Spring 2016

Watch Dog for a Plant House

Vamshider Reddy Dasari
Governors State University

Yashwanth Kumar Kalwa
Governors State University

Vinay Keerthipati
Governors State University

Ranjith Kiran Motru
Governors State University

ABSTRACT

The determination in selecting this project is to reduce the effects by the attributes of the nature on the plants. The proposed project we will help the user to check and maintain all the essential things like temperature, moisture in air, water content in soil and salinity in the soil required for an optimal growth of a plant. The Plant House maintenance by the user in a dynamical process is the goal of our project.

In the existing system there should be a person in-charge in the room where the trees are planted and grown, to check the temperature and the other attributes which are to be maintained. To reduce the human presence in the process, in the existing system we construct a Greenhouse application which will respond to the changes in the attributes of the environment. The optimal growing environment of the different plants are stored in to the database. The application consists of a main home page where we can see the different room, if the user need to add a room he can add it. The user is recommended to enter required values for the Temperature, Humidity, Water, Fertilizer, Plant Bed, and Lightning, which are the essential needs for a plant growth. Sensors are maintained by the user, values which should match with the optimal values of the plant given by the user in the database. The fields can also be updated and also the room can also be deleted.

Table of Content

List of Figures

1 Project Description

In the process of fulfillment of the graduate capstone project here we propose a project entitle Watch Dog for a Plant House. This project goal is to maintain the Planthouse by the user dynamically. Comparing with the existing system the proposed system has many advantages which reduces the level of risk. Here we develop a web application through which we can we can monitor the attributes of the nature like Temperature, Soil Acidity, Humidity, Sunlight and fertilizer level.

1.1 Competitive Information

We haven't got any appropriate software through which we can maintain a plant house. So our competition is mainly with the challenges we have in building the application. We are building a project which has higher potential, which can be implemented as a real world application. Maintaining a plant house is a really hard job, so as we develop a solution from which we can state the potentiality of our team. We believe in our team work which helps us to be first in the market.

1.2 Relationship to Other Applications/Projects

We are developing an independent project which doesn't have any relationship with the other project.

1.3 Assumptions and Dependencies

- The values of the sensor are considered samples values, the sample values are stored into the database for each attribute to check the execution of the program.
- This project needs support with the external persons who have good knowledge about the plants and instruments used in developing the plant house.

1.4 Future Enhancements

This application which we are developing will be a web service. In the future we would like to develop a Mobile Application which can run on multiple mobile operating systems like Android, IOS, and Windows. Furthermore, any additional features requested by the users are also taken into consideration for our future step. Forget password option will be provided at the user interface.

1.5 Definitions and Acronyms

Plant house: - It is a closed house where plants are grown under certain conditions which help the plant to grow. Our project helps in maintaining these conditions.
Attributes: - These are the factors which are essential for a plant to grow. They are Temperature, Soil Acidity, Humidity, Sunlight, Fertilizer and Water content.
Admin: - The Admin will have the privileges to add a room, deleted a room and update the attribute values. The admin will be given a level code and description of his job.
User: - The person who access the web page and don't have privileges to add or deleted a room. He will be given a unique ID.

2 Project Technical Description

In this project we have 6 Web Pages which include a Login page, Registration page, Add Room, Main Page, Modify Room, and Room Details. These web pages are designed to be responsive for every screen using Bootstrap. The Database should be created containing all the tables which can store the User data like Name, Email id, Phone number, Address. Details about the room, attributes of the room and example values of the attributes are Stored in the database. Spring framework is to control the web application. Different modules are used in the application. They are Spring Core Container, Aspect-oriented programming, Model–view–controller. Hibernate framework is used to connect to the database.

3 System Requirements.

Hardware Requirements.

1) Hard disk 250 GB

2) Ram 4 GB and above

3) Processor i3 and above.

Software Requirements.

1) Eclipse

2) My SQL Database

3) Postman

4) Operating System Windows 2008 and above

5) Java

3.1 Application Architecture

Our Plant house project three components User, Room, Attributes of the Room. Describe system architecture of your project. The first component User are of two kinds, they are Admin and normal user. The Admin have more privileges compare to the normal user. Second Component Room will contain the information of the specific room. The attribute in are project are divided into three categories like Range, Interval and Boolean. Temperature, Humidity and Soil Acidity comes under Range, it has the minimum and maximum value. Sunlight comes under Boolean, required or not required. Fertilizer and Water spraying comes under interval, specifies at what time the requirement is there. The Admin makes the user register and allows him to check the values.

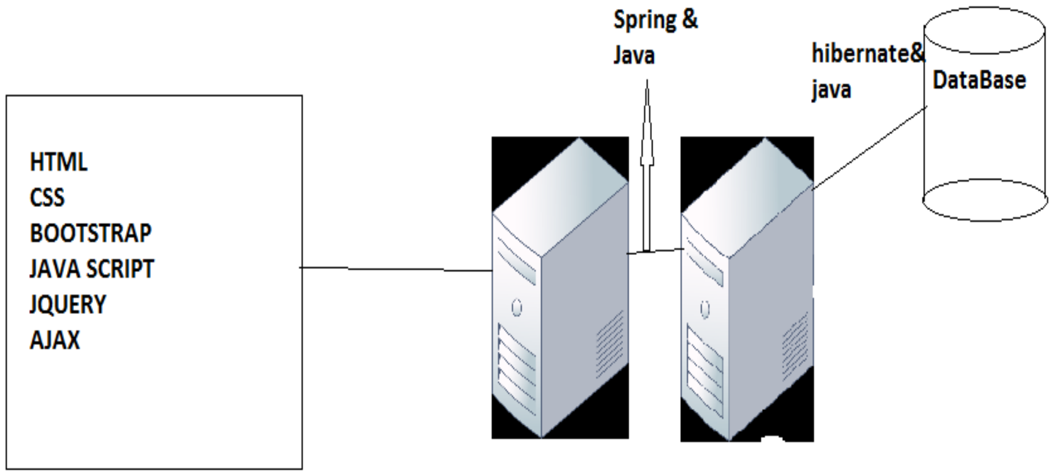


FIGURE 1 Application Architecture

3.2 Application Information flows

The User main component of the architecture, the information flow is shown in the below diagram.

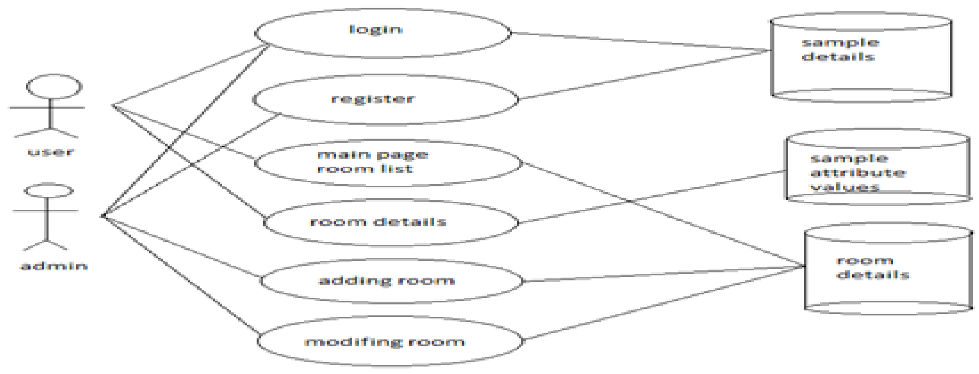


FIGURE 2 Information flow.

