

**An-Najah National University**

**Faculty of Graduate Studies**

# **A Cloud Application for Smart Agricultural Irrigation Management System**

**By**

**Mustafa Mohammad Younes**

**Supervisor**

**Dr. Adnan Salman**

**This Thesis is Submitted in Partial Fulfillment of the Requirements for  
the Degree of Master of Advanced Computing, Faculty of Graduate  
Studies, An-Najah National University - Nablus, Palestine.**

**2018**

# **A Cloud Application for Smart Agricultural Irrigation Management system**

**By**

**Mustafa Mohammad Younes**

**This thesis was defended successfully on 27/11/2018, and approved by:**

**Defense Committee Members**

**Signature**

- <b>Dr. Adnan Salman /Supervisor</b>	.....
- <b>Dr. Sobhi Samhan /External Examiner</b>	.....
- <b>Dr. Baker Abdelhaq / Internal Examiner</b>	.....

### III

## **Dedication**

*To my mother a strong and gentle soul who taught me to trust in Allah....*

*To the spirit of my dear father in heaven .....*

*To my dear wife who bore me the burdens of studying and her distress.....*

*To my brothers and my sister and their children all respect and love....*

*To my dear children Toleen & Yazan.....*

## **Acknowledgement**

*Great thanks and great to my supervisor Dr. Adnan Salman for his helpful and continual encouragement.*

*Special thanks to faculty members working in Computer science, Dr. Baker Abdel Haq, Dr. Fadi Draidí , Dr. Abdul Razaq Natsheh, and Mathematics department members Dr. Mohammad Najeeb, Dr. Sameer Matar, Dr. Ali Barakat, for their help and guidance.*

*Great thanks Also to my University" Al-Quds Open University", Prof. Younes Amro, and Dr. Basem Shalash for their support and facilities.*

*Finally, Big thanks to Dr. Sobhi Samhan and Dr. hazem Kittaneh for their experience and qualitative addition to my thesis.*

## الإقرار

أنا الموقع أدناه مقدمة الرسالة التي تحمل العنوان

### **A Cloud Application for Smart Agricultural Irrigation Management system**

أقر بأن ما اشتملت عليه هذه الرسالة إنما هو نتاج جهدي الخاص، باستثناء ما تمت الإشارة إليه حيثما ورد، وأن هذه الرسالة ككل أو من جزء منها لم يقدم من قبل لنيل أية درجة أو بحث علمي أو بحثي لدى أية مؤسسة تعليمية أو بحثية أخرى.

### **Declaration**

The work provided in this thesis, unless otherwise referenced, is the researcher's own work, and has not been submitted elsewhere for any other degree or qualification.

**Student's name:**

**اسم الطالب:**

**Signature:**

**التوقيع:**

**Date:**

**التاريخ :**

## Table of Contents

Dedication .....	III
Acknowledgement.....	IV
Declaration .....	V
Table of Contents .....	VI
List of Figures .....	VIII
List of Symbols and Abbreviations.....	X
Abstract .....	XI
Chapter One.....	1
1. Introduction .....	1
1.1. Root Zone Management.....	3
1.2. Gathering data and triggering action in the field .....	5
1.3. Cloud computing.....	6
1.4. Motivation .....	8
1.5. Thesis Objectives .....	9
Chapter Two .....	10
Literature Review .....	10
Chapter Three.....	19
System Architecture .....	19
3.1. Wireless Sensor Network.....	21
3.2. Cloud Computing .....	24
3.3. Web Services.....	25
3.3.1. Restful web services .....	26
3.4. Sensors Location .....	27
3.5. Farm Application .....	29
3.6. Cloud Application .....	29
3.7. Database .....	32
Chapter Four.....	38

## VII

System Implementation.....	38
4.1. Oracle SQL Developer.....	38
4.2. Oracle Database Expression Edition.....	39
4.3. Oracle Web Logic Server.....	40
4.4. IDE JDeveloper.....	40
4.5. Web Service Implementation.....	41
4.5.1. Web Services Methods .....	43
4.5.2. Benefits of web services .....	46
4.6. Zones and WSN .....	47
4.7. Login Page Implementation.....	48
4.8. Database Implementation.....	48
Chapter Five .....	52
System Evaluation.....	52
Chapter Five .....	71
Conclusion.....	71
References .....	72
الملخص .....	ب

## List of Figures

Fig. (1.1): Fielding Capacity and Wilting Point.....	4
Fig (2.1): Using Cloud in Smart Farming .....	15
Fig. (2.2): Cyper-Physical Management Cycle of Smart Farming .....	16
Figure (3.1): Smart Agricultural Irrigation Management System Architecture ..	20
Figure (3.2): Wireless Sensor Network.....	22
Fig.(3.3) : Sensor location.....	28
Fig.(3.4) : Software architecture of the cloud application .....	30
Figure(3.5) : Data model for the smart irrigation system .....	34
Figure (4.1): Oracle Database XE.....	39
Fig. (4.2): Database Tables .....	49
Figure (5.1) : Login Page .....	52
Figure (5.2) : Home Page of the Project .....	52
Figure (5.3) : Sensors Readings .....	53
Figure (5.4) : Cultivation Information .....	54
Figure(5.5) : Adding new cultivation.....	55
Figure (5.9): Adding Plant Type .....	57
Figure (5.10) : Soil Types .....	57
Figure (5.11): Adding Soil Type.....	58
Figure (5.12): Soil Specifications .....	58
Figure (5.13): Adding Soil Specifications .....	59
Figure (5.14): List of Irrigation Best Values.....	59
Figure (5.15): Adding New Best Value .....	60
Figure (5.16): List of Sensors.....	60
Figure (5.17): Adding New Sensor .....	61
Figure (5.18): List of Nodes.....	61
Figure (5.19): Adding New Node .....	62
Figure (5.20) : List of Zones .....	62
Figure (5.21): Adding New Zone.....	63
Figure (5.22) : Zone Sensor Management .....	63



## IX

Figure (5.23): Connect Sensors to Zones .....	64
Figure (5.24): Node Sensor Management .....	64
Figure (5.25): Connect Sensor to Node .....	65
Figure (5.26) : User Management Icon .....	65
Figure (5.27): Adding New User .....	66
Fig.(5.28): Sensors Readings .....	66
Figure(5.29): Total Readings based on Sensors.....	67
Figure (5.30): Total Readings based on Nodes.....	67
Figure (5.31) : Total Readings based on Zones .....	68
Figure (5.32): Moisture Level of one zone .....	68
Figure (5.33) : Generate Random Sensor Readings.....	69
Figure (5.34) : Sensor Reading Table .....	69
Figure (5.35) : List of Zones and their requirements of water.....	70
Figure (5.36) : List of JSON Objects .....	70

## List of Symbols and Abbreviations

IoT	Internet of Things
WSN	Wireless Sensor Network
ICT	Information and Communication Technology
GPS	The Global Positioning System
CC	Cloud Computing
DPU	Data Processing Unit
RS	Request Subscriber
IAMU	Identity and Access Management Unit
DR	Data Repository
PA	Precision Agriculture
GSM	Global System for Mobile Communication
SDC	Symmetrical Double Chain
SaaS	Software -as-a- Service
PaaS	Platform-as-a-Service
IaaS	Infrastructure-as-a-Service
SOAP	Simple Object Access Protocol
REST	Representational State Transfer
XML	eXtensible Markup Language
HTTP	Hypertext Transfer Protocol
CSV	Comma Separated Value
JSON	Java Script Object Notation
RSS	Really Simple Syndication
URI	Uniform Resource Identifier
API	Application Programming Interface
JAX-RS	Java API for RESTful Web Services
MVC	Model, View, and Controller

XI  
**A Cloud Application for Smart Agricultural Irrigation Management  
system  
By  
Mustafa Mohammad Younes  
Supervisor  
Dr. Adnan Salman**

**Abstract**

Agriculture is one of the most important sectors of the Palestinian economy and it is the main consumer of fresh water. It is the broadest economic sector and plays an important role in the economic development of any nation. Several factors have a major impact on agriculture activities includes water availability, soil type, climate condition, fertilizers, and diseases. In conventional farming, farmers have to make decisions about all these factors. This include: what to grow, how to use the irrigation schedule, the type of fertilizers, and the best method to control pest and diseases. Farmer's decisions are based solely upon their experience, which can result in wasting valuable resources such as water, fertilizers, time, labor, etc. Furthermore, conventional farming experience can result in growing plants that are not the most suitable for a particular soil and climate, which can cause less yield and profit.

In this thesis, we provide a cloud-based software application that is (combined with IoT devices) able to automates the irrigation schedule based on information obtained from agricultural experts and environmental data collected from the field by using sensors technology through Wireless Sensor Network (WSN). The application can easily be extended to automate fertilization as well as provide recommendation for weed and pest control.

# Chapter One

## 1. Introduction

Demands on water resources are constantly increasing due to natural growth in population. It is predicted that world population will double in the next 50 years. In response, greater yields must be extracted from the current agricultural areas and more marginal lands should be prepared. Generally, and particularly in Palestine region, agriculture is the largest consumer of fresh water, accounting for more than 50% of water consumption [1]. A pressing challenge facing the agriculture sector is how to boost the production with less land, less water, and without damage to the environment.

To meet these requirements, there is a strong need to create new unconventional agricultural techniques with the focus on productivity and better management of the available resources--productive farming. The focus of these techniques is to maximize the agricultural productivity [2] by increasing the yield and minimizing the cost and losses. One main component in productive farming is the employment of Information and Communication Technologies (ICT) [3] into agricultural management systems. These systems are called Smart Farming systems [4] In smart farming, multiple recent emerging technologies such as Internet of things (IoT) [5], Wireless Sensors Network [6] and Cloud Computing [7] [8] combined with machine learning techniques are integrated to create an environment that advance productivity in farming practice. Smart farming

is expected to have a major impact on the advancement of this development in the near future. This thesis describes the design and implementation of the software component of a smart farming system with the focus on the management of water use in agriculture, irrigation.

Irrigation is one of the main consuming factors of agricultural resources. To get optimal irrigation schedule, it is necessary to deliver to the plant the required quantity of water taking into consideration other parameters, such as the environment condition, the texture of the soil, the type of crops and its growth stage. In Palestine, the irrigation schedule is mainly conventional based on the farmer experience. This conventional irrigation approach causes inefficient use of water and over or under irrigation. This can reduce the crops yield and increase wasted fresh water due to non-beneficial evaporation, drainage, leaching, or ineffective water delivery. Further, more losses in fertilizers which is delivered through water is likely to happen. Even though, there is a significant effort in establishing additional surface water resources, saving water and improving operational skills can have a major impact on agricultural productivity [9].

One challenge faces farmers to practice productive farming is the lack of knowledge about the state-of-art productive farming. Inexperienced farmer needs better knowledge of the texture of the soil, the characteristics of the field, and the development of the crops. One goal of the application described in this thesis is to gather and store this knowledge in the environment.

### **1.1. Root Zone Management**

Irrigation, can be defined as the proper application of water to the plant root zone at the proper time. To get optimal irrigation schedule, it is necessary to deliver the plant with a required quantity of water, taking into consideration several parameters includes:

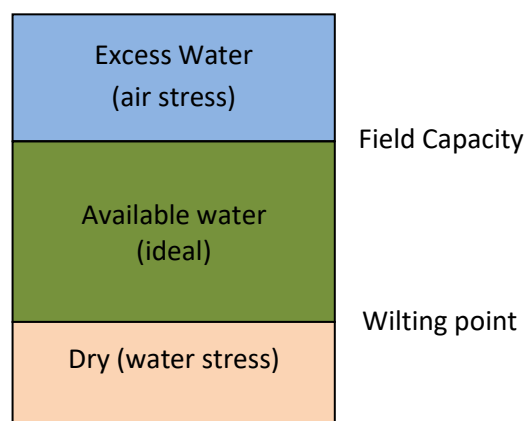
- 1) the type of plant, for instance, the irrigation requirement for tomato is different from the irrigation requirement of cucumbers.
- 2) the stage of growth of a plant, plant requires more water as they grow.
- 3) the texture of the soil, the fraction of sand, silt and clay in the soil mix. Different soil texture has different water holding capacity.

Water is necessary for proper growth and development of plants. In addition, to its own need for the plant, it acts as a delivery medium for transporting necessary nutrition to the root. These nutrient are typically delivered into a plant through irrigation. Therefore, optimal water management can help achieving optimal nutrient management. A plant demand for water must be available in the root zone in proper quantity for proper growth.

In addition to water, plants need optimal air concentration in the soil. Field Capacity, refers to a moisture level where all excess water are drained out due to the force of gravity. At this level small pores space of soil are full of water while large pores are full of air. This is the ideal condition for plant growth. It is easy for a plant to pull water and nutrition from the soil and air is available as well.

Lack of air in the soil due to excess water in the root zone, a condition called waterlogging or saturation, where soil pores are filled with water. If this condition happened for a long period of time, it can cause harm (plant air stress) [10] to the plant and reduce yield due to the lack of Oxygen. This will limit the growth of the root that causes the plants susceptible to diseases and other deficiencies. Further, it causes losses in water, fertilizers, and labor.

When the moisture level in the root zone drops below a threshold value called wilting point (where plant cannot pull water from the soil), it causes plant water stress. Under this condition water is strongly bounded by capillary to the soil particles and it is hard to be extracted by the crop. If this happens for a long period of time, the plant will wilt and possibly die. Field capacity [11] and the wilting point [12] depends on the soil texture and the kind of crops. For optimal growth to maximize the yield and to reduce wasting resources, water content should be maintained in the available water range. At this level, it is easy for the plant to absorb water and air is available in the soil as well. Figure (1.1) shows these cases.



**Fig. (1.1):** Fielding Capacity and Wilting Point

This study summarizes that maintaining soil moisture at optimal level in the root zone can enhance farming productivity. Too much moisture can cause root diseases and wasted water. While too little moisture can cause yield loss and plant death. There are several challenges to achieve optimal irrigation schedule includes, when to irrigate? and how much? The required amount of irrigation must, at least, meet the crop water loss through evapotranspiration. Both timing of irrigation and the amount of irrigation has a major impact on productive farming. However, timing of irrigation has a significant influence on crop yield and quality. In some growth stage, delayed irrigation can reduce the potential yield and quality significantly.

The application described in this thesis provides the software component of an automation technique to manage the root zone moisture. The goal is to control the water and air concentration in the root zone to their optimal level for a given crop and soil texture. Application of such a system will help increase the productivity of farming by increasing the yield and decreases losses in resources such as water, nutrient and human labor. The application uses recent emerging technologies which are described in the following sections.

## **1.2. Gathering data and triggering action in the field**

Recent advances in computing power, storage, and network communications have enabled Machine-to-Machine communication and shifted the roll of computing devices from passive devices to proactive devices. Based on these technology advancements, new systems are



emerging includes Wireless Sensor Network (WSN). The main components of these systems are small and cheap sensing elements and actuators linked together through wireless connections to perform sensing and actuation tasks. Sensors are able to collect information about the physical environment around them and communicate this information wirelessly to a base station. Based on this information, a decision-support system application running on the base station can be used to trigger events that execute actuation responses in the environment. Accessing IoT devices such as GPS, soil moisture sensors become cost effective and widely available to be deployed in the field in a large scale. By gathering data from the field using these devices, the use of agricultural resources such as fertilizer and water become more effective and wasting become less. Further, controlling the delivery of these resources to the plant will keep them at optimal level which results in increasing the yield and the quality. Also, using smart farming, land field can be divided into zones, in which each zone can be managed separately. This makes each zone better monitored and the resources will be better managed. For example, different zones can be managed differently based on their location and Topography. Flat zones will be managed differently from those on down a hill zones. [13].

### **1.3. Cloud computing**

The amount of data generated by WSN systems is significantly large and requires high storage and processing power, which is beyond the capability of a personal computer. Further, usually more information from domain

experts in the targeted application is needed, and feedback to the decision-support system is useful in improving the future decisions based on previous experience through the application of machine learning techniques. Typically, those experts are not available on site and their remote access to the system is essential to include their experience. Therefore, leveraging the recently emerging Cloud Computing Paradigm provides an effective solution to these issues. First, Cloud Computing provides a scalable computing and storage resources as needed by the system. Second, Cloud Computing provides most of its resources as services which allows sharing data and information [14] .

The goal of this thesis is to design a Cloud-based application which manages the data coming from the sensors in a WSN for smart farming and water management. Using Web Services [15] [16] [17] interface, the application will gather data collected by the sensors in the field, which includes environment conditions (e.g. soil moisture, temperature). Furthermore, the application will gather data through a web interface from agricultural experts about best agricultural practice which includes: 1) knowledge about the amount of water needed for a particular kind of crops as a function of age of the plant and the nature of the soil, and 2) the kind of fertilizers needed for each kind of plant at different age, and 3) the different types of soil and their specifications. This knowledge will be saved in a database. Based on this information, the framework will issue on different things in smart farming such as irrigation schedule for the plants that needs it, and fertilizer needed as a function of age and time, and it will

communicate this information to the base station in the field. Based on this information, the base station will trigger actuation in the field. The field is divided into different zones, which is a part of the whole field which is managed in particular, every zone will have managed automatically through irrigation and fertilizers, amount and period, and all other automated processes in smart farming [18].

The main goal of this thesis is to provide a software framework to enable building a smart farming system which is based on expert's knowledge, environment condition, and the type of plants. The framework together with decision-support system will allow inexperienced farmer to practice best farming in a semi-automated manner, by only deciding what farmer want to plant, and the type of soil farmer has, and the date of planting, then the smart farming process will progress with output profits.

By providing this application software, the smart farming achievement will be done, which means that there are efficient factors will be performed such as minimizing the amount of water needed by the plants, minimizing fertilizer needed, and minimizing of energy inputs and maximizing the yields. These all factors will improve productivity.

#### **1.4. Motivation**

We provide a cloud-based software application that is able to automates the irrigation schedule based on information from agricultural experts by web forms and environmental data collected from the field by using Wireless Sensor Network.

This cloud-based software application enables building a decision-support system to allow inexperienced farmer to practice best farming in a semi-automated manner.

### **1.5. Thesis Objectives**

#### **1) General Objectives:**

The general objective of this thesis is to provide a cloud-based software application that enables building a decision-support system to allow inexperienced farmer to practice best farming in a semi-automated manner.

#### **2) Specific objectives:**

1. Provide a Cloud-based application to manage data coming from the sensors in Wireless Sensor Network and information supplied by agricultural experts in order to create smart farming and water management system.
2. Provide communication interfaces to trigger actuation in the field.

#### **3) Research question and identified problems:**

- How to gather, structure, and store the data and information coming from the sensors and experts?
- How to use the data coming from the sensors and agricultural experts in constructing a decision-support system that decides the optimal irrigation schedule?

## **Chapter Two**

### **Literature Review**

There are several works done to integrate Wireless Sensor Network with Cloud Computing, for example. V. Paul [19] proposed framework for WSN-CC integration with the following steps:

For each cluster of WSN, there is a sensor gateway, which collect data from different sensor and process the sensory data, then the cloud gateway receives the sensory data from the sensor gateway of WSN. The cloud gateway processes the received data by decrypting the data with the data decryption unit and then decompressing the data with the data decompression unit. After that, the decrypted and decompressed sensory data from the cloud gateway are stored and processed by the powerful servers in the cloud. cloud encrypts the required sensory data with the encryption unit at the cloud gateway when the user requests it. Finally, the cloud gives feedback to the WSN manager, the feedback contains feature information. In this work there is encryption and decryption processes to the data, but there is no integration of the data from WSN and another data in comparison with our theses which is integrate data from WSN and data from experts or web sites.

Another work of integration is done, N. Moise [20] proposed another architecture of WSN integrated with CC as follow:

Integrate wireless sensor networks with cloud computing consists of combining advantages of the data gathering capability of WSN and the

powerful data storage and processing power of cloud computing. In this paper most of the work focus on the advantages of the combination between WSN and CC.

Another framework discussed here, K. Ahmed [21] show another framework of WSN –CC integration, this framework consists of components such as: Data Processing Unit (DPU), Pub/Sub Broker, Request Subscriber (RS), Identity and Access Management Unit(IAMU), and Data Repository (DR).

Data which collected from WSN moves to DPU through a gateway, and then send the data to DR. Users connect to the cloud through the secured IAMU and will give access. This framework established secured path for the data.

H. S. Guruprased [22] discusses both wireless sensor network and cloud computing technologies with their applications and mention overview of architectural extension to wireless sensor network. The applications of cloud computing enhance the reliability and availability of wireless sensor networks with real time applications.

C. R. Chowodhury [23] gives a brief survey of sensor cloud integration. There are three areas of sensor cloud integration: Sensor Cloud Database, different cloud based sensor data sharing platform and cloud based sensor data processing approaches. In this survey the data gathered from WSN will stored in Database in the cloud, but in our work the data from WSN will integrated with data from experts then stored in Database in the cloud.

W. Kurschi, [24] do another work in combining and discussing the services from combine WSN and CC, he presents a model which combines the concept of wireless sensor networks with the cloud computing paradigm, and the benefits from this combination. The architecture system combines sensor networks with the cloud computing paradigm, which consists of several basic services with these requirements:

Receive and manage sensor data from sensor nodes, then manage a set of filters that perform on-line analysis on sensor data, and then run filters offline on a given set of sensor data, after that run filters on a given set of sensory data in order to provide online analysis.

S. Janani, [25] show the proposed system for the integration of wireless sensor network with cloud computing for patient monitoring. The proposed system architecture has four agents; they are aggregator agent, patient agent, doctor agent, and nurse agent.

A group of patients has been connected to the cluster head which act as patient agent. Different cluster head is used to send the information to the access point which is used to access the patient information. The information of the patient agent has been transferred to the base station where aggregator agent presents. The aggregator agent is used to receive the information and check for denotations of anomalous readings which is sent by the patient agent and start alerting to doctor agent and nurse agent. Then the aggregator agent transmits the patient's parameters to the cloud computing and storing in the database. So, the doctor and nurse can deal with the patients using this proposed system by sending alerts to the duty

doctor and nurse. The differences between this work and our work are that our work is done in the agriculture, not for patients, and our work integrates more than one kind of data from different places.

A. Botta, [26] reviews the literature about the integration of Cloud and IoT by analyzing and discussing the need for integrating them, the challenges deriving from such integration, and how these issues have been tackled in literature, and then describe application scenarios that have been presented in literature, as well as platforms. From this work we know the advantages from the integration between WSN and CC.

N. Alharbe, [27] has worked in the similar area, he shows the proposed smart hospital network system architecture “Use of Cloud Computing with wireless Sensor Networks in an Internet of Things Environment for a Smart Hospital Network”: a system for the purpose of detecting, locating and monitoring the object. This architecture divided into six layers system structure: a Data Processing Layer, a Data Integration Layer, a Cloud Computing Layer, a Network structure, a Knowledge Reasoning layer, and a Visualization layer.

