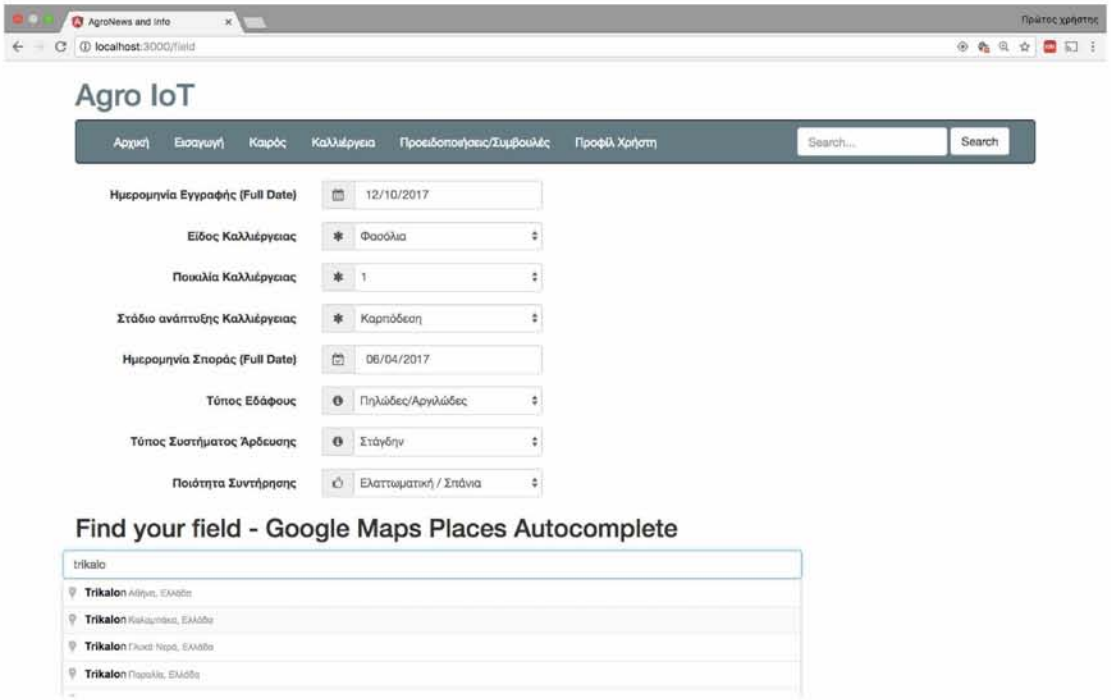


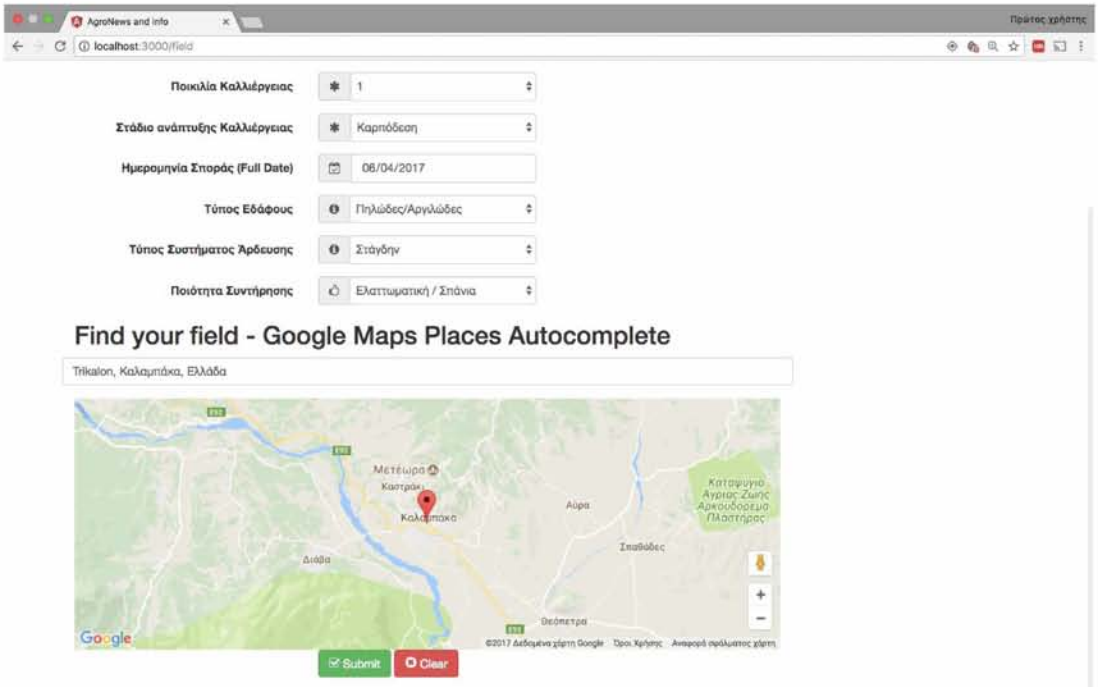
Πανεπιστήμιο Θεσσαλιάς





Another useful information for the cultivation is the position or the coordinates of the field. For bigger convenience there is an autocomplete search field so it is not necessary for the user to manually enter the coordinates or the area. After area selection there is google map with a pin in field's location.



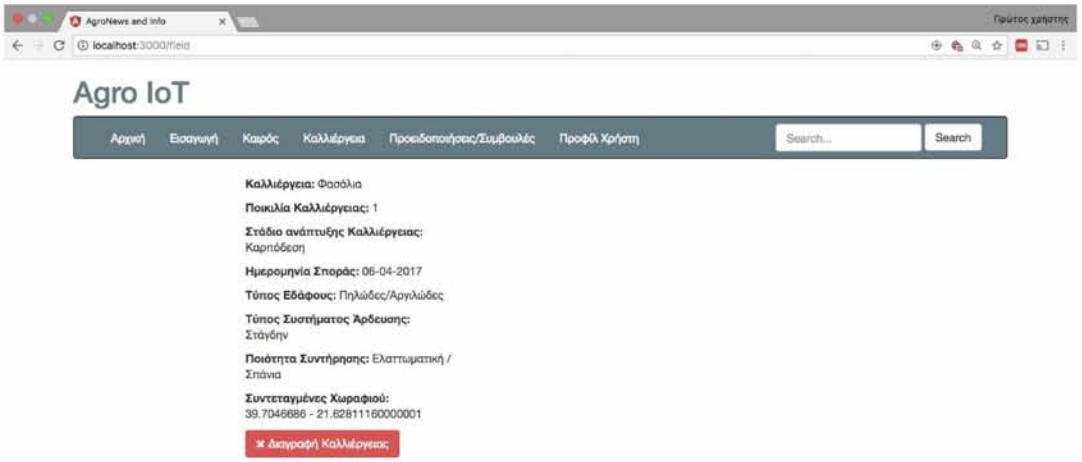


After the submit button being pressed an import query is sent to backend with a cultivation object in json format. An http post request with a token as authorization header and a json body with the cultivation is send to the server to be inserted into the database and assigned to the user with this authentication token . The declaration of a cultivation object looks like this:

warningId.component.ts | cultivation.service.ts × | warnings.component.ts | w

```
1   import { Injectable } from '@angular/core';
2
3   @Injectable()
4
5   export class MyCultivation {
6   cultivation: any ;
7
8   constructor() {
9   this.cultivation = {
10                      registrationDate: <Date> null,
11                      selectedCultivation: <string> null,
12                      selectedCultivar: <string> null,
13                      selectedStage: <string> null,
14                      sowDate: <Date> null,
15                      selectedSoilType: <string> null,
16                      selectedIrrigation: <string> null ,
17                      selectedQuality: <string> null,
18                      latitude: <string> null,
19                      longitude: <string> null
20                    };
21    }
22
23
24  }
```

If the response succeeds the following screen appears with the information of the imported crop or crops.



This web applications gives the farmer the ability to be informed about the prevailing weather conditions in his field through graph charts. This is feasible using an open weather rest api interface (http://api.openweathermap.org/data/2.5/forecast)[22] that returning the weather conditions for specific coordinates. To achieve this we make an http get request to this address "http://api.openweathermap.org/data/2.5/forecast" with coordinates as parameters, latitude and the final request turns into this: http.get("http://api.openweathermap.org/data/2.5/forecast?lat={lat}&lon={lon}") where {lat} is the cultivation's latitude and {lon} the cultivation's longitude from the field's form.

