



MIDDLE EAST TECHNICAL UNIVERSITY
COMPUTER ENGINEERING DEPARTMENT



CENG 491 – Computer Engineering Design – I

SMART HOME PROJECT

INITIAL DESIGN REPORT

v1.1



Abdullah Hasan Taher Bayrakdar

Anıl Ulutürk

Şerafettin Öztürk

Zeynep Mavuş

Sponsored by



List of Changes

Date	Revision	Comment
03/12/2012	1.0	Created
18/01/2013	1.1	References updated
18/01/2013	1.1	Tools updated
18/01/2013	1.1	Traceability Matrix added
18/01/2013	1.1	User initiated events added
18/01/2013	1.1	Figure: “Component Diagram of the System” Changed
18/01/2013	1.1	GUI updated
18/01/2013	1.1	Data Design updated
18/01/2013	1.1	Constraints updated – security constraints added

Table of Contents

1. Introduction	5
1.1 Problem Definition.....	5
1.2 Purpose	6
1.3 Scope	6
1.4 Overview	7
1.5 References	7
1.6 Definitions and Acronyms.....	9
2. System Overview	10
3. Design Considerations	12
3.1. Design Assumptions, Dependencies and Constraints	12
3.1.1. Assumptions, Dependencies	12
3.1.2. Constraints	12
3.2. Design Goals and Guidelines	13
3.2.1. Availability & Reliability	13
3.2.2. Extensibility	13
4. Data Design	14
4.1. Data Description	14
4.1.1. Description of Data Entities	15
4.2. Data Dictionary	17
4.2.1. Statistics Table	17
4.2.2. Device Entity	17
4.2.3. Sensor Data.....	18
4.2.4. Commands Table	18
4.2.5. Adjustment Data	18
5. System Architecture.....	19
5.1. Architectural Design	19
5.2. Description of Components	20
5.2.1. GUI.....	20
5.2.2 Authentication	22
5.2.3 User	24
5.2.4 Back End Applications.....	25
5.2.5 Data Retrieval from Data Storage.....	26
5.2.6 Data Retrieval from Raspberry PI	27
5.2.7 Data Storage	29
5.2.8 Raspberry PI	31
6. User Interface	32
6.1. Overview of the User interface	32
6.2. Screenshots	32
6.2.1 Sign In	32
6.2.2 Select Category	33
6.2.3 Adjust Configurations	34
6.2.3 Sign Out.....	39
6.2. Screen Objects and Actions	39
7. Dictionary of User Initiated Events.....	40
8. Traceability Matrix	40

9. Time Planning.....	41
9.1. Term 1 Gantt Chart	41
9.2. Term 2 Gantt Chart	42
10. Libraries and Tools	43
10.1 MPLAB X IDE	43
10.2 NetBeans IDE	43
10.3 PHP	43
10.4 HTML5.....	43
10.5 JavaScript	43
10.6 CSS.....	44
10.7 XAMPP	44
10.8 MySQL.....	44
10.9 StarUML.....	44
10.10 SmartDraw.....	44
11. Conclusion.....	45

1. Introduction

This documentation has aim to indicate initial design strategies and structural properties of the Smart Home project. This document contains system architecture and implementation phases in terms of software components, interfaces, and data. All information about these steps is explained clearly for better code development.

1.1 Problem Definition

In this project, a general purpose wireless controller hardware that controls the home appliances and various sensors, and master controller software working on an embedded Linux installed board will be developed. Master controller will collect data, show data in a user friendly interface, let user send control commands to appliances, push data to web services if necessary, and also provide web based control of devices. Zigbee is the wireless protocol to use. Master controller is developed on a Linux box – Raspberry PI.

Moreover, efficiency and usability of home automation systems will be simplified and designed in a generic manner. In order to apply “smart home” ideas to any house, certain requirements of electronic devices at home, light fixtures exist and they will probably cost much to renovate than “smart home” system devices themselves. Our system will be lowering the cost of these expenses. Network constituted of ZigBee compatible devices will serve as data medium, whereas PIC embedded devices with a collection of sensors will be responsible of collecting and reporting statistical data to master controller. Advantage of our approach from the ones in market is more general compatibility ability meaning that a more general communication protocol will be used to address much more devices to make our design more valuable. Also, master controller will possibly realize simple machine learning techniques to provide home the ability of learning efficiency strategies over time. In order to achieve this, we need to combine a more generic sensor collection, more than smart homes systems currently on the market. Additionally, we will work on a user friendly interface after finishing up the hardware part of the project.

Statistical data collected from end devices will be collected in master device and will be pushed to a web server for user friendly display and control of certain properties of those end devices. Local data storage in case of emergency and internet connection loss will be provided in Linux Box with certain limitations, as well.

In short, our project will address certain issues in smart home industry and these ideas can be listed as follows:

- **Usability:** Generic set of sensors will provide a proper set of information to achieve from home appliances.
- **Ease of application:** With a fixed box of end device, every home appliance will be addressed in same way, in turn allowing for ease in application to any home.
- **Machine learning:** Coordinator box will work as a reflexive agent, adapting to the tendencies of home owners, applying efficiency plans accordingly.
- **Safety and Reliability:** Necessity for emergency case plans in smart homes will be addressed in this project with a low cost.

1.2 Purpose

Purpose of the document is to help reader understand and visualize solution process to the problem presented previously in SRS document. Stakeholders can also use the document to check design constraints satisfied in the final product.

As a predecessor to implementation phase, document is intended to provide a stronger ground through development process and to increase quality of the final product, using the SDD document guidelines from IEEE 1016-2009.

1.3 Scope

This document includes all design elements, which will be existent in delivered project. A set of design views and viewpoints will be presented in order to demonstrate design and development of the project in closely related yet discrete perspectives.

Document also constitutes the last step before implementation phase and will cover development and design phases, and aimed to guide especially implementation phase of project.

1.4 Overview

This document may be used with Software Requirements Specification Document to grasp functionalities provided by project and their representative design entities for fulfilling those functionalities. Software Developers and System Designers can use this document to gain insight about fundamental aspects of this project and how its purpose is achieved by structural means, based on certain constraints such as user interface, system architecture and timing of the development progress of the project.

In the next chapter we will give a general overview of the system. Then the Design Considerations follows in which assumptions, dependencies, constraints and guidelines of our system will be explained. The chapter after that is the Data Design which will include information and data domain of the system and its organization. In the fifth chapter, we will architecture of our system in detail with description of components. Then we will present images for our user interface. After, libraries and tools will be specified. Gantt charts of term1 and term2 will be presented and document is finished with conclusion, where we sum up intentions and information within this document.