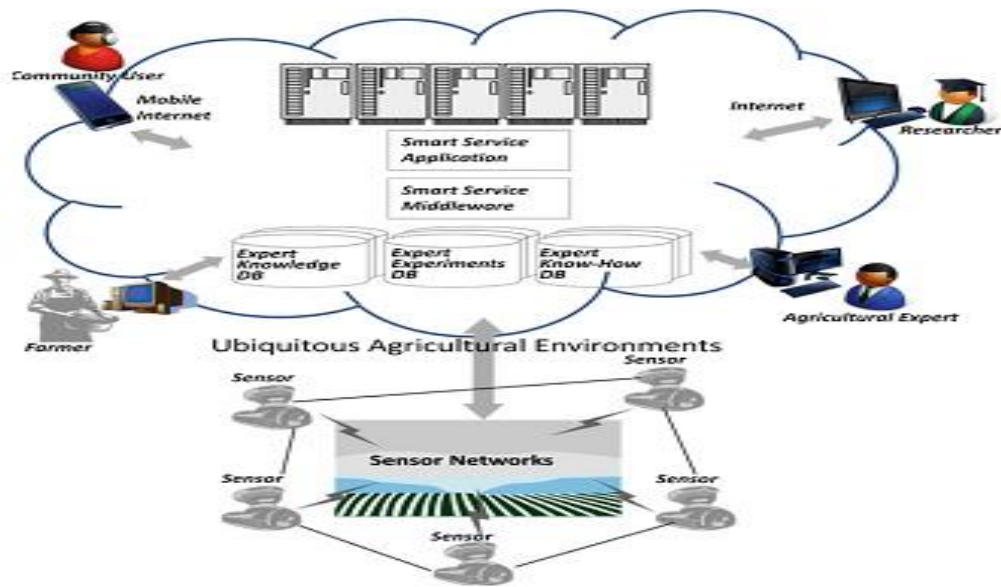
Now, we have proposed the work which is focus on agriculture, M. H. Anisi, [28] present a survey of wireless sensor network approaches and their energy consumption for monitoring farm fields in Precision Agriculture, (PA) is the use of information and communication technology together with best agricultural practice for farm management. PA requires the acquisition, transmission and processing of large amount of data from farm fields.



Another paper on a Precision Agriculture is done by A. Awasthi, [29] he explores the potential of WSN in the area of agriculture in India on sugarcane crop. System is designed based on low-power ZigBee wireless communication technology for system automation and monitoring. In this paper, AnjumAwasthi does not use cloud computing, so the data which is gathered from wireless sensor nodes transmitted to base station using ZigBee. Data is received, saved and displayed to base station to achieve soil temperature, soil moisture and humidity monitoring. If the data exceeds the base station limit, message is sent to the farmer mobile through GSM network to do an action.

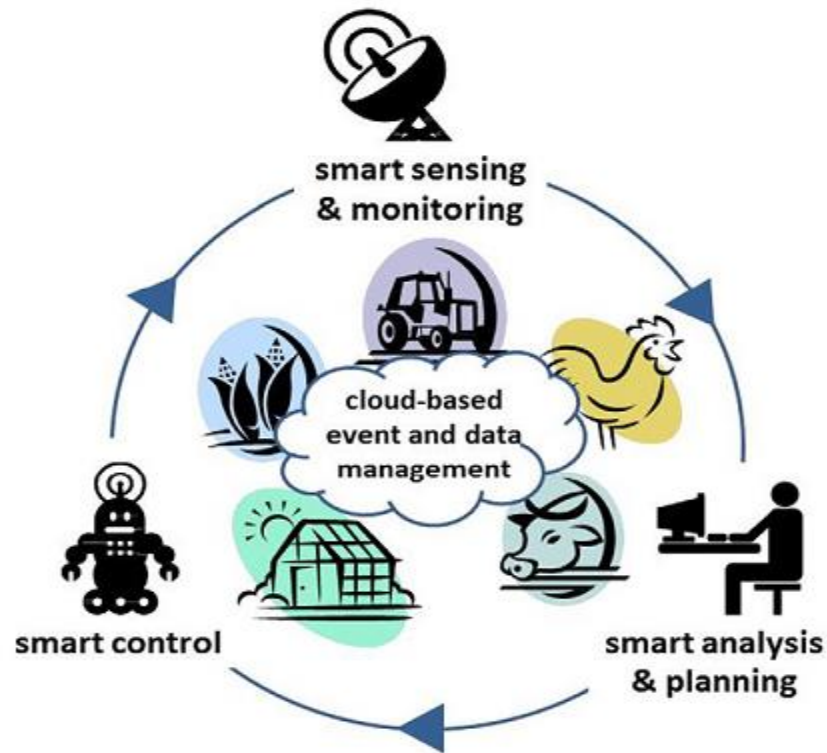
Another work is done on a Precision Agriculture area by J. Li, [30] he optimize topology for improving the network life time, differ from the traditional topology, he designed a Symmetrical Double-Chain (SDC) topology which is suitable to be deployed in farmland and the lifetime become longer than traditional tree topology, the system that designed will greatly help farmers to make more informed decisions on the efficient use of resources and hence improve black pepper productivity which is used in this paper. We notice that there is no use of Cloud Computing in this paper too.

We consider that smart farming is an important field, there are more and more works done on it, Y. Cho, [31] mention paper "An Agricultural expert Cloud for a Smart Farm" he gains valuable knowledge and data from experts and put them in cloud to help farmers and any one to use them in smart farm services. So, user can successfully and efficiently use this expert cloud to cultivate any crops in agricultural environments as figure (2.1):



**Fig (2.1):** Using Cloud in Smart Farming

S. Wolfert, [32] presents a review on big data and using it in smart farming. Using internet of things and cloud computing make the agriculture smart. The author mentions cyper-physical management cycle of smart farming, the smart device which connected to the internet can control smart farming, sensors which spread among the field also can add data to the database of smart farming, robots and humans can control and manage the smart farming by modifying and adding or deleting some data of database as figure (2.2):



**Fig. (2.2):** Cyber-Physical Management Cycle of Smart Farming

These all things that effect on farming make the data collected is large and large because sensors and robots producing big number of images and videos, so the need of big data become important to facilitating agriculture in different fields.

Big data can predict farming operations and make some decisions for real-time operations. So, this review focus on big data applications which related to smart farming. So, big data applications handle global issues such as: food safety and security, efficiency, and sustainability.

In this review, the authors developed framework for analysis from a chain network, and there are network management between the stockholders in this network. So, this framework can provide a systematic classification of issues and concepts for the big data application analysis in smart farming.

Next paper discusses the development of smart farming technologies and their application in Brazil, D. Pivoto [33], mention that cloud computing and internet of things expected to advance the development of smart farming which involves merge communication and information technologies into equipment, machinery, and sensors for use in farming production systems. The main aims of this paper are to describe smart farming in Brazil from the perspective of exports in agriculture field, and to describe the scientific knowledge about smart farming that is available in the world.

The authors of this paper depend on two stages of research: interviews with four experts of smart farming and scientific and bibliometric analysis using text mining. From these methods, the authors see that there are need to developing technology for smart farming, then manage these technologies and integrate them in supply chains and in farms. Then impact these technologies on production system and the environment.

In other hand, there are some limitation in using smart farming, such as education of farmers, skills and ability of farmers to understand and use smart farming tools. The other limitation is the lack of integration between the different systems with the supply chains.

Then Q. Khan [34], make a research about advance modeling of agriculture farming and its techniques using internet of things. He developed smart framework which has centralize control for water sprinkle and developed another module to check soil conditions such as soil composition and water holding capacity of the soil.

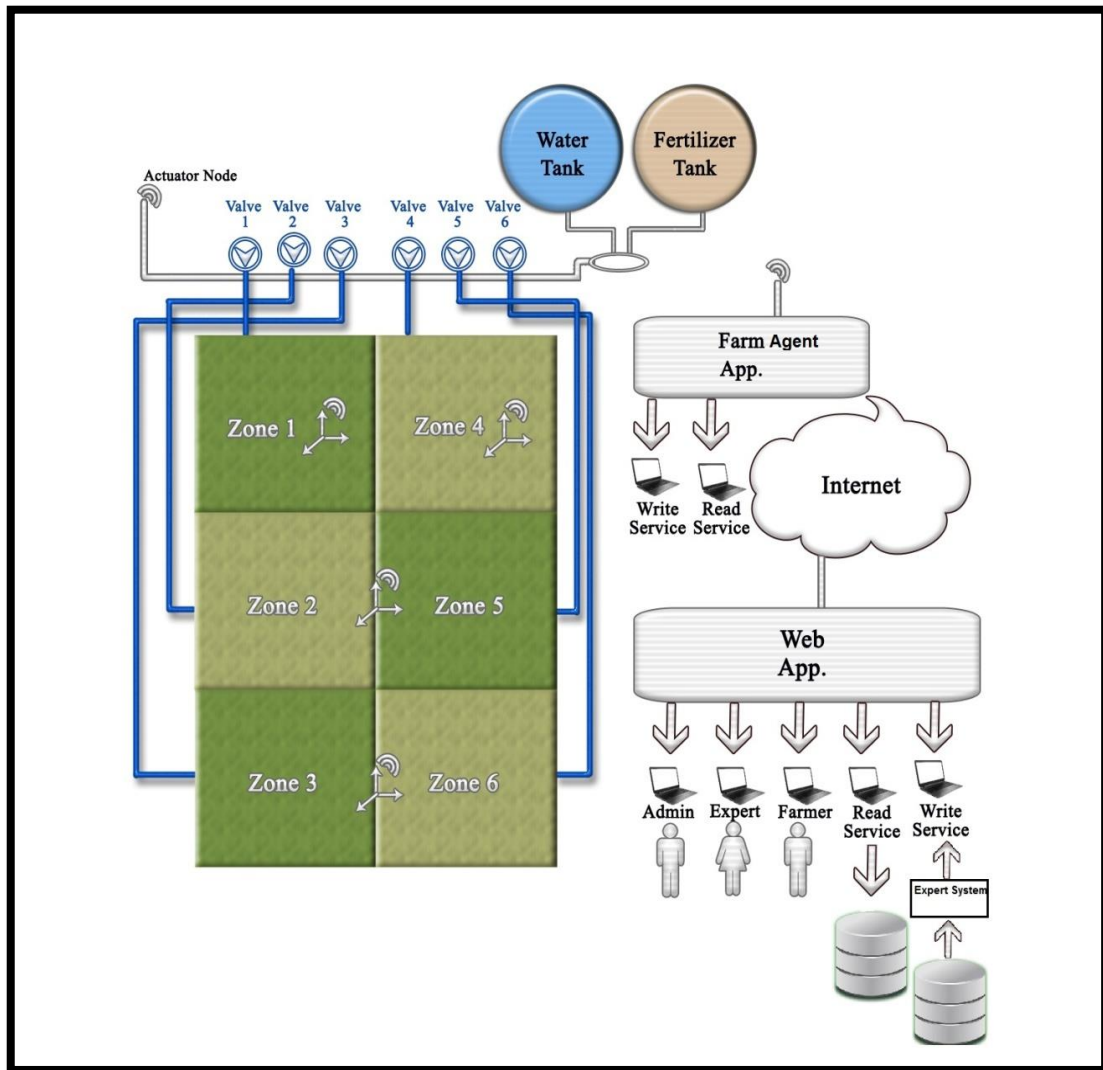
The framework which developed in this paper consists of six modules: module of the sensors to take inputs required for processing. Then, the input data want to be processed, so he uses processor module to process the data coming from the sensory module and analyze it. Actuator to perform any action, i.e. data collection and action. Memory module used to store any data gathered from the sensory inputs and data which processed in processing module. Then the application management module which is the core of the framework, it has a server that handle all the tasks of the internet of things smart farming. Finally, decision support system which take a final decision to actuator module.

## **Chapter Three**

### **System Architecture**

In this chapter we describe the architectural design of a Cloud-based application which manages the data coming from the sensors in a WSN for smart farming and water management. The application gathers data collected by the sensors in the farm, which includes environment conditions (e.g. soil moisture, temperature). Furthermore, the application gathers data through a web interface from agricultural experts about best agricultural practice which includes knowledge about the amount of water needed for a particular kind of crops as a function of age of the plant and the type of the soil. This knowledge and the sensors data is saved in a database. Based on this information, the application determines an irrigation schedule for the plants that needs it, and communicates this schedule to the base station in the field. Consequently, the base station triggers irrigation events in the field. The application, not only, schedule irrigation but also helps the farmer make decisions about some of these factors in order to automate the decision making process.

The architecture of the management system as shown in Figure(3.1) consists of two main components. The hardware component, and the software components.



**Figure (3.1):** Smart Agricultural Irrigation Management System Architecture

The hardware component consists of IoT devices distributed in the field. These devices gather data from the environment through sensors and communicate this data to a base station through a wireless network in the field. Other IoT devices (actuators) are connected to the base station to trigger action in the field such as open and close water valves. The software component consists of two applications, the farm application, and the cloud application. The farm application is executed on a computer in the farm. It has two main tasks, 1) It gathers the data from WSN and puts it

in a specific format and communicate this data to the cloud application through a web service and 2) It receives decision from the cloud application in a timely manner about which valve should be open for irrigation. The cloud application, is a web application that runs on a cloud server. It saves the data obtained from the farm application and from agricultural experts in a database. Further, it has a decision support system that determines when to irrigate each zone of the farm based on the moisture sensors data and the agricultural expert's data. The focus of this thesis is only the cloud application.

The rest of the chapter is organized as follow. Sections 3.1 and 3.2 give a brief background about the WSN technology and cloud computing. Section 3.3 gives a brief review about web services. Section 3.4 provides the architectural design of the smart farming web application design. Finally, Section 3.5 gives a description about the database for the smart farming application.

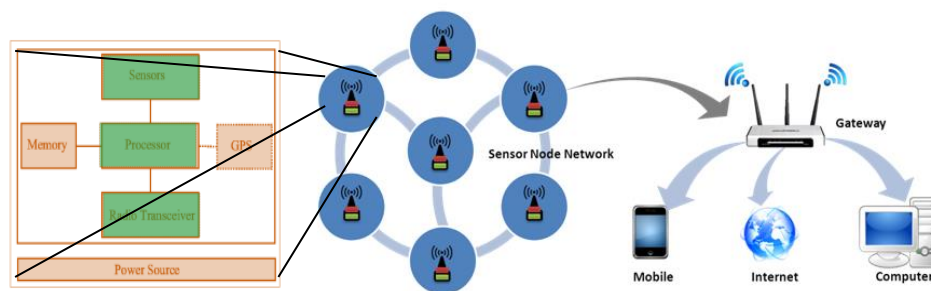
### **3.1. Wireless Sensor Network**

Wireless Sensor Network (WSN) [35] consist of a large number of sensor nodes that collect data from their environment (real world) through sensors and send these data over wireless links to a central computer called base station. WSN is a recent important technology in which sensors are placed in a distributed manner to monitor and control the physical conditions in the environment around us. These networks are capable of measuring changes in many environmental parameters (e.g., temperature, pressure,



sound) periodically. Recent falling in the cost of sensors, communication, and processing has motivated scientist and engineers to adopt WSNs in the automation of different large scale processes in several domains includes industry and agriculture.

Typically, in these systems, a large number of sensor nodes are deployed in different regions. Each node monitors multiple environment parameters in its assigned region periodically. The sensor nodes can communicate among themselves using radio signals. The data collected by all the nodes needs to be saved, shared, and processed for a decision-support system to trigger actions in the environment or notify users in order to make them take necessary actions [36]. This results in a large amount of collected data periodically that requires a large storage place, significant processing power, and the capability of sharing this information, see figure (3.2) . The basic elements of a WSN are, sensor node, gateway, and base station described below.



**Figure (3.2):** Wireless Sensor Network

**Sensor node:** A wireless sensor node contains sensing and computing devices, radio transceiver and power components.

It gathers environmental properties through attached sensors, translates them into digital form and sends them wirelessly via other sensor nodes to a base station. Sensors can be viewed as transducers that sense characteristics of environment and translate them into electrical signals. Further, actuator can be connected to a wireless node to trigger actions in the environment (e.g. turning a pump on or off).

**A base station:** A base station is a computer that interfaces between users and the wireless sensor network. It executes applications that are responsible for receiving and processing data collected by the sensor nodes. Also, it is responsible for triggering actions (e.g. turning a pump on or off) in the environment by sending control messages to actuators connected to wireless sensor nodes. Typically, the amount of data collected by the sensors and the amount of necessary computation is large and beyond the capability of a personal computer. Therefore, it is more practical that the base station is connected to a cloud based servers through web services where the data can be shared and processed.

**Gateways:** Gateway is a networking device that interconnect two different networks running different communication protocols. The gateway in the WSN is necessary to interconnect the communication between a WSN using zigbee protocol [37] and the wifi/ethernet protocol.

### 3.2. Cloud Computing

Cloud Computing CC is a recent emerging technology. It offers a scalable solution to manage, process, and share large amount of data. In cloud computing, a large number of computer systems are connected in private or public networks to provide dynamically scalable infrastructure for applications and data storage as services over the Internet. Some of these services are:

**Software-as-a-service (SaaS):** The SaaS concept provides a broad market solution that may include anything from web based email to inventory control and database processing. End users can access the service over the Internet, like collaboration, business processes, industry applications and e-Health

**Platform-as-a-Service (PaaS):** In PaaS consumers can host applications using platforms which include the runtime software necessary to host consumer developed applications

**Infrastructure-as-a-Service (IaaS):** IaaS provides consumers with an opportunity to consume processing, storage, network, and other resources. There are multiple vendors of cloud computing infrastructure includes Google [38], Microsoft [39], Amazon [40], Oracle[41], etc. The list is expected to grow in the near future. Cloud computing provides rapid access to flexible and low cost IT resources. It eliminates the need to purchase large hardware and spend time for installation and managing this hardware. Therefore, with cloud computing we can access as many resources as the application demands, and only pay for what we use.

### 3.3. Web Services

The communication between the cloud-based framework and the base station in the field will be based on web services. A web service is a piece of software that is available remotely over the Internet. Web services are not tied to any operating system or any programming language which makes them loosely coupled. They are self describing discoverable via a simple find mechanism. They provide a reliable way for publishing data and applications over the Internet for sharing to allow others to arrive to these data sources, applications and services to benefits from them. The architecture of a web service consists of three major roles:

- Service provider: the provider creates the web service and makes it available on the Internet.
- Service requestor: any consumer of the web service and needs to contact a web service.
- Services Registry: the registry provides a central place where developers can publish new services, or find existing ones.

There are two types of web services, the Simple Object Access Protocol (SOAP) [42] and the Representational State Transfer (REST) [43].

SOAP is a standard-based protocol that has been developed earlier than REST. It depends on XML for messaging services and from the beginning it is designed to be extendible. SOAP is not tied to any particular transport protocol (even though HTTP is popular). Another important feature of SOAP is its built-in error handling. Unfortunately, the XML-based

communication can become very complex in some programming languages.

Unlike SOAP, REST, the successor of SOAP, is not a protocol. It is a network-based architectural style developed to simplify the access to web services. It provides a lighter way approach. Instead of using XML, REST uses URL for messaging services and uses the HTTP1.1 verbs (GET, POST, PUT, and DELETE) to perform its tasks. REST web services doesn't have to use XML for messaging. It can communicate the data in Comma Separated Value (CSV), JavaScript Object Notation (JSON), XML, and Really Simple Syndication (RSS) [44]. These features free the programmer to choose the data format that suites the used programming language.

Both SOAP and REST has their own advantages and disadvantages based on the requirement of the application. We choose the REST web services over the SOAP web services for several reasons includes: 1) Easier to use and develop, 2) Efficient (Can use smaller message format, and 3) Faster, since there is no intensive processing required.

### **3.3.1. Restful web services**

REST web services [45] are built to work best on the web. REST is an architectural style which is based on web-standards and the HTTP protocol. It has some properties such as performance, scalability, and modifiability that enable services to work best on the web. In REST based architecture everything is a resource. A resource is accessed via a common interface

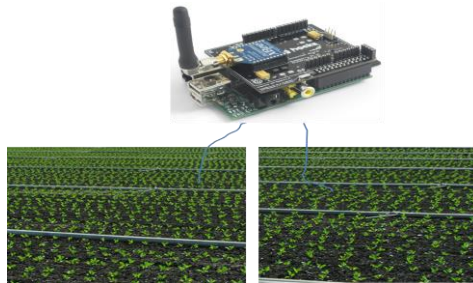
based on the HTTP standard Methods. The REST architectural style is based on four principles:

- Resource identification through URI: A REST web services exposes a set of resources that identify the targets of the clients. These resources are identified by URIs (Uniform Resource Identifier) which is a global addressing space for resource.
- Uniform interface: Resources are manipulated using a fixed set of four operations, create, read, update, and delete
  - PUT creates a new resource, which can be then deleted using DELETE.
  - GET retrieves the current state of a resource in some representation.
  - POST transfers a new state onto a resource.
- Self descriptive messages: Resources are decoupled from their representation so that their content can be accessed in different formats (HTML, XML, PDF, etc).
- Stateful interactions through hyperlinks: Every interaction with a resource is stateless.

### **3.4. Sensors Location**

As shown in figure (3.1), the field is divided into several irrigation zones. A zone can be considered as an irrigation management unit that has semi-uniform characteristics, such as topography, type of soil, kind and age of the growing crops. Each zone corresponds to an area for irrigation. A zone

can be irrigated by turning on and off an electric valve controlled by the farm application. In each zone, several sensors can be placed to get the value of different parameters necessary for optimal growth of the plant such as pH and Salinity levels. For optimal irrigation [46], soil moisture level in the root zone must be maintained between the field capacity and the wilting point as discussed before [introduction]. Since the size of the root zone increases as the plant grow, moisture sensors are placed at different locations and at different depth in the field. Each sensor is connected to a wireless node. Sensors placed in different zones can be connected to the same wireless node. Therefore, each sensor is identified by its zone and to which wireless nodes it is connected. Figure (3.3) demonstrate the connection of multiple sensors from different zones to a wireless node.



**Fig.(3.3) :** Sensor location

The data gathered by the wireless nodes are transmitted through an RF antenna to a gateway node, where it receives all different kind of sensed data in a timely manner. This data will be passed from the gateway to a computer in the field where it then can be transmitted to the cloud application.

### **3.5. Farm Application**

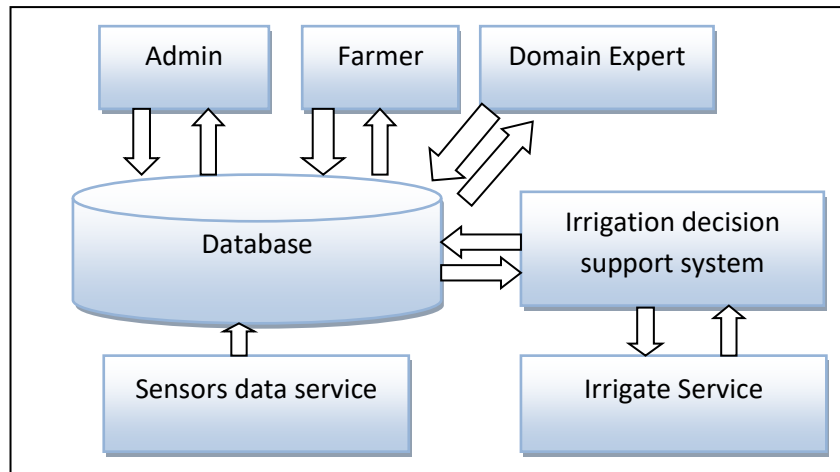
The farm application is a software runs on a computer in the field. The main tasks of the farm application are:

- 1) In a timely manner the application reads the sensor data from the sensor nodes deployed in the field. Then, it puts the data in a specific format and communicate this data to the cloud application by invoking the cloud application web service "upload Sensors Reading".
- 2) In a timely manner the farm application invokes a web service "zones to Irrigate" provided by the cloud application to get the list of zones ids and which ones to turn on and which ones to turn off.
- 3) Then, based on the data received from the cloud application, it sends messages to the wireless nodes that controls the water valves to turn each water valve on or off.