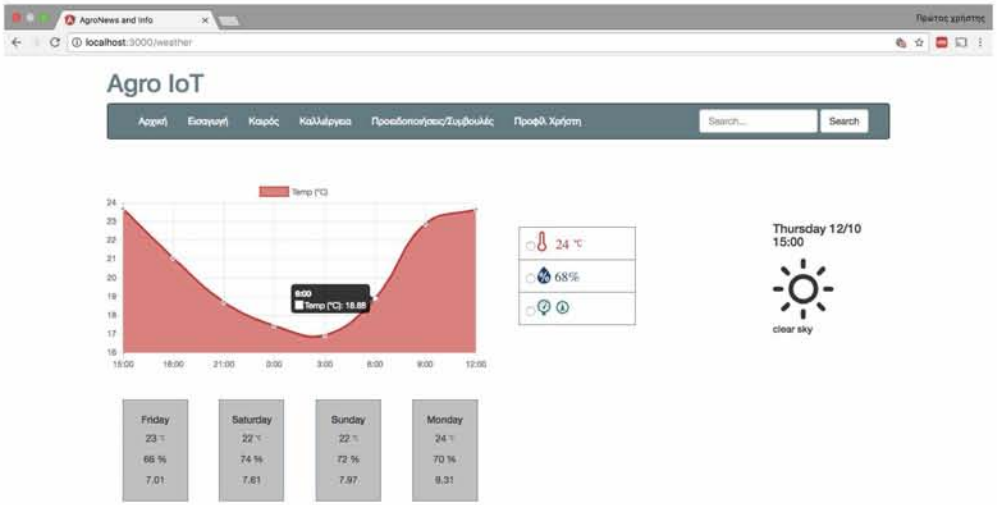
```
10
11  public lat : string;
12  public lon : string;
13  public params :string;
14
15       constructor(private http:Http) {
16           let params = new URLSearchParams();
17      }
18
19      getCoord() {
20          var units = "metric";
21          var appid = "a21046ab57b31843c68391741fe7270f";
22          // var params = "units=" + units + "&appid=" + appid;
23
24
25          return this.http.get('http://ip-api.com/json')
26           .map((res:Response) =>
27              {
28                  var data = res.json();
29                  console.log('GetCoordinates from Service:' )
30                  console.log(data);
31                  this.lat = data.lat;
32                  this.lon = data.lon;
33                  this.params = "units=" + units + "&appid=" + appid + "&lat=" + this.lat + "&lon=" + this.lon;
34                  return data;
35              });
36      }
37
38      weatherForecast(){
39          // var units = "metric";
40          // var appid = "a21046ab57b31843c68391741fe7270f";
41          // this.params = "units=" + units + "&appid=" + appid + "&lat=" + this.lat + "&lon=" + this.lon;
42          console.log('Params:' +  this.params);
43          return this.http.get('http://api.openweathermap.org/data/2.5/forecast', { search: this.params } ).map((res:Response) =>
44                          {
45                                  res.json();
46                                 console.log('weatherForecast from Serice:');
47                                 console.log(res.json());
48                                 return res.json();
49                          }
50                          );
51
52
53      }
54
55
```

The response from the weatherForecast() function is a json format with 5 days weather forecast and each day includes weather data every 3 hours. This json response consists of an array of 40 elements, 5 different days start from current day and end 4 days after,and for each day has 8 different weather data.

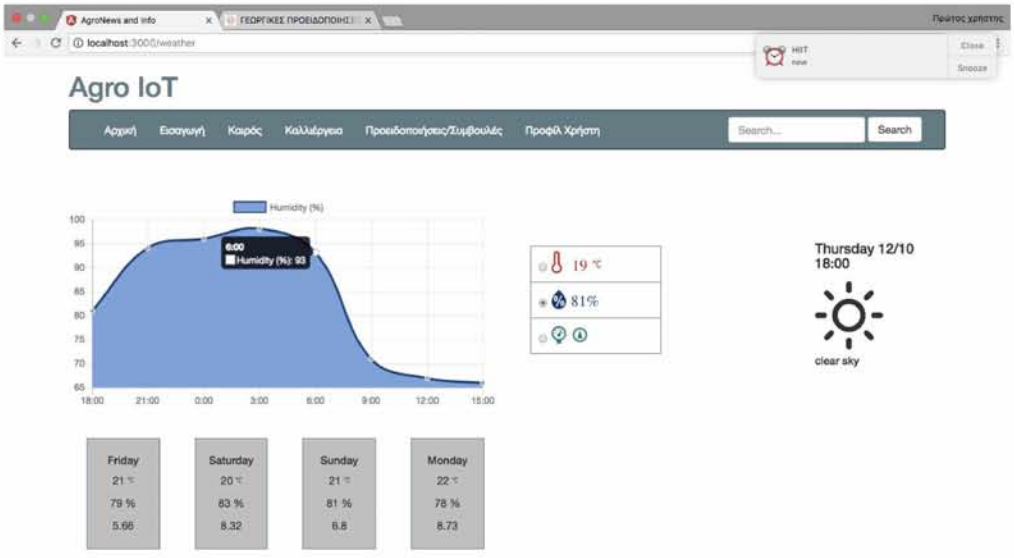
```
{
    "cod": "200",
    "message": 0.0143,
    "cnt": 40,
    "list": [
        {
            "dt": 1507831200,
            "main": {
                "temp": 19.37,
                "temp_min": 19.37,
                "temp_max": 20.23,
                "pressure": 1029.6,
                "sea_level": 1036.27,
                "grnd_level": 1029.6,
                "humidity": 81,
                "temp_kf": -0.86
            },
            "weather": [
                {
                    "id": 800,
                    "main": "Clear",
                    "description": "clear sky",
                    "icon": "01n"
                }
            ],
            "clouds": {
                "all": 0
            },
            "wind": {
                "speed": 5.36,
                "deg": 24.5013
            },
            "sys": {
                "pod": "n"
            },
            "dt_txt": "2017-10-12 18:00:00"
        },
        {
            "dt": 1507842000,
            "main": {
                "temp": 17.5,
                "temp_min": 17.5,
                "temp_max": 18.15,
                "pressure": 1029.53,
                "sea_level": 1036.14,
                "grnd_level": 1029.53,
                "humidity": 94,
                "temp_kf": -0.65
            },
            "weather": [
                {
                    "id": 800,
                    "main": "Clear",
                    "description": "clear sky",
```


After receiving this reply, frontend processes the list and represents the data in charts.

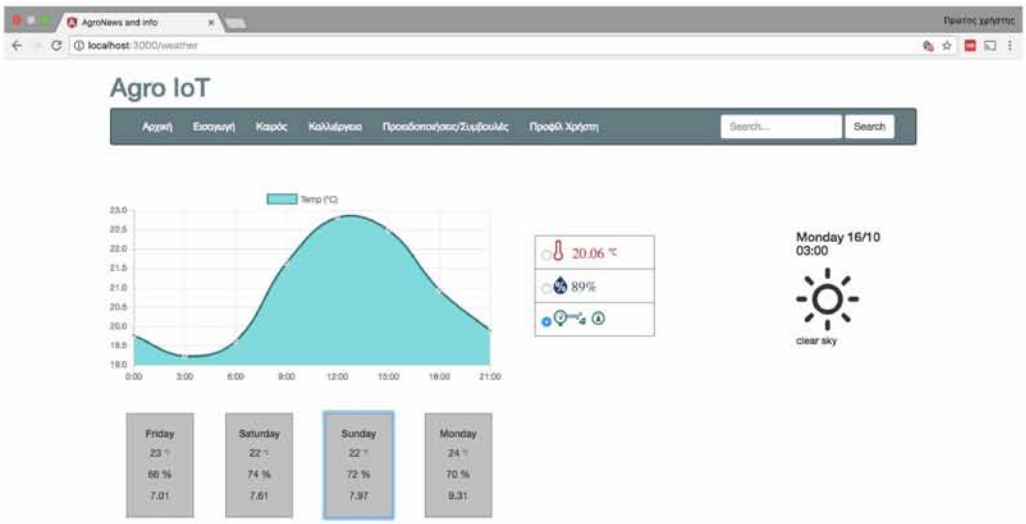
Temperature



Humidity

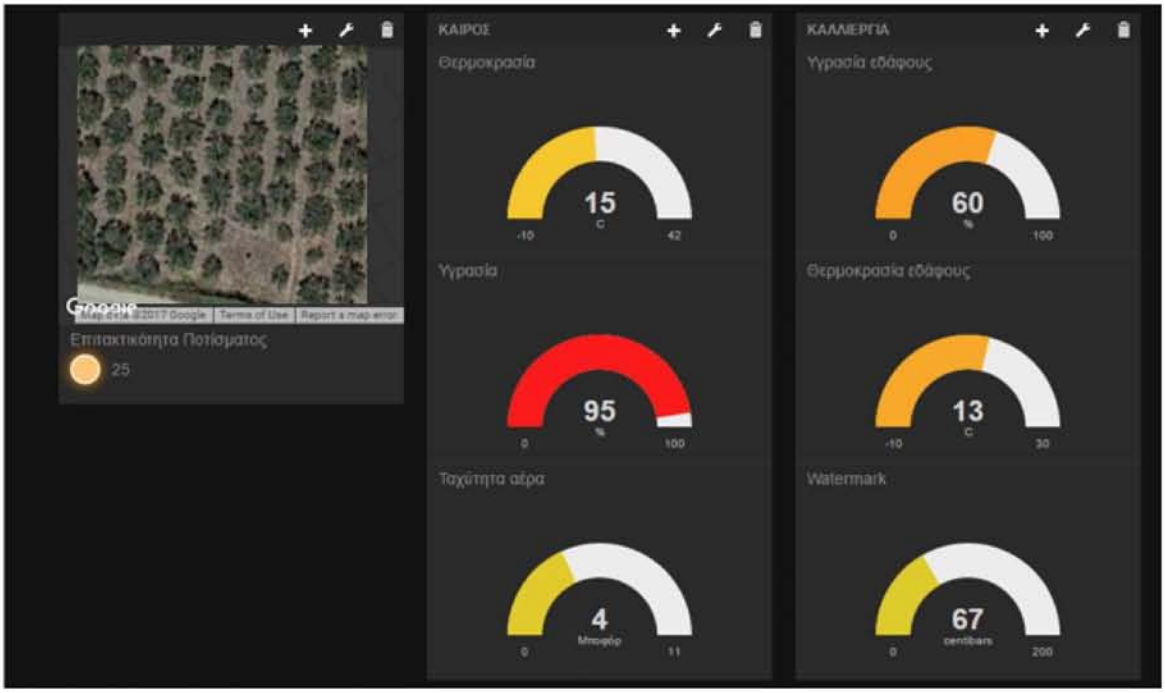


Wind



Except from the weather forecast, farmer wants to see the data measurements from his field. For this reason a remote portal has developed for monitoring all system data and decisions. After the data collection from GW and the process through the backend service is time to display the results to the user. This knowledge is important for irrigation scheduling recommendations and actions.