3.3 Interactions with other Applications

Since we are developing a web based application which is indeed an independent application. The plant house can be maintained by comparing the attribute to the sample values stored into the database. Our project does not contain any payment system or human resource maintained so the interactions with the other application is null.

3.4 Capabilities

In this application we have to users they are Admin and the normal user. The Admin will have the capability add a user, to add a room, to edit a room, give permissions to the room, delete the user. The database which we created has a capability of adding deleting, retrieving the user's data, room data, attribute values through front end. The front end has some capability to provide the validations. Validations are provided in all the web pages created using jQuery. Apache tomcat server used as the server.

3.5 Risk Assessment and Management

Risks are common when we start any project. Identification of those risk at the requirement and designing stage will be very easy to handle. Maintaining the session for the use and implementing that through connecting through the database has a few risk. This can be eliminated by normalizing the database. The other risk is to maintain the data for the room. If the data is more regarding the attributes it may result to data complexity this is overcome by getting the backup for a certain interval period.

4 Project Requirements

4.1 Identification of Requirements

This is the first process we have started in the development of the project. We considered the requirement collection as the critical task and scheduled more time for it because the requirements acts as the basic for what we develop, test and deploy. We have gathered 64 requirements. Here we list her requirements.

WPH-001-0.1-User-000101
Every user should have a User id to login to the system.

WPH-001-0.1-User-000102
The user should have a Password to log in to the system

WPH-001-0.1-User-000103
User email Id must be unique, as the user will be logging in using email Id.

WPH-001-0.1-User-000104
User login form must be validated

WPH-001-0.1-User-000105
The first name of the user should be stored

WPH-001-0.1-User-000106
The last name of the user should be stored

WPH-001-0.1-User-000107
The contact number of the user should be stored

WPH-001-0.1-User-000108
The E-mail ID of the user should be stored

WPH-001-0.1-User-000109
The Secondary Mobile number of the user should be stored

WPH-001-0.1-User-000110
The date of birth of the user should be stored.

WPH-001-0.1-User-000111
The registration date of user must be stored

WPH-001-0.1-User-000112
The Address of the user should be stored which consists of the address line1, address line 2, City, State, Country, Pin code.

WPH-001-0.1-User-000113
The State information should be stored in a separate table and should be connected to the user address table

WPH-001-0.1-User-000114
The City information should be stored in a separate table and should be connected to the user address table

WPH-001-0.1-User-000115
The Pin code information should be stored in a separate table and should be connected to the user address table

WPH-001-0.1-User-000116
Their maybe three types of address for each user. The Mailing address of the user must be stored

WPH-001-0.1-User-000117
Current address of the user must be stored

WPH-001-0.1-User-000118
A new user must be able to register with all the user details

WPH-001-0.1-User-000119
User registration form must be validated

WPH-001-0.1-Room-000120
Every Room should have a unique ID called as a room ID

WPH-001-0.1-Room-000121
Temperature is considered as an attribute in a room which has to be maintained

WPH-001-0.1-Room-000122
Temperature Attribute is of type Range which has minimum and maximum Values to me maintained

WPH-001-0.1-Room-000123
Soil Acidity Is considered as an attribute in a room which has to be maintained and has a range value

WPH-001-0.1-Room-000124
Soil Acidity Attribute is of type Range which has minimum and maximum Values to me maintained

WPH-001-0.1-Room-000125
Humidity is considered as an attribute in a room which has to be maintained and has a range value

WPH-001-0.1-Room-000126
Humidity Attribute is of type Range which has minimum and maximum Values to me maintained

WPH-001-0.1-Room-000127
Sunlight is considered as an attribute in a room which has to be maintained and has a Boolean value

WPH-001-0.1-Room-000128
Sunlight Attribute is of type Boolean which has only two possibilities required or not

WPH-001-0.1-Room-000129
Water is considered as an attribute in a room which has to be maintained and has a Interval value

WPH-001-0.1-Room-000130
Water Attribute will be of type interval, and interval type (daily, weekly) should be stored

WPH-001-0.1-Room-000131
Fertilizers is considered as an attribute in a room which has to be maintained and has an Interval value

WPH-001-0.1-Room-000132
Fertilizer Attribute will be of type interval, and interval type (daily, weekly) should be stored

WPH-001-0.1-Room-000133
All the attributes are classified into three types- Range, Interval, Boolean

WPH-001-0.1-Room-000134
Room Attributes of type range will have minimum value

WPH-001-0.1-Room-000135
Room Attributes of type range will have maximum value

WPH-001-0.1-Room-000136
The Range attribute last modified date must be stored.

WPH-001-0.1-Room-000137
The table contain the range details should have a column contain an unique ID called - range id

WPH-001-0.1-Room-000138
The interval Attribute has two types daily and monthly.

WPH-001-0.1-Room-000139
The room's value of the interval is also stored

WPH-001-0.1-Room-000140
The room may have a type attribute Boolean

WPH-001-0.1-Room-000141
The room's value of the Boolean type is also stored

WPH-001-0.1-Room-000142
Each attribute must have its unique id

WPH-001-0.1-Room-000143
Every User will have different level of access permission for the room’s allotted

WPH-001-0.1-Room-000144
User with admin level access permission only can change the attribute values of the room

WPH-001-0.1-Room-000145
User with user level access permission can only check and maintain the room values and attributes but cannot change the attribute values

WPH-001-0.1-Room-000146
Description of the access level must be stored

WPH-001-0.1-Room-000147
Access level should be given unique id and should be related to the user for permissions

WPH-001-0.1-Room-000148
Admin must be able to add/inactivate new user

WPH-001-0.1-Room-000149
Admin must be able to assign room to the user

WPH-001-0.1-Room-000150
Admin must be able to add/inactivate a room

WPH-001-0.1-Room-000151
Add New Room Form must be validated according to room attributes

WPH-001-0.1-Room-000152
Temperature values from the sensor must be stored

WPH-001-0.1-Room-000153
Temperature sensor values are populated with current time

WPH-001-0.1-Room-000154
Soil acidity values from the ph meter must be stored

WPH-001-0.1-Room-000155
Soil acidity values are populated with current time

WPH-001-0.1-Room-000156
Humidity values from the hygrometer must be stored

WPH-001-0.1-Room-000157
Humidity values are populated with current time

WPH-001-0.1-Room-000158
Water must be sprinkled with certain interval provided by the admin

WPH-001-0.1-Room-000159
Last watered record and interval must be maintained

WPH-001-0.1-Room-000160
Fertilizers are applied with certain interval provided by the admin

WPH-001-0.1-Room-000161
Fertilizers applied record with time must be maintained

WPH-001-0.1-Room-000162
Current sunlight maintenance whether required or not must be shown

WPH-001-0.1-Room-000163
Current shades position must also be shown

4.2 Operations, Administration, Maintenance and Provisioning (OAM&P)

When the new system is proposed according to the requirements we have to types of users, one is the Admin and the second will be the Normal User. Each have different access codes they are ADM and USR respectively. Maintenance of the database of the project is done by the Admin. The admin will have the privileges to add a normal user, to add a new room and to edit the room details. The normal user has the only right to check and maintain the software. The fraud detection is easily done by maintaining and comparing the details of the customer in the Database. We will implement the recovery plan in further steps.

4.3 Security and Fraud Prevention

The Security of the developed system is maintained by providing a User ID and Password for all the users. The Admin level User can only create a room, change the values of the attributes for a certain room. The admin authorizes the Normal User of the application so that it increases the security and no other than the authorized users can uses the application. Using spring framework will also increases the security to the system.