1.5 References

- [1] IEEE Std 1016-2009 IEEE Recommended Practice for Software Design Description. IEEE Computer Society, 2009.
- [2] MPLAB X IDE. Retrieved 01.12.2012 from
<http://www.microchip.com>
- [3] Netbeans Integrated development environment. Retrieved 01.12.2012 from
<http://netbeans.org>
- [4] w2schools.com PHP tutorial. Retrieved 01.12.2012 from
<http://www.w3schools.com/php>
- [5] HTML5 definition. Retrieved 01.12.2012 from
<http://www.comparebusinessproducts.com/fyi/wtf-is-html5>
- [6] Javacript description. Retrieved 01.12.2012 from
<http://www.roseindia.net/javascript/what-is-javascript.shtml>

- [7] Cascading Style Sheets description. Retrieved 01.12.2012 from
<http://www.wellesley.edu/Computing/Dreamweaver/CSS/cssMain.html>
- [8] Apache HTTP Server. Retrieved 01.12.2012 from
<http://www.apachefriends.org/en/index.html>
- [9] MySQL definition. Retrieved 01.12.2012 from
<http://www.bluemooosetech.com>
- [10] StarUML - The Open Source UML/MDA Platform. Retrieved 01.12.2012 from
<http://staruml.sourceforge.net/en/>
- [11] Smartdraw visual processor. Retrieved 01.12.2012 from
<http://www.smartdraw.com/>
- [12] Advantages of ZigBee. Retrieved 17.01.2013 from
<http://www.versalinks.com/FAQ.html>
- [13] Features of Raspberry PI. Retrieved 17.01.2013 from
<http://www.raspberrypi.org/faqs>

1.6 Definitions and Acronyms

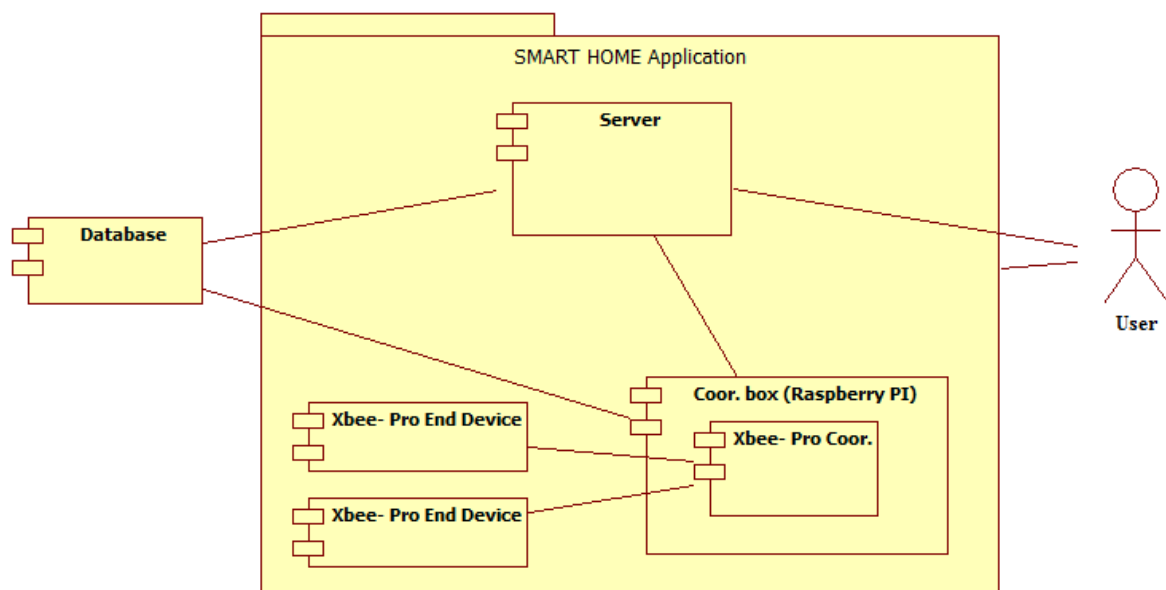
Term	Definition
IDR	Initial Design Report
User	The person(s) who use the SMARTHOME application
XBee	Is the brand name from Digi International for a family of form factor compatible radio modules.
PIC	Is a family of modified Harvard architecture microcontrollers made by Microchip Technology.
SRS	Software Requirements Specifications
Raspberry PI	Is a credit-card sized computer that plugs into your TV and a keyboard. It's a capable little PC which can be used for many of the things.
Zigbee	A specification for a suite of high level communication protocols using small, low-power digital radios based on an IEEE 802 standard
IEEE	Institute of Electrical and Electronics Engineers
GNU	Unix-like computer operating system developed by the GNU Project
GUI	Graphical User interface

2. System Overview

SMARTHOME is a generic solution for home automation enhanced with machine learning techniques. It uses a well-known wireless platform Zigbee for communication between devices and allows you to collect data and control home appliances from a single web interface. Main program running on Coordinator Box (Raspberry PI) is open source with a GNU General Public License.

The system is mainly consists of the following components:

- Environment and device sensors collection (PIC boards)
- Wireless network(*One*XBee-Pro coordinator and *Many*XBee-Pro End devices)
- Coordinator Box (Raspberry PI)
- User Interface



As home appliances provide various data from their own work conditions and status, sensor collection provided as a generic device analyzes the data and sends it through wireless network to coordinator box. Receiving the data through local wireless system, coordinator box collects and stores the data on an online server. According to users' choices or previous statistics, it can act as a smart agent to provide certain reflexes in case of emergencies and possible threats. All scenarios of actions and status information can be viewed through internet (web server) in which statistical data are accumulated. As well as viewing, user can send orders to the coordinator box in order to change home appliances' activities in real time.

3. Design Considerations

3.1. Design Assumptions, Dependencies and Constraints

3.1.1. Assumptions, Dependencies

Smarthome system is a platform dependent. Embedded Linux working on a Raspberry PI device or PC with linux OS is a necessity.

It is assumed that users of the SmartHome system are familiar with the features of home devices and know how to adjust appliances like refrigerator, washing machine ..etc. In addition, the users should have basic computer skills to use the system interface.

3.1.2. Constraints

The network throughput of the system should be at least 250 Kbps.

The HomeSmart application will be interoperable. This means that it would be able to work with other products or systems using zigbee communication protocol in the environment without any restricted access or implementation.

The response time of the system should be less than 1 second. For instance, the user would like to see the adjustments he/she made on the device just after setting the new values.

Security is another design constraint and this constraint can be considered in 2 different aspects of the system.

The first aspect is the closed circuit electronic hardware running at home. The system should encrypt the data transmitted between the Xbee devices. As well as checking Zigbee API frames' checksum control implemented in both home embedded device control hardware and Control Box (Raspberry PI), system will be protected against unauthorized access by logging of Zigbee device API's associated with every home appliance at home.

The second one is the authentication associated with remote control interface built into the server. Several distinct user ids and passwords can be assigned to system such that every user can have different privileges over devices at home, in turn provides additional security for specific device types.

3.2. Design Goals and Guidelines

3.2.1. Availability & Reliability

The system should service with the 7 days / 24 hours availability. The communication between the devices should be stable and the database connection should always exist. To be sure that the system will run without any bugs or problems a general test application will be prepared for this purpose.

3.2.2. Extensibility

This project designed to be extensible. It means any updates and changes have to be done easily. Our code implementation will be done with regard of this feature.

Our goal is to develop a Smart Home Automation system with general compatibility ability meaning that more general communication protocol will be used to address much more devices. Also, master controller would realize simple machine learning techniques to provide home the ability of learning efficiency strategies over time.

4. Data Design

Data structures of the project are generally defined by kinds of information must be transferred to and from home appliances regarding their work status, progress and real time conditions.