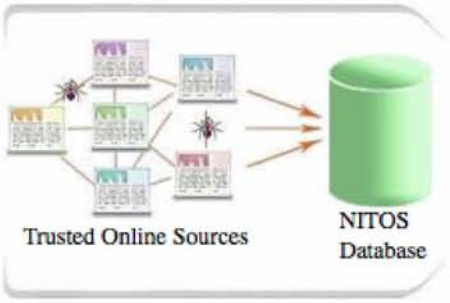
All data measurements from the WSN are available for monitoring through this view.



Part of this master thesis is not only collecting and displaying data from the wireless sensor network but also providing farmers with warnings, updates and information to avoid possible contamination or pest in its cultivation as well as saving resources and money through useful advice.A lot of official and unofficial, but very accurate, information is scattered across the Internet, thus left unexploited.

- Ministry of Rural Development & Food reports.
- Airport weather stations.
- Agriculture and weather forecast blogs.

Through web crawling and pattern matching the backend service with the help of an agronomist collects useful information which is incorporated in the data analysis process, resulting in more accurate and localized suggestions and warnings for the end user.

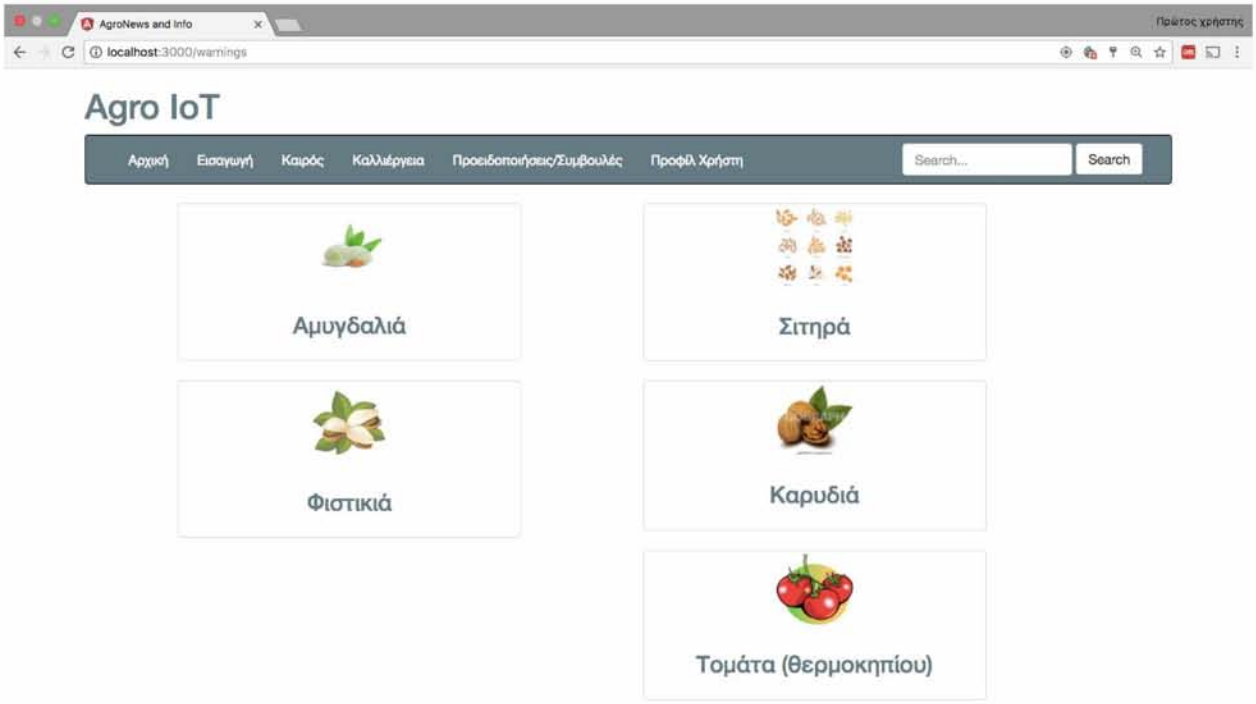


In this management and visualization platform the user can be up to date with the latest news and early warnings related to his crops. There is a tab where user can see all his crops ,with a simple http get request in http://194.177.207.12:3000/warnings/categories and a hash table with a categoryId as key and a cultivation as value is returned. All cultivations belonging to a user has a unique identifier id,called categoryId, an http get request with categoryId as query is used to retrieve warnings from the service.An example of this get request is:

http.get('http://194.177.207.12:3000/warnings/categories?categoryId=5918220a5c948d14d65c8824') and the code for these requests is below.

```
  ngOnInit(): void {
    console.log('OnInit');
    this.getCategories();
  }

  imageWidth: number = 80;
  showImage: boolean = true;
  public data : Object;

  toggleImage(): void {
    this.showImage = !this.showImage;
  }

  constructor(
    private _WarningsService: WarningsService,
    private http: Http
  ){

  }

  getCategories(){

    let url = "http://194.177.207.12:3000/warnings/categories";
            console.log(url);
            return this.http.get(url)
               .map((res:Response) => res.json())
               .subscribe(
                    data =>{
                        this.data = data; console.log(data);
                    }
                    // err => console.log(err),
                    // () => console.log(this.weather)
                );

        // this._WarningsService.getCategories().subscribe(
          //  data => { this.data = data }
          //  (err) => console.error(err),
          //  () => console.log(this.data)
        // );
    }
```

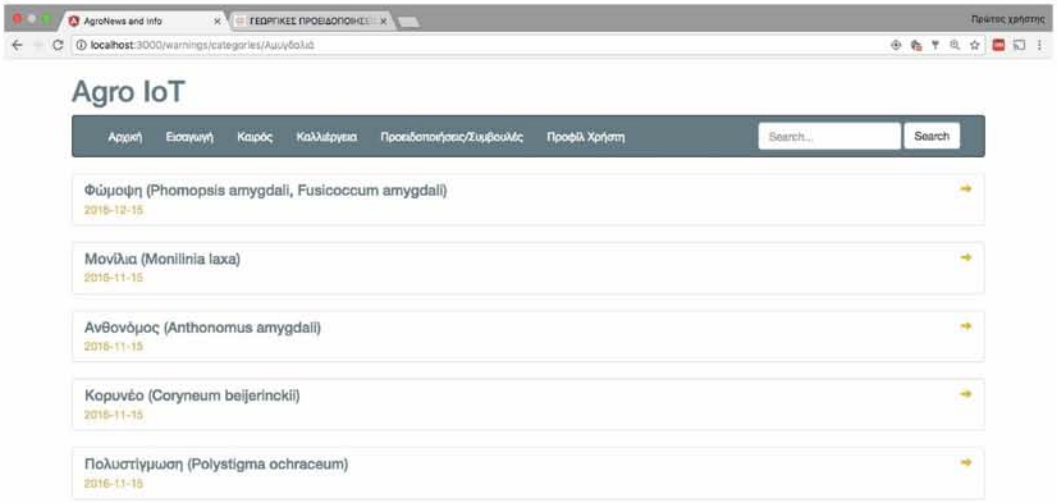
After getCategories() function the view looks like this:



On button click on one of the type cultivations a new get request is called to retrieve the appropriate warnings for the chosen cultivation. The getData() function takes the categoryId from the chosen cultivation and uses it from the next get request.

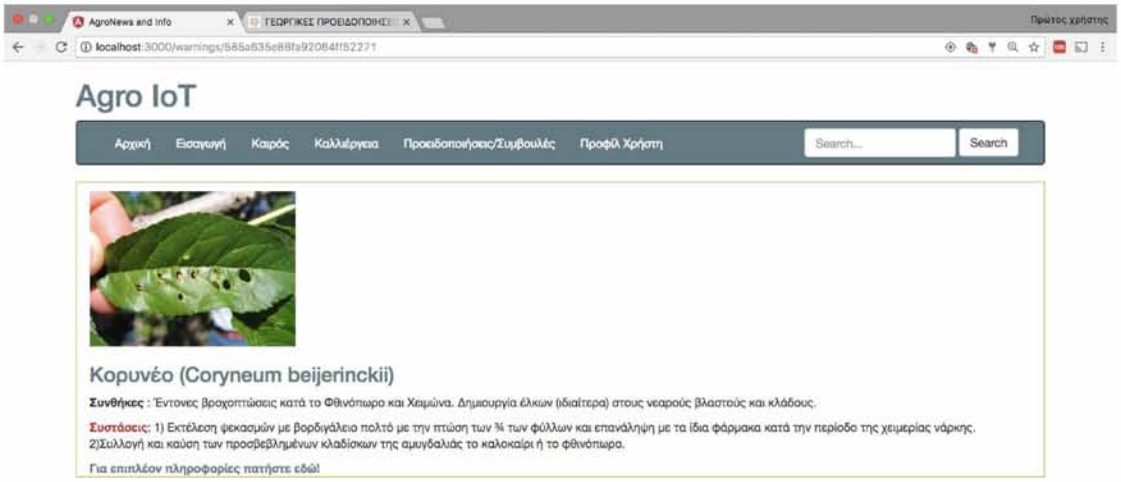
```
@Component ({
    moduleId: module.id,
    selector: 'category-id',
    templateUrl: 'warningId.component.html'
})

export class WarningIdComponent implements OnInit{

    public data : Object;
    imageWidth: number = 100;
    categoryid: string;


    constructor(
        private _WarningsService: WarningsService,
        private _infectionIdService: InfectionIdService,
        private http: Http,
        private route: ActivatedRoute
        ) { }

    ngOnInit() {
        console.log("Hello");
        this.getData();

    }

    getData(){
        this.categoryid = this.route.snapshot.params['id'];
            console.log('The choosen category is:' + this.categoryid);
            let url = "http://10.64.92.26:3000/warnings/categories/" + this.categoryid.toString();
            console.log(url);
           return this.http.get(url)
                .map((res:Response) => res.json())
                .subscribe(data => { this.data = data; console.log(this.data); });

    }


}
```