4.4 Release and Transition Plan

The execution of the program is shown on the local host on the personal computer and for the further transition plan the complete project will be deployed in the cloud by using the Amazon Web Services which will provide a free account for the students for six months.

5 Project Design Description

After the requirement collect phase when we started the designing phase we have divided the project into three parts like developing the Front End pages using Html and CSS. These pages are designed responsive so that the page will display accordingly to the size of the screen. The second part of the development phase is designing the Database of the project. We have Used My SQL Work Bench to develop the database. The total database consists of 17 tables which are interconnected to each other using the foreign key relation. In the database we can store the user's data, room data, attributes data, access permissions, types of attributes and the sample values for the attributes.

Designing the front end we have designed six responsive web pages which also have the validations created. These pages are linked with each other, so that the user can be redirected from one page to another by clicking the respective Button. In the database we are writing Stored Procedures so that the duplication of database code in reduced and increases the performance of the quires.

The following is a diagram which shows the design of the application.

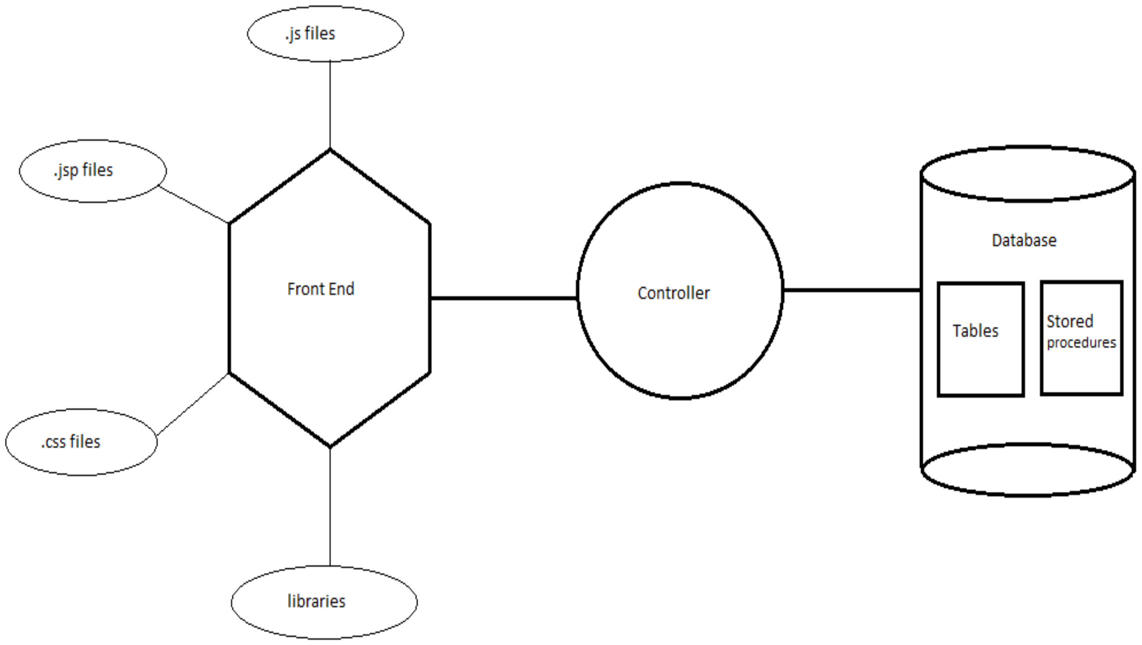


FIGURE 3 BASE DESIGN

After the designing of both front end and database is done both are connected using the spring and hibernate frame works. The spring framework will control the program like where to get redirected and also to provide web services. While the Hibernate helps in converting the database queries to the java objects. Eclipse is the tool used. Here are the following tables which shows what files does the project contain and which part the files belong to. This is our proposed design for the project.

.jsp files
1. Login.jsp
2. Registration.jsp
3. Homepage.jsp
4. Room.jsp
5. Addroom.jsp
6. Updateroom.jsp
7. Addnewuser.jsp

.js files
1. Login.js
2. Homepage.js
3. Common.js
4. Room_details.js
5. Wph_utility.js

Libraries
1. Bootstrap.css
2. Bootstrapmain.js
3. Handlebars.js
4. Jquery.min.js
5. Jqueryvalidate.min.js

.css files
1. Registration style.css
2. Main page style.css
3. Modify room.css
4. Add room style.css
5. Room details.css
6. Style.css

Controller
1. Addroomcontroller.java
2. Homepagecontroller.java
3. Logincontroller.java
4. Modifycontroller.java
5. Registrationcontroller.java
6. Roomdetailscontroller.java

Tables	
1. User	10. Access_level
2. Address	11. interval
3. City	12. Room_attribute_range
4. State	13. Room_attribute_boolean
5. Zip code	14. Room_attribute_interval
6. Room	15. Room_value_range
7. Attribute	16. Room_value_attribute
8. Attribute_type	17. Room_value_interval
9. Room_permissions	

Stored Procedures
1. Get_room_boolean_attr
2. Get_room_details_by_admin
3. Get_room_interval_attr
4. Get_room_range_attr
5. Get_user_details
6. Get_value_room_boolean_attr
7. Get_value_room_interval_attr
8. Get_value_room_range_attr

6 *Stored Procedures.*

Here we show the Store Procedures which are created to avoid the repetitions of the SQL queries.

*1) **get_room_boolean_attributes:***

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_room_boolean_attr`(IN room_id int)
BEGIN
SELECT
room_attribute_boolean.room_attribute_boolean_id,
room_attribute_boolean.attribute_id,
room_attribute_boolean.room_id,
room_attribute_boolean.value,
attribute.name,
attribute_type.attribute_type_id,
attribute_type.code,
attribute_type.type
FROM
room_attribute_boolean
INNER JOIN
attribute ON attribute.attribute_id = room_attribute_boolean.attribute_id

INNER JOIN
attribute_type ON attribute_type.attribute_type_id = attribute.attribute_type_id
WHERE
room_attribute_boolean.room_id = room_id;
END
```

2) ***get_room_details_by_admin***

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_room_details_by_admin`(IN user_id int)
BEGIN
SELECT room.room_id, room.room_name, room.created_date, room.modified_date
FROM mydb.room
inner join room_permission
on room_permission.room_id = room.room_id
where room_permission.user_id=user_id;
END
```

3) ***get_room_interval_attributes***

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_room_interval_attr`(IN room_id int)
BEGIN
SELECT
room_attributes_interval.room_attributes_interval_id,
room_attributes_interval.room_id,
room_attributes_interval.attribute_id,
room_attributes_interval.interval_id,
room_attributes_interval.value,
room_attributes_interval.modified_date,
mydb.`interval`.interval_type,
room.room_name,
attribute_type.attribute_type_id, attribute_type.code, attribute_type.type,
attribute.name
FROM
room_attributes_interval
INNER JOIN
room ON room.room_id = room_attributes_interval.room_id
INNER JOIN
mydb.`interval` ON mydb.`interval`.interval_id = room_attributes_interval.interval_id
inner join attribute
on attribute.attribute_id = room_attributes_interval.attribute_id
inner join
attribute_type on attribute_type.attribute_type_id=attribute.attribute_type_id
WHERE
room_attributes_interval.room_id = room_id;
END
```

4) ***get_room_range_attribute***

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_room_range_attr`(IN room_id int)
BEGIN
```

```
SELECT
room_attributes_range.room_attributes_range_id,
room_attributes_range.room_id,
room_attributes_range.attribute_id,
room_attributes_range.min_value,
room_attributes_range.max_value,
room_attributes_range.modified_date,
room.room_name,
attribute.name,
attribute_type.code
FROM
mydb.room_attributes_range
INNER JOIN
room ON room_attributes_range.room_id = room.room_id
INNER JOIN
attribute ON room_attributes_range.attribute_id = attribute.attribute_id
INNER JOIN
attribute_type ON attribute_type.attribute_type_id = attribute.attribute_type_id
WHERE
room.room_id = room_id;
END
```

7 Hibernate

Hibernate is a java framework which is used for mapping relational database to object-oriented domain. It uses high level object handling functions to resolve impedance mismatch problems. The primary function of hibernate is to map java classes into database tables. It has some facilities like data retrieval and data query. Even the relationships like one-to-one and many-to-many can be facilitated between the classes. Hibernate provides Hibernate Query Language and allows queries to be written on data objects. It provides persistence for Plain Old Java Objects. A no-argument constructor is the main requirement for persistent class. All the data objects are stored in the java classes as lists and sets. It is used in single java app as well as java EE application using EJB session beans, servlets, and JBI services. It is used as a feature in other languages in programming.