### **3.6. Cloud Application**

The cloud application is a software runs on the cloud. It gathers sensors data from the field through web service interface, knowledge about the soil and plant characteristics through a web form interface, the arrangement of zones and sensors through a farmer web form. The cloud application based on this information triggers irrigation based on this data through the irrigation service.





**Fig.(3.4) :** Software architecture of the cloud application

The software architecture of the cloud application is shown in Figure (3.4). As shown in the figure, there are three different users of the web application, admin user, expert user, and farmer user, in addition to two web services for programs to access. All users have different privileges to access the backend database.

**Farmers interface:** Farmer interface provides a web form, where the farmer can select the crop he wants to plant and the cultivation date, also, through this interface, the farmer sets zones identifiers, different kind of sensors identifiers and to which wireless node each sensor is connected and in what zone.

**Admin interface:** The admin user has full privileges on all tables of the database, and can create, modify, and delete any user he wants.

**Domain expert interface:** This interface allows agricultural expert to add, modify, and delete information about different kinds of plants and soil growth characteristics. This includes for each plant type and age, the optimal soil moisture upper and lower thresholds for different soils types.

This information is saved in the irrigation optimal values table with the appropriate data for different kind of plants and different types of soil.

Save sensors data web service: This web service is used to upload the readings of the sensors in the field and save this data in the backend database. The farm application gathers the sensors data from the field, put the data in a JSON object and invokes the service in a timely manner. Once the service gets the data, it will save it in a backend database. The interface of this service is upload Sensors Reading (JSON object). The JSON object hold all sensors reading.

Irrigation web service: The second web service is to generate a list of zones and their requirements of water," update zone moisture". In this web service, we calculated the average zone moisture depending on the readings comes from the zones, and compare this average zone moisture with the moisture best values level which are entered by the expert of farming, if the calculated moisture values less than the best value of the moisture level, then this zone must be irrigated until the moisture level become within the range.

Irrigation decision support system: The irrigation decision support system decides whether a zone is to be irrigated or not based on zones' moisture level, the type of soil, and the kind and age of the plant. The moisture level of a zone is calculated by taking the weighted average of all sensors readings in the zone. Weights based on the age of the plant can be used to give lower weights for deep sensors when the plant is young and increase the weights for these sensors as the plant grows. However, we didn't

implement this approach in this thesis. We just calculated the average moisture of all sensors in the zone and call it zone moisture.

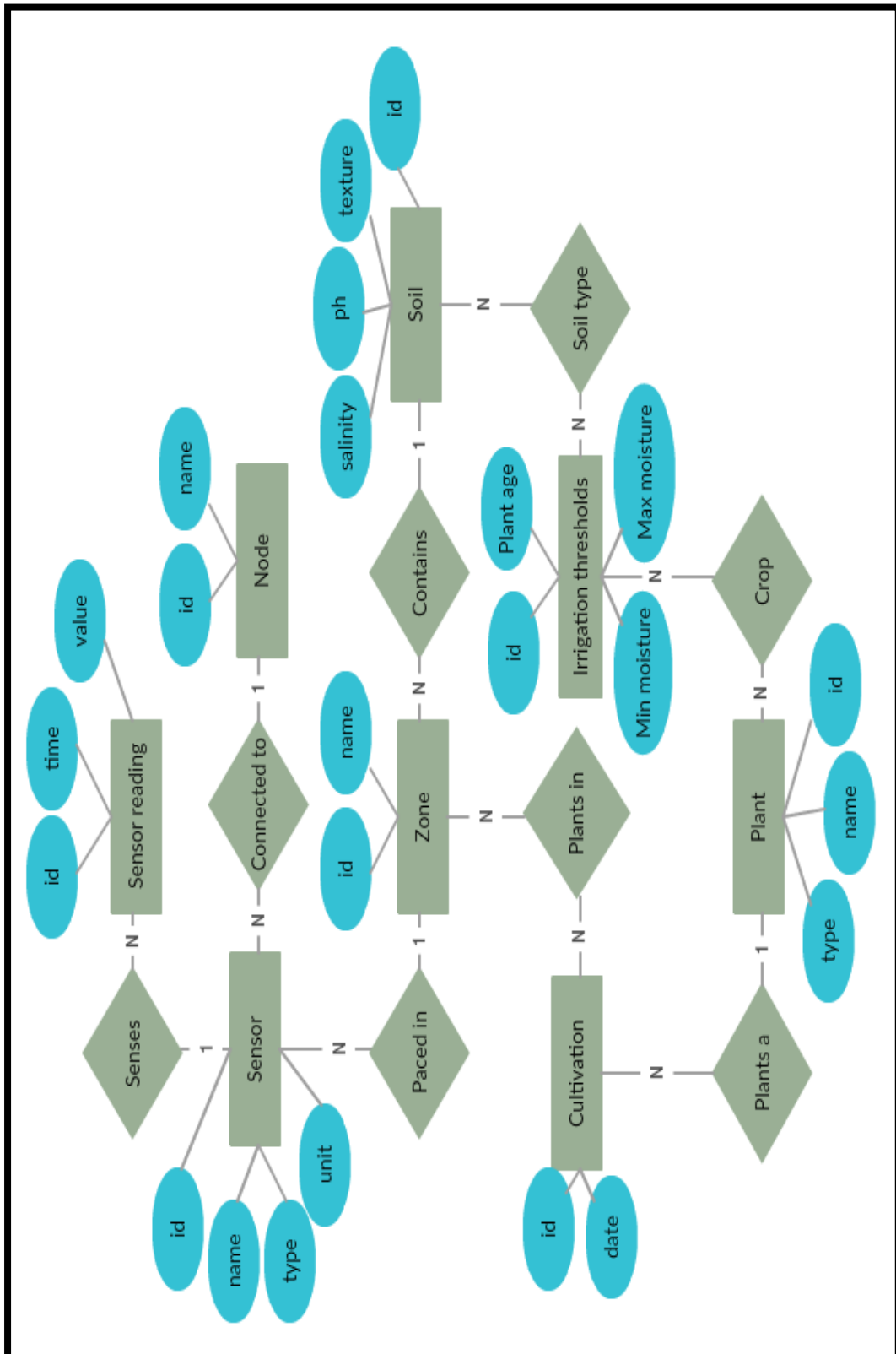
Once the zone moisture level is calculated for each zone, it is compared with the threshold data stored in the database for each crop and soil type. If the moisture value is within the optimal range or exceeds the upper threshold, the irrigation is set off for that zone. If the moisture level is lower than the threshold value, then the irrigation is set to on. Once, all zones irrigation is set, the web service return them to the caller (the farm application).

### **3.7. Database**

The Cloud Application uses a backend database to manage the data gathered from the sensors in the field and the data gathered from agricultural experts. The moisture level of a zone depends on the amount of water added, the type of the soil in the zone, and the type and age of the plant. The goal of the system is to maintain the moisture level in each zone to stay between the lower and upper levels as discussed in the introduction. When the moisture level for a particular zone goes below the lower threshold, the system will response by triggering the Farm Application to open the corresponding valve for irrigation. The irrigation of the zone continues until the moisture level exceeds the maximum moisture threshold for that zone. At this point the system triggers the farm application to close the valve. The irrigation process continue in this manner. The lower and upper threshold for different plant types and soil texture is set manually by

an agricultural expert. In future work we are planning to make these parameters to be learned from the data.

In the following, we discuss the schema of the database system. Again as discussed before, each cultivation consists of one or more zones. Each zone can be irrigated using an electric valve and it contains multiple different sensors. These sensors are connected to wireless node to communicate the data to the base station. A wireless node also controls the valves and which valve to open and which one close.



**Figure(3.5) :** Data model for the smart irrigation system

The data model of the smart irrigation system is shown in Figure(3.5). It consists of the following entities.

1. Cultivation entity: Each cultivation consists of one or more farming zones and one kind of plant. Each cultivation has its own id, and date of planting. The age of the plant can be calculated from the cultivation date.
2. Sensor entity: Several kinds of sensors that measure different quantities can be used in this system. Each sensor has attributes: id, name, type, and unit. The type attribute of a sensor is the physical quantity that it measures, such as moisture, temperature, humidity, etc. The unit attribute is the unit of the measured data. For example, degree Celsius for temperature. Data collected by the sensors in the system will be saved in the Sensor Reading table. Since each physical sensor is connected to a wireless sensor node in the field. The sensor entity is connected to the node entity as shown the schema. The relationship is many-to-one, since many sensors can be connected to a particular node, but a particular sensor can only be connected to one particular node. Also, since sensors are placed in zones in the field. The sensor entity is related to the zone entity by a many-to-one relationship since multiple sensors are placed in a zone. While, a particular sensor can only be in one zone.
3. Node entity: A node entity corresponds to a physical wireless node in the Wireless Sensor Network in the field as discussed in Section[3.1]. Each node has the attributes: name and id. These attributes should

match labels on the nodes in the field. The node entity is useful to allow the system communicate with actuator, or detect failure.

4. Zone entity: The cultivation farm is divided into one or more management units that we call zones. The system treats each zone as it has uniform characteristics such as topography, soil texture, soil salinity, etc. Sensors measurements in a particular zone are averaged to obtain a zone quantity. For example, the zone moisture is the average of the reading of all moisture sensors in that particular zone. The zone entity is related to the sensor entity by one-to-many relationship. Since a particular zone can have many sensors, also, the zone entity is related to the cultivation entity by many-to-many relationship.
5. Soil entity: The soil entity contains information about the characteristics of the soil and its specifications such as water holding capacity and salinity. The attributes of soil entity are: id, name, texture, and Salinity. Other attributes can regarding soil characteristics can be added. This entity has a many-to-one relationship with the zone entity since each zone has one particular type of soil, but several zones can have same soil types.
6. Irrigation thresholds: This entity holds values of the upper and lower moisture levels for a particular plant and soil type. In this thesis, agricultural expert provides these values through a web interface. This entity has the attributes: id, plant age, minimum moisture level, and maximum moisture level. Other thresholds can be included as well such as minimum temperature, maximum temperature, etc. This table is

connected with two tables: plants table and soil table as shown Figure (3.5).

7. Plant entity: The Plant entity holds information about different plants. This entity is related to the irrigation threshold entity and the cultivation entity. The Plant table contains a list of plants which is filled also by an experts of farming. The attributes of this Plant entity are: id, name, and type.

More detail about the schema shown in Figure (3.5) is provided in the implementation chapter which includes other entities such as login entity and plant types entity.



## **Chapter Four**

### **System Implementation**

This chapter discusses how we implement the web services used in our thesis using web services methods in java environment, how to run and call these web services. And how we implement the web interfaces, and the implementation of the database, tables and attributes.

#### **4.1. Oracle SQL Developer**

Oracle SQL developer [47] used in our application to create database, tables and attributes in wizard styles which is easy to implemented and developed.

By oracle SQL developer there is no need to use command line. So we used oracle SQL developer for create, modify, and delete all database objects.

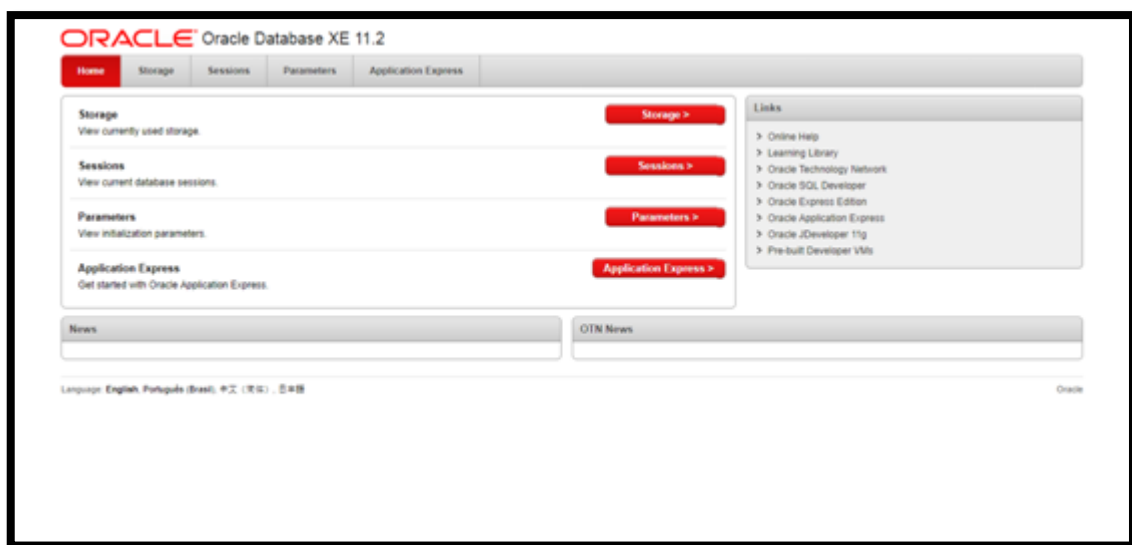
It is a free graphical tool that improve productivity and simplifies database development tasks. Oracle SQL developer simplifies development cycles and there is no need to buy non-oracle tools for developing and debugging SQL code. Using SQL developer, users can create, modify and browse database objects, run SQL statements. Also users can create database connections for non-oracle databases, such as MySQL. Also it can import the data and objects from these platforms to oracle easily.

SQL Developer can run on Windows, Linux, and Mac OSx. It is developed in Java leveraging the oracle JDeveloper IDE.

We chose this environment mainly for its ease and our familiarity with it. SQL Developer allows users to create the connections of stored database using a simple dialog, and they used these connections to browse the database, execute SQL statements, and create schema objects.

## 4.2. Oracle Database Expression Edition

We used oracle database expression edition [48] as database management system, which contain our application database, all creation of database objects were done in this environment, it is a free version of relational database, it is easy to install, easy to manage, and easy to developed. With Oracle Database XE like figure (4.1), you can easily create database objects, also you can import, export, and view table data, run queries and also run SQL scripts. Using Oracle database XE, you can also generate reports and creating different database users with different privileges.



**Figure (4.1):** Oracle Database XE

### **4.3. Oracle Web Logic Server**

It is a web server used to deploy our application to the web using JDeveloper. So by using web logic server [49] users can see our application on the web.

Web logic server is the best application server for building and deploying Java applications with supported many features, with low cost, more scalability, and improving performance.

The web logic server infrastructure supports many distributed applications. The web logic server complete implementation of the java specification provides a standard set of APIs for creating java applications, that can access many services, such as databases, connections to external enterprise systems, and messaging services.

Web logic server includes several utilities that can used to create, monitor, and manage domains such as, configuring active domains, starting and stopping servers, monitoring application performance, monitoring server health and performance, and viewing server logs. We can also use web logic server's management API's to create custom management utilities.

### **4.4. IDE JDeveloper**

We used Oracle IDE Jdeveloper [50] to implement our application. By Jdeveloper we wrote the codes of our web services used in this application by java programming language. Also by JDeveloper we make connections with our database, also we wrote the codes of the web interface and login

pages, and to deploy our application to the oracle web logic server to become available to the users.

Java is a programming language which was developed by Sun Microsystems which was initiated by James Gosling and released in 1995.

Java is an object oriented, which means that everything in java is an object, so it can be extended because it based on the object model. Java is Platform Independent, which is mean, that when java is compiled, it is compiled into platform independent byte code, not compiled into platform specific machine. So, byte code is distributed over the web and interpreted by the virtual machine (JVM). Java is a simple programming language; it is easy to learn. Also, java is secure, and architecture-neutral which means, that java compiler generates object file format, which makes the compiled code executable on many processors, which is improve the runtime.

In this thesis we used Jersy RESTful web service framework which is open source, good quality for developing RESTFul web service in java that provides support for APIs such as JAX-RS ones.

We used MVC architecture which is short of model, view, and controller. It is easy way for organizing our code. It is used for developing user interfaces. So MVC consists of three components, the model which is the central of the pattern. A view is any output representation of information. And the controller converts input to some commands.

#### **4.5. Web Service Implementation**

Now, we have arrived to system implementation, we mention before in system architecture chapter that the architecture of the management system consists of two main components, the hardware component and the

software component. In this implementation chapter we focus on software component. The software component consists of two applications: the farm application and the cloud application. The farm application is executed on a computer in the farm. First, farm application call the first web service "sensors data service" in automated way from the web cloud application to send the sensors data collected from environment by wireless sensor network and write them on database of the web cloud application.

We have used in this first web service PUT method to create a resource in this case, so by using PUT method we updated database by inserting into sensor node readings table these attributes: node id, sensor id, description, reading time, reading value, and sensor node id. This link can be used to open this web service: `restwebservice/sensorReading`, this appear in the sample code (A.1) in appendix A:

The second main task of farm application is to receives decision from the cloud application in a timely manner about which valve should be opened for irrigation, to do this farm application call another web service "Irrigation Service" from the web application which send a list of zones and their requirements of water, if the zone needs to be irrigated a list has "ON" value beside that zone, else the zone has "OFF" value which means that the water in that zone is enough and there is no need to irrigated.

The second web service is to update zone moisture, we use PUT method also, which is used to update capabilities. This web service method used to check reading average within time best values and update zones new table ON/OFF. We get average before ten minutes from system date and

calculate reading average or by providing date interval, after calculating average, we compare if it lays between minimum and maximum moisture value from system best values, if less than minimum, turn it ON, else turn it OFF. We did this by getting date and age by weeks from cultivation plant table, also get zone and get from it soil id, and get plant id, get minimum and maximum values from best values based on plant id, soil id, and age, then compare and update. We notice this in the sample code in Appendix A.2:

We added in implementation another web service that query a list hierarchy, we used in this web service GET method which is used to retrieve a representation of resources. This web service used to give a list of JSON objects. This list first elements level is zones, and inside each zone there are a list of child list which is either node, or sensor, and inside the nodes there are another level of sensors. The list returned has id, type and child list and other attributes, as appear in the code in Appendix (A.3):

#### **4.5.1. Web Services Methods**

GET:

The HTTP GET method is used to retrieve (or read) a representation of a resource. In a non-error path, GET returns a representation in XML or JSON and an HTTP response code of 200 (ok). In an error case, it most often returns a 404 (NOT FOUND) or 400 (BAD REQUEST). SO GET requests are used only to read data and not change it, which is safe way because there is no risk of data modification or corruption. So GET never modify any resources on the server.

**PUT:**

PUT is used for update capabilities; it can also be used to create a resource in the case where the resource ID is chosen by the client instead of by the server.

On successful update, return 200 (or 204 if not returning any content in the body) from a PUT. If using PUT for creates, return HTTP status 201 on successful creation. PUT is not a safe operation, it modifies (or creates) state on the server, but it is idempotent.

**POST:**

The POST verb is used often for creation of new resources, and it's used to create subordinate resource which belongs to other parent resource. On successful creation return HTTP status 201, returning a location header with a link to the newly created resource with the 201 HTTP status.

If the client knows what the resulting URI (or resource ID) will be , use PUT at that URI , otherwise , use post when the server or service is in charge of deciding the URI for the newly , created resource .

**DELETE:**

It is used to delete a resource identified by a URI. On successful deletion, return HTTP status 200 (ok) a long with a response body, or return HTTP status 204 (NO CONTENT) with no response body.

So, we have two web services in our smart agricultural irrigation management system , also we added third one. In normal, any web service has a link, by using this link any application can call this web service from any another application. After calling this web service, we see that the

purpose of this web service is done during to its location in the system. The first web service function is to send the values of all sensors in all zones and save these values in sensor readings table in data base. We make this service to generate some of random numbers as we want and send them to the sensor node readings table to save them. We use random numbers now until the real readings of the sensors become ready.

In the first web service sensor readings, we use PUT method to update the data and insert these data to the sensor node readings table, ( Node\_Id, Sensor\_Id, Description, Reading\_time, Reading, Sensor\_Node\_Id).

Finally, the link of the first web service is:

*<http://localhost:7101/AgriCultureApplication-RESTWebService-context-root/resources/restwebservice/sensorReadings>*

The second web service of our thesis is to generate a list of zones and their requirements of water," update zone moisture", which use PUT method also, so this list has the zone number and his status: On or Off, I.e. this zone need to open water valve on it or not.

This service compares the best values of soil moisture with the average reading of soil moisture from the sensors, if the average readings less than optimum range, then the zone need to irrigated, then the valve must have opened.

So, this web service checks reading average within time, by get date and age from cultivation plant table, and get soil id from zone table, and get plant\_id, get min/max values from best values based on plant id, soil id, and age, then compare it with best values from best values table. If it is less



than minimum moisture values from system best values, turn it on else return it off.

The link of this second web service is:

<http://localhost:7101/AgriCultureApplication-RESTWebService-context-root/resources/restwebservice/updateZoneMoisture>

By using first web service which send all readings of all sensors in the whole zones to database, then by using these readings some calculations done to determine which zone need to be irrigated or not, all these calculation and zones moisture list can be appear in the sample code (A.4):

The additional third web service we used in this thesis is Get List Hierarchy which use GET method, this web service generates a list of JSON objects. This list first elements level is zones, and inside each zone there is a list of child list, which is either node, or sensor.

We have to add that inside the node there is another level of sensors.

The link of the third web service is:

<http://localhost:7101/AgriCultureApplication-RESTWebService-context-root/resources/restwebservice/GetListHierarchy>

#### **4.5.2. Benefits of web services**

- Exposing the function on the network:  
a web service is a managed code that can be remotely invoked using HTTP. So, it can be activated by requests from service requestors, once the web service is exposed on the network, all other applications can use it remotely.

- **Interoperability:**  
using web services, various applications can talk to each other, and share different data among themselves. so, for example, Java web service can talk to VB or .NET and vice versa.
- **Standardized Protocol:**  
For the communication, web services use standardized industry protocol, these layers: service transport layer, XML messaging layer, service description layer, and service discovery layer use well defined protocols in the web service, so this standardization of protocol gives the business many advantages, such as reducing cost, increasing quality, and wide range of choices.
- **Low cost communication:**  
we can use low cost internet for implementing web services.
- **Code re-use:**  
multi uses of the code of web services can be done in different fields.

#### **4.6. Zones and WSN**

The whole field is divided into different regions called zones, every zone has its own specification, and own electric valve which has own name belong to its zone. Every valve opened and closed upon the moisture calculated by the web service which produce a list of zones which is needed to irrigate or not.

Wireless Sensor Network consist of a large number of sensor nodes spread among different zones in our field, each node monitors multiple environment parameters in its region, these nodes collect different data

from their real world, and by a web service these collected data moved to some table in our data base.

These data become large and large, so cloud computing offers a scalable solution to manage, process, and share these large amount of data.