4.1. Data Description

For the sake of data consistency and safety, there will be two different databases built on two discrete components of the project.

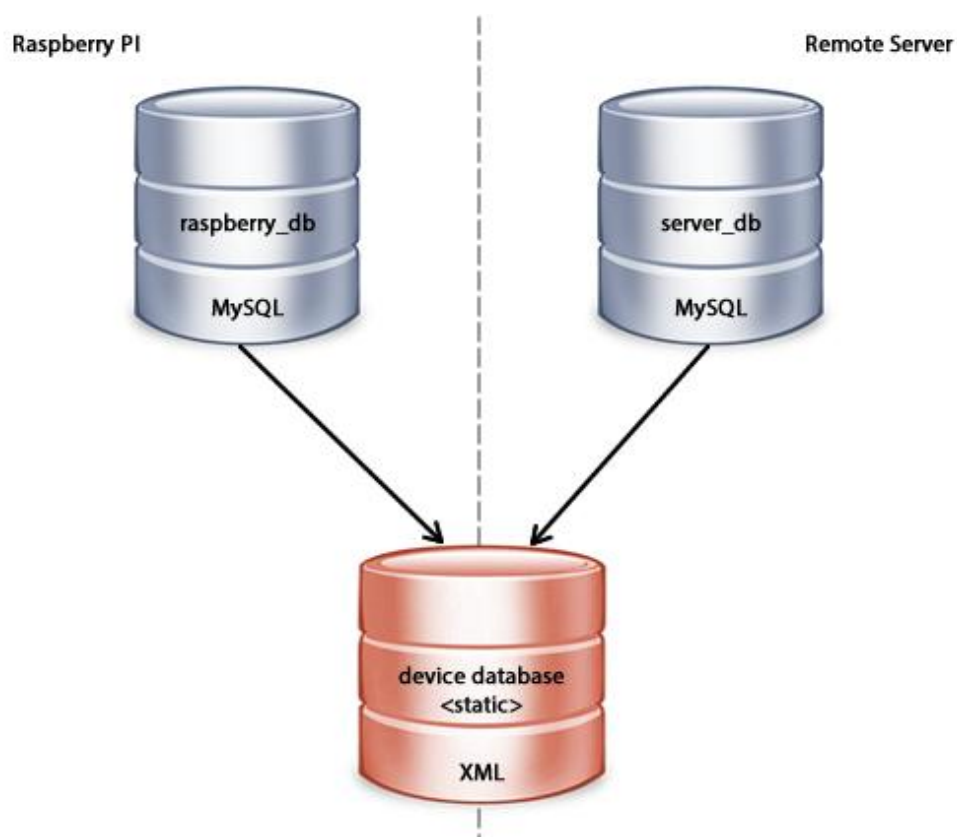


Figure: Database Relationship Diagram

First one of these databases will be working on server side, which will be providing statistical data collection about home appliances and will serve as user interface (client) for remote control of home appliances. This database will hold time stamped data received from Coordinator Box at home and display this data to user through a web interface accessible through internet. Commands and adjustments of home appliances in remote connection will be done through this interface described previously. In this database; devices, statistics, data_packages tables will be present.

Second database will be working on Coordinator Box, holding a collection of adjustments, commands and control orders addressing home appliances. In case of internet connection loss and/or malfunction on Ethernet connection, coordinator device will continue remaining command items previously received from Server, provided by user interface. In this database; devices, commands, current_status, adjustment tables will be present.

Both database structures will be dependent on a static db which will be responsible of holding defined home appliance models, features and available sensor data correspond to these devices. This way, raw data collected from home appliances will be identified and displayed in a meaningful form for statistical purposes. Also, commands which are sent to home appliances will be encoded in a device and model specific manner using the static data provided by this database. In this database; device list, sensor data types in conjunction will be present.

4.1.1. Description of Data Entities

4.1.1.1. Statistics Table

In the database running on remote server, various kinds of sensor and status data will be collected from home appliances. For the sake of simpler design, every kind of appliance will have various number of data entities (“Sensor Data” entity in ER diagram below), which will be attached to a specific device. Appliances such as thermostat will be sending heat data in float form, lighting will be sending only on-off status information and refrigerator will provide heat adjustments and real time sensor data regarding current heat to the database, as well as status about proper workstate of the appliance. All these data will be kept in “Sensor Data” entity, consisting of an identification name (data property) and associated value of it.

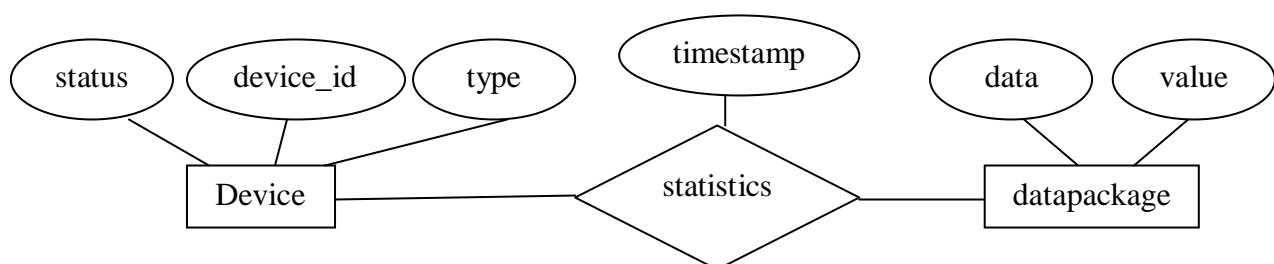


Figure: ER Diagram – Statistics Table

4.1.1.2. Commands Table

As user can turn on/off any device through web interface, one kind of basic command will be on/off. Thermostat and refrigerator can be issued with adjustment commands, and washing machine can be issued with various setting selections regarding spin speed, cloth type etc. To support these various operations, database will hold every distinct setting to be adjusted in a separate command and issues will be given in terms of id values associated with these selections' values in range of device compatibility. For instance; spin speeds 800, 1000, 1200 will be represented as integer values 0, 1, 2 in a database entry, combined with an identifier string addressing specific setting of a selected device. All of these particular commands will be represented as a relation between “Device” and “Sensor Data”, issued with a predefined priority.