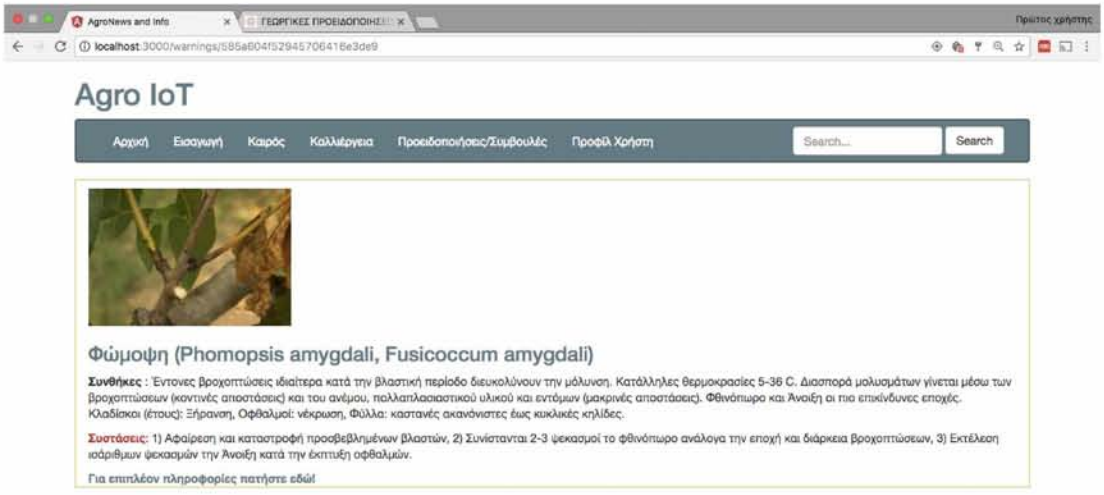
List of possible plant diseases and pests.



On click in one of the list items a small paragraph with conditions,warnings and suggestions appears.

```
getImage(url : string){
    let headers = new Headers({ 'Accept': 'application/json'});
    console.log('The image url in GetImage is: ' + url);
    //let temp = encodeURIComponent(url);
    return this.http.get(url)
        .map((res:Response) => {res.json();});
        // .subscribe(
        //     data => {
        //         //this.img = JSON.parse(JSON.stringify(data || null ));
        //         console.log('getImage function with response');
        //     }
        //);

}

  getCategoryData() {
        this.categoryId = this.route.snapshot.params['id'];
            console.log('The choosen list item has id:' + this.categoryId);
            let url = "http://194.177.207.12:3000/warnings/" + this.categoryId;
            console.log(url);
            return this.http.get(url)
               .map((res:Response) => res.json())
               .subscribe(
                    data =>{
                        this.weather = data; console.log(data); this.icon = this.weather[0].photo;
                        this.imgUrl = "http://194.177.207.12:3000" + this.icon;
                        this.pdfUrl = "http://www.minagric.gr" + data.link;
                        console.log('The image url in GetCategoryData is: ' + this.imgUrl);
                        this.getImage(this.imgUrl);
                    }
                    // err => console.log(err),
                    // () => console.log(this.weather)
                );
    }


}
```





In the description there is a link for redirection to a pdf file with more details.

ΓΕΩΡΓΙΚΕΣ ΠΡΟΕΙΔΟΠΟΙΗΣΕΙΣ

ΠΕΡΙΟΧΗ ΚΕΝΤΡΙΚΗΣ ΕΛΛΑΔΟΣ ● ΤΕΧΝΙΚΟ ΔΕΛΤΙΟ

ΥΠΟΥΡΓΕΙΟ ΑΓΡΟΤΙΚΗΣ ΑΝΑΠΤΥΞΗΣ ΚΑΙ ΤΡΟΦΙΜΩΝ
Δ/ΝΣΗ ΑΠΟΚ. ΥΠΗΡΕΣΙΩΝ ΘΕΣΣΑΛΙΑΣ & ΣΤΕΡΕΑΣ ΕΛΛΑΔΟΣ
ΤΜΗΜΑ:ΠΕΡ/ΚΟ ΚΕΝΤΡΟ ΠΡΟΣΤΑΣΙΑΣ ΦΥΤΩΝ ΚΑΙ ΠΟΙΟΤΙΚΟΥ ΕΛΕΓΧΟΥ ΒΟΛΟΥ
Ταχ.Δ/νση: Τορούζια-Νικολαΐδη, 38334 Βόλος
Τηλ.: 2421066525 - Fax: 2421069545

Η αλόγιστη χρήση φυτοφαρμάκων :
- βλάπτει την υγεία των καταναλωτών
- επιβαρύνει το περιβάλλον
- αυξάνει το κόστος παραγωγής

Πληροφορίες : Φυτοπούλου Αννέτα

ΑΜΥΓΔΑΛΙΑ Ν°9 16/11/2016

ΑΜΥΓΔΑΛΙΑ

ΑΝΘΟΝΟΜΟΣ *(Anthonomus amygdali)*

Ο ανθονόμος της αμυγδαλιάς έχει μήκος 3-4mm και χρώμα σκούρο καστανό. Έχει μία γενιά το έτος είναι δραστήριο το χειμώνα και την άνοιξη. Το καλοκαίρι τα ενήλικα άτομα βρίσκονται σε προστατευμένες θέσεις πάνω ή κοντά στα δένδρα (κάτω από ξερούς φλοιούς, σε ρωγμές του φλοιού, μέσα στο έδαφος και αλλού).
Τα ενήλικα δραστηριοποιούνται από τις αρχές Νοεμβρίου. Αφού τραφούν για δύο εβδομάδες σε βάρος των οφθαλμών, συζεύγνυνται και αρχίζουν να ωοτοκούν. Η ωοτοκία γίνεται από τον Δεκέμβριο έως τέλος Φεβρουαρίου, αναλόγως της περιοχής. Το θηλυκό εναποθέτει σε κάθε οφθαλμό ένα αυγό.
Οι προσβεβλημένοι οφθαλμοί ξεραίνονται χωρίς να ανοίξουν και τελικά πέφτουν στο έδαφος.
ΣΥΣΤΑΣΕΙΣ: Σε περιοχές όπου υπάρχει ένδειξη ότι το φθινόπωρο ο ενήλικος πληθυσμός του εντόμου πάνω στα δένδρα είναι πυκνός συνίσταται ένας ψεκασμός των δένδρων τον Νοέμβριο πριν πέσουν τα φύλλα και επανάληψη του ψεκασμού μετά την πτώση των ¾ των φύλλων μέχρι και τον Ιανουάριο μήνι, με εγκεκριμένα φυτοπροστατευτικά σκευάσματα.

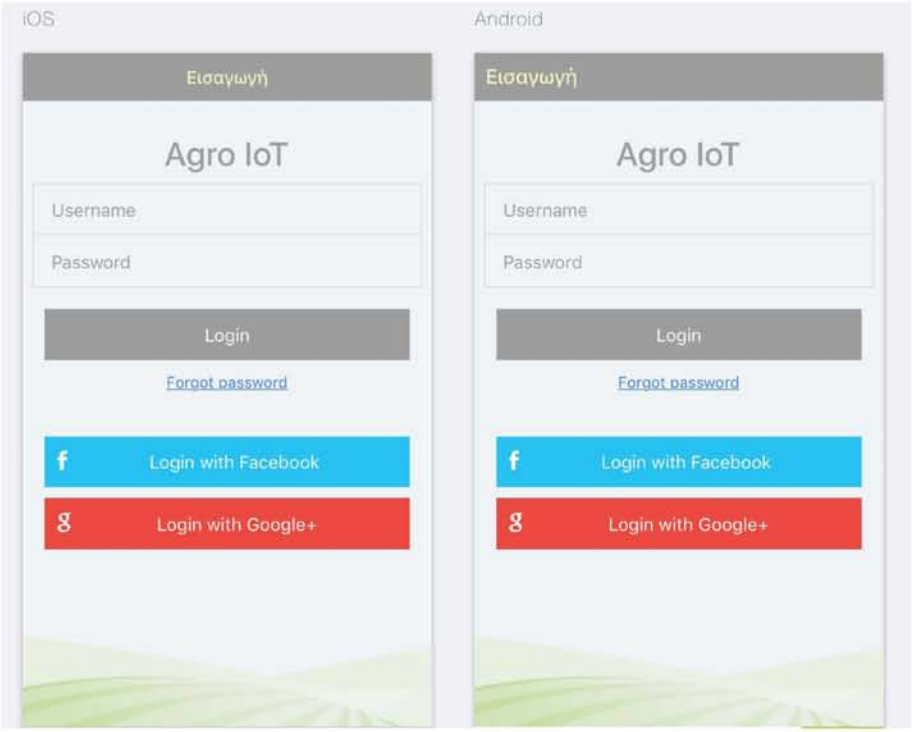
8.2. Mobile application

The same implemented backend service with small changes in previous Rest API has been used for the communication with the mobile application. The frontend application developed in my postgraduate work supports 2 types of platforms, ANDROID and iOS, so it's a hybrid application. A hybrid application (hybrid app) [23] is one that combines elements of both native and Web applications. Native applications are developed for a specific platform and installed on a computing device. Web applications are generalized for multiple platforms and not installed

locally but made available over the Internet through a browser. Hybrid apps are often mentioned in the context of mobile computing.