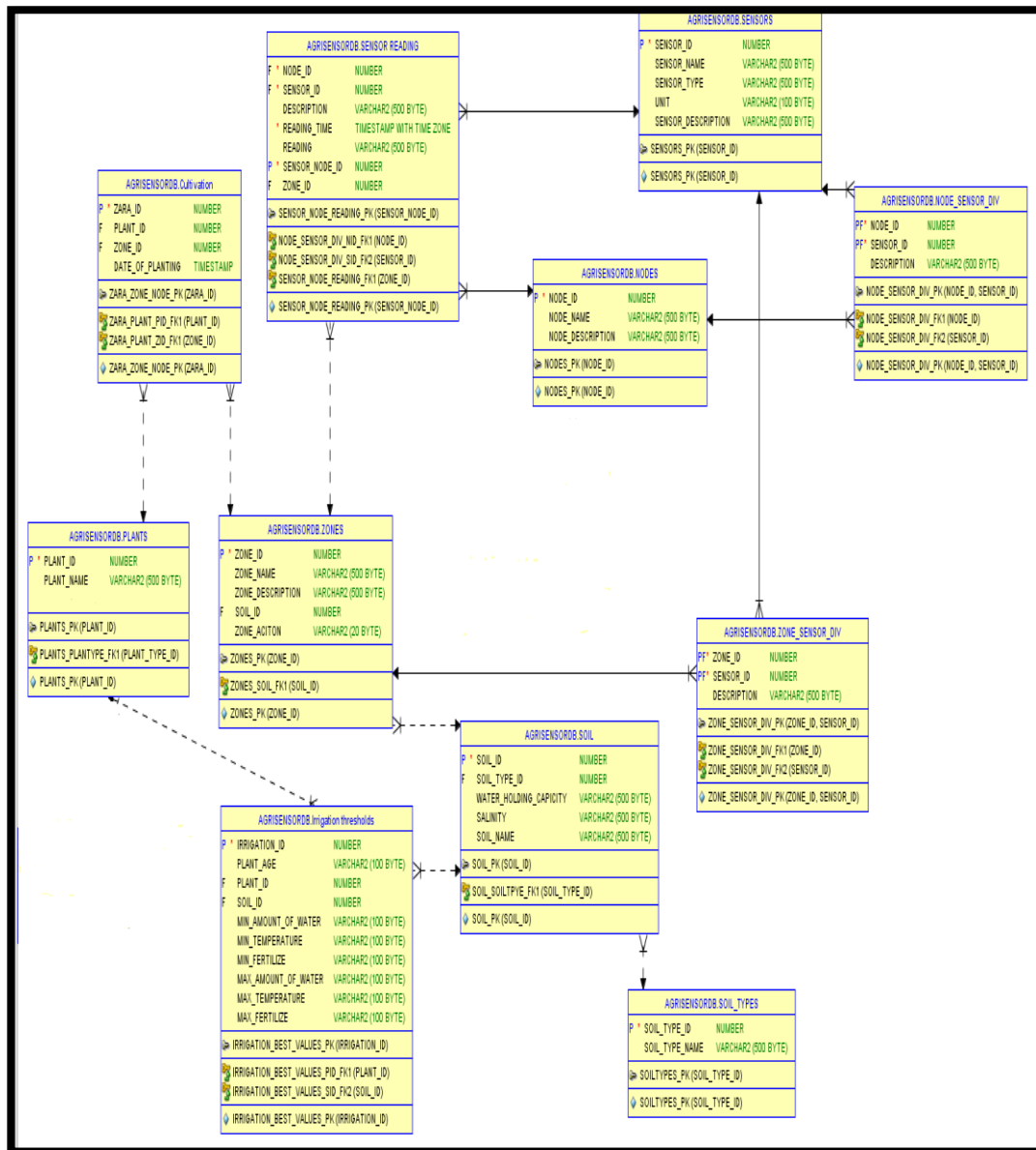
#### **4.7. Login Page Implementation**

In Cloud Application, different user's login to the system. In the login page, system need username and password for authentication, every user has different privileges which are decided through the system about the function of every user.

The implementation of the login page appears in the code in Appendix (A.5).

#### **4.8. Database Implementation**

Now, we will discuss all these attributes and entities in our database. We have in our Smart Agricultural Irrigation Management System eleven tables (entities) which are: Irrigation Thresholds, Plants, Cultivation, Zones, Soil, Soil Types, Nodes, Node Sensor Div, Sensors, Readings, and Zone Sensor Div as illustrated in the Figure ( 4.2) and described below:



**Fig. (4.2):** Database Tables

Irrigation thresholds table, only administrator and agricultural expert can access this table. The agricultural expert can fill this table with the thresholds values of the upper and lower moisture levels in the root zone for each kind of plant and soil, and the age of plants in weeks. This table has the following attributes: Irrigation Id, Plant Age, Plant Id, Soil Id, Min Amount of Water, Max Amount of Water, Min Temperature, Max

Temperature, Min Fertilizer, and Max Fertilizer. This table is connected with two tables: plants and soil tables.

The Soil table: Also, this table is filled by agricultural experts. It contains information about the characteristics of the soil and its specifications such as water holding capacity and salinity. The attributes of soil table are: Soil Id, Soil Name, Soil Type Id, Water Holding Capacity, and Salinity. This table connected with Soil Types table which define the soil types, and has these two attributes: Soil Type Id, and Soil Type Name. Soil table connected also with zones table.

The Plants table: the plants table connected with two tables: irrigation thresholds, and cultivation, The Plants table has a list of plants which filled also by an experts of farming, so the attributes of this table are: Plant Id, and Plant Name.

The Cultivation table: The farmer can fill this table. This table contains information about a cultivation. This information includes, the cultivation plant from the plant table and the cultivation date. The attributes of this table are: Cultivation Id, Plant Id, Zone Id, and Date of Planting. This table also connected with plants table and zones table.

The Zones table: A land zones is a piece of land that has a uniform physical characteristics, such as soil type and topography. The zones table holds the zones ids and each zone name. The zones table is connected with soil table, cultivation table, sensor reading, and zone sensor div. The attributes of this table are: Zone Id, Zone Name, Zone Description, Soil Id, and zone Action.

The Nodes table: defines the nodes spread among all the zones, every node manages defined number of sensors. The Nodes table consists of these attributes: Node Id, Node Name, and Node Description. The nodes table connected with two tables: sensor reading, and node sensor div.

The Node Sensor Div table: which connect the nodes table with the sensors table, and have these attributes: Node Id, Sensor Id, and Description. By this table we can connect any sensor to any node.

Then the Sensors table: which defines the sensors and their types and the unit of every sensor value, this table has the following attributes: Sensor Id, Sensor Name, Sensor Type, Unit, and Sensor Description. This table was connected with node sensor div table, zone sensor div table, and sensor reading table.

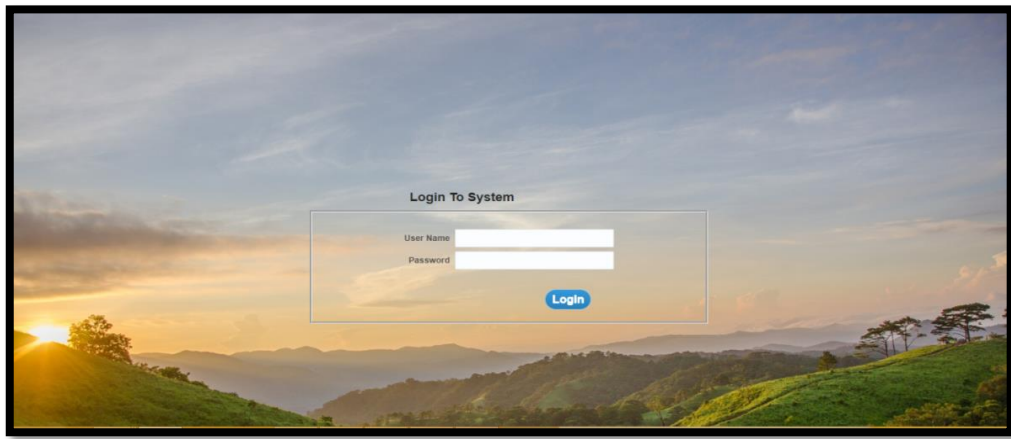
The Sensor Reading table: which save all the readings of all the sensors in all zones, and the time stamp of every reading, so the attributes of this table are: Node Id, Sensor Id, Description, Reading Time, Reading, Sensor Node Id, and Zone Id. This reading table connected with sensors table, nodes table, and zones table.

Finally, Zone Sensor Div table: which connect zones table with sensors table, by this table we can connect any sensor in wireless sensor network to any zone in our field, this table also has the following attributes: Zone Id, Sensor Id, and Description.

## Chapter Five

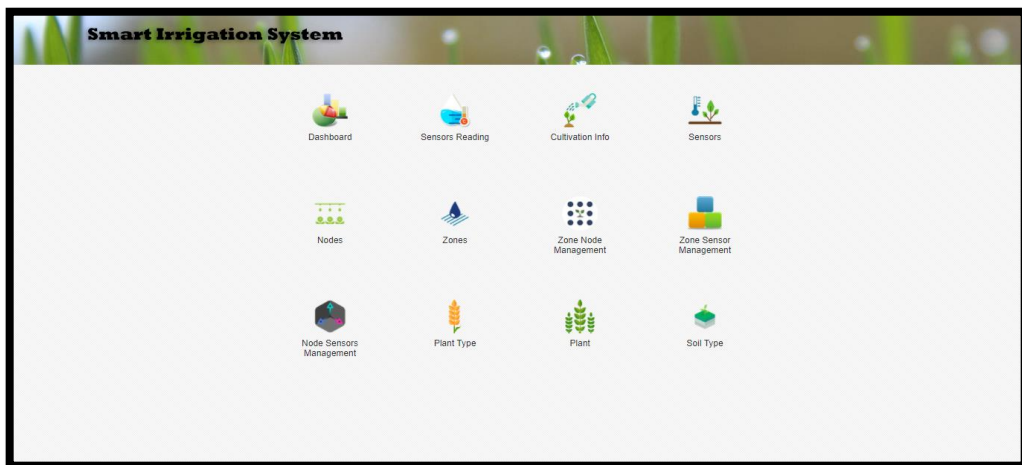
### System Evaluation

Finally, I have presented the last interface of our project, Smart Irrigation Management System, this interface allow the users control all the farm as their allowed permissions. First of all, figure (5.1) appear the login page to the project which include three types of users: administrator, expert of farming, and farmer:



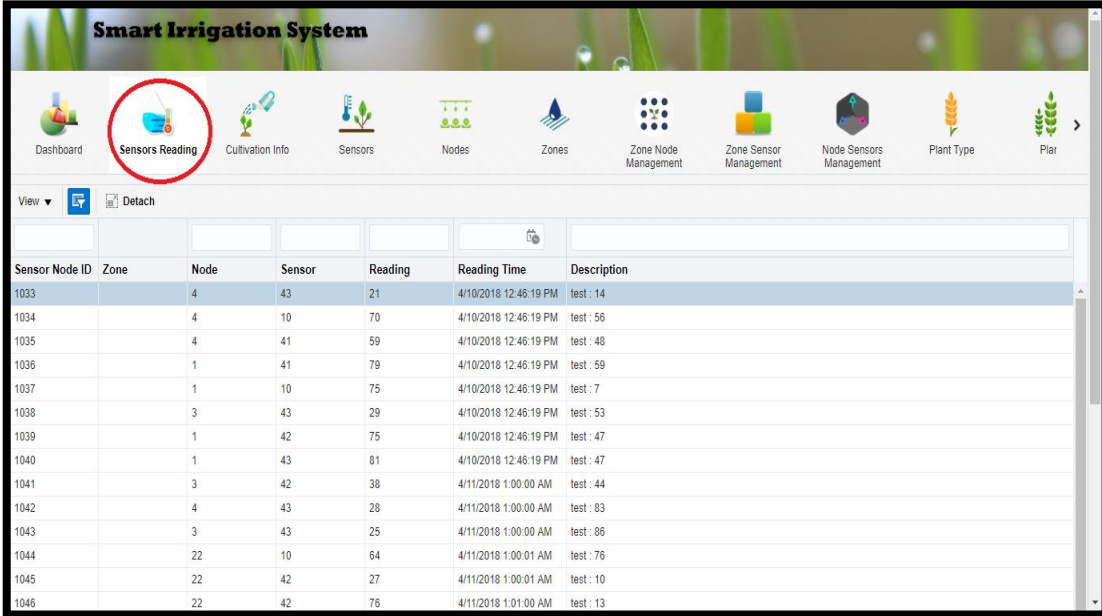
**Figure (5.1) : Login Page**

The following figure (5.2) is the home page of the project which appear after the login of administrator user who has all the privileges:



**Figure (5.2) : Home Page of the Project**

By clicking on sensors reading icon in the homepage of our project, it appears a list of readings belongs to the sensors in the zones that transferred automatically to this table by the web service we used to transfer collecting data. This table show the zone, node, sensors , readings, readings time, and description as figure (5.3):

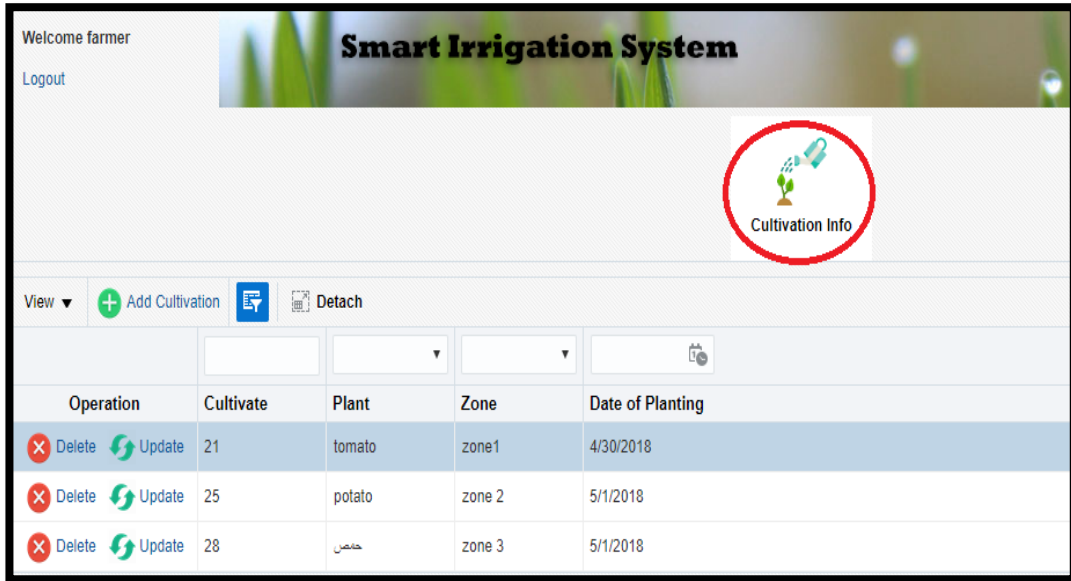


Sensor Node ID	Zone	Node	Sensor	Reading	Reading Time	Description
1033		4	43	21	4/10/2018 12:46:19 PM	test : 14
1034		4	10	70	4/10/2018 12:46:19 PM	test : 56
1035		4	41	59	4/10/2018 12:46:19 PM	test : 48
1036		1	41	79	4/10/2018 12:46:19 PM	test : 59
1037		1	10	75	4/10/2018 12:46:19 PM	test : 7
1038		3	43	29	4/10/2018 12:46:19 PM	test : 53
1039		1	42	75	4/10/2018 12:46:19 PM	test : 47
1040		1	43	81	4/10/2018 12:46:19 PM	test : 47
1041		3	42	38	4/11/2018 1:00:00 AM	test : 44
1042		4	43	28	4/11/2018 1:00:00 AM	test : 83
1043		3	43	25	4/11/2018 1:00:00 AM	test : 86
1044		22	10	64	4/11/2018 1:00:01 AM	test : 76
1045		22	42	27	4/11/2018 1:00:01 AM	test : 10
1046		22	42	76	4/11/2018 1:01:00 AM	test : 13

**Figure (5.3) : Sensors Readings**

Now, we suppose that our farm is divided into four zones and we have two nodes among these four zones, and have a lot of sensors spread among these zones and belongs to the two nodes.

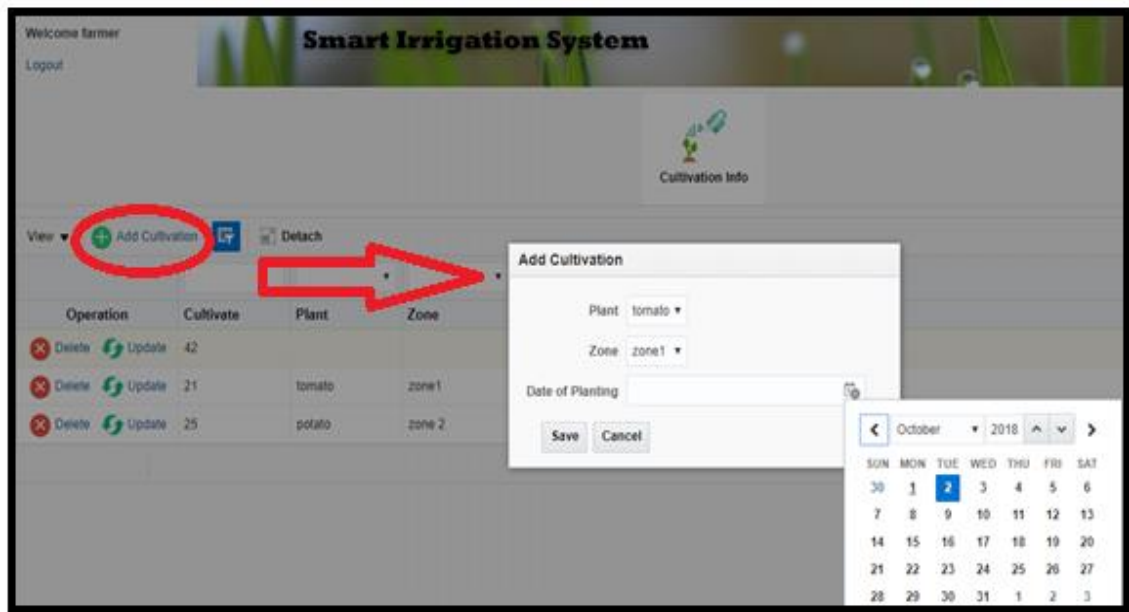




**Figure (5.4) : Cultivation Information**

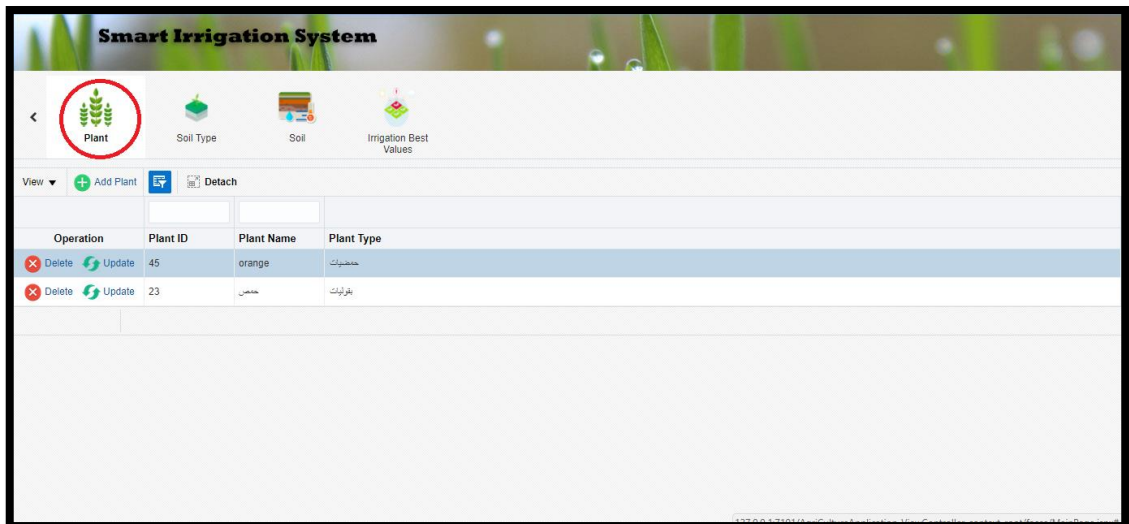
First of all, the farmer decided what he want to plant, So, when the farmer login to the system by his username and password which provided by the administrator, he can only see the cultivation info icon which has the cultivation and the zone that it planted in, and the date when the farmer planted this cultivation, as the following figure (5.4)

In this part we suppose that the farmer want to plant tomato cultivation, so by clicking on Add Cultivation bottom from the last figure, farmer can choose from a list what he want to plant which is tomato, and also can determine in which zone he want to plant, like zone 1, and also must determine the date of planting this cultivation as the following figure (5.5):



**Figure(5.5) :** Adding new cultivation

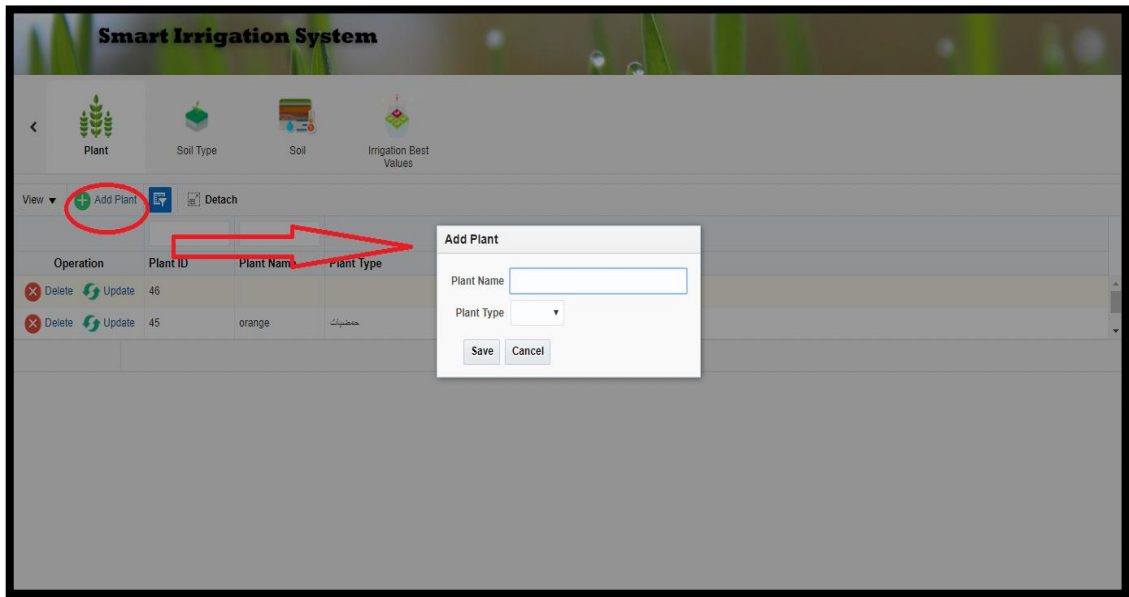
As we mention before, the farmer choose the cultivation that he want to plant from the plant table which has plant id, plant name, and plant type as the figure (5.6):