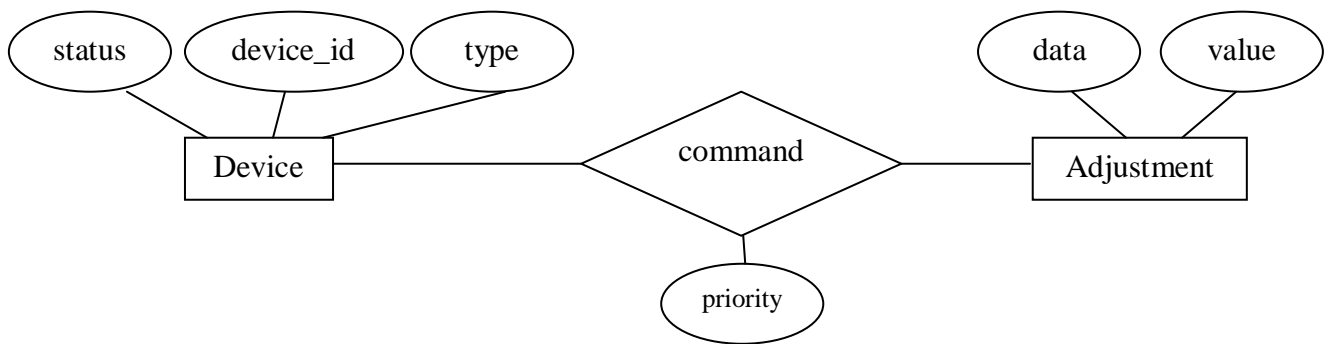


Figure: ER Diagram – Commands Table

4.1.1.3. Static Device Table

In order to interpret every single command or raw data package received from a device, a static database will be kept in XML format. This database will work as a lookup table for encoding “statistics” and decoding “commands” on Raspberry PI and server side. An example data structure of this database can be seen in following ER diagram.

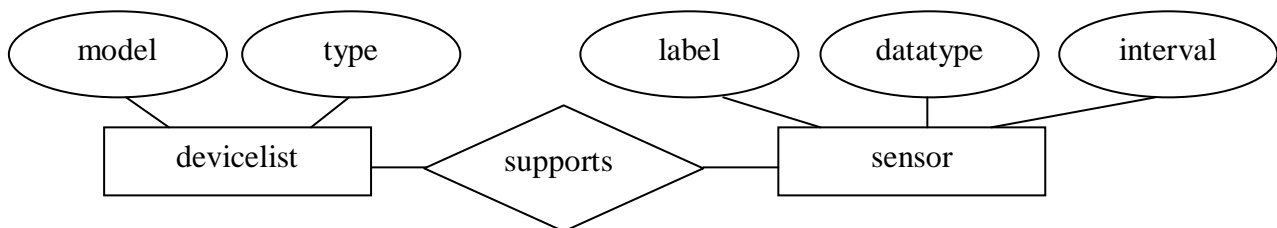
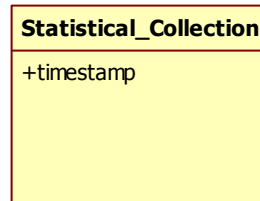


Figure: ER Diagram – Static Device Table

4.2. Data Dictionary

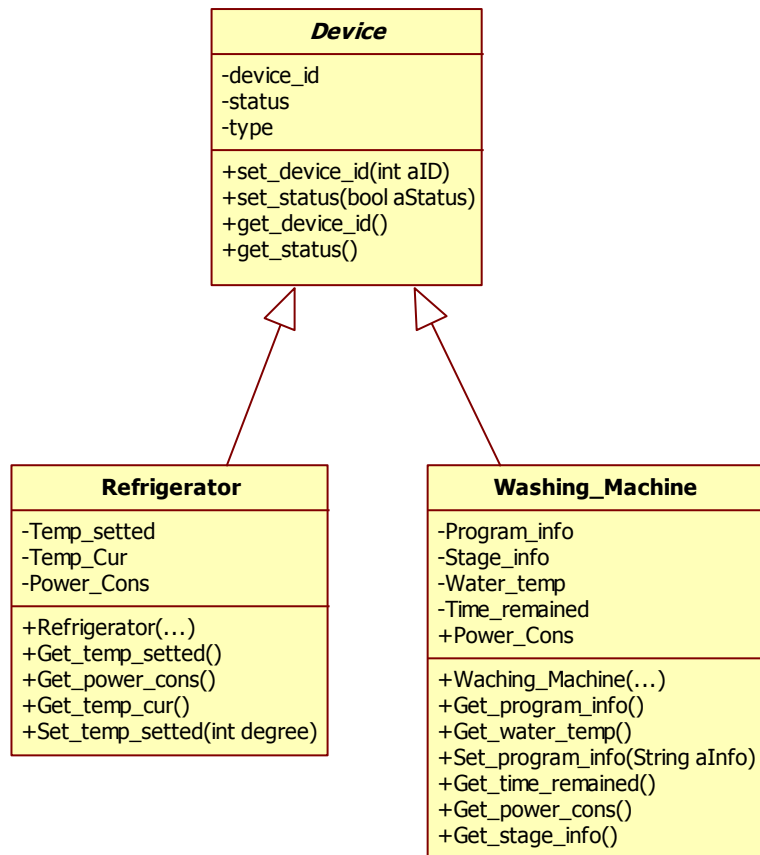
4.2.1. Statistics Table

A relational database collection based on “Device” entity and “Sensor Data” associated with one another, attributed with a “timestamp” as a unique identifier. Logically, located on remote server accessed by user, through internet.



4.2.2. Device Entity

Abstract entity of a home appliance.



status: A flag variable containing current status of the appliance. May indicate a malfunction or problem in device, or proper workstate.

device_id: A unique id which is used when identification of the home appliance.

type: Type of home appliance; may be a lighting, thermostat, refrigerator etc.

4.2.3. Sensor Data

Abstract entity of a dictionary like data id and value mapping.

-

<i>Sensor_Data</i>
<div><div>-data</div><div>-value</div><div>-timestamp</div><div>+get_data()</div><div>+set_data(string aData)</div><div>+update_value(int aValue)</div><div>+get_all_timestamp()</div><div>+get_value()</div></div>

Contains the following attributes;

data: Identification (in type of string) of sensor data to be stored.

value: Corresponding value of the sensor data.

timestamp: Signifies issue time of a received statistical data.

4.2.4. Commands Table

Another relational database collection based on device entity and “Adjustment”. Priority of those commands will be important when processing their communication orders in Zigbee network.

4.2.5. Adjustment Data

Real time issue regarding adjustable settings.

5. System Architecture

A general description of the Smart Home system architecture is presented in the following sections.

5.1. Architectural Design

Smart Home is a web and embedded base software which is composed of seven components; GUI, Authentication, User, Data Retrieval, Back End Application, Data Storage, Raspberry PI. The component diagram illustrates structure of the system. Each component will be explained separately in the next section.