The framework that has been used in this deployment is called Ionic [24] and is a complete open-source SDK for hybrid mobile app development. The original version was released in 2013 and built on top of AngularJS and Apache Cordova. The more recent releases, known as Ionic 2 or simply "Ionic", are built on Angular. Ionic provides tools and services for developing hybrid mobile apps using Web technologies like CSS, HTML5, and Sass.

The experience that the user acquires in this application is the same as the previous one. Below there will be some snapshots with corresponding descriptions where is needed.



The login process needs username and password for register,authentication and then login. On Login button press 3 steps follows:

1. Register http post to "http://194.177.207.12:3000/auth/register" with json body the user's credentials, {username: {username},password: {password}}.
2. Login http post to "http://194.177.207.12:3000/api/login" with the same json body.
3. After the last server's response the app stores the authentication token to the local storage.

The json response after success registration and login is:

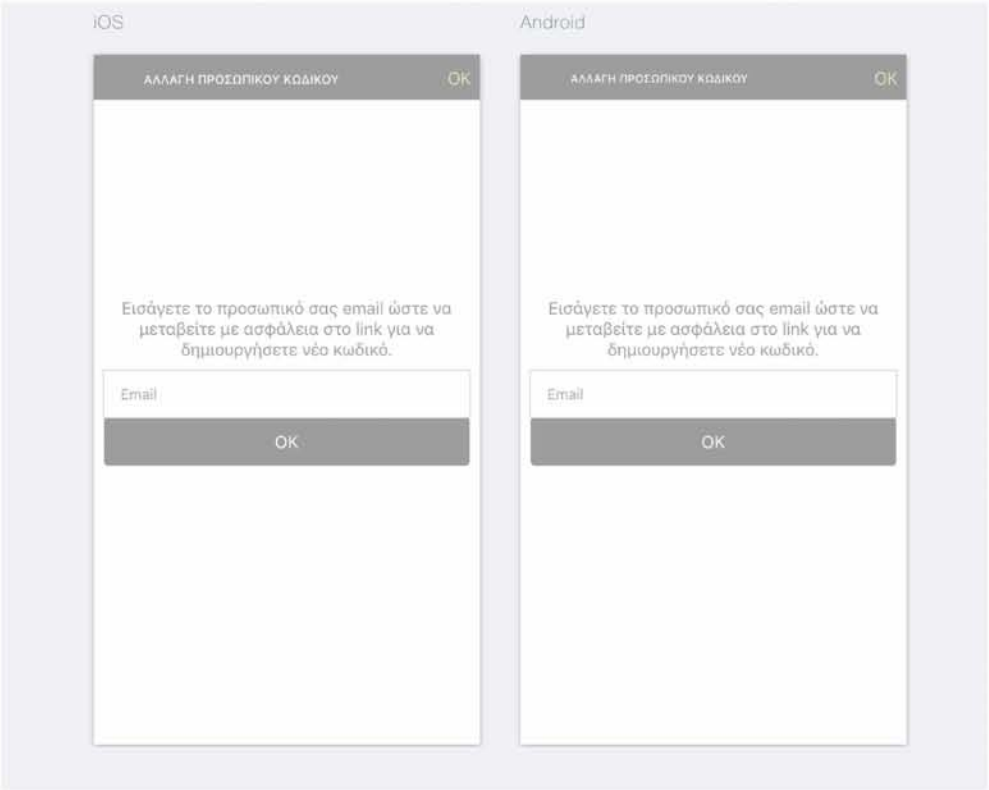
{message:'User successfully created!', token: 'JWT eyJhbGciOiJIUzI1NiIsInR ...' user: { username: 'ellie', password: '123'}}

```
// factory for authentication register with user's info
.factory('RegisterService', function($http,server) {
    return {
      register: function(user) {

        var url = server + '/auth/register'

        // post structure
        var register_user = {
            method: 'POST',
            url: url,
            headers: {
                      'Content-Type': 'application/json'
                     },
           data: user
        }
        console.log('Register Service');
        console.log(register_user);
        return $http(register_user);
      }
    }
})
```

```
225
226  // factory for login user
227  .factory('loginService', function($http,server) {
228      return {
229        login: function(credentials,token) {
230
231          var url = server + '/api/login'
232
233          // post structure
234          var req_login = {
235              method: 'POST',
236              url: url,
237              headers: {
238                        'Content-Type': 'application/json'
239                        },
240              data: JSON.stringify(credentials)
241          }
242
243          console.log(req_login);
244          return $http(req_login);
245        }
246      }
247  })
```

Login response handling code

```
2436    $scope.loginFunc = function(){
2437         //register for push notifications
2438          $ionicPush.register().then(function(t) {
2439              return $ionicPush.saveToken(t);
2440              }).then(function(t) {
2441              console.log('Token saved:', t.token);
2442              // alert(t.token);
2443              $scope.credentials.pushNotificationToken = t.token;
2444         //             });
2445         // });
2446              var valid = $scope.validate();
2447              if (valid) {
2448                  var login = loginService.login($scope.credentials, $scope.token);
2449                  $scope.text_error = ""
2450                  // console.log($scope.credentials);
2451                  login.then(
2452                  function(payload) {
2453                      //success
2454                      $ionicHistory.nextViewOptions({
2455                        disableBack : true,
2456                        historyRoot: true
2457                      });
2458                      $scope.hide($ionicLoading);
2459                      $scope.response = payload.data;
2460                     console.log('Response after post json data:' , $scope.response);
2461                     $window.localStorage['token'] = angular.toJson($scope.response.token);
2462                     $window.localStorage['role'] = angular.toJson($scope.response.user.role);
2463                      // var alertPopup = $ionicPopup.alert({
2464                      //     title: 'Η σύνδεση ολοκληρώθηκε επιτυχώς!',
2465                      //     template: 'Πιέστε OK για να πλοηγηθείτε στην εφαρμογή.',
2466                      //     cssClass: 'codeAlert'
2467                      // });
2468                      $state.go('home');
2469                  },
2470                  function(errorPayload) {
2471                      //fail
2472                      //message in greek
2473                      $scope.hide($ionicLoading);
2474                      // console.log(errorPayload);
2475
2476                      if ((errorPayload.data != undefined ) || (errorPayload.data != null )) {
2477                          console.log(errorPayload.data);
2478                          var error = errorPayload.data.error;
2479                      } else {
2480                          if (errorPayload.status == 400) {
2481                              $scope.text_error = "Ωχ , κάτι πήγε στραβά!"
2482                          }else if (errorPayload.status == 401){
2483                              $scope.text_error = "Τα στοιχεία που εισάγατε δεν είναι σωστά!"
2484                          }
2485                      }
2486
2487                      var alertPopup = $ionicPopup.confirm({
2488                          title: "Προσοχή!",
2489                          template: 'Τα στοιχεία που εισάγατε δεν είναι σωστά! Πιέστε OK για προσπάθεια ξανά.'
2490                      });
2491
2492                  });  // errorPayload
2493              }
2494          });
2495      }
```



In case user forgets his password there is a get http request in '/auth/reset-password/:token' where "token" is the authentication token as a parameter to the request.

After login user needs to insert his areas with cultivations. For the selection there are popup menus. Each item of the below view (Region, Module, Municipality, Unit, Community) displays a menu when pressed. The user, starting from top to bottom, selects the circumference first from the options menu that appears to him, then the Regional Section, and so on, until he reaches Municipal Community, thus completing the process. All five divisions (Region, Module,

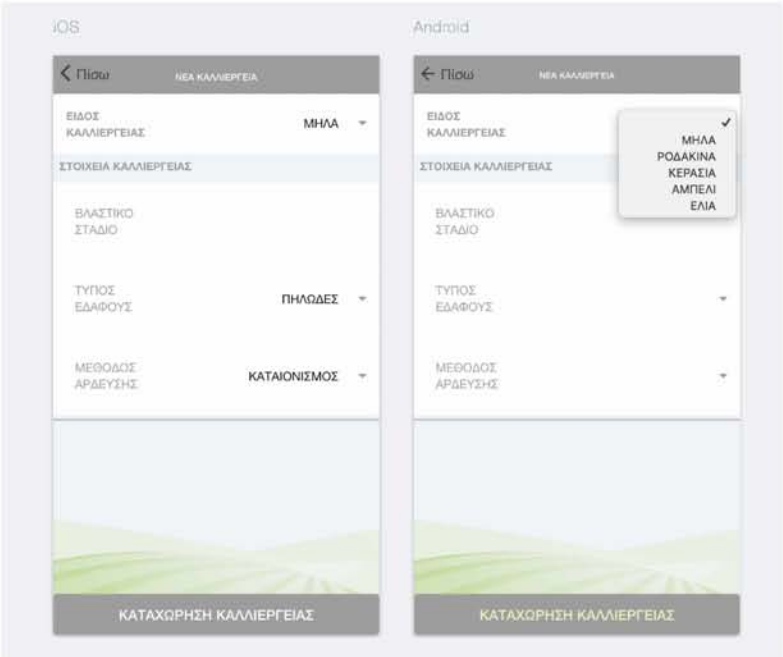
Municipality, Unit, Community) are identified with a unique divisionId. Every time a menu is displayed after press, a get http request is executing to fill up the menu. For this request the route is "/areas/:division" where division takes one of these values: regions, regional_units, municipalities, municapal_units, municipal_communities according to the selection. For a success request one more parameter is required as a query to the call, the divisionId. The service for the get request is GetAreaService.