**Figure (5.6) :** Plant table

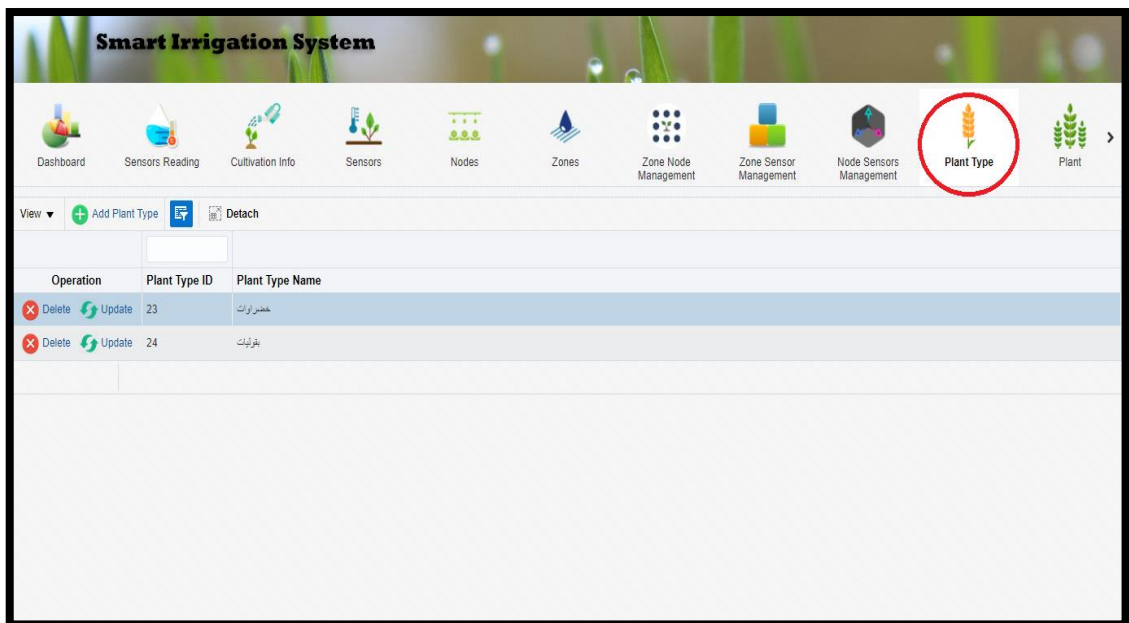
The last figure appears in the expert of farming window, not to farmer, so expert of farming can have decided another issues in this smart farming process,

he can add plant by clicking on add plant, this figure (5.7) will appear:



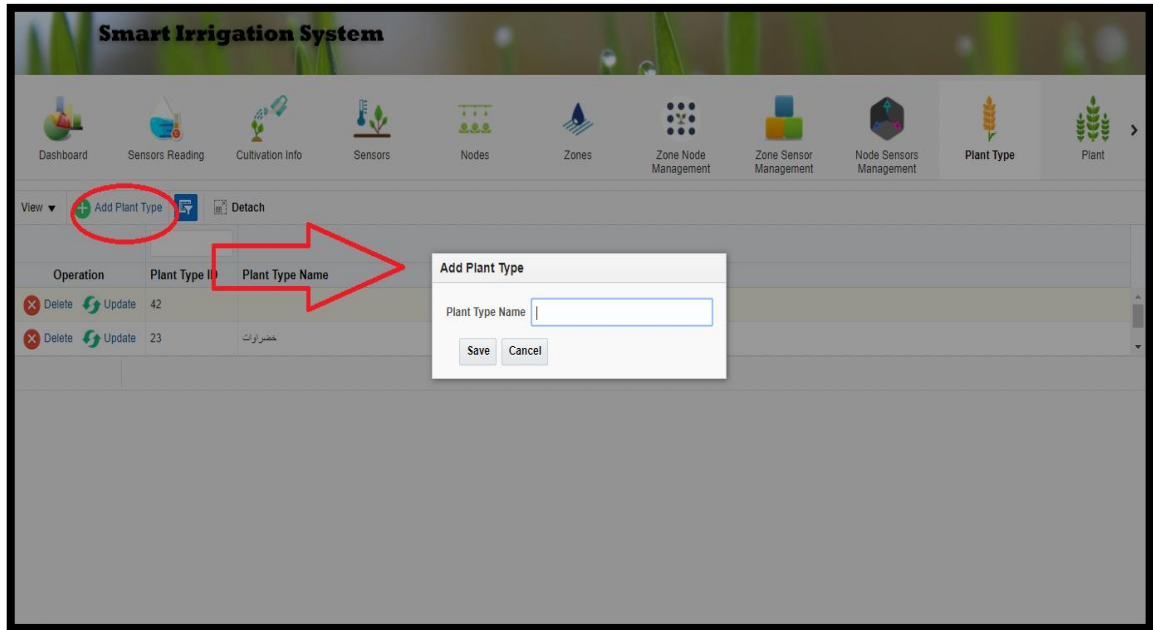
**Figure (5.7) :** Adding new plant

Also, the expert of farming can add data to plant type table, the attributes of this table are: plant type id, and plant type name, as the figure (5.8):



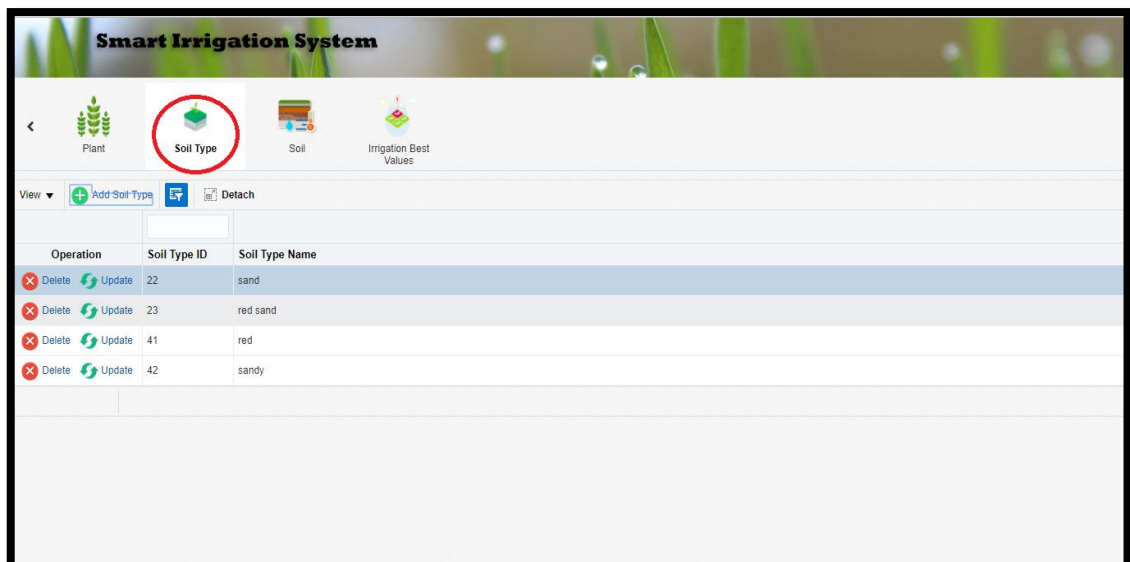
**Figure (5.8):** Plant Type

Also, expert can add plant type by clicking on Add plant type icon to add it on plant type table, like the figure (5.9):



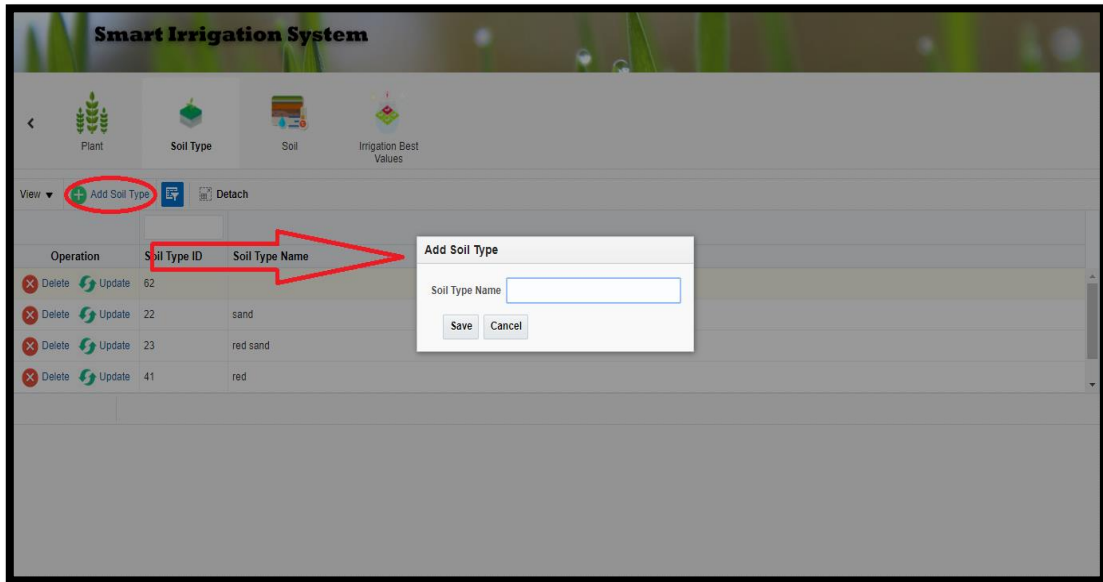
**Figure (5.9):** Adding Plant Type

Another information that the expert can determined is the soil types, and the soil specifications. So, soil type table has a list of soil type names, as this figure (5.10):



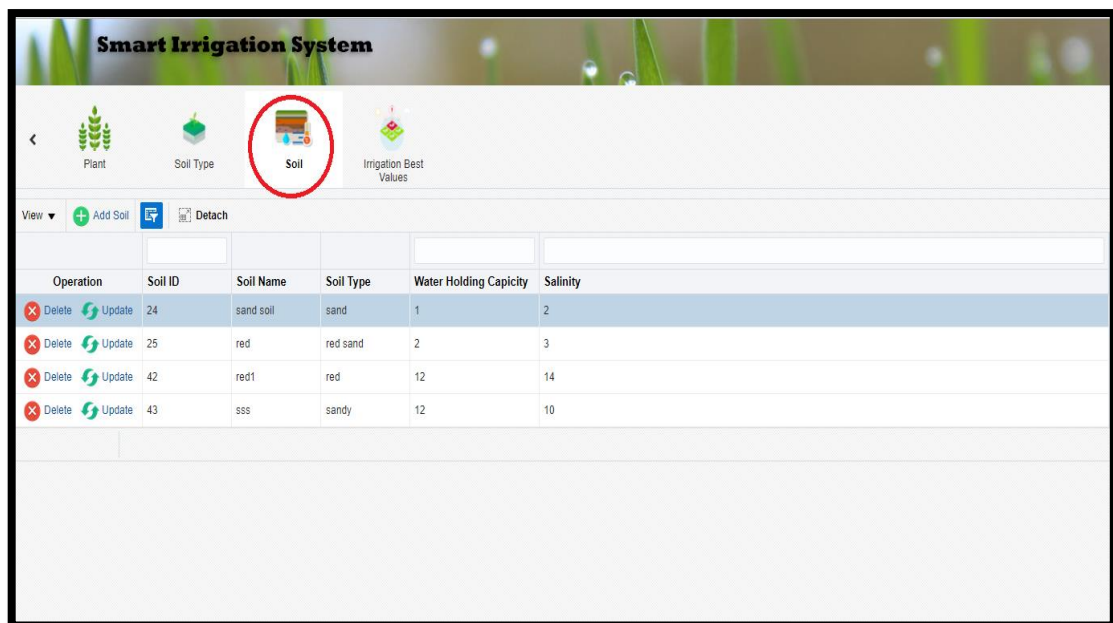
**Figure (5.10) :** Soil Types

The expert can add any type of soil by clicking on add soil types, so the figure (5.11) will appear:



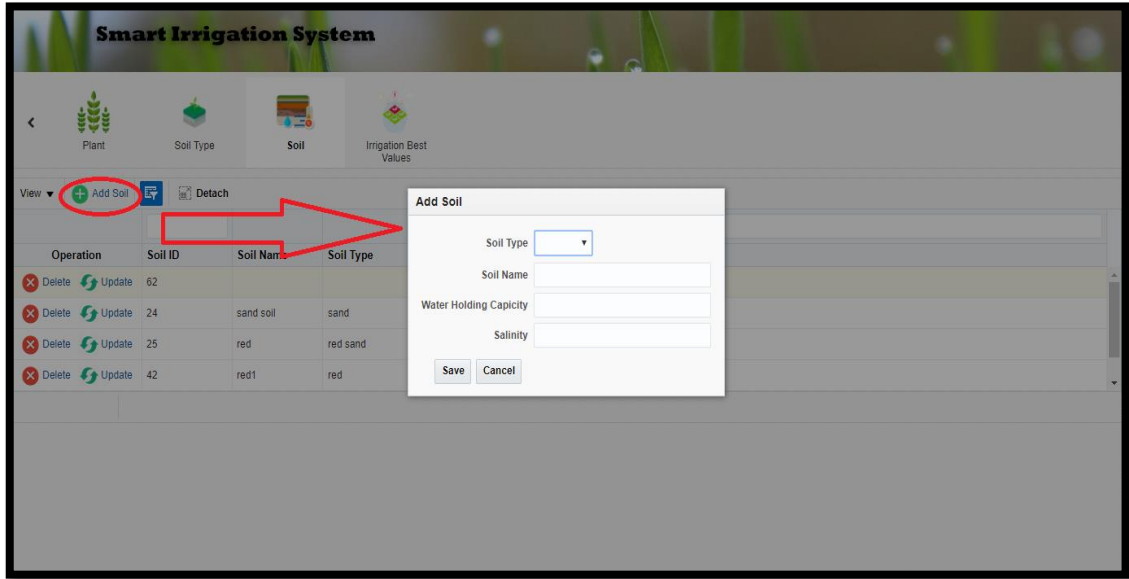
**Figure (5.11):** Adding Soil Type

Then the expert can add the soil specifications from the soil table which has soil id and name, and can choose soil type from a list, water holding capacity, and salinity as the figure (5.12):



**Figure (5.12):** Soil Specifications

When the expert of farming click on add soil, the figure (5.13) will appear:



**Figure (5.13):** Adding Soil Specifications

The last thing of expert function is irrigation best values which has the best range values of moisture, temperature and fertilizer for every age period of plants as the following figure (5.14):

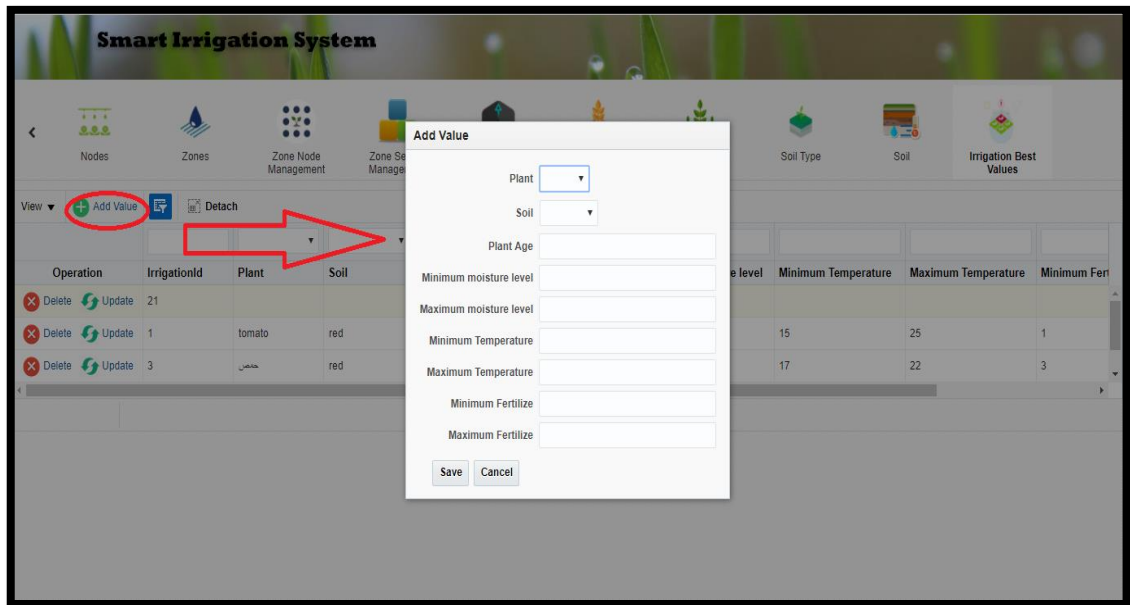
The screenshot shows the 'Smart Irrigation System' interface with the 'Irrigation Best Values' icon circled in red. Below the navigation bar, there's a table with columns for Operation, IrrigationId, Plant, Soil, Plant Age, Minimum moisture level, Maximum moisture level, Minimum Temperature, Maximum Temperature, and Minimum Fertilizer. The table contains three rows of data for tomato, cucumber, and potato.

Operation	IrrigationId	Plant	Soil	Plant Age	Minimum moisture level	Maximum moisture level	Minimum Temperature	Maximum Temperature	Minimum Fertilizer
Delete Update	1	tomato	red	1	10	50	15	25	1
Delete Update	3	خوخ	red	2	30	50	17	22	3
Delete Update	4	potato	red1	3	12	41	14	26	1

**Figure (5.14):** List of Irrigation Best Values

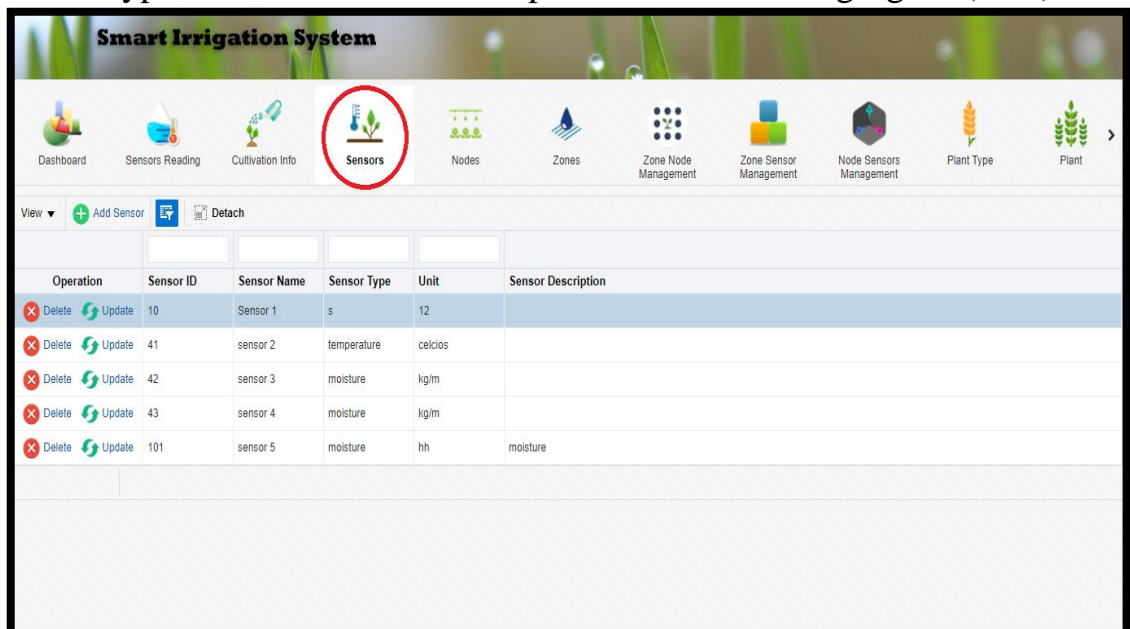
So the expert of farming can fill this table by clicking on add values as the following figure (5.15):





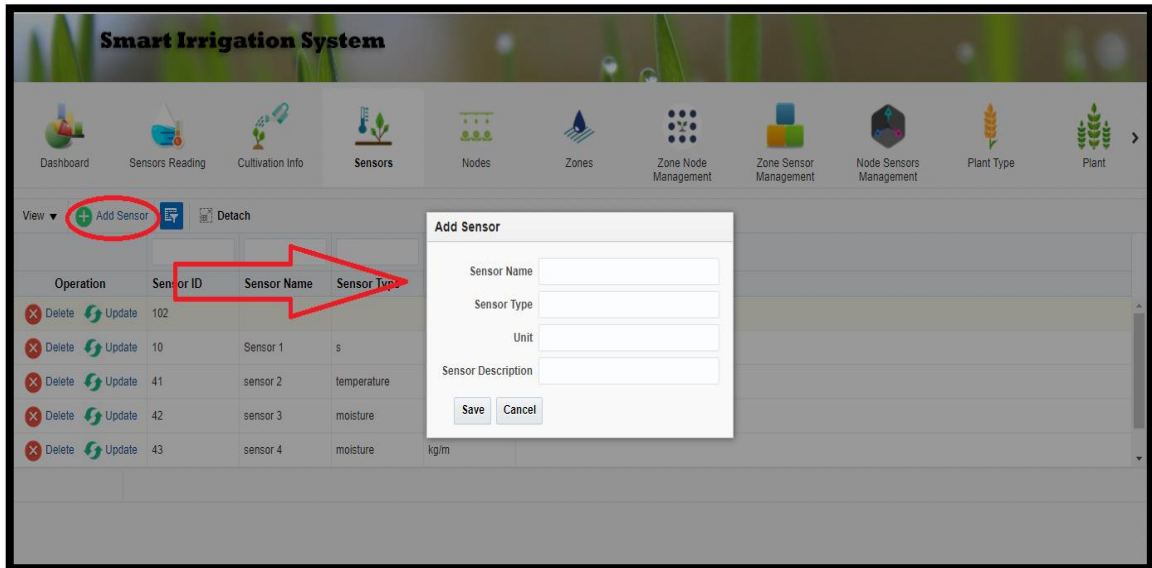
**Figure (5.15):** Adding New Best Value

Now, we arrive to administrator functions, so the first thing is sensors, the administrator has full control to sensors, show, add, modify, and delete. So the following icon of the interface is sensors which has sensor id and name, sensor type, unit, and sensor description as the following figure (5.16):



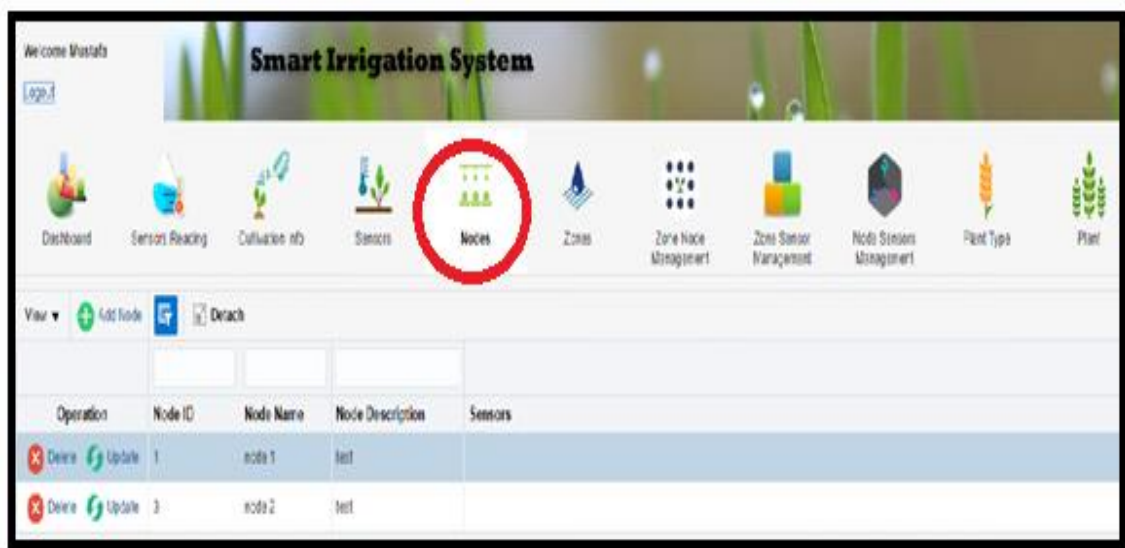
**Figure (5.16):** List of Sensors

The administrator can add any sensor by clicking on add sensor , so this figure (5.17) will appear:



**Figure (5.17):** Adding New Sensor

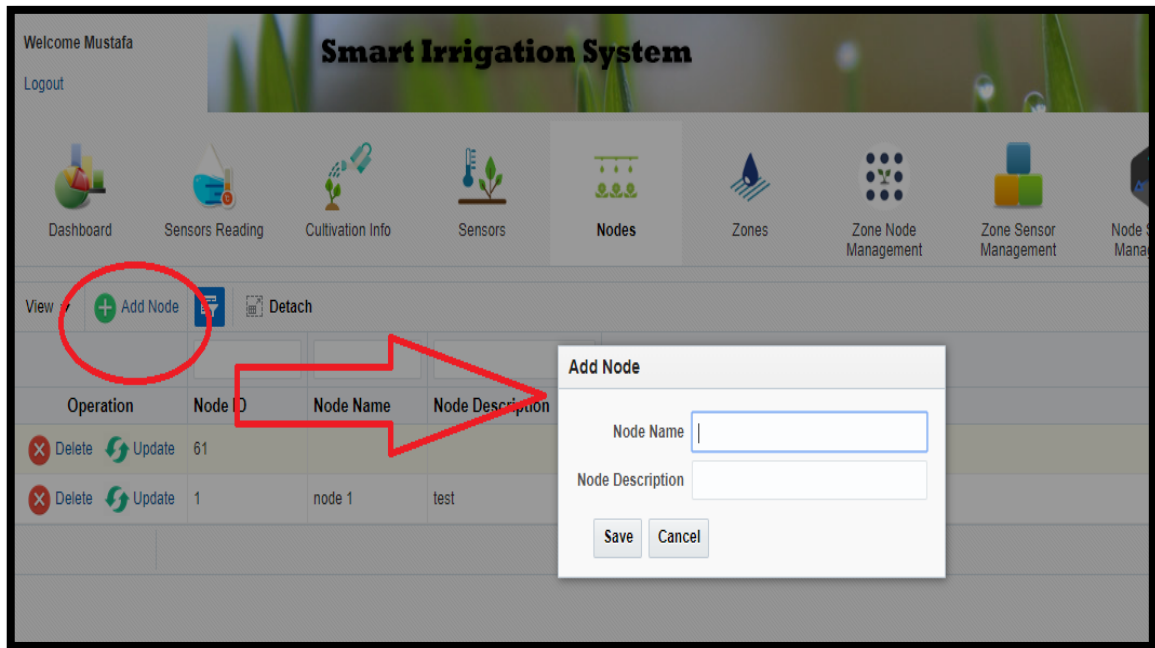
The nodes icon, which has all the nodes in all zones, and their description, as the figure (5.18):



**Figure (5.18):** List of Nodes

From the bottom: Add Node, the administrator can add any node like the figure (5.19):





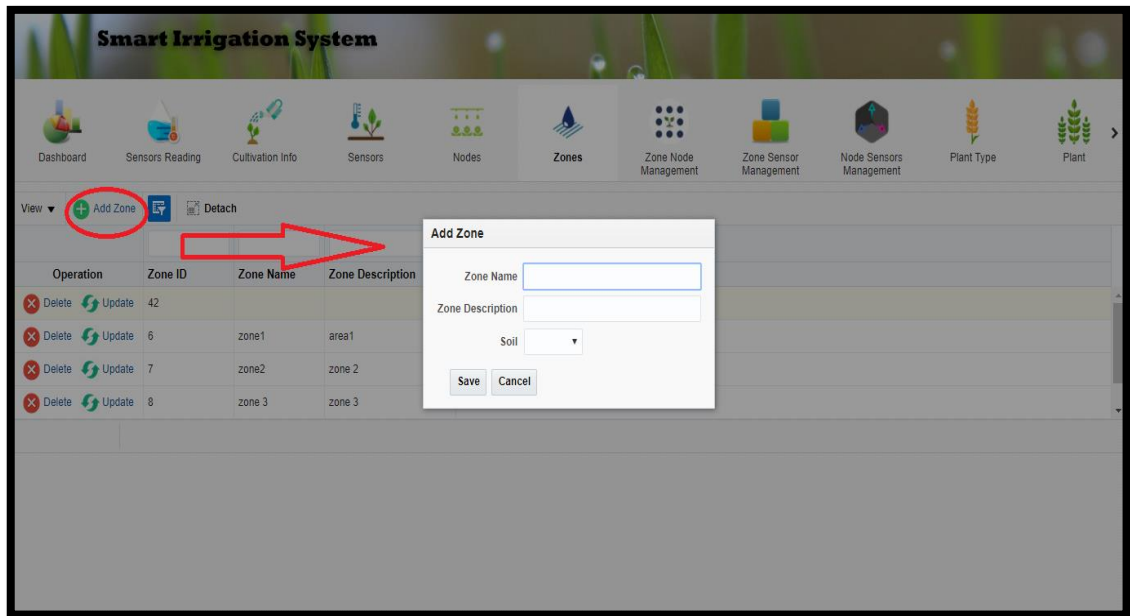
**Figure (5.19):** Adding New Node

After that the zones icon, which has these attributes: zone id, zone name, zone description, and soil as the following figure (5.20) :



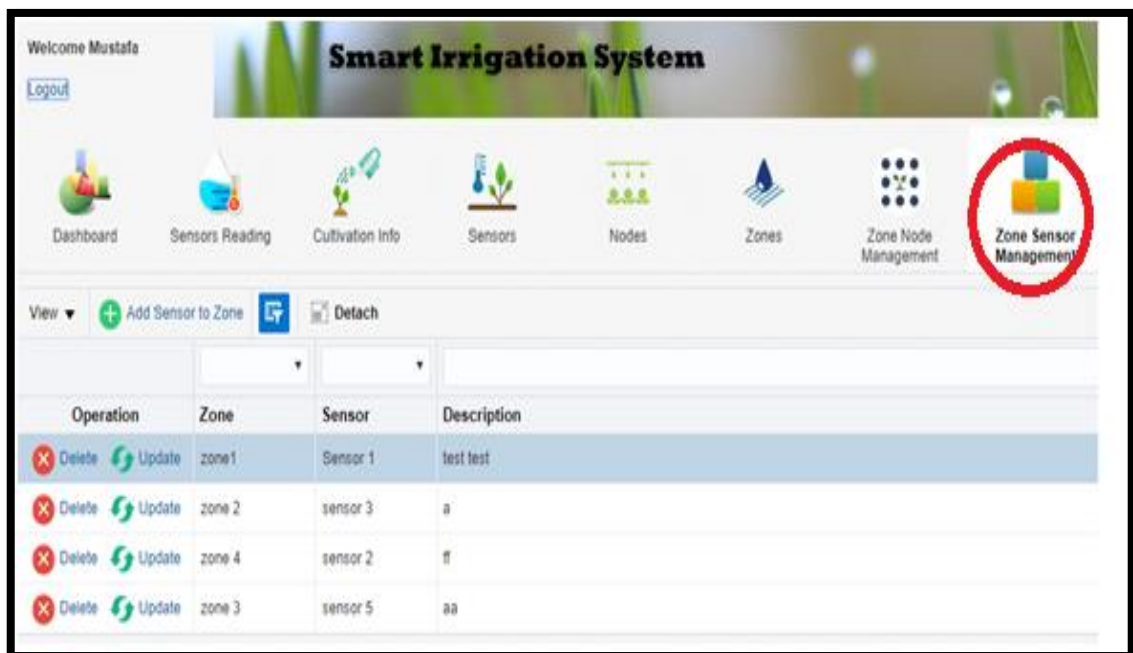
**Figure (5.20) :** List of Zones

Then from add zone, the administrator can add any zone and can choose the type of soil in that zone as the figure (5.21):



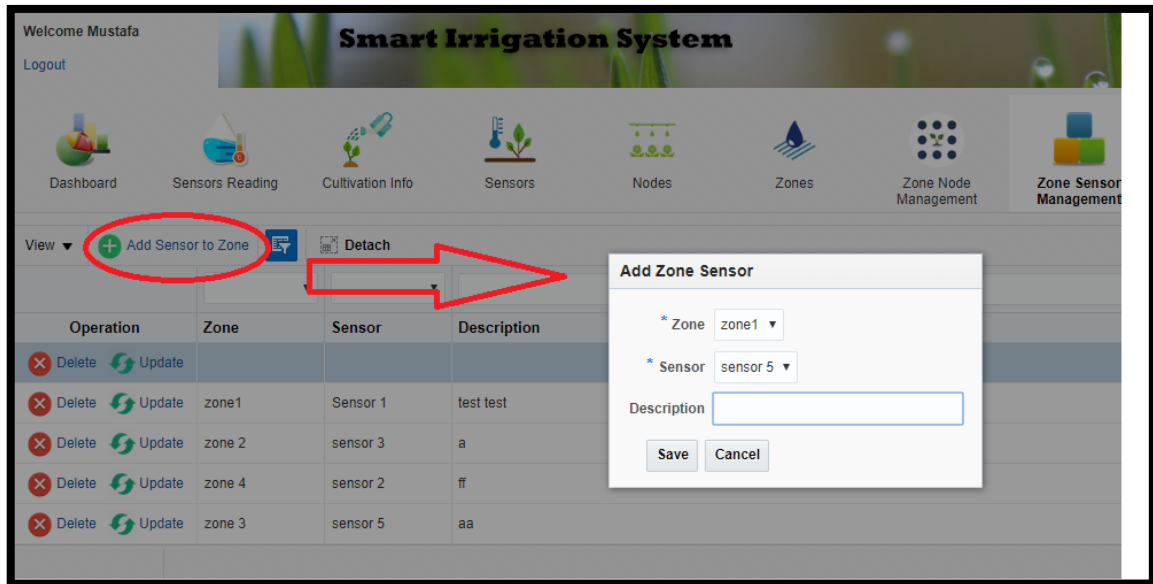
**Figure (5.21): Adding New Zone**

After that, to connect the sensors with the zones, there is a zone sensor management icon, which has a zone, sensor, and a description, as the following figure (5.22):



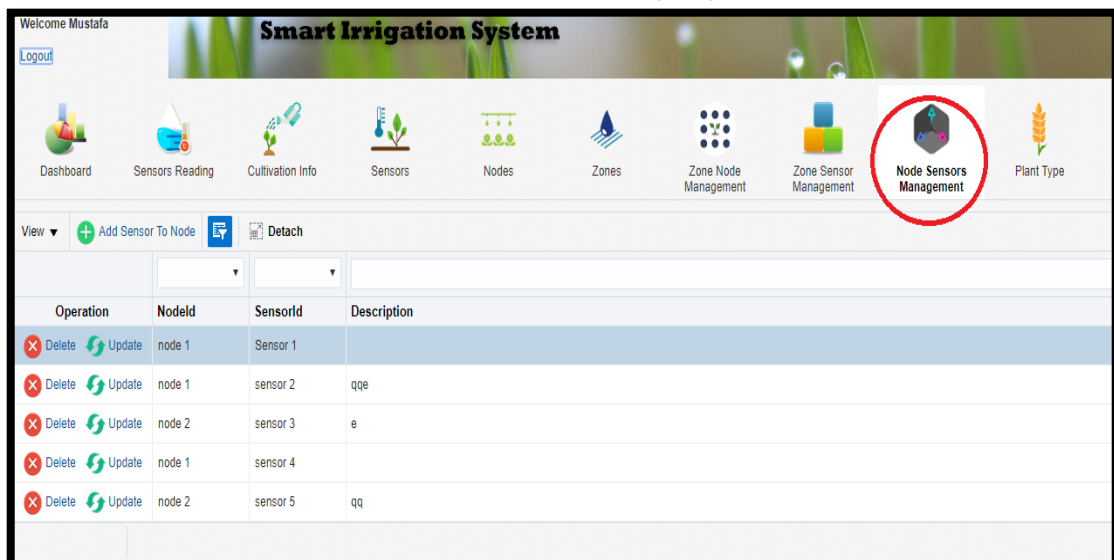
**Figure (5.22) : Zone Sensor Management**

In this management, we can connect any sensor to any zone we want by clicking on Add sensor to zone to show this figure (5.23):



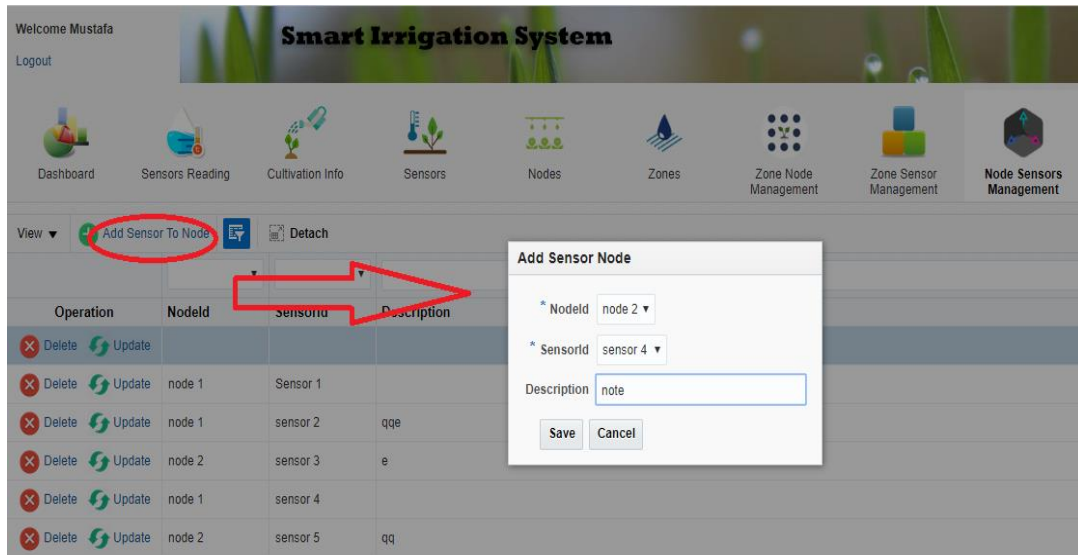
**Figure (5.23):** Connect Sensors to Zones

The last management icon is node sensor management which connect the sensors to determined nodes as the following figure (5.24):



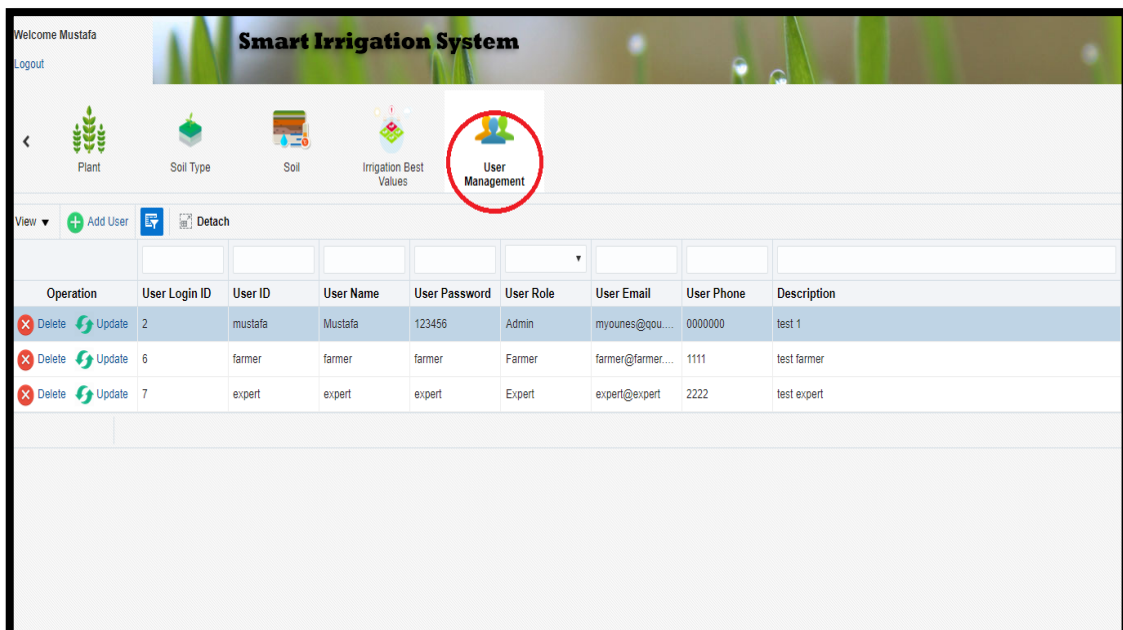
**Figure (5.24):** Node Sensor Management

Then, we can add any sensor to any node by clicking on add sensor to node, like the figure (5.25):



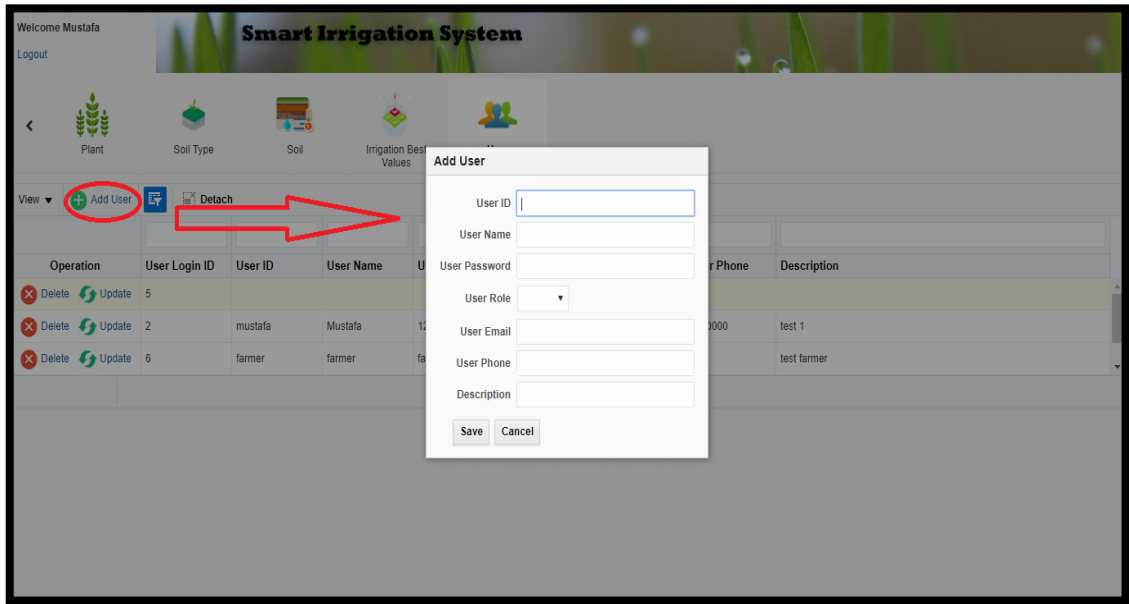
**Figure (5.25):** Connect Sensor to Node

Then we can see the user management icon, that we can add, modify, or delete any user using this icon, and also we can choose the role that the user need to perform, as the following figure (5.26):



**Figure (5.26) :** User Management Icon

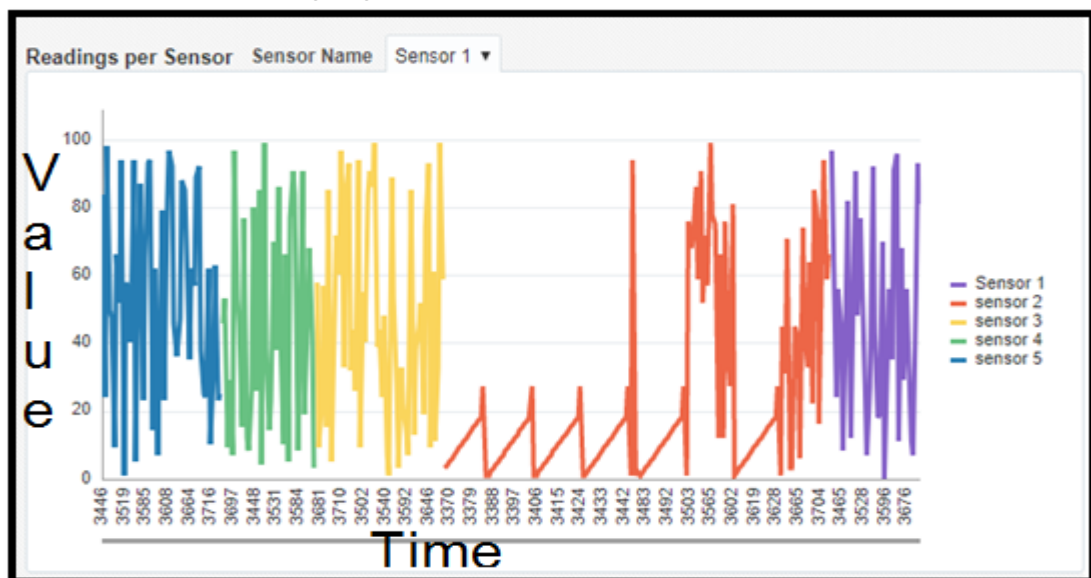
so the administrator can add any user with his own privileges and specifications by this figure (5.27):



**Figure (5.27):** Adding New User

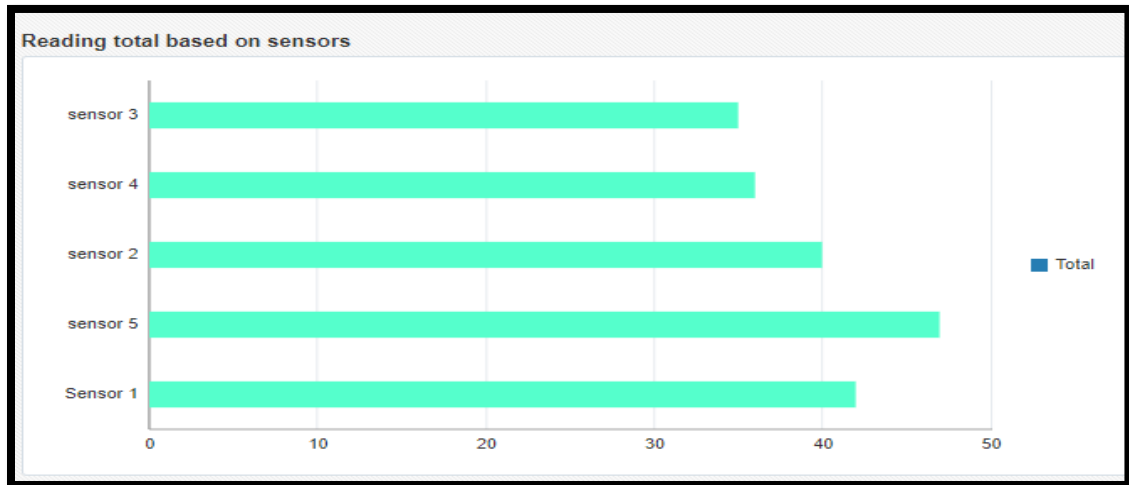
The last thing is the dashboard, which gives us some figures that illustrate all things in our farm and monitor everything in the farm.