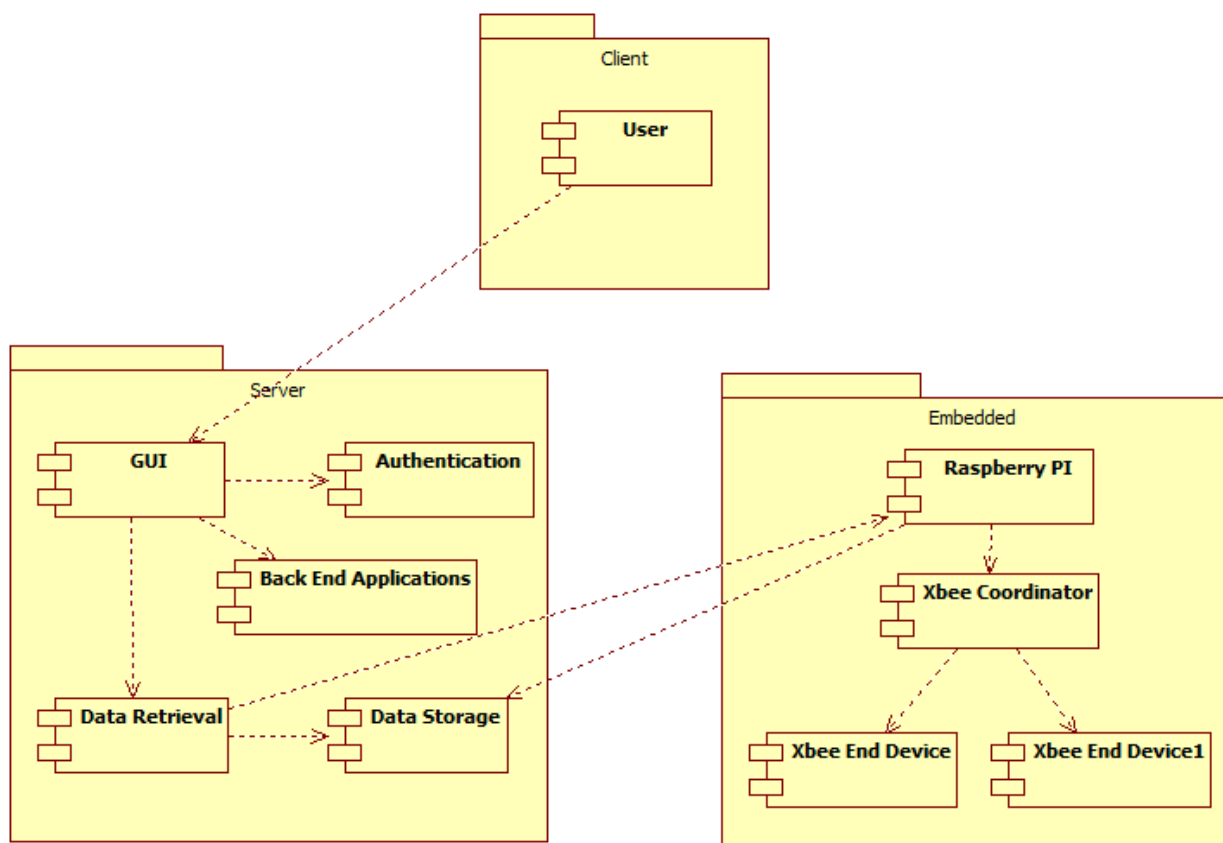


Figure: Component Diagram of the System

5.2. Description of Components

In this section, components mentioned in 5.1 are described individually in subsections. Each description contains short definition, processing narrative for component, interface description, component processing detail and dynamic behavior.

5.2.1. GUI

Graphical user interface provides user to use system easily. This is obtained by using button, textbox..etc.

5.2.1.1. Processing Narrative for GUI

This component comprises all the objects and shows them as appropriate kind. This component works in client side through HTML. Server side sends desired information and client side interpret as desired. This component manages this overflow by creating dynamic HTML webpage with the use of PHP,JavaScript and CSS.

5.2.1.2 GUI Interface Description

There are inputs from client side. Then, this input stack sends to server side with necessity information. Server side processes this information and also makes some information transformation with other components and return desired information as output. On the other hand, this output stack reaches the client side successfully. Client side creates dynamically desired webpage through using output information.

5.2.1.2 GUI Processing Detail

The complete procedural activities related to this component are as follow;

1. User/client requests a page from the system through the internet.
2. Server captures the request successfully.
3. Server evaluates this request inside.
4. Server gets necessary information from data storage component.
5. Server returns information as output.
6. Client side takes information successfully.
7. Client evaluates the server's output.
8. Client creates webpage dynamically and shows it to the user.

5.2.1.4 GUI Dynamic Behavior

This component has relation with authentication and user components. This component interacts with authentication component as authentication component calls this component. User component also calls this component to show itself on webpage. This component depends on receiving information from these two components.

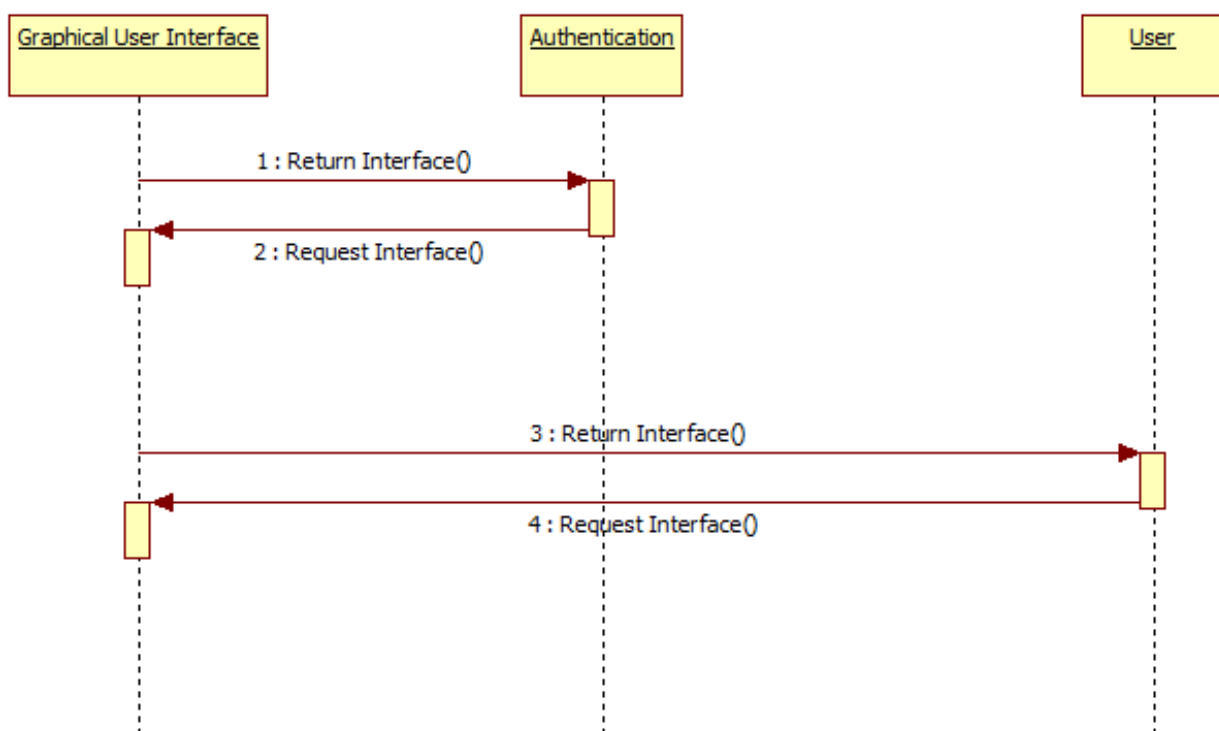


Figure: Sequence Diagram of GUI

5.2.2 Authentication

This component provides to separate people who have permission to enter system and also separate people who are administrator.

5.2.2.1 Processing Narrative for Authentication

This component is responsible for checking personal requests with the permission of the clients. This prevents people who are not related with the system. This is implemented as username and password. At this point, everybody who is related with system have unique username and password. On the other hand, this component separates profile of people who is related with system.