Hibernate Configuration XML File:

```
<?xml version='1.0' encoding='utf-8'?>
        <!DOCTYPE hibernate-configuration PUBLIC
            "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
            "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
        <session-factory>
                <!-- Database connection settings -->
                <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>

                <property name="hibernate.connection.username">root</property>
                <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/mydb?autoReconnect=true</property>
                <property name="hibernate.connection.password">varshan123</property>

                <property
name="connection.provider_class">org.hibernate.connection.C3P0ConnectionProvider</property>
                <property name="hibernate.c3p0.min_size">10</property>
                <property name="hibernate.c3p0.max_size">50</property>
                <property name="hibernate.c3p0.acquire_increment">1</property>
                <property name="hibernate.c3p0.idle_test_period">1800</property>
                <property name="hibernate.c3p0.max_statements">50</property>
```

```
		<property name="hibernate.c3p0.timeout">1800</property>

		<property name="dialect">org.hibernate.dialect.MySQLDialect</property>
		<property name="current_session_context_class">thread</property>
		<property name="show_sql">true</property>
		<property name="format_sql">true</property>
		<property name="use_sql_comments">true</property>

		<mapping package="com.planthouse.DTO" />
		<mapping class="com.planthouse.DTO.User" />
		<mapping class="com.planthouse.DTO.RoomDetails" />
		<mapping class="com.planthouse.DTO.RoomAttributeRange" />
		<mapping class="com.planthouse.DTO.RoomAttributeInterval" />
		<mapping class="com.planthouse.DTO.RoomAttributeBoolean" />
		<mapping class="com.planthouse.DTO.RoomValueRange" />
		<mapping class="com.planthouse.DTO.RoomValueInterval" />
		<mapping class="com.planthouse.DTO.RoomValueBoolean" />
		<mapping class="com.planthouse.DTO.Room" />
		<mapping class="com.planthouse.DTO.RoomAttributeRangeTable" />
		<mapping class="com.planthouse.DTO.RoomAttributeIntervalTable" />
		<mapping class="com.planthouse.DTO.RoomAttributeBooleanTable" />
		<mapping class="com.planthouse.DTO.RoomPermission" />
		<mapping class="com.planthouse.DTO.ValueRangeTable" />

	</session-factory>
</hibernate-configuration>
```

Data Transfer Object:

```
package com.planthouse.DTO;

import java.sql.Timestamp;
import java.util.Date;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

@Entity
@Table(name = "room")
public class Room {
	@Id
	@GeneratedValue(strategy = GenerationType.AUTO)
	@Column(name = "room_id")
	private int roomId;

	@Column(name = "room_name")
	private String roomName;

	@Column(name = "created_date")
	private Date createdDate;

	@Temporal(TemporalType.TIMESTAMP)
	@Column(name = "modified_date", nullable = false, columnDefinition = "TIMESTAMP default
CURRENT_TIMESTAMP on update CURRENT_TIMESTAMP")
	private Date modifiedDate = new Date();

	public Room() {

	}

	public Room(int roomId, String roomName, Timestamp createdDate, Timestamp modifiedDate) {
		super();
		this.roomId = roomId;
		this.roomName = roomName;
		this.createdDate = createdDate;
		this.modifiedDate = modifiedDate;
	}

	public int getRoomId() {
		return roomId;
	}

	public void setRoomId(int roomId) {
		this.roomId = roomId;
	}

	public String getRoomName() {
		return roomName;
	}

	public void setRoomName(String roomName) {
		this.roomName = roomName;
	}

	public Date getCreatedDate() {
		return createdDate;
	}

	public void setCreatedDate(Date createdDate) {
		this.createdDate = createdDate;
	}

	public Date getModifiedDate() {
		return modifiedDate;
	}

	public void setModifiedDate(Timestamp modifiedDate) {
		this.modifiedDate = modifiedDate;
	}

}
```

Database Connection Code:

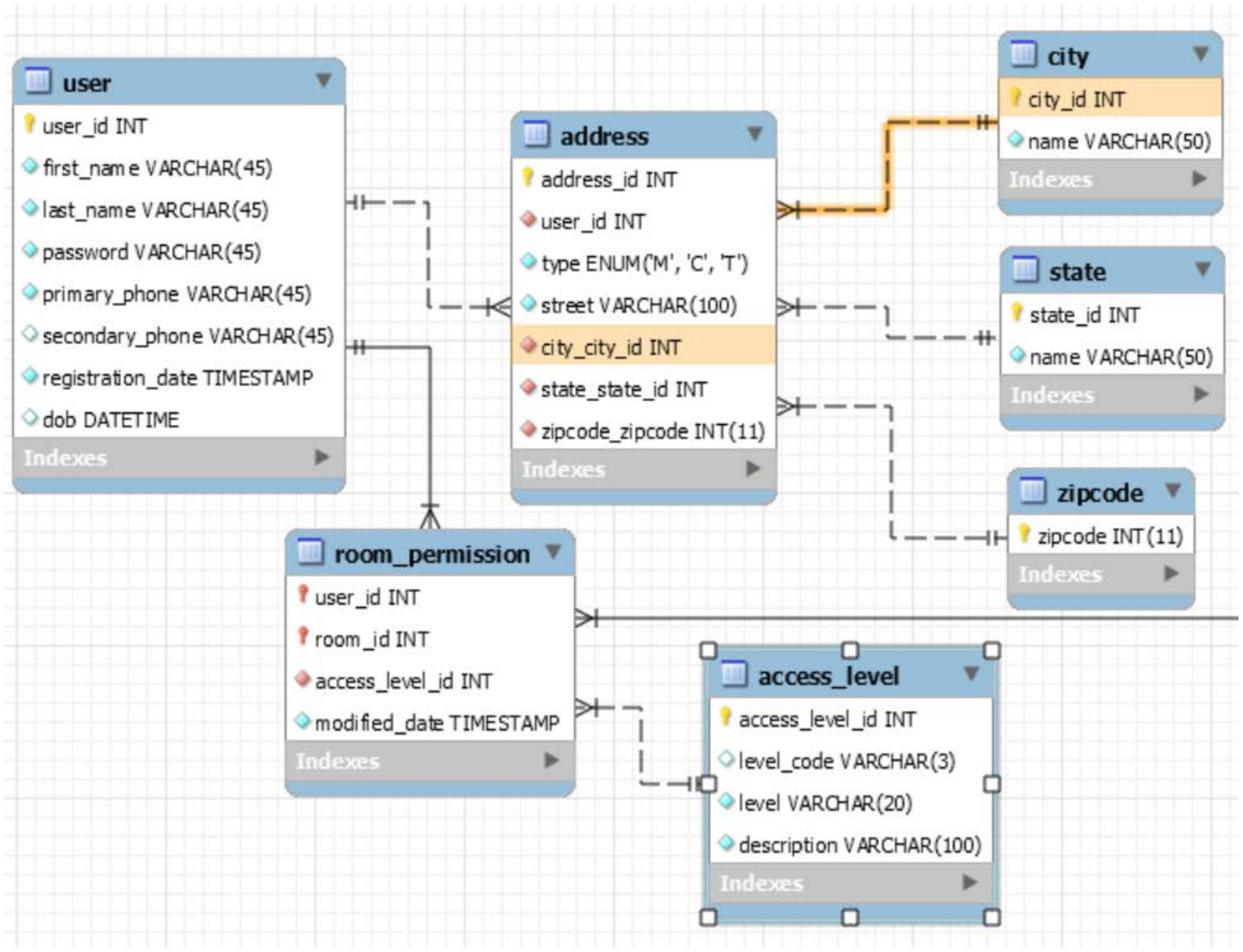
```
package com.planthouse.util;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
import org.hibernate.cfg.Configuration;

public class ApplicationContext {
        private static Configuration configuration;
        private static Session sess;
        private static SessionFactory sessionFactory;
        private static StandardServiceRegistryBuilder builder;
        public static SessionFactory getHiberSession() throws Exception{
                if(sessionFactory!=null){
                        System.out.println("sessionFactory is conected "+ sessionFactory.isClosed());

                }

                if(sessionFactory==null){
                        if(sessionFactory!=null)
                                System.out.println("the session is not null"+sessionFactory);
                        configuration = new Configuration().configure("hibernate.cfg.xml");
                        builder = new StandardServiceRegistryBuilder().applySettings(configuration.getProperties());
                        sessionFactory = configuration.buildSessionFactory(builder.build());

                }
                return sessionFactory;
        }

        public static void cleanUp(){
                try{
                if(sess!=null)sess.close();
                }catch(Exception e){
                        e.printStackTrace();
                }
                System.out.println("Cleanup done");
        }
}
```