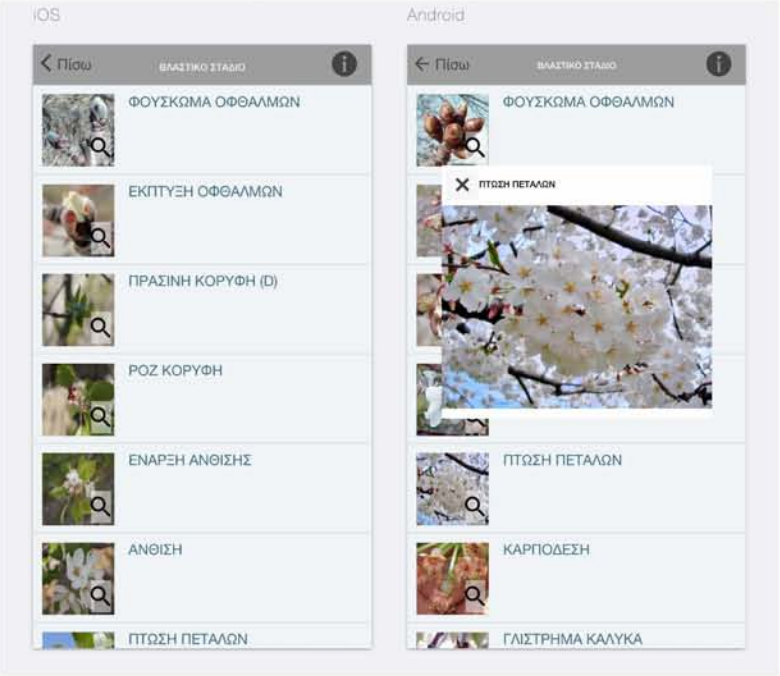
```
80  // factory for get regions and areas
81  .factory('GetAreaService', function($http,server) {
82      return {
83        getArea: function(areaId,division) {
84
85          var url = server + '/areas/' + division
86
87          // post structure
88          var req_get_reg_unit = {
89              method: 'GET',
90              url: url,
91              params: {id:areaId}
92          }
93
94          console.log(req_get_reg_unit);
95          return $http(req_get_reg_unit);
96        }
97      }
98  })
```

Below are the functions which fill up the drop down menus.

```
3360
3361    $scope.updateRegion = function(option) {
3362        // $rootScope.region = option;
3363        $window.localStorage['region'] = angular.toJson(option);
3364        console.log(option);
3365            var areaId = option;
3366            var division = 'regional_units';
3367            var req_get_reg_unit = GetAreaService.getArea(areaId,division);
3368            req_get_reg_unit.then(
3369            function(payload) {
3370                //success
3371                $scope.regional_units = payload.data;
3372                console.log('Response after select perifereia:' , $scope.regional_units);
3373            },
3374            function(errorPayload) {
3375                //fail
3376                //message in greek
3377                console.log(errorPayload.data);
3378                $ionicPopup.alert({
3379                    title: 'Κάτι πήγε στραβά',
3380                    template: 'Πιέστε OK για προσπάθεια ξανά!',
3381                    cssClass: 'codeAlert'
3382                })
3383                });  // errorPayload
3384    }
3385
3386    $scope.updateRegUnit = function(option) {
3387        // $rootScope.regional_unit = option;
3388        $window.localStorage['regional_unit'] = angular.toJson(option);
3389        var areaId = option;
3390        var division = 'municipalities';
3391        var req_get_reg_unit = GetAreaService.getArea(areaId,division);
3392        req_get_reg_unit.then(
3393        function(payload) {
3394            //success
3395            $scope.municipalities = payload.data;
3396            console.log('Response after select perifereiaki enothta:' , $scope.municipalitie
3397        },
3398        function(errorPayload) {
3399            //fail
3400            //message in greek
3401            console.log(errorPayload.data);
3402            $ionicPopup.alert({
3403                title: 'Κάτι πήγε στραβά',
3404                template: 'Πιέστε OK για προσπάθεια ξανά!',
3405                cssClass: 'codeAlert'
3406            })
3407            });  // errorPayload
3408    }
3409
3410    $scope.updateMun = function(option) {
3411        // $rootScope.municipality = option;
3412        $window.localStorage['municipality'] = angular.toJson(option);
3413        var areaId = option;
3414        var division = 'municipal_units';
3415        var req_get_reg_unit = GetAreaService.getArea(areaId,division);
3416        req_get_reg_unit.then(
3417        function(payload) {
3418            //success
3419             angular.forEach( payload.data, function(value, key) {
3420                  console.log(value.name.capital);
3421                  value.name.capital = (value.name.capital.split('ΕΝΟΤΗΤΑ')[1]);
3422                  // $scope.municipal_communities
3423            });
3424            $scope.municapal_units = payload.data;
3425            console.log('Response after select dhmo:' , $scope.municapal_units);
3426        },
3427        function(errorPayload) {
3428            //fail
3429            //message in greek
3430            console.log(errorPayload.data);
3431            $ionicPopup.alert({
3432                title: 'Κάτι πήγε στραβά',
3433                template: 'Πιέστε OK για προσπάθεια ξανά!',
3434                cssClass: 'codeAlert'
3435            })
3436            });  // errorPayload
3437    }
```

When an area has been selected the next step is to insert area's cultivations. A cultivation is defined from the type of cultivar, the growth stage, the type of soil and the irrigation system.





View of available growth stages for the selected cultivation with zoom in option.

If the user has finished with his cultivations the app undertakes to save data as a cultivation object with saveCultivar() function and send them with postCultivars() function through http post request and use cultivation data as json body.

The cultivation object has this format:

{

Cultivar: “ΜΗΛΑ”,

Area: [“12”, “10”, “1002”, “100202”, “10020202”],

Stage: “ΠΡΑΣΙΝΗ ΚΟΡΥΦΗ”,

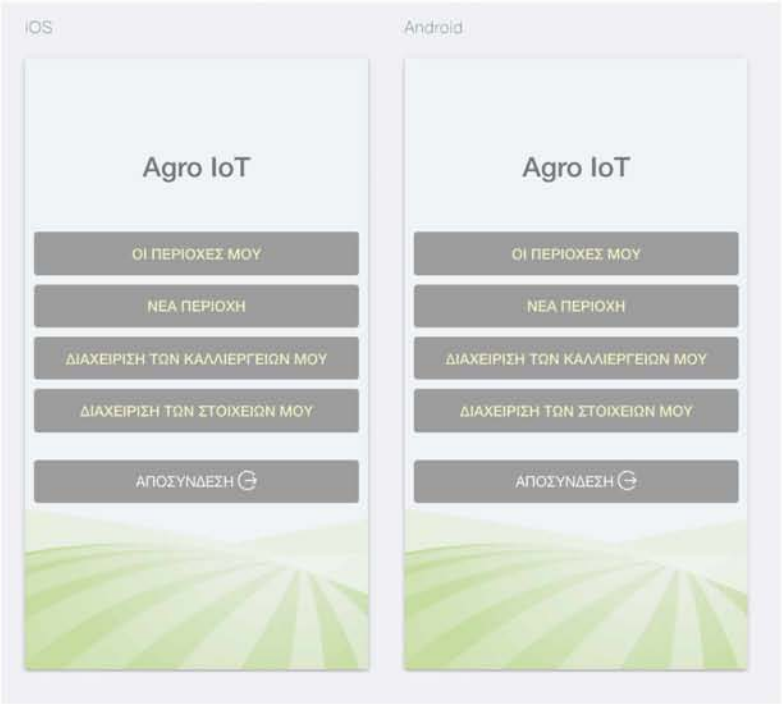
Irrigation system: “ΚΑΤΑΙΟΝΙΣΜΟΣ”,

Soil: “ΑΜΜΩΔΕΣ”

}

```
    $scope.saveCultivar = function() {
          console.log('STOIXEIA NEAS KALLIERGEIAS');
          console.log($scope.cultivar );
          console.log($scope.kairos);
          console.log($scope.futoprostasia);
          console.log($scope.nero);
          console.log($scope.stadio);
          console.log($scope.edafos);
          console.log($scope.ardeusi);
          console.log($scope.servicekairou + ' ' +  $scope.servicefutoprostasia +  ' ' + $scope.servicenerou)

          if ($scope.validation()) {
              console.log('CULTIVATION IS VALID');
              $scope.saveCultivationJson();
              console.log('FIRST CULTIVAR ADDED!')
              console.log($rootScope.jsonData);
              $window.localStorage['cultivationJSON'] = angular.toJson($rootScope.jsonData);
               var alertPopup = $ionicPopup.alert({
                      title: 'Η καλλιέργεια προστέθηκε επιτυχώς!',
                      template: 'Πατήστε OK για να συνεχίσετε να προσθέτετε καλλιέργειες.',
                      cssClass: 'codeAlert'
              });
              alertPopup.then(function(res) {
              if (res) {
                $rootScope.stadio = '';
                $scope.stadio = '';
                $rootScope.$ionicGoBack();
                // $rootScope.kalliergia = '';
                // $rootScope.edafos = '';
                // $rootScope.ardeusi = '';
                // $rootScope.futoprostasia = '';
                // $rootScope.kairos = '';
              }
          });
          }

    }
}])
```

```
$scope.postCultivars = function(){

    $scope.token = angular.fromJson($window.localStorage['token']);
    console.log($scope.token);
    if(($scope.token != undefined) && ($scope.token != null) && ($scope.token != '')) {
    $scope.show($ionicLoading);
    var post_cultivations = PostCultivationsService.postCultivars($scope.data, $scope.token);
        post_cultivations.then(

        function(payload) {
            //success
            $scope.hide($ionicLoading);
            $scope.response = payload.data;
            // console.log('Response after post json data:' , $scope.response.message);
            // console.log('clear localstorage and rootscope variables');
            $window.localStorage.clear();
            $rootScope.region = '';
            $rootScope.regional_unit = '';
            $rootScope.municipality = '';
            $rootScope.municipal_unit = '';
            $rootScope.minicipal_community = 'ΠΕΡΙΟΧΗ';
            $rootScope.regionArray = [];
            $rootScope.region_community = '';
            $rootScope.region_community_id = '';

            var alertPopup = $ionicPopup.alert({
                title: 'Ολοκληρώθηκε το πρώτο βήμα της εγγραφής.',
                template: 'Πατήστε OK για να συνεχίσετε με την πληρωμή.',
                cssClass: 'codeAlert'
            });
            alertPopup.then(function(res) {
                // $window.location.reload(true);
              $window.localStorage['token'] = angular.toJson($scope.token);
              $state.go('page13');
            });
        },
        function(errorPayload) {
            $scope.hide($ionicLoading);
            // console.log(errorPayload);
            if (errorPayload.status == 422) {
                $scope.text_error = errorPayload.statusText;
                var alertPopup = $ionicPopup.alert({
                title: 'Κάτι πήγε στραβά ',
                template: "Κάποιο στοιχείο δεν έχει επιλεγεί σωστά!",
                cssClass: 'codeAlert'
            });
            }else if (errorPayload.status == -1) {
             var alertPopup = $ionicPopup.alert({
                title: 'Κάτι πήγε στραβά ',
                // template: 'Ο χρήστης υπάρχει ήδη.',
                cssClass: 'codeAlert'
            });
        }else if (errorPayload.status == 400) {
            $scope.text_error = errorPayload.data.errors;
             var alertPopup = $ionicPopup.alert({
                title: 'Κάτι πήγε στραβά ',
                template:  $scope.text_error,
                cssClass: 'codeAlert'
            });
         }
        });  // errorPayload
        }
}
```