5.2.2.2 Authentication User Interface

This has two areas. First one is username, second one is password. Also there is log in button under these two text boxes. Log in button sends all information in the text box to server side to check. Output depends on whether information is correct or not. User can go next step or try again.

5.2.2.3 Authentication Process Detail

The complete procedural activities related to this component are as follow;

1. User/client fills the username and password area.
2. User/client clicks the log in button under this area.
3. Client side sends these information to server side to check.
4. Server side takes these information successfully.
5. Server side checks information from database.
6. Server side return different information stack whether correct or not.
7. Client side gets information successfully.
8. Client side interprets received information.
9. Client side creates dynamically webpage according to received information.
10. User sees the desired webpage.

5.2.2.4 Authentication Dynamic Behavior

This component has relation with GUI component and user component. This calls GUI component to show itself as webpage. On the other hand, this component directs client to user component according to receiving information from server side.

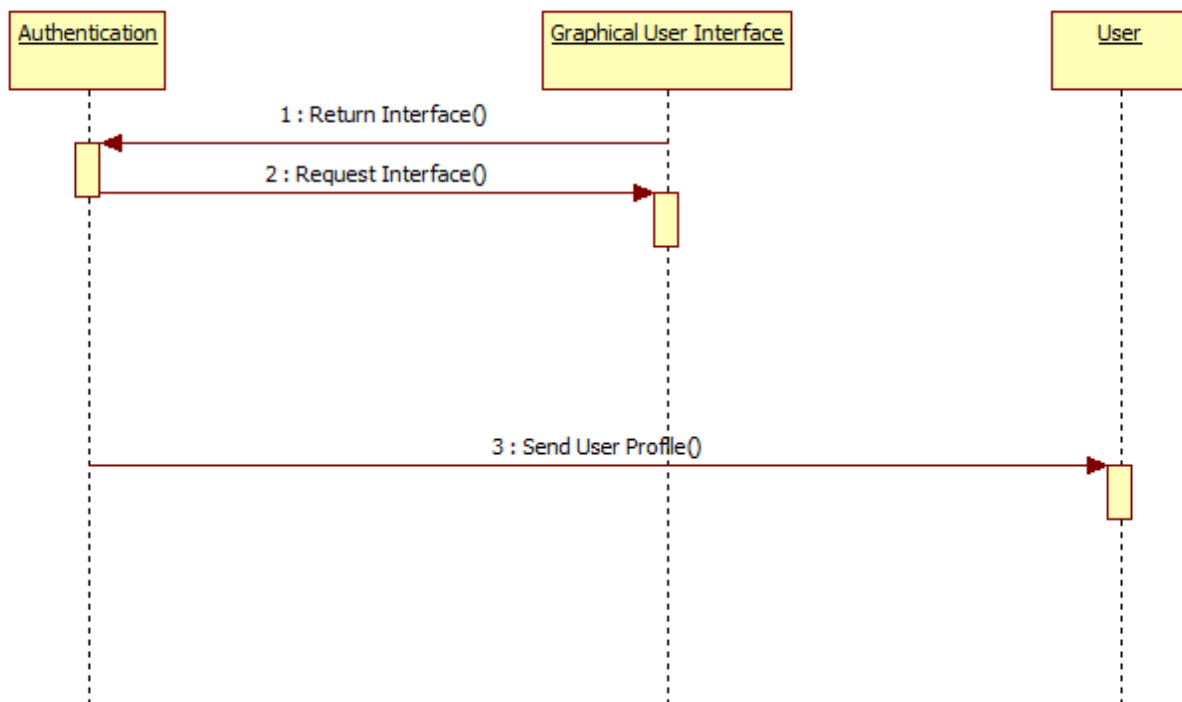


Figure: Sequence Diagram of Authentication

5.2.3 User

5.2.3.1 Processing Narrative for User

This component is responsible for sending requests to Data Retrieval. Only the authenticated components become user components. It has 2 kinds of sub-components as administrator and user. User component has corresponding back end applications. It acts as a bridge between back end applications and data retrieval. User can request a data from Data Retrieval.

5.2.3.2 User Interface Description

It generates data requests for Data Retrieval. By using back end applications requests are processed with corresponding program.

5.2.3.3 User Processing Detail

It works as follows;

1. It generates requests for loading from or storing to database by using backend applications.
2. Results of requests are returned to user component from Data Retrieval.

5.2.3.4 User Dynamic Behavior

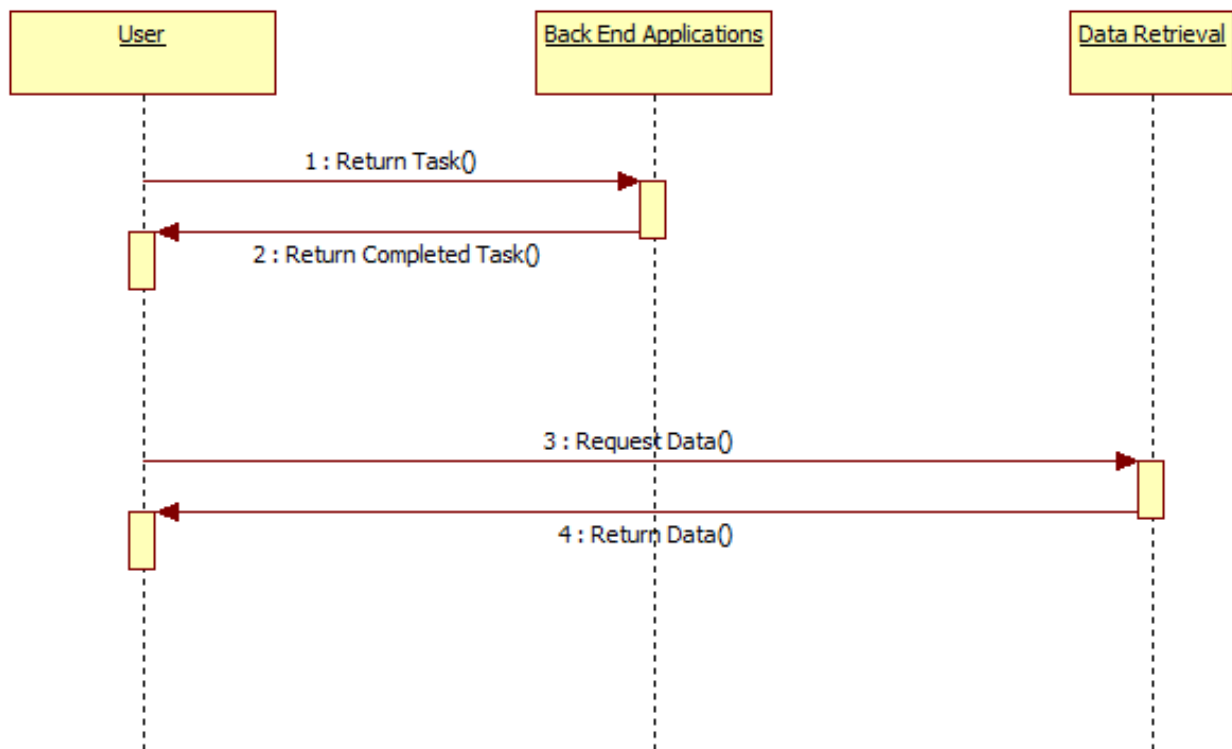


Figure: Sequence Diagram of User

5.2.4 Back End Applications

5.2.4.1 Processing Narrative for Back End Applications

This component is for handling different tasks using technologies available. These technologies include PHP, JavaScript, HTML and CSS.