Data Access Code:

```
public int isValidUser() {
                Session session = null;

                try {
                        session = new ApplicationContext().getHiberSession().openSession();
                        Query qry = session.createQuery("from User u where u.userId=:userId");
                        qry.setString("userId", "1");

                        User user = (User) qry.uniqueResult();

                } catch (Exception e) {
                        e.printStackTrace();
                } finally {
                        session.close();
                }
                return 1;
        }
```

8 Design Units Impacts

The project we develop will have only one design unit which consists of the three sub unites. They are View, Controller and model. This design totally covers all the requirements and fulfils the project. The View will have all the front end files which will be developed by using the HTML and CSS. We are doing the responsive page designing using Bootstrap. The controller will contain the connection files. The connection from the front end and database is done by using the spring and Hibernate framework. Model will contain the database developed by using My SQL workbench tool. The database is totally normalized. The following are the Database screen shots.



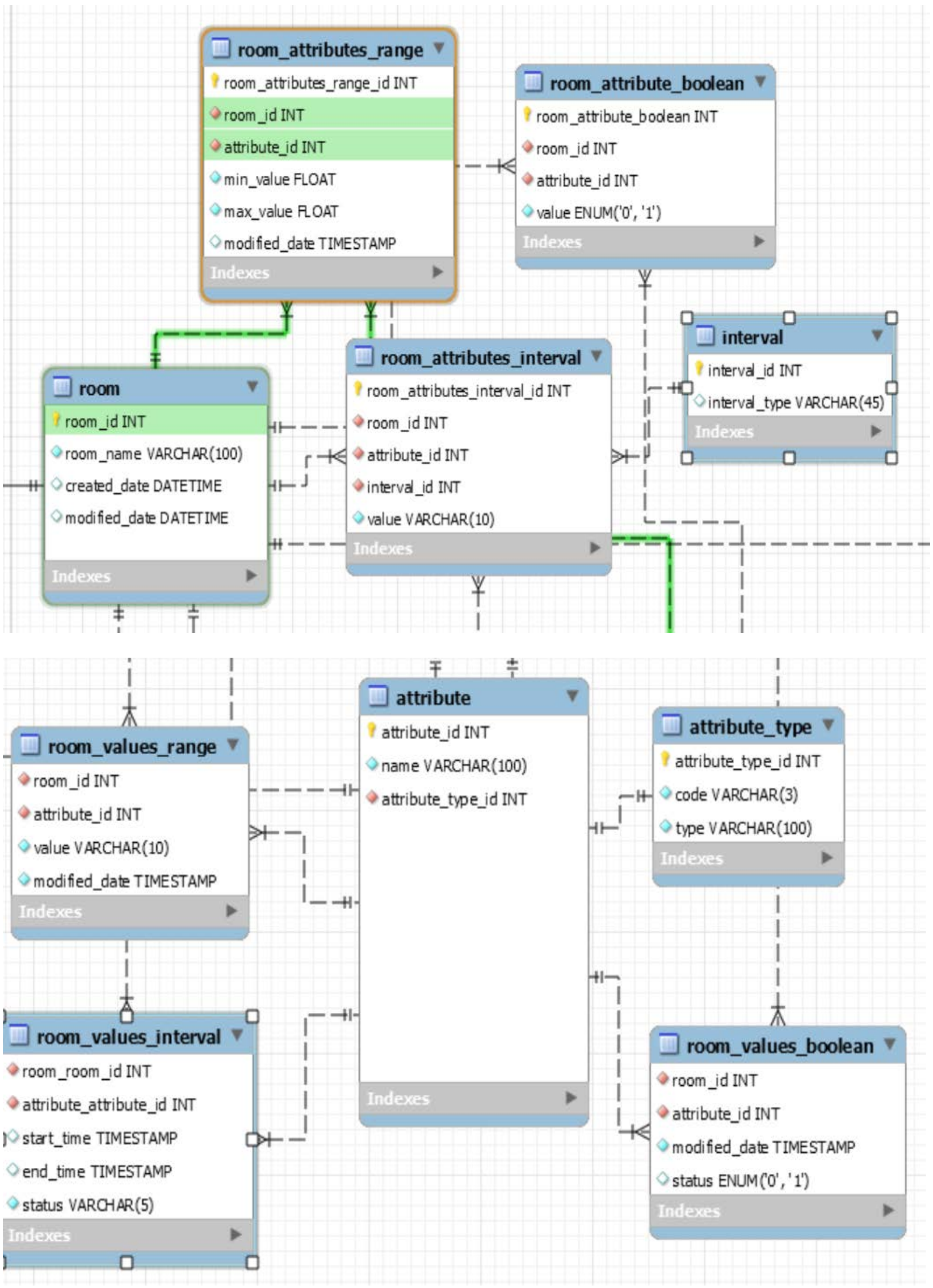


FIGURE 3 DATA MODEL DIAGRAMS.

8.1 Functional Area A/Design Unit A

8.1.1 Functional Overview

The three main parts for the development phase are present in this design unit. These impact are considered essential and the requirements are written accordingly.

8.1.2 Impacts

The complete project depends on this part because we have only one design unit. This design unit consist of the parts where we create a database and a front end application and connect both of them by using the controllers. Connecting the front end and the database will be a challenging task in this design.