The following figure show us all the range of readings for every sensor, as shown in the following figure:



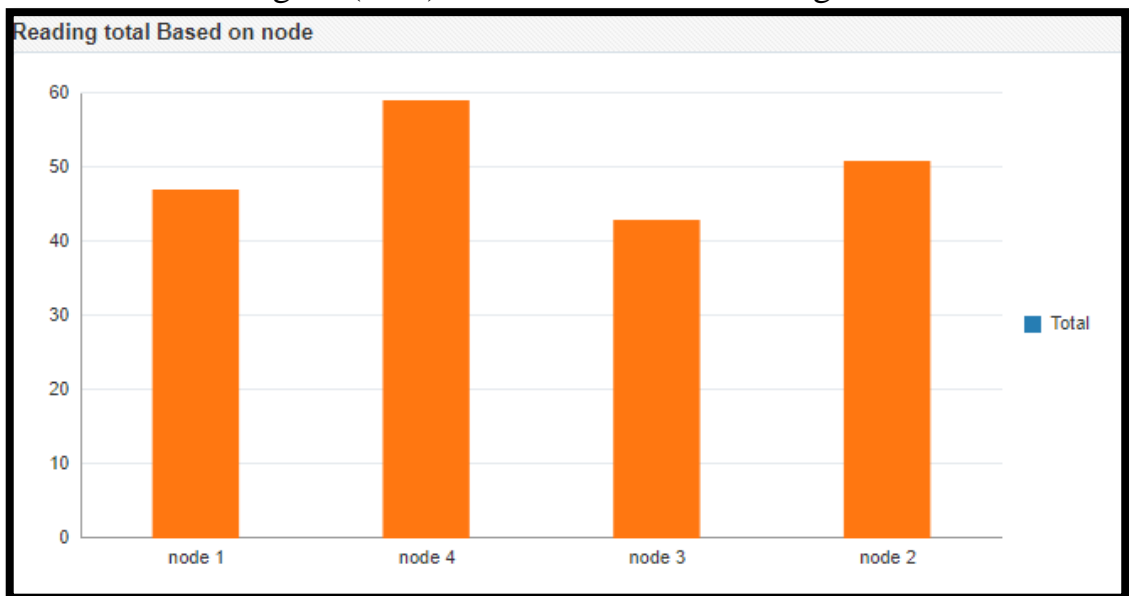
**Fig.(5.28):** Sensors Readings

The following figure appear the total readings for each sensors, as this figure (5.29):



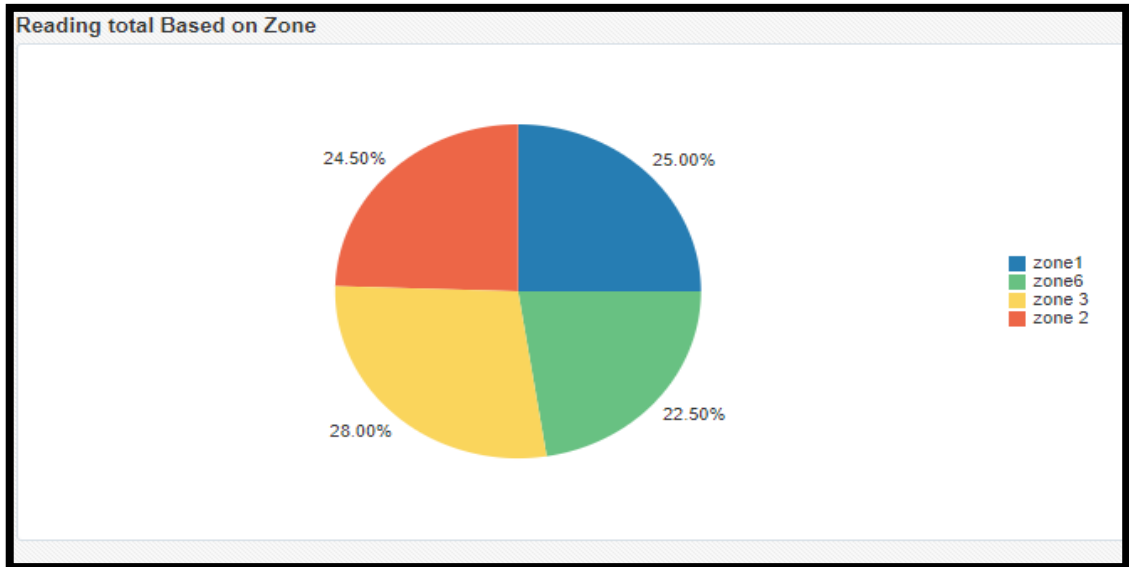
**Figure(5.29):** Total Readings based on Sensors

Then we have a figure (5.30) that show us total readings based on nodes:



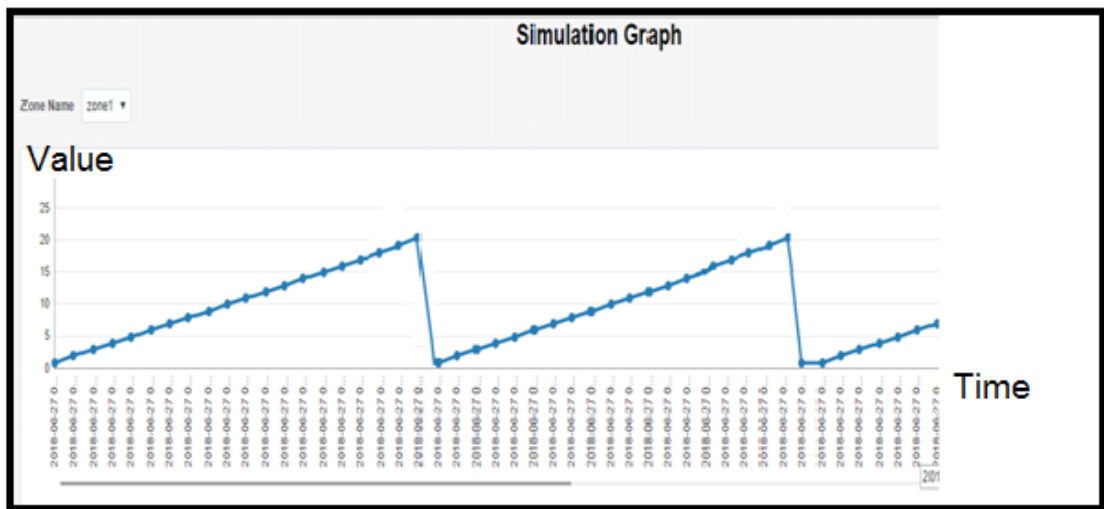
**Figure (5.30):** Total Readings based on Nodes

The last figure of dashboard figures illustrate total readings based on zones, which appear in the figure (5.31):



**Figure (5.31) :** Total Readings based on Zones

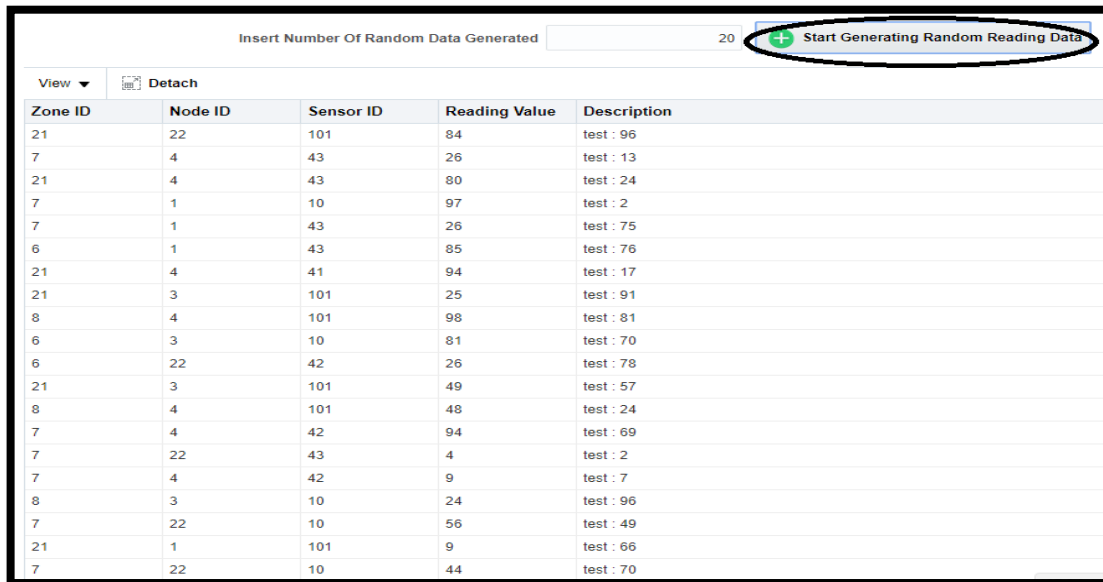
The moisture level of any zone depends on the amount of water added and the time when the zone was irrigated, and the type of the soil in the zone.



**Figure (5.32):** Moisture Level of one zone

From the last figure (5.32), when the moisture level goes out the optimum range, then the valve opened and irrigated that zone until the moisture level return to the optimum, then the irrigation process stopped and so on.

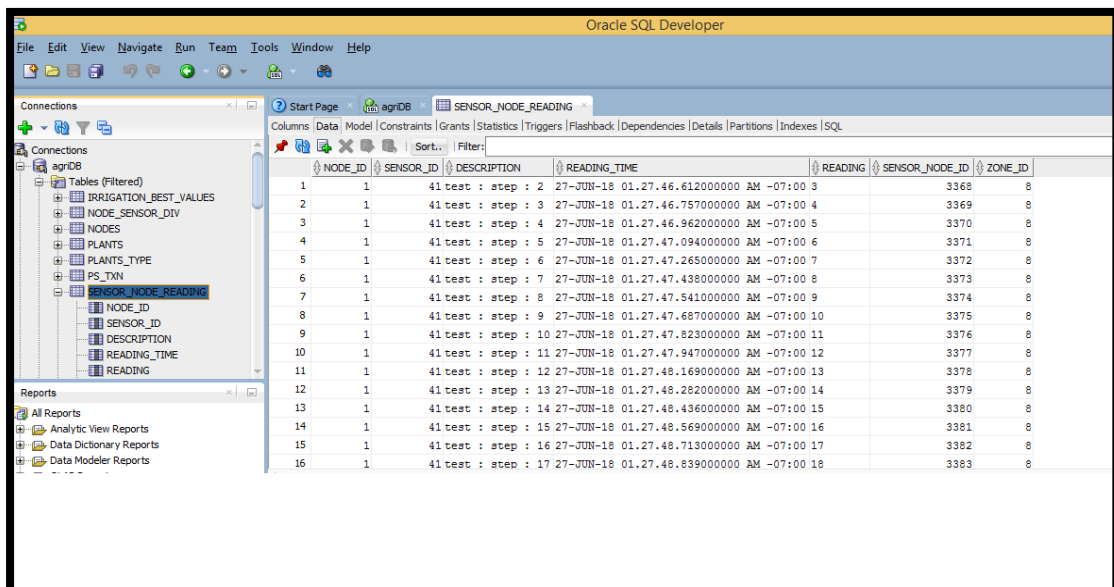
Now, we will see the figures of web services evaluation. In the first web service we use random numbers to write them in the sensor readings table in the database as readings of the sensor like this figure (5.33):



Zone ID	Node ID	Sensor ID	Reading Value	Description
21	22	101	84	test : 96
7	4	43	26	test : 13
21	4	43	80	test : 24
7	1	10	97	test : 2
7	1	43	26	test : 75
6	1	43	85	test : 76
21	4	41	94	test : 17
21	3	101	25	test : 91
8	4	101	98	test : 81
6	3	10	81	test : 70
6	22	42	26	test : 78
21	3	101	49	test : 57
8	4	101	48	test : 24
7	4	42	94	test : 69
7	22	43	4	test : 2
7	4	42	9	test : 7
8	3	10	24	test : 96
7	22	10	56	test : 49
21	1	101	9	test : 66
7	22	10	44	test : 70

**Figure (5.33) : Generate Random Sensor Readings**

The sensor readings table become like this figure (5.34):



NODE_ID	SENSOR_ID	DESCRIPTION	READING_TIME	READING	SENSOR_NODE_ID	ZONE_ID
1	1	41 test : step : 2	27-JUN-18 01.27.46.612000000 AM -07:00 3		3368	8
2	1	41 test : step : 3	27-JUN-18 01.27.46.757000000 AM -07:00 4		3369	8
3	1	41 test : step : 4	27-JUN-18 01.27.46.962000000 AM -07:00 5		3370	8
4	1	41 test : step : 5	27-JUN-18 01.27.47.094000000 AM -07:00 6		3371	8
5	1	41 test : step : 6	27-JUN-18 01.27.47.265000000 AM -07:00 7		3372	8
6	1	41 test : step : 7	27-JUN-18 01.27.47.438000000 AM -07:00 8		3373	8
7	1	41 test : step : 8	27-JUN-18 01.27.47.541000000 AM -07:00 9		3374	8
8	1	41 test : step : 9	27-JUN-18 01.27.47.687000000 AM -07:00 10		3375	8
9	1	41 test : step : 10	27-JUN-18 01.27.47.823000000 AM -07:00 11		3376	8
10	1	41 test : step : 11	27-JUN-18 01.27.47.947000000 AM -07:00 12		3377	8
11	1	41 test : step : 12	27-JUN-18 01.27.48.169000000 AM -07:00 13		3378	8
12	1	41 test : step : 13	27-JUN-18 01.27.48.282000000 AM -07:00 14		3379	8
13	1	41 test : step : 14	27-JUN-18 01.27.48.436000000 AM -07:00 15		3380	8
14	1	41 test : step : 15	27-JUN-18 01.27.48.569000000 AM -07:00 16		3381	8
15	1	41 test : step : 16	27-JUN-18 01.27.48.713000000 AM -07:00 17		3382	8
16	1	41 test : step : 17	27-JUN-18 01.27.48.839000000 AM -07:00 18		3383	8

**Figure (5.34) : Sensor Reading Table**



The result of second web service " Update Zone Moisture" become like this figure (5.35):



**Figure (5.35) :** List of Zones and their requirements of water

Finally, in the third web service " Get List Hierarchy", the JSON object of this web service appear like this figure (5.36):



**Figure (5.36) :** List of JSON Objects

## **Chapter Five**

### **Conclusion**

In this thesis we provided design and implementation of Cloud-based application manage data coming from the sensors in Wireless Sensor Network located in the field, and information supplied by agricultural experts from some different forms in order to create smart farming and water management system. And provided communication interfaces to trigger actuation in the field.

## References

- [1] **The Applied Research Institute - Jerusalem ( ARIJ), " *Palestinian Agricultural Production and Marketing between Reality and Challenges* ", March 2015.**
- [2] **Kiminori Matsuyama," *Agricultural productivity, comparative advantage, and economic growth* ", Volume 58, Issue 2, Pages 317-334, December 1992.**
- [3] **Jabir Ali, Sushil Kumar," *Information and communication technologies (ICTs) and farmers' decision-making across the agricultural supply chain* ", Volume 31, Issue 2, Pages 149-159, April 2011.**
- [4] **Achim Walter, Robert Finger, Robert Huber, Nina Buchmann, " *Smart farming is key to developing sustainable agriculture* " PNAS , vol. 114, no. 24, PP. 6148-6150, June 13, 2017.**
- [5] **Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, Marimuthu Palaniswami, " *Internet of Things (IoT): A vision, architectural elements, and future directions* ", Volume 29, Issue 7, Pages 1645-1660, September 2013.**
- [6] **I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci " *Wireless Sensor Networks: A Survey* ", Volume 38, Issue 4, PP. 393-422, 15 March 2002.**
- [7] **Qi Zhang, Lu Cheng, Raouf Boutaba, " *Cloud computing: state – of – the – art and research challenges* ", Journal of Internet Services and Applications, May 2010, Volume 1, Issue 1, PP 7-18.**

- [8] Vrishali Wagaj, Vinayak Pottigar, and Prakash Gadekar, "***Implementation of Load Balanced Data Gathering of Nodes in Wireless Sensor Network***", **International Journal of Recent and Innovation Trends in Computing and Communication**, Volume (3), Issue (8), ISSN : 2321-8169 , PP: 5122-5126, August 2015.
- [9] J. Girona, M. Mata, J. Marsal, "***Regulated deficit irrigation during the kernel-filling period and optimal irrigation rates in almond***", **Agricultural Water Management**. Vol. 75, Issue 2, PP: 152-167, July 2005.
- [10] Mohammad Valipour, "***Drainage, waterlogging, and salinity***", Volume 60, Pages 1625-1640, 2014.
- [11] Tsz Him Lo, Derek M. Heeren, Luciano Mateos, Joe D. Luck, Derrel L. Martin, "***Field Characterization of Field Capacity and Root Zone Available Water Capacity for Variable Rate Irrigation***", Vol. 33(4), PP: 559-572, 2017.
- [12] M.B. Kirkham, "**Principles of Soil and Plant Water Relations Book**", "***Chapter 10: Field Capacity, Wilting Point, Available Water, and the Nonlimiting Water Range***", December 2005.
- [13] Camille Cobb, Samuel Sudar, Nicholas Reiter, Richard Anderson, Franziska Roesner, Tadayoshi Kohno, "***Computer Security for Data Collection Technologies***", Vol. 3, PP: 1-11, 2018.
- [14] Ling Qian, Zhiguo Luo, Yujian Du, and Leitao Guo, "***Cloud Computing- An Overview***", **IEEE International Conference on Cloud Computing**, vol. 5931, PP: 626 – 631, 2009.

- [15] Ye Wint Maung Maung, Aung Aung Hein, " *Colored Petri-Nets (CPN) based Model for Web Services Composition*". **International Journal of Computer and Communication Engineering research**, Volume 2, Issue 5, PP. 169-172, September 2014.
- [16] Snehal Mumbaikar, Puja Padiya, " *Web Services Based On SOAP and REST Principles*", Vol. 3, Issue 5, May 2013.
- [17] Sana Baccar, Mohsen Rouached, " *A Web Services based Approach for Resource-Constrained Wireless Sensor Networks* ", **International Journal of Computer Science Issues**, Vol. 9, Issue 3, No. 2, PP. 63-72, May 2002.
- [18] Heike Bach, Wolfram Mauser, " *Sustainable Agriculture and Smart Farming* ", Volume 15, ISSI, PP 261-269, 24 January 2018.
- [19] Vince Paul, Sukanya C.M, Priya K.V, Mr. Sankaranarayanan P.N," *Integration of Wireless Sensor Networks and Mobile Cloud- a Survey*", (IJCSIT) **International Journal of Computer Science and Information Technologies**, Vol.6(1), PP: 159-163, 2015.
- [20] NdayishimyeMoise, N.J. Uke," *A Study on Sensory data processing Framework in integration of WSN and Cloud Computing*", **International Journal of Advanced Research in Computer and Communication Engineering**, Vol. 5, Issue 6, June 2016.
- [21] KhandakarEntenamUnayes Ahmed, Mark A Gregory," *Integration Wireless Sensor Networks with Cloud Computing* ", **Seventh International Conference on Mobile Ad-hoc and Sensor Networks, Beijing**, PP. 364-366, December 2011.

- [22] H S Guruprased, Swathi B.S.," ***Integration of Wireless Sensor Networks and Cloud Computing*** ", Vol 2, Issue 5, May 2014.
- [23] Chandrani Ray Chowodhury, "A ***Survey on Cloud Sensor Integration***", Vol. 2, Issue 8, August 2014.
- [24] Werner Kurschi, Wolfgang Beer, "***Combining Cloud Computing and Wireless Sensor Networks***", **Conference paper**. PP. 512-518, December 2009.
- [25] S. Janani Devi, G.S.Streetha Devi, G.M. Tamil Selvan, "***Wireless Sensor Network Integrating with Cloud Computing for Patient Monitoring***", Volume 6, Issue 3, pp:316-323,Dec.2013.
- [26] AlessioBotta, Walter de Donato, Valerio Persico, Antonio Pescape, "***On the Integration of Cloud Computing and Internet of Things***", **International Conference on future Internet oh Things and Cloud**, PP.23-30, August 2014.
- [27] NawafAlharbe, Anthony S. Atkins, Justin Champion," ***Use of Cloud Computing with wireless Sensor Networks in an Internet of Things Environment for a Smart Hospital Network*** ", **The Seventh international Conference on eHealth, Telemedicine, and Social Medicine**, November 2015.
- [28] Mohammad Hossein Anisi, Gaddafi Abdul-Salaam, Adbul Hanan Abdullah, "***A survey of wireless sensor network approaches and their energy consumption for monitoring farm fields in precition agriculture***", Volume 16, Issue 2, PP. 216-238, April 2015.

- [29] Anjum Awasthi, S.R.N Reddy, " *Monitoring for Precision Agriculture using Wireless Sensor Network-A Review*", **Global Journal of Computer Science and Technology Network, Web and Security**, Volume 13 Issue 7 version 1.0 Year 2013.
- [30] Jing Li, Chong Shen, " *An Energy Conservative Wireless Sensor Networks Approach for Precision Agriculture* ". PP.387-399,2(4), December 2013.
- [31] Yongyun Cho, Kyoungryong Cho, Changsun Shin, Jangwoo Park, Eun-Ser Lee, " *An Agricultural Expert Cloud for a Smart Farm* ", vol. 164, PP. 657-662, 5 June 2012.
- [32] Sjaak Wolfert, Lan Ge, Cor Verdouw, Marc-Jeroen Boraardt, " *Big Data in Smart Farming- A review*", **Agricultural Systems**, vol. 153, PP. 69-80, May 2017.
- [33] Andrea Oradnini, Paolo Pasini, " *Service co-production and Value co-creation : The case for a Service – Oriented Architecture (SOA)* ", volume 26, Issue 5, PP. 289-297, October 2008.
- [34] Qura-Tul-Ain Khan, Sagheer Abbas, Atifa Athar, " *Advance Modeling of Agriculture Farming Techniques Using Internet of Things* ", **IJCSNS International Journal of Computer Science and Network Security**, Vol. 17, No. 12, PP. 114-119, December 2017.
- [35] M. A. M. Vieira, C. N. Coelho, D. C. da Silva and J. M. da Mata, " *Survey on wireless sensor network devices* ", **IEEE Conference on**

**Emerging Technologies and Factory Automation** , Volume 1, PP. 537-544, 2003.

- [36] Shrikant D.Dhamghere, and Shanthi K. Guru, " ***Robust Data Collection in Wireless Sensor Networks with Mobile Sinks*** ", (IJCSIT) **International Journal of Computer Science and Information Technologies**,vol. 5(4),PP:4999-5002, 2014.
- [37] Thoraya Obaid, Haleemah Rashed, Ali Abou-Elnour, Mohammad Rehan, Mussab Muhammad Saleh, Mohammed Tarique, " ***ZigBee Technology and Its Application in Wireless Home Automation Systems : A survey***", Vol. 6, No. 4, July 2014.
- [38] Krishnan S.P.T., Jose L. Ugia Gonzalez, " ***Building Your Next Big Thing With Google Cloud Platform Book***", **Getting Started With Google Cloud Platform Chapter**, January 2015.
- [39] Michael Collier, Robin Shahan, " ***Microsoft Azure Essentials, Fundamentals of Azure*** ", 2015.
- [40] David Palma, Joseph Snow, " ***Amazon Web Services Student Tutorials*** ", October 2012.
- [41] **Oracle Cloud**, " ***Getting Started With Oracle Cloud***", November 2018.
- [42] Tom Wennerstrom, Jens Jespersen, Magnus Lundquist, " ***Simple Object Access Protocol A basic overview***", September 2002.
- [43] R. Khare, R. N. Taylor, " ***Extendingthe Representational State Transfer (REST) architectural style for decentralized systems*** ",



**International Conference on Software Engineering**, PP.428-437, 2004.

- [44] Teh Phoey Lee, Abdul Azim Abdul Ghani, Hamidah Ibrahim, Rodziah Atan, " *Coalescence of XML- Based Really Simple Syndication (RSS) Aggregator for Blogosphere* ", January 2009.
- [45] CesarePaulasso , Olafzimmermann , Frank Leymann , " *RESTFUL Web Services vs . “big” web services: Making the Right Architectural Decision* ", April 2008.
- [46] Heping Zhang, Theib Oweis, " *Water - yield relations and optimal irrigation scheduling of wheat in the Mediterranean region* ", Vol. 38, Issue 3, PP: 195-211, January 1999.
- [47] An Oracle White Paper, " *Oracle SQL Developer for Database Developers*", September 2008.
- [48] **Getting Started Guide**, " *Oracle Database Express Edition*", May 2014.
- [49] **Oracle Fusion Middleware**, " *Introduction to Oracle Web Logic Server* ", November 2010.
- [50] **An Oracle White Paper**, " *Oracle JDeveloper Overview* ", January 2008.