5.2.4.2 Back End Applications Interface Description

It generates result for User using the User's request as input and produces output.

5.2.4.3 Back End Applications Processing Detail

It works as follows;

1. User requests a task and triggers an appropriate task for back end application.
2. The application returns processed task's result to user.

5.2.4.4 Back End Applications Dynamic Behavior

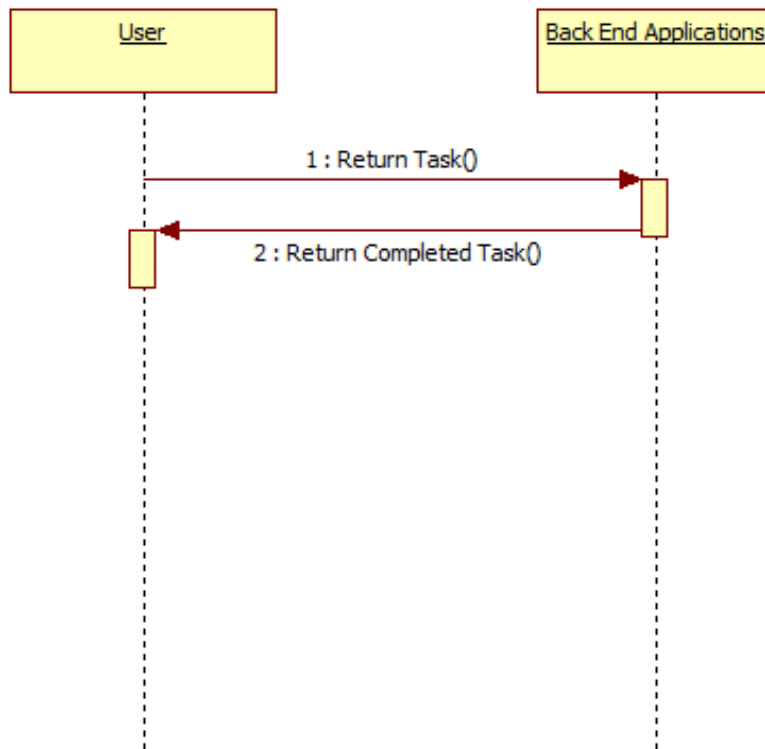


Figure: Sequence Diagram of Back End Applications

5.2.5 Data Retrieval from Data Storage

5.2.5.1 Processing Narrative for Data Retrieval from Data Storage

Data Retrieval is responsible for accessing data from Data Storage. It provides a connection between user and database.

5.2.5.2 Data Retrieval from Data Storage Interface Description

It receives data requests from users of the system. Then, these requests are converted into SQL commands and sent to Data Storage. The result obtained from database is reported to user or component which sent the request.

5.2.5.3 Data Retrieval from Data Storage Processing Detail

It works as follows;

1. Receive a request from User component.
2. Request is converted into SQL commands.
3. Send commands to Data Storage
4. Received object from Data Storage is reported to the user or component which sent the request.

5.2.5.4 Data Retrieval from Data Storage Dynamic Behavior

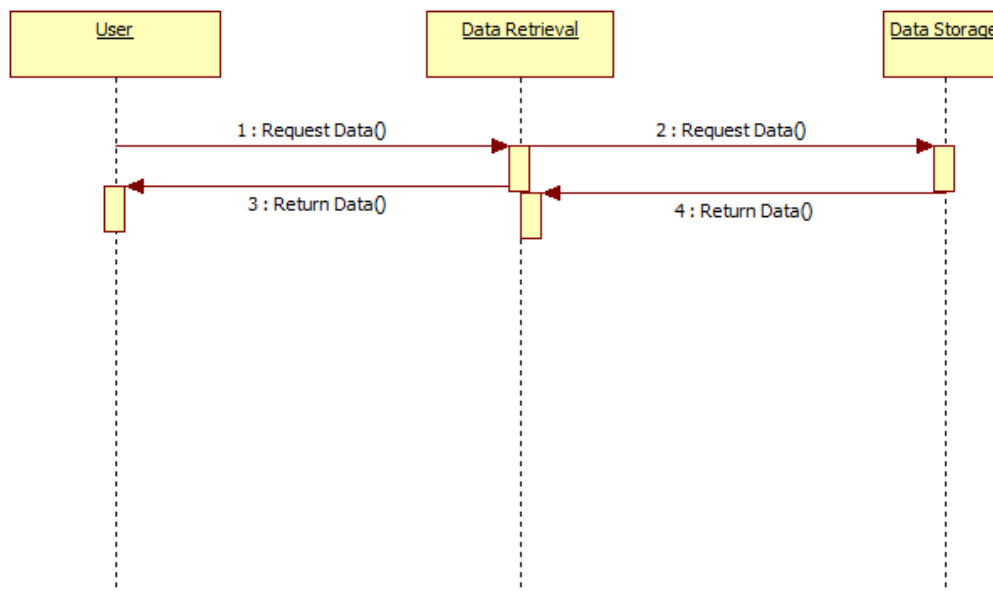


Figure: Sequence Diagram of Data Retrieval from Data Storage

5.2.6 Data Retrieval from Raspberry PI

5.7.2.1 Processing Narrative for Data Retrieval from Raspberry PI

Data Retrieval is responsible for accessing data which contains current information of home appliances from Raspberry PI. It provides a connection between user and Raspberry PI.

5.2.6.2 Data Retrieval from Raspberry PI Interface Description

It receives data requests from users of the system. Then, these requests are sent to Raspberry PI. The result obtained from Raspberry PI is reported to user or component which sent the request.

5.2.6.3 Data Retrieval from Raspberry PI Processing Detail

It works as follows;

1. Receive a request from User component.
2. Send commands Raspberry PI.
3. Received object from Raspberry PI is reported to the user or component which sent the request.