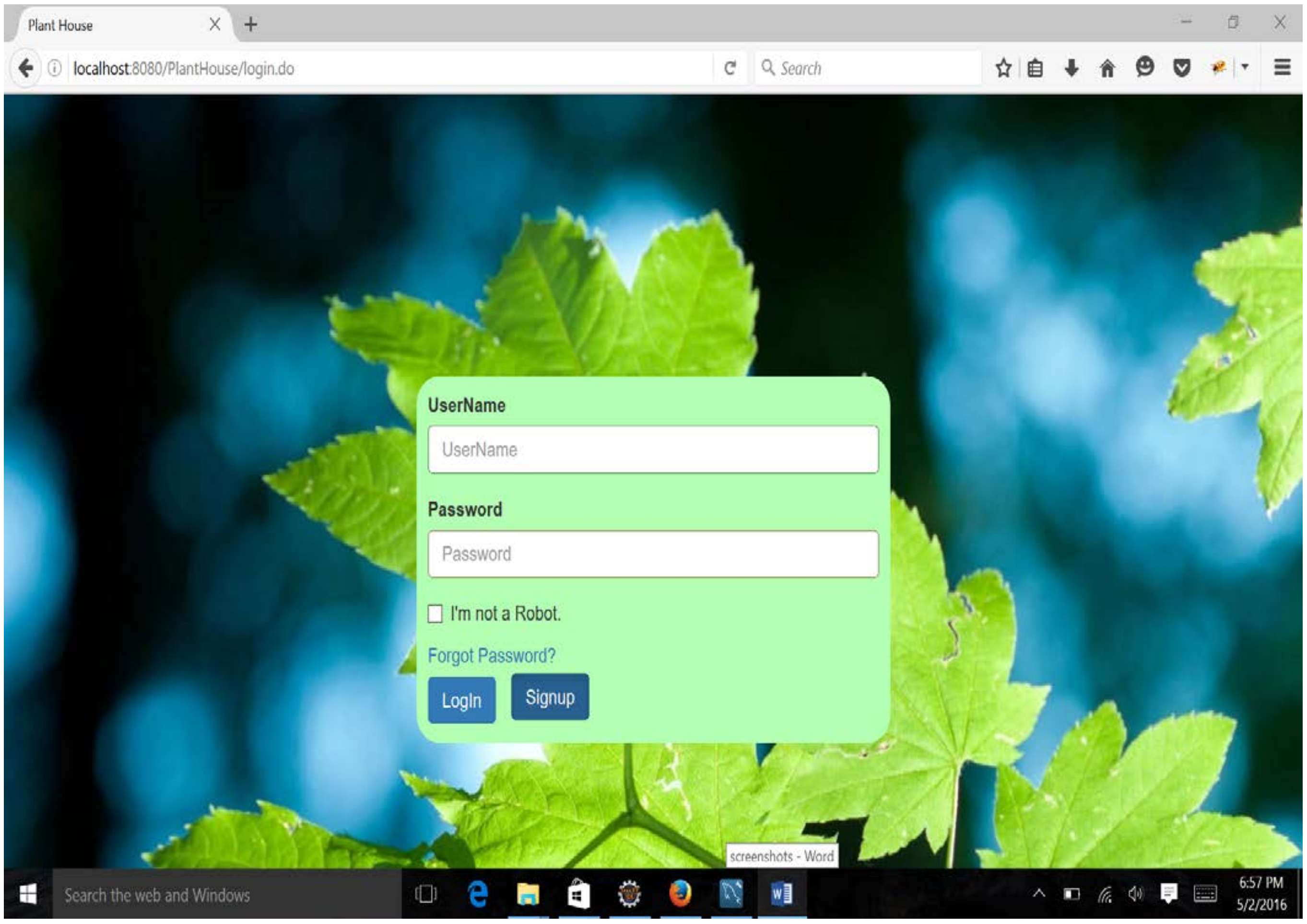
8.1.3 Requirements

We have totally 63 requirements all the requirements are covered under this functional unit. We have already listed the requirements in the above section. To go through the requirements you can please refer to the section 3.1

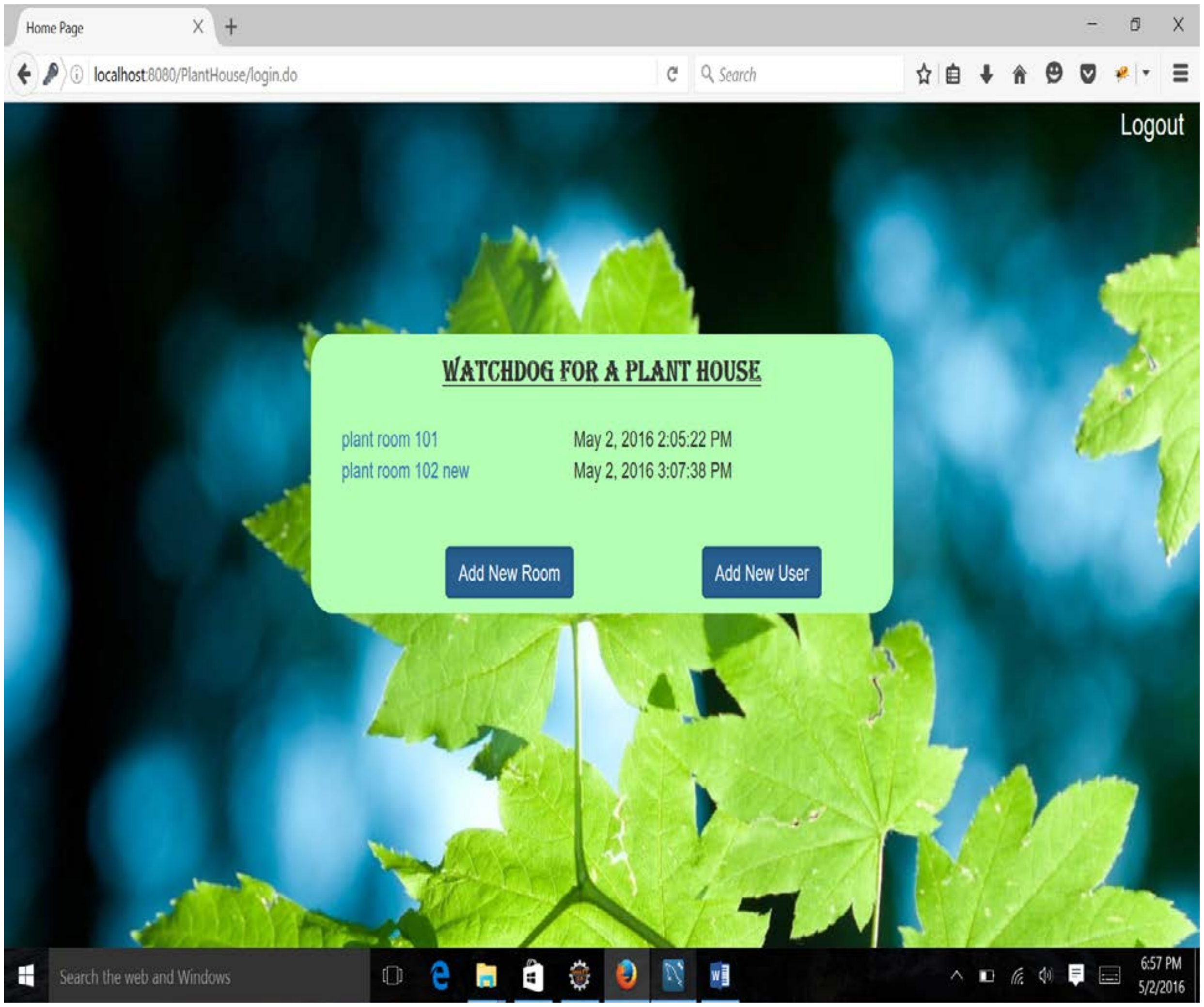
9 *Screen Shots*

In this section we show the screen shots of our application user interface.