## Appendix A

### (A.1) The sample code of first web service "sensors data service":

```

@Path("restwebservice")
public class SensorNodeReadingsSC {
    public SensorNodeReadingsSC() { }
    @PUT
    @Consumes("application/json")
    @Produces("text/plain")
    @Path("/sensorReadings")
    public String insertSensorNodeReadings(SensorNodeReadingObject sensorReadings)
    {
        String msg = " ";
        try {
            msg = connectingToDBandInsert(sensorReadings);
            return msg;
        } catch (Exception e) {
            e.printStackTrace();
            return "Exception Has been Happend !! "; } }
    private String connectingToDBandInsert(SensorNodeReadingObject sensorReadings)
    throws Exception {
        Connection connection = null;
        Statement stmt = null;
        String msg = " ";
        String sql = " ";
        try {
            javax.naming.Context initialContext = new javax.naming.InitialContext();
            javax.sql.DataSource dataSource =
                (javax.sql.DataSource)
initialContext.lookup("java:comp/env/jdbc/AgriCultureDBDS");
            connection = dataSource.getConnection();
            stmt = connection.createStatement();
            //Insert into SENSOR_NODE_READING
            (NODE_ID,SENSOR_ID,DESCRIPTION,READING_TIME,READING,SENSOR_N
ODE_ID) values (.,",to_timestamp(", 'DD-MON-RR HH.MI.SSXXF AM'),",");
            sql =
                "Insert into SENSOR_NODE_READING values (" + new
BigDecimal(sensorReadings.getNodeID()) + ", " +
                new BigDecimal(sensorReadings.getSensorID()) + ", " +
sensorReadings.getDescription() +

```

```

        ",systimestamp," + sensorReadings.getReadingValue() +
        ",SEN_NOD_READ_SEQUENCE.nextVal," +
        new BigDecimal(sensorReadings.getZoneID()) + ")";
        System.out.println("sql: " + sql);
        stmt.executeUpdate(sql);
        connection.commit();
        msg = "Created Success";
    } catch (Exception e) {
        e.printStackTrace();
        msg = "Failed with Exception";
    } finally {
        //finally block used to close resources
        if (stmt != null)
            try {
                connection.close();
            } catch (SQLException e) {
                e.printStackTrace();
                connection.rollback();    }
    }
    return msg;    }

```

**(A.2) The sample code of second web service "Irrigation service":**

```

@PUT
@Produces("text/plain")
@Path("/updateZoneMoisture")
public String updateZoneMoisture() {
    String msg = "";
    try {
        msg = connectingToDBandUpdate();
        return msg;
    } catch (Exception e) {
        e.printStackTrace();
        return "Exception Has been Happend !! ";
    }
}

private String connectingToDBandUpdate() throws Exception {
    Connection connection = null;
    String msg = "";
    CallableStatement cStmt = null;
    try {
        javax.naming.Context initialContext = new javax.naming.InitialContext();
        javax.sql.DataSource dataSource =
            (javax.sql.DataSource)
initialContext.lookup("java:comp/env/jdbc/AgriCultureDBDS");
        connection = dataSource.getConnection();
        connection.setAutoCommit(true);
        // calling stored procedure in oracle data based which modifying zverage and
water
        cStmt = connection.prepareCall("{?=call moisture_update_zone()}");
        cStmt.registerOutParameter(1, java.sql.Types.VARCHAR);
        cStmt.execute();
        System.out.println("output: " + cStmt.getString(1));
        connection.commit();
        cStmt.close();
        msg = "updated Success";
    } catch (Exception e) {
        e.printStackTrace();
        msg = "Failed with Exception";
    } finally {
        //finally block used to close resources
        if (cStmt != null)
            try {
                connection.close();
            }

```

```
    } catch (SQLException e) {  
        e.printStackTrace();  
        connection.rollback(); } }  
return msg; }
```

**(A.3) The sample code of third web service "Get List Hierarchy service":**

```

@GET
@Produces("application/json")
@Path("/GetListHierarchy")
public ListZNSResponseObject getListHierarchy() {
    ListZNSResponseObject finalList = new ListZNSResponseObject();
    List<ListZNSObject> herList = new ArrayList<ListZNSObject>();
    try {
        herList = generateHierarchyList();
        finalList.setMainObjectList(herList);
        finalList.setResponseMsg("Success");
        return finalList;
    } catch (Exception e) {
        e.printStackTrace();
        finalList.setResponseMsg("Exception Has been Happend !!");
        return finalList;    } }

private List<ListZNSObject> generateHierarchyList() throws Exception {
    List<ListZNSObject> zonHerList = new ArrayList<ListZNSObject>();
    ListZNSObject firstListComp = new ListZNSObject();
    Connection connection = null;
    PreparedStatement preparedStatement = null;
    try {
        javax.naming.Context initialContext = new javax.naming.InitialContext();
        javax.sql.DataSource dataSource =
            (javax.sql.DataSource)
initialContext.lookup("java:comp/env/jdbc/AgriCultureDBDS");
        connection = dataSource.getConnection();
        String selectSQL = "select ZONE_ID,ZONE_NAME from zones";
        preparedStatement = connection.prepareStatement(selectSQL);
        ResultSet rs = preparedStatement.executeQuery();
        while (rs.next()) {
            firstListComp = new ListZNSObject();
            firstListComp.setType("ZONE");
            String id = rs.getString("ZONE_ID");
            String name = rs.getString("ZONE_NAME");
            System.out.println("Zone ID: " + id);
            firstListComp.setId(id);
            firstListComp.setName(name);
            firstListComp.setMsg("Success");
        }
    }
}

```



```

    } nodeObj.setChildList(sensorHerList);    } }
    //getting sensor under zone direct
    selectSQL = "SELECT ZONE_ID, SENSOR_ID FROM
ZONE_SENSOR_DIV where ZONE_ID=?";
    preparedStatement = connection.prepareStatement(selectSQL);
    preparedStatement.setInt(1, Integer.parseInt(masterId));
    rs = preparedStatement.executeQuery();
    while (rs.next()) {
        firstListComp = new ListZNSObject();
        firstListComp.setType("SENSOR");
        firstListComp.setMsg("Success");
        String id = rs.getString("SENSOR_ID");
        System.out.println("SENSOR_ID under Zone " + id);
        firstListComp.setId(id);
        nodeHerList.add(firstListComp); }
    masterObj.setChildList(nodeHerList); } }
} catch (Exception e) {
    e.printStackTrace();
    firstListComp = new ListZNSObject();
    firstListComp.setMsg("Failed with Exception");
    zonHerList.add(firstListComp);
} finally {
    //finally block used to close resources
    if (preparedStatement != null)
        try {
            preparedStatement.close();
            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
            connection.rollback(); } }
    return zonHerList;}

public boolean setAttributeValue(AttributeContext p0, Object p1) {
    return false; }

public Object createRowData(RowContext p0) {
    return null; }

public Object registerDataProvider(RowContext p0) {
    return null;}

public boolean removeRowData(RowContext p0) {
    return false; }

public void validate() {
}
}

```



**(A.4): calculation and zones moisture list can be appear in the sample****code :**

```

package view;
import com.sun.jersey.api.client.Client;
import com.sun.jersey.api.client.ClientResponse;
import com.sun.jersey.api.client.WebResource;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashSet;
import java.util.Map;
import java.util.Random;
import java.util.Set;
import javax.faces.event.ActionEvent;
import javax.faces.event.ValueChangeEvent;
import oracle.adf.share.ADFContext;
import oracle.adf.view.rich.context.AdfFacesContext;
import oracle.jbo.Row;
import oracle.jbo.ViewObject;
import org.apache.myfaces.trinidad.event.PollEvent;
import restconsumeobjects.SensorNodeReadingObject;
public class SimulationGraphMB {
    private String plant_id = "";
    private String plant_age = "";
    private String soil_id = "";
    private String min_moist = "";
    private String max_moist = "";
    private Map pfs = ADFContext.getCurrent().getPageFlowScope();
    public SimulationGraphMB() {
        if (!AdfFacesContext.getCurrentInstance().isPostback()) {
            ViewObject vo = ADFUtils.getAm().findViewObject("ZoneReadingsPVO1");
            String where = "ZONE_NAME = '-11'";
            vo.setWhereClause(where);
            vo.executeQuery();
        }
    }

    public void selectZoneValueChangeLis(ValueChangeEvent valueChangeEvent) {
        String zoneName = valueChangeEvent.getNewValue() + "";
        ViewObject vo = ADFUtils.getAm().findViewObject("ZoneReadingsPVO1");
        String where = "ZONE_NAME = " + zoneName + "";
        vo.setWhereClause(where);
    }
}

```

```

vo.setOrderByClause("READING_TIME ASC");
System.out.println(vo.getQuery());
vo.executeQuery();
String zone_id = "";
if (vo != null && vo.getRowCount() > 0 && vo.getRowAtRangeIndex(0) != null)
    zone_id = vo.getRowAtRangeIndex(0).getAttribute("ZoneId") + "";
vo = ADFUtils.getAm().findViewObject("plantBasedZonePVO1");
where = "ZONE_ID = " + zone_id + "";
vo.setWhereClause(where);
vo.executeQuery();
if (vo.getRowCount() > 0) {
    Row row = vo.getRowAtRangeIndex(0);
    if (row != null) {
        plant_id = row.getAttribute("PlantId") + "";
        plant_age = row.getAttribute("Age") + "";
    }
}
vo = ADFUtils.getAm().findViewObject("soilBasedZonePVO1");
where = "ZONE_ID = " + zone_id + "";
vo.setWhereClause(where);
vo.executeQuery();
if (vo.getRowCount() > 0) {
    Row row = vo.getRowAtRangeIndex(0);
    if (row != null) {
        soil_id = row.getAttribute("SoilId") + "";
    }
}
vo
ADFUtils.getAm().findViewObject("avgReadingBasedZonePVO1");
vo.executeQuery();
vo = ADFUtils.getAm().findViewObject("bestValuesPVO1");
vo.executeQuery();
vo = ADFUtils.getAm().findViewObject("bestValuesPVO2");
where = "";
if (plant_id != null && !plant_id.isEmpty())
    where = "PLANT_ID= " + plant_id;
if(!where.isEmpty()){
if (soil_id != null && !soil_id.isEmpty())
    where += " and SOIL_ID=" + soil_id;
}else{
    if (soil_id != null && !soil_id.isEmpty())
        where = "SOIL_ID= " + soil_id;
    }
if(!where.isEmpty()){
if (plant_age != null && !plant_age.isEmpty())

```

jjjjs=

```

        where += " and PLANT_AGE=" + plant_age;
    }else{
        if (plant_age != null && !plant_age.isEmpty())
            where = "PLANT_AGE=" + plant_age;
    }
    vo.setWhereClause(where);
    System.out.println("Query: " + vo.getQuery());
    vo.executeQuery();
    if (vo.getRowCount() > 0) {
        Row row = vo.getRowAtRangeIndex(0);
        if (row != null) {
            min_moist = row.getAttribute("MinMoisture") + "";
            max_moist = row.getAttribute("MaxMoisture") + "";
            if (max_moist != null && !max_moist.isEmpty() &&
!max_moist.equalsIgnoreCase("null"))
                pfs.put("max_moist", max_moist);
            System.out.println("max_moist: " + max_moist);
        }
    }
    updateZonesAction();
    vo = ADFUtils.getAm().findViewObject("ZonesView3");
    vo.executeQuery(); }
public String updateZonesAction() {
    Client client = Client.create();
    WebResource webResource =
        client.resource("http://localhost:7101/AgriCultureApplication-
RESTWebService-context-root/resources/restwebservice/updateZoneMoisture");
    ClientResponse response =
webResource.type("application/json").put(ClientResponse.class);
    if (response.getStatus() != 200) {
        System.out.println("Failed : HTTP error code : " + response.getStatus());
        String error = response.getEntity(String.class);
        System.out.println("Error: " + error);
        return error;
    }
    System.out.println("Output from Server .... \n");
    String output = response.getEntity(String.class);
    System.out.println(output);
    return output; }
public void pollList(PollEvent pollEvent) {
    // Add event code here... }
private void checkNeedToUpdateAvg(ArrayList<String> zoneOnId, String
max_moist) {

```

```

boolean stop = false;
while (!stop) {
    Random randnum;
    int numberOfData = 5; //number of insert per time to get the avg
    int max_mo = Integer.parseInt(max_moist);
    Random ran = new Random();
    // need to put sensor and nodes ID that is defined on the system
    Set<String> sensorHashSet = new HashSet<String>(Arrays.asList("42", "10",
"41","101","43"));
    Set<String> nodeHashSet = new HashSet<String>(Arrays.asList("1", "4",
"3","22"));
    Set<String> zoneHashSet = new
HashSet<String>(Arrays.asList(zoneOnId.toArray(new String[zoneOnId.size()])))

    SensorNodeReadingObject newDataToSent;
    int size = nodeHashSet.size();
    randnum = new Random();
    int item = randnum.nextInt(size);
    int i = 0;
    for (int count = 0; count < numberOfData; count++) {

        newDataToSent = new SensorNodeReadingObject();

        size = nodeHashSet.size();
        item = randnum.nextInt(size);
        i = 0;
        for (String obj : nodeHashSet) {
            if (i == item) {
                newDataToSent.setNodeID(obj);
                break;    }
            i++;    }
        size = sensorHashSet.size();
        item = randnum.nextInt(size); // In real life, the Random object should be
rather more shared than this
        i = 0;
        for (String obj : sensorHashSet) {
            if (i == item) {
                newDataToSent.setSensorID(obj);
                break;    }
            i++;    }
        size = zoneHashSet.size();
        item = randnum.nextInt(size);
        i = 0;

```

```

        for (String obj : zoneHashSet) {
            if (i == item) {
                newDataToSent.setZoneID(obj);
                break;
            }
            i++;
        }
        ran = new Random();
        newDataToSent.setReadingValue(ran.nextInt(max_mo) + "");
        newDataToSent.setDescription("test : " + ran.nextInt(max_mo));
        //send data to webservice
        insertReadingData(newDataToSent);
    }
    updateZonesAction();
    ViewObject vo = ADFUtils.getAm().findViewObject("ZonesView3");
    vo.executeQuery();
    int rowCount = vo.getRowCount();
    zoneOnId = new ArrayList<String>();

    if (rowCount > 0) {
        for (int j = 0; j < rowCount; j++) {
            Row row = vo.getRowAtRangeIndex(j);
            if (row != null) {
                String zoneAction = row.getAttribute("ZoneAciton") + "";
                if (zoneAction.equalsIgnoreCase("ON")) {
                    zoneOnId.add(row.getAttribute("ZoneId") + "");
                }
            }
        }
        if (zoneOnId.size() > 0) {
            stop = false;
        } else {
            stop = true;
        }
    } else {
        stop = true;
    }
}

private String insertReadingData(SensorNodeReadingObject
sensorNodeReadingObject) {
    Client client = Client.create();
    WebResource webResource =
client.resource("http://localhost:7101/AgriCultureApplication-RESTWebService-
context-root/resources/restwebservice/sensorReadings");

```

```

        ClientResponse response =
webResource.type("application/json").put(ClientResponse.class,
sensorNodeReadingObject);
        if (response.getStatus() != 200) {
            System.out.println("Failed : HTTP error code : " + response.getStatus());
            String error = response.getEntity(String.class);
            System.out.println("Error: " + error);
            return error;
        }
        System.out.println("Output from Server .... \n");
        String output = response.getEntity(String.class);
        System.out.println(output);
        return output;
    }
    public void startIrrigation(ActionEvent actionEvent) {
        ViewObject vo = ADFUtils.getAm().findViewObject("ZonesView3");
        vo.executeQuery();
        // to insert data to table within avg
        int rowCount = vo.getRowCount();
        ArrayList<String> zoneOnId = new ArrayList<String>();
        if (rowCount > 0) {
            for (int i = 0; i < rowCount; i++) {
                Row row = vo.getRowAtRangeIndex(i);
                if (row != null) {
                    String zoneAction = row.getAttribute("ZoneAciton") + "";
                    if (zoneAction.equalsIgnoreCase("ON")) {
                        zoneOnId.add(row.getAttribute("ZoneId") +
                        "");
                    }
                }
            }
        }
        max_moist = pfs.get("max_moist") + "";
        System.out.println("max_moist: " + max_moist);
        if (max_moist != null && !max_moist.isEmpty()) {
            System.out.println("max_moist: " + max_moist);
            if (zoneOnId.size() > 0) {
                checkNeedToUpdateAvg(zoneOnId, max_moist);
                System.out.println("zoneOnId.size: " + zoneOnId.size());
            }
        }
        vo = ADFUtils.getAm().findViewObject("avgReadingBasedZonePVO1");
        vo.executeQuery();
    }
}

```

**(A.5): The sample code of the implementation of the login page:**

```

package view;
import java.util.Map;
import javax.faces.application.Application;
import javax.faces.application.NavigationHandler;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.event.ValueChangeEvent;
import model.EViewObjects.UserLoginViewRowImpl;
import oracle.adf.share.ADFContext;
import oracle.jbo.Row;
import oracle.jbo.ViewObject;
import oracle.security.idm.providers.ad.ADUtills;
public class LoginPageMB {
    private Map pfs = ADFContext.getCurrent().getPageFlowScope();
    private Map sfs= ADFContext.getCurrent().getSessionScope();
    private String Error_MSG_login="User Name or password is Incorrect";
    public LoginPageMB() { }
    public void userNameValueChangeLis(ValueChangeEvent valueChangeEvent) {
        pfs.put("userName", valueChangeEvent.getNewValue()); }
    public void passwordValueChangeLis(ValueChangeEvent valueChangeEvent) {
        pfs.put("password", valueChangeEvent.getNewValue()); }
    public void loginToSystemBtn(ActionEvent actionEvent) {try{
        String userName="";
        String password="";
        userName=pfs.get("userName")+"";
        password=pfs.get("password")+"";
        if(userName!=null && !userName.isEmpty() && password!=null &&
!password.isEmpty()){
            ViewObject usersVo=
ADFUtils.getAm().findViewObject("UserLoginView2");
            String where="USER_ID = '"+userName+"' AND
            USER_PASSWORD='"+password+"'";
            usersVo.setWhereClause(where);
            usersVo.executeQuery();
            if(usersVo!=null && usersVo.getRowCount(>0){
                Row row= usersVo.getRowAtRangeIndex(0);
                if(row!=null){
                    sfs.put("userFullName", row.getAttribute("UserName"));
                    sfs.put("userRole", row.getAttribute("UserRole"));
                    handleNavigation(); }else{

```

```

        pfs.put("loginErrorMsg",Error_MSG_login);
    pfs.put("loginErrorVisible", "Y");    }    }else{
        pfs.put("loginErrorMsg",Error_MSG_login);
    pfs.put("loginErrorVisible", "Y");    }    }else{
        pfs.put("loginErrorMsg",Error_MSG_login);
        pfs.put("loginErrorVisible", "Y");    }
    }catch(Exception e){ e.printStackTrace();
        ADFUtils.showMessageError("Error Has been occured while loggin check your
Admin");    }}
    public void handleNavigation() {
        FacesContext context = FacesContext.getCurrentInstance();
        Application app = context.getApplication();
        NavigationHandler handler = app.getNavigationHandler();
        handler.handleNavigation(context, null, "toMain");
    }}

```



جامعة النجاح الوطنية

كلية الدراسات العليا

# تطبيق الحوسبة السحابية على نظام إدارة الري الزراعي الذكي

إعداد

مصطفى محمد يونس

إشراف

د. عدنان سلمان

قدمت هذه الأطروحة استكمالاً لمتطلبات الحصول على درجة الماجستير في الحوسبة المتقدمة  
بكلية الدراسات العليا في جامعة النجاح الوطنية في نابلس، فلسطين.

2018

## ب تطبيق الحوسبة السحابية على نظام إدارة الري الزراعي الذكي

إعداد  
مصطفى محمد يونس  
إشراف  
د.عدنان سلمان

### الملخص

تعتبر الزراعة من العوامل الرئيسية على تطور الدول ونهضتها لذلك يجب العمل على العوامل المؤثرة على الزراعة وتطويرها ومن اهم العوامل التي تؤثر على الزراعة طبيعة التربة والمناخ وكمية المياه المطلوبة لكامل فترة نمو النبتة وكمية السماد المطلوبة حتى تنمو النبتة بشكل جيد بعيد عن الامراض .

تم التركيز في هذه الرسالة على العوامل المؤثرة على الزراعة من حيث نوع التربة ومعرفة خصائص كل نوع وكمية المياه المطلوبة لكل نوع نبات في كل مراحل العمر بحيث تصبح عملية الري عملية اوتوماتيكية بناء على متطلبات النبتة والتربة، وأيضا كمية السماد المطلوبة في مراحل عمر النبتة المختلفة.

من خلال هذه الرسالة يقوم المزارع بتحديد نوع النبات المراد زراعته ونوع التربة الموجودة وتاريخ زراعة النبتة فقط وبعد ذلك يقوم النظام بعمل كل الخطوات اللاحقة واللازمة للنمو الصحيح لهذه النبتة بحيث يتم عمل كل الحسابات اللازمة والمطلوبة لمعرفة كمية المياه المطلوبة بناء على المعطيات التي قام بتحديد المزارع.

ولتسهيل عملية الزراعة تم تقسيم الارض الزراعية الى مناطق محدودة وتم وضع حساسات مختلفة في كل منطقة زراعية وتم ربطها مع محبس الكتروني ليتم فتحه وتسكيه بناء على القرارات القادمة من تطبيق الانترنت الذي يقوم بعمل كامل الحسابات المختلفة وإعطاء القرارات الصحيحة التي تساعد في نمو النبتة بأقل كمية مياه مطلوبة وبدون جهد بشري لمتابعة سير عملية الزراعة.

ولتسهيل عملية ادارة الري والزراعة يقوم الخبير الزراعي بتعبئة المعلومات المطلوبة عن مجموعة من النباتات بأعمار مختلفة من حيث كمية المياه المطلوبة بناء على نوع التربة في جدول معين موجود في قاعدة البيانات وأيضا يقوم بتعبئة جدول اخر يحتوي على انواع التربة المختلفة

ت

وخصائص كل نوع من هذه الانواع. ويقوم المزارع بتعبئة جدول خاص بالنبته من حيث نوع النبتة ونوع التربة الموجودة وتاريخ الزراعة . وبإمكان مسئول النظام الاطلاع على جميع الجداول والتعديل عليها وإضافة الحسابات المختلفة.