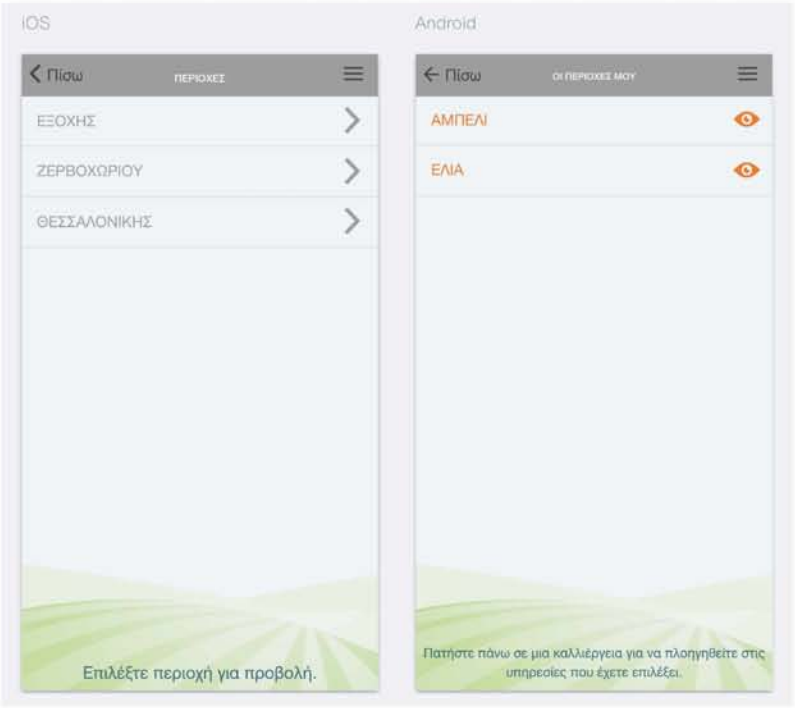


After success registration the user can see the home screen with these options:

1. Receive warnings,suggestions / monitor data collection / check forecast
2. Insert new region and new cultivation
3. Manage his cultivations
4. Manage his profile account
5. Logout

In the following paragraphs we will describe the most complicated and demanding of the five options,the first one. The others just reuse views and functions that we analysed in previous chapters.

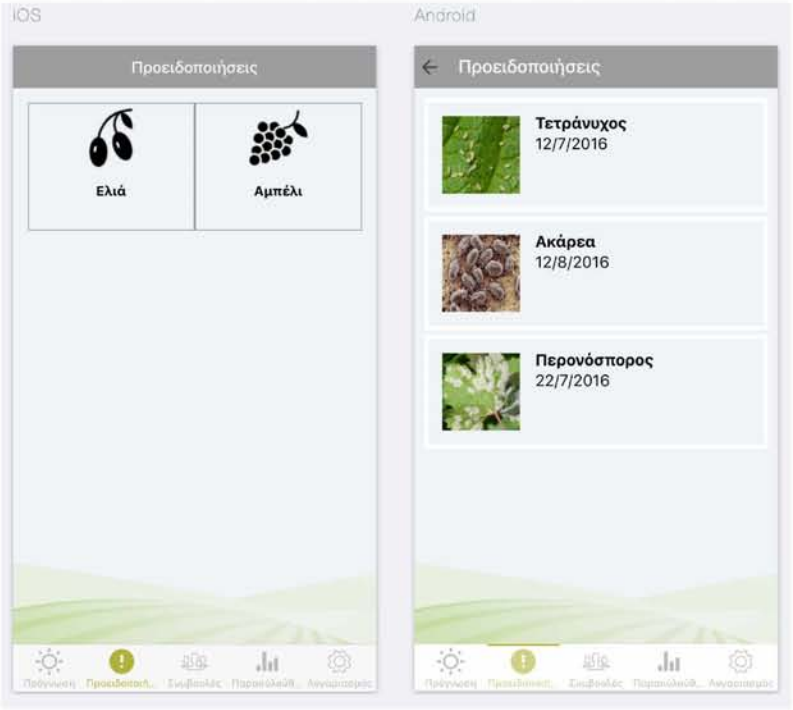
8.2.1. Warnings

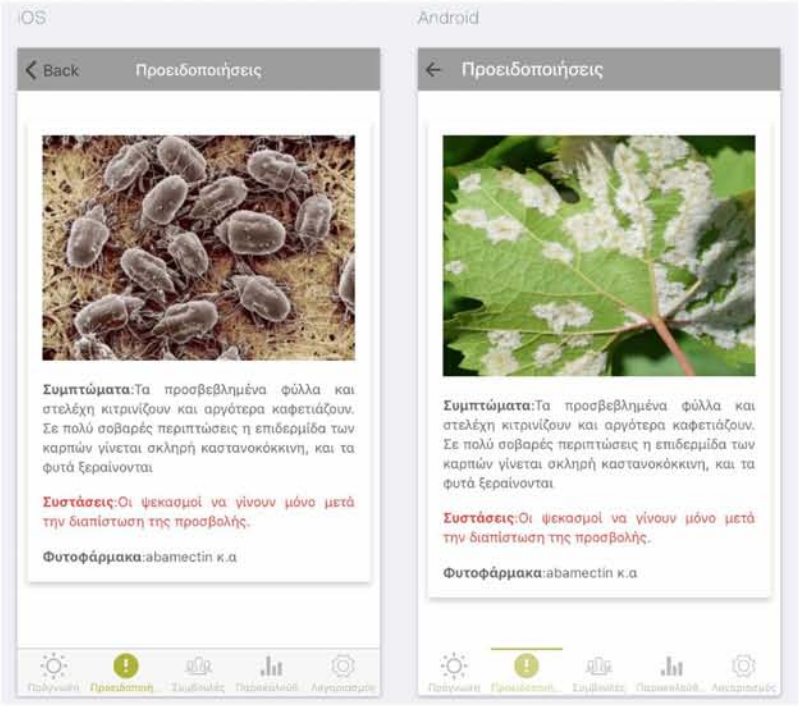


In the view above there is a list with all user's areas, an http get request in "http://194.177.207.12:3000/warnings/areas" is executed and a response with the list is returned. Clicking in one of the areas, a list with cultivations, belong to this area, appears. The corresponding get request for this view is "http://194.177.207.12:3000/warnings/areas/:areaId" . To reveal warnings, user redirects after the click of one item, from the previous view, to the view which is depicted below, with a tab menu on the bottom. In this screen the selected tab is "warnings". In order to display the warnings for a specific cultivation of one area another http get request is used in "http://194.177.207.12:3000/warnings/areas/:areaId/:cultivationId".

Πανεπιστήμιο Θεσσαλιάς

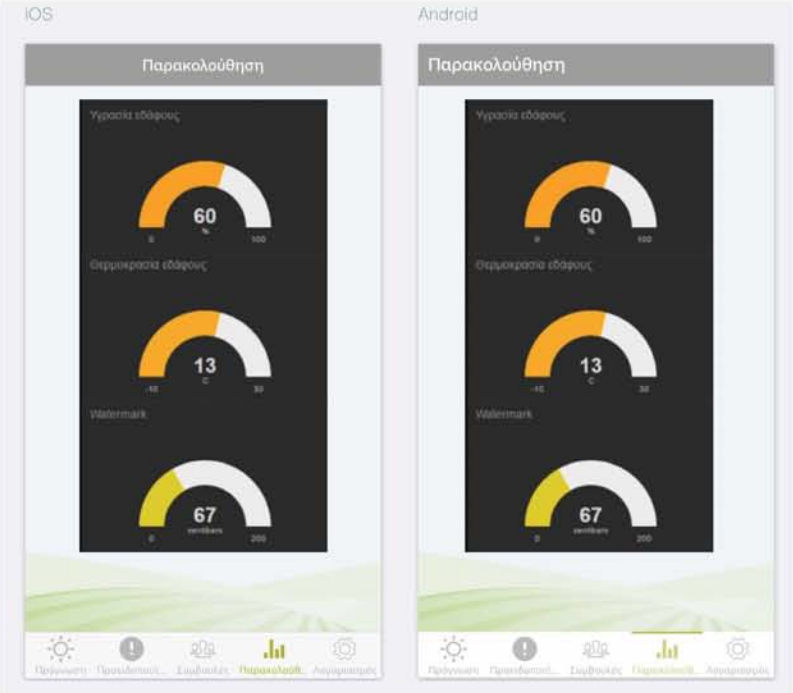
Often the farmer's hard work are destroyed by pests and they suffer huge monetary losses. To prevent such situation this app, through web crawling and pattern matching, can collect useful information which is incorporated in the data analysis process, resulting in more accurate and localized suggestions and warnings for the farmer, which is capable of predicting pest behavior. This information can be used by farmers to reduce damage done by pests on a large scale.