User page:

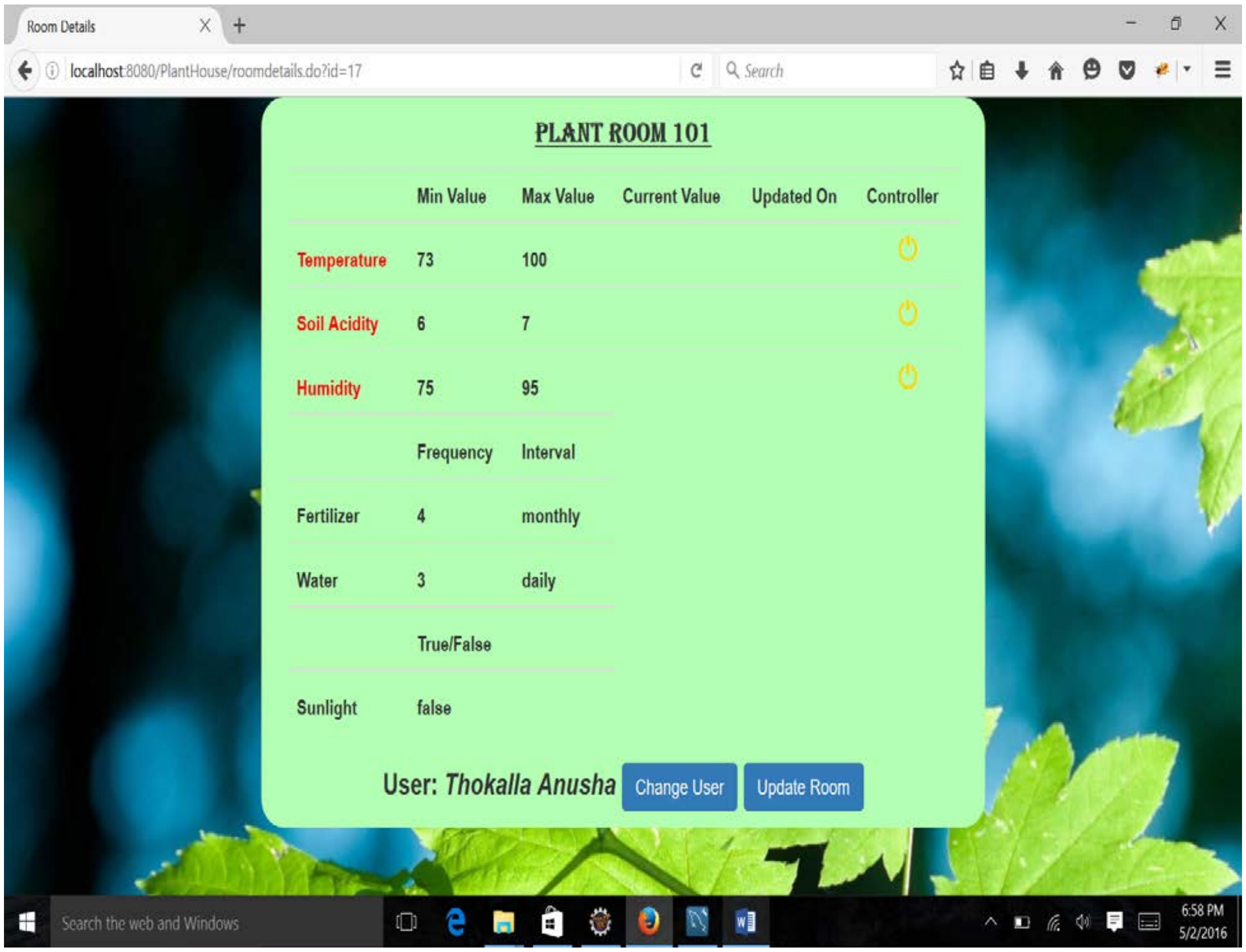


Home page:


Home Page
localhost:8080/PlantHouse/login.do
Search
Logout
WATCHDOG FOR A PLANT HOUSE
Add New Room
Add New User
Search the web and Windows
6:57 PM
5/2/2016

plant room 101	May 2, 2016 2:05:22 PM
plant room 102 new	May 2, 2016 3:07:38 PM

Room Details:



Room Details

localhost:8080/PlantHouse/roomdetails.do?id=17

PLANT ROOM 101

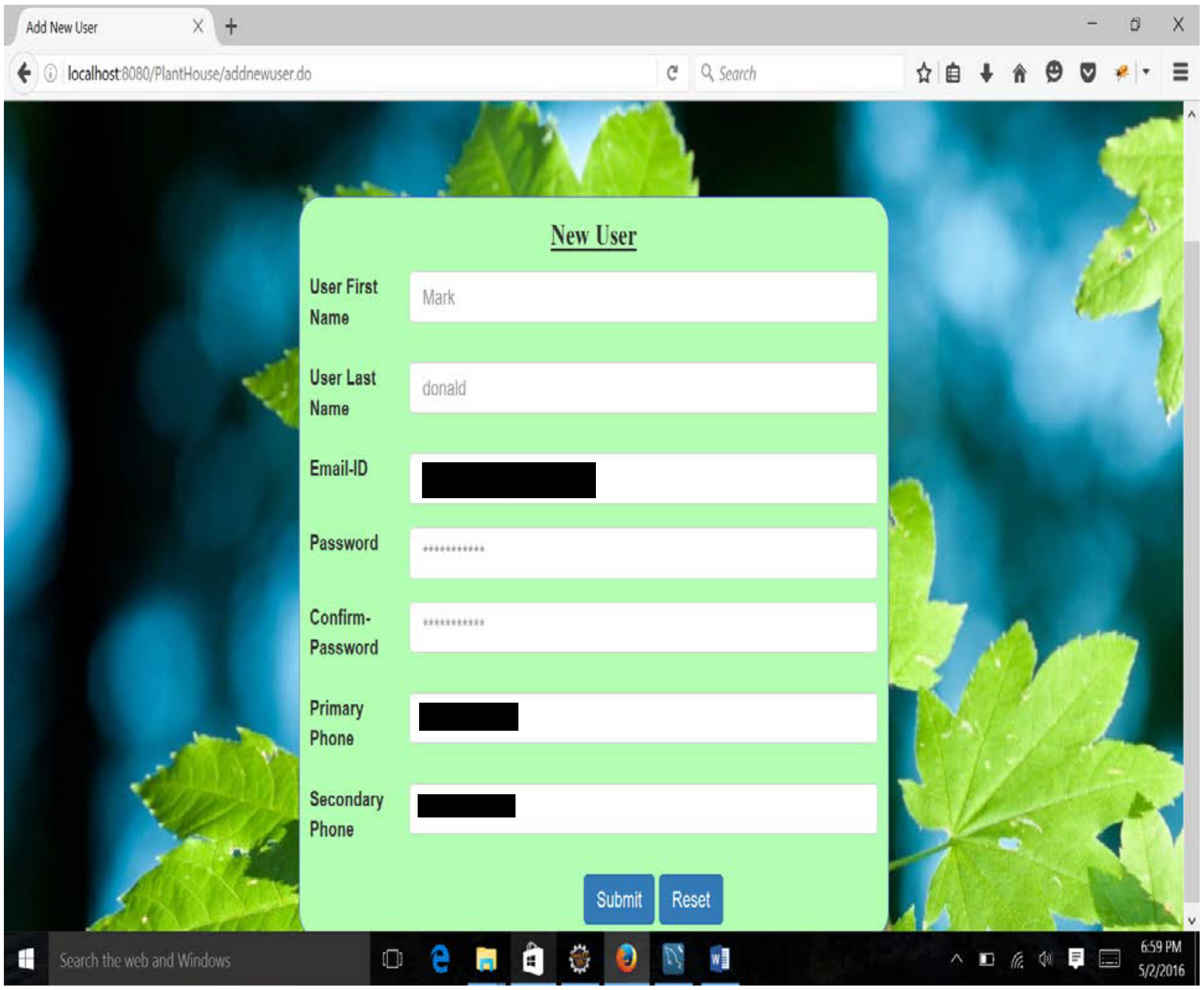
	Min Value	Max Value	Current Value	Updated On	Controller
Temperature	73	100			⏻
Soil Acidity	6	7			⏻
Humidity	75	95			⏻

	Frequency	Interval
Fertilizer	4	monthly
Water	3	daily

	True/False
Sunlight	false

User: *Thokalla Anusha*

Add new user:

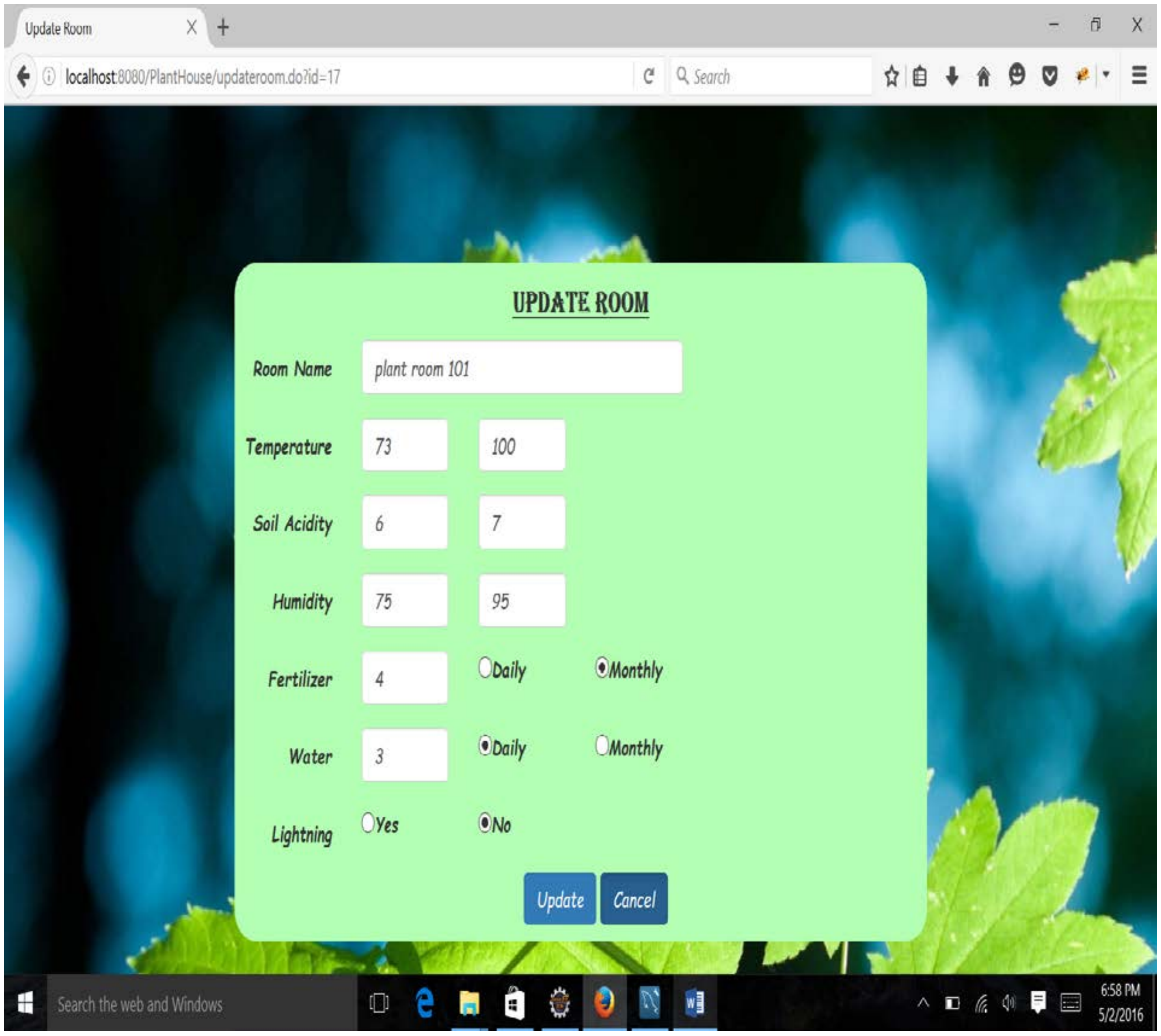

Add New User
localhost:8080/PlantHouse/addnewuser.do
Search
Search the web and Windows
6:59 PM
5/2/2016

New User

User First Name	Mark
User Last Name	donald
Email-ID	[redacted]
Password	***********
Confirm-Password	***********
Primary Phone	[redacted]
Secondary Phone	[redacted]

Submit Reset

Update room:



Update Room

localhost:8080/PlantHouse/updateroom.do?id=17

UPDATE ROOM

Room Name	plant room 101		
Temperature	73	100	
Soil Acidity	6	7	
Humidity	75	95	
Fertilizer	4	Daily	[✓ Monthly]
Water	3	[✓ Daily]	Monthly
Lightning	Yes	[✓ No]	

10 Acknowledgements

We take this opportunity thank each and every person who has involved in this project and supported us in the way a completing this project successfully. We thank Dr. Soon Ok Park for guiding us throughout the project work. We have received and immense support from through her valuable advices, suggestions and correction in the process of completing our application.

We are also grateful to our College of Arts and Science, Computer Science Department, how were encouraging us in finding the new solutions for the existing system. Our Gratitude to our Governors State University to accepting our project and providing resources for completing it.

We thank our self as a team, we have achieved this success through complete team work and dedication toward our field.

11 References

All references should include, author, title of document, doc ID# and issue date.

[1]. Green House, April 2016, Wikipedia Green House, Retrieved from:

https://en.wikipedia.org/wiki/Greenhouse#cite_ref-2

[2]. Sommerville I., Software Engineering 9th edition, Delhi: Pearson Education, Ltd,

[3]. Temperature Sensor. Maximum Integrated 2015, Retrieved from:

https://www.maximintegrated.com/en/glossary/definitions.mvp/term/Temperature%20Sensor/gpk/846#

[4]. Soil pH, From Wikipedia, Retrieved from:

https://en.wikipedia.org/wiki/Soil_pH

[5]. Spring Framework – Overview, Jan 2016, .Tutorialspoint, Retrieved from:

http://www.tutorialspoint.com/spring/spring_overview.htm

[6]. Hibernate ORM, Idiomatic persistence for Java and relational databases. LGPL V2.1, Retrieved from:

http://hibernate.org/orm/

[7]. Postman, Improve API workflow, blog, Postdot Technologies, retrieved from:

http://blog.getpostman.com/

[8]. Maven, Apache Maven Project, The Apache Software Foundation. Retrieved from:

https://maven.apache.org/what-is-maven.html