8.2.2. Monitoring

If user selects the monitoring tab,a view with data measurement indications reveals.

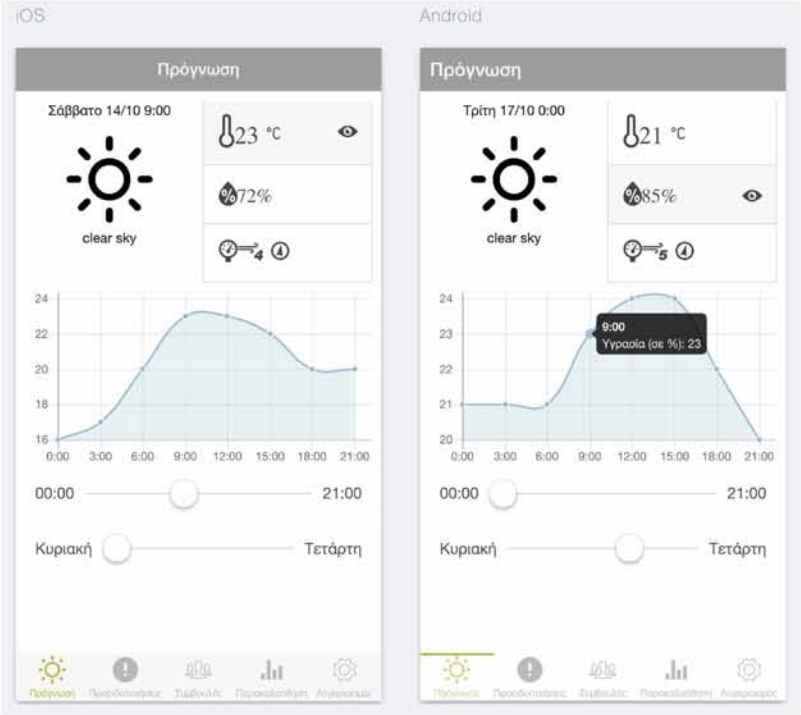


All sensory data associated with the soil quality are presented in this tab with a curved bar. This function ensures accurate and timely communication of real time data or information related to dynamic agricultural processes. Farmer's with availability of such important real time information in advance are able to plan their course of activities pre-hand and take corrective/preventive measures for future contingency.

All these data are returned in a json format, from the backend service through an http get request in the "http://194.177.207.12:3000/monitoring/areas/:areaId/:cultivationId" .

8.2.3. Forecast

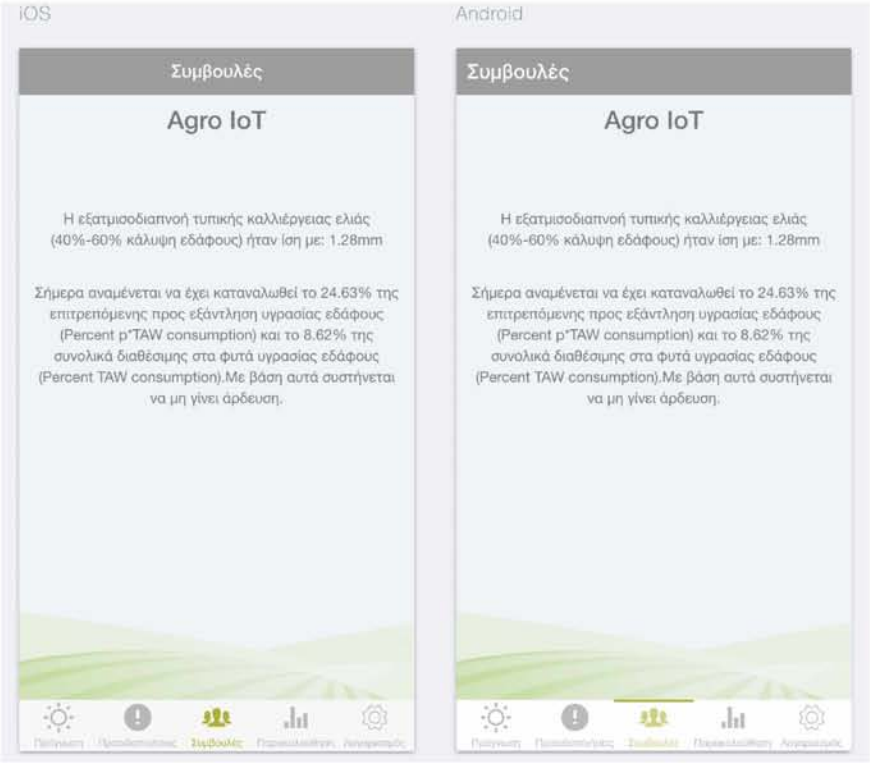
A five day forecast for temperature, humidity and wind appears in the “forecast” tab. The api that has been used is the same as the forecast in web application.



8.2.4. Suggestions

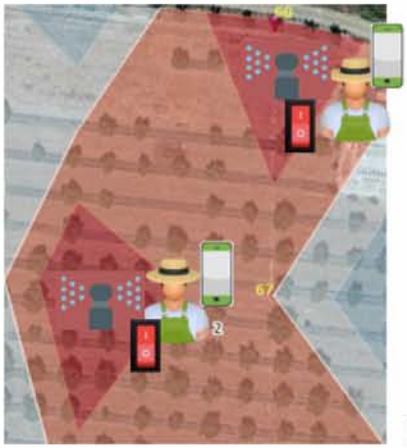
All users are given information on the reference evapotranspiration, as well as indicative evapotranspiration values for basic crops, yesterday, and an available related record. Based on a water balance for the collected data measurements from WSN (chapter 7,paragraph 7.1), the system gives a recommendation on when to make the next irrigation and how long it should last to restore the soil to its water-resistance. In any case, the accuracy of the calculations is subject to

possible errors due to the microclimate and the specific characteristics of each crop (density, physical state, etc.) and the soil in which it develops.



A simple http get request in "http://194.177.207.12:3000/suggestions/areas/:areaId/ :cultivationId" with the id of selected cultivation as parameter , from the view of cultivations , is executed and the app handles the response which is a list with suggestions and information that derives from irrigation processing.

The farmer takes into account the system's recommendations and conducts irrigation manually by switching on the valves by himself. Afterwards he can observe the new data on the Web or on the mobile monitoring interface.



Manual irrigation of dry areas

9. Agriculture IoT Future (2050)

It is forecasted that the Agriculture Internet of Things in 2050 will increase the food production by 70% and would be feeding to 9.6 billion of people globally by the year 2050.

Therese Cory, Beecham Research analyst, & Towards Smart Farming, author of a new report, stated: "The demand for more food has to be set against the challenges of rising climate change

and more extreme weather conditions, along with the environmental impact resulting from intensive farming practices."

References

[1] Wireless Sensor Networks: https://www.cs.virginia.edu/~stankovic/psfiles/wsn.pdf

[2] Institute of Electrical and Electronics Engineers : http://www.ieee.org/index.html

[3] IEEE 802.15.4: http://en.wikipedia.org/wiki/IEEE_802.15.4

[4] ZigBee protocol : http://www.zigbee.org/

[5] Wireless HART : http://www.hartcomm2.org/index.html

[6] ISA-SP100 : http://www.isa.org/MSTemplate.cfm?MicrositeID=1134&CommitteeID=6891

[7] IETF IPv6 LoWPAN : http://ietfreport.isoc.org/ids-wg-6lowpan.html

[8] MiWi:

http://www.microchip.com/pagehandler/enus/technology/personalareanetworks/technology/home.html

[9] Digi Mesh : http://www.digi.com/technology/digimesh/

[10] Arduino : http://www.arduino.cc/

[11] NITOS Wireless Sensor Mote:

http://nitlab.inf.uth.gr/NITlab/index.php/hardware/sensors/nitos-wireless-sensorplatform

[12] Atmega 32U4 Microcontroller :

http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Dev/Arduino/Boards/ATMega32U4.pdf

[13] Xbee s8 : https://researchdesignlab.com/wireless/xbee/xbee-s8.html

[14] Zigbee Device Objects:

http://ftp1.digi.com/support/images/APP_NOTE_XBee_ZigBee_Device_Profile.pdf

[15] Watermark sensor: https://farmsolutions.com/works-with/sensors/

[16] Watermark guideline for centibars : http://www.irrometer.com/basics.html

[17] Dallas, DS18B20:

https://www.tautvidas.com/blog/2015/09/measuring-temperature-with-ds18b20-temperature-sensor/

[18] SHT21 Sensor: http://www.farnell.com/datasheets/1780639.pdf

[19] Wind Monitor : http://www.youngusa.com/Brochures/05103(0106).pdf

[20] Precision Agriculture : https://en.wikipedia.org/wiki/Precision_agriculture

[21] Evapotranspiration: https://en.wikipedia.org/wiki/Evapotranspiration

[22] Open Weather Api: https://openweathermap.org/api

[23] Hybrid App :

http://searchsoftwarequality.techtarget.com/definition/hybrid-application-hybrid-app

[24] Ionic Framework: https://en.wikipedia.org/wiki/Ionic_(mobile_app_framework)

[25] IoT: https://en.wikipedia.org/wiki/Internet_of_things