5.2.6.4 Data Retrieval from Raspberry PI Dynamic Behavior

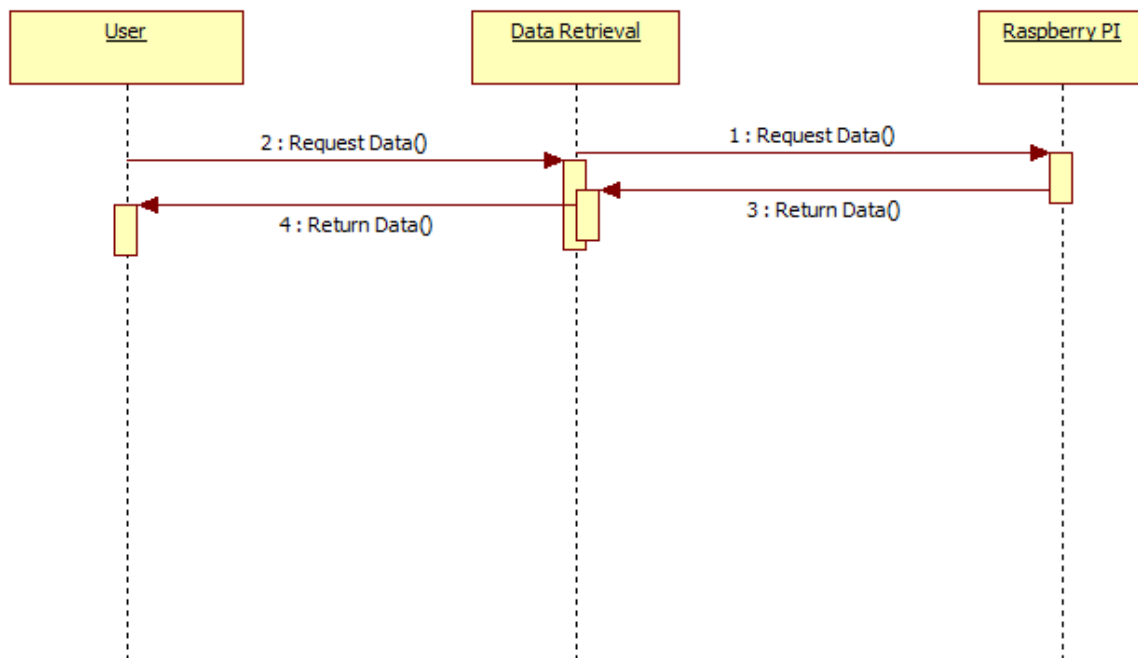


Figure: Sequence Diagram of Data Retrieval from Raspberry PI

5.2.7 Data Storage

5.2.7.1 Processing Narrative for Data Storage

Data Storage component is responsible for creating and storing data objects. It provides the data which is requested from Data Retrieval.

5.2.7.2 Data Storage Interface Description

Data Retrieval sends data requests to Data Storage for further processing. Then, Data Storage processes these commands and returns the results of these queries.

5.2.7.3 Data Storage Processing Detail

It works as follows;

1. Receive a data request from Data Retrieval component.
2. Converted SQL queries are processed.
3. Results of queries are returned.

5.7.2.4 Data Storage Dynamic Behavior

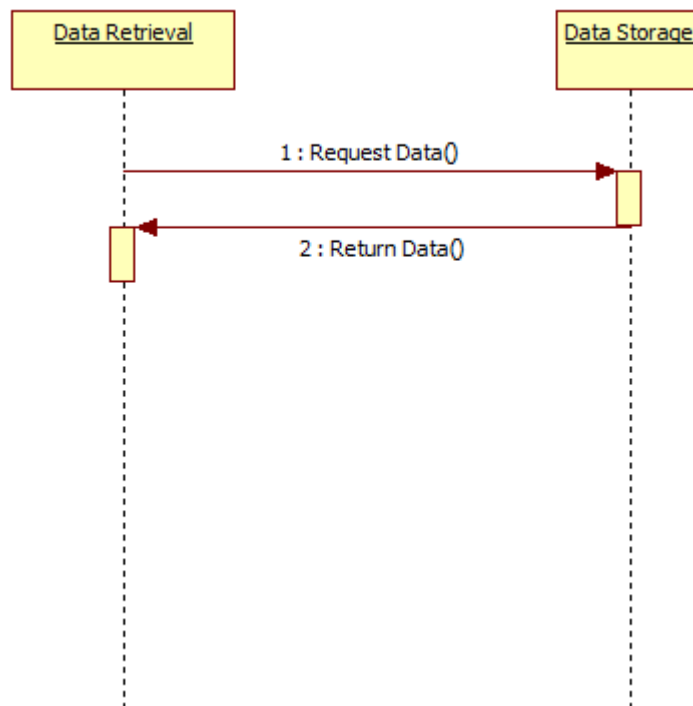


Figure: Sequence Diagram of Data Storage

5.2.8 Raspberry PI

5.2.8.1 Processing Narrative for Raspberry PI

Raspberry PI is responsible for receiving data from home appliances and either sending them to user or storing them to Data Storage. It provides connection between home appliances.

5.2.8.2 Raspberry PI Interface Description

According to time or user request, Raspberry PI sends data request to home appliances. Returning data will send to either user or Data Storage.

5.2.8.3 Raspberry PI Process Detail

It works as follows;

1. Receive data request or interrupt from timer.
2. Request data from home appliances.
3. Send received data accordingly.

5.2.8.4 Raspberry PI Dynamic Behavior

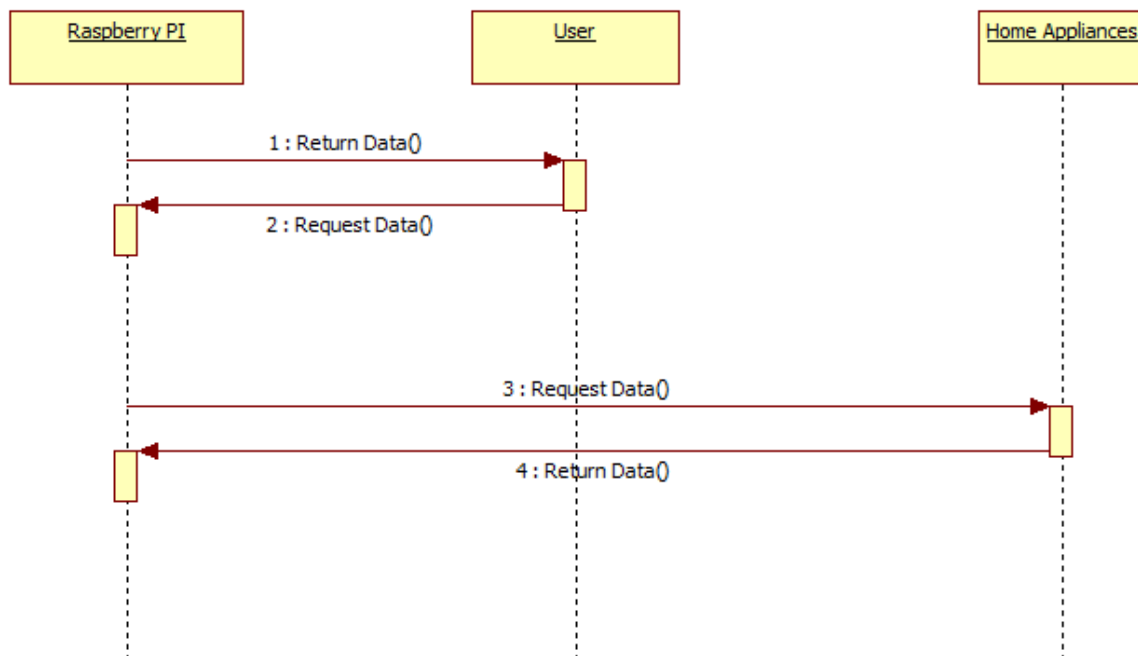


Figure: Sequence Diagram of Raspberry PI

6. User Interface

In this section, user interface of Smart Home application is explained in detail. The details will be explained on the user interface templates in a simple way in the following subtitles.

6.1. Overview of the User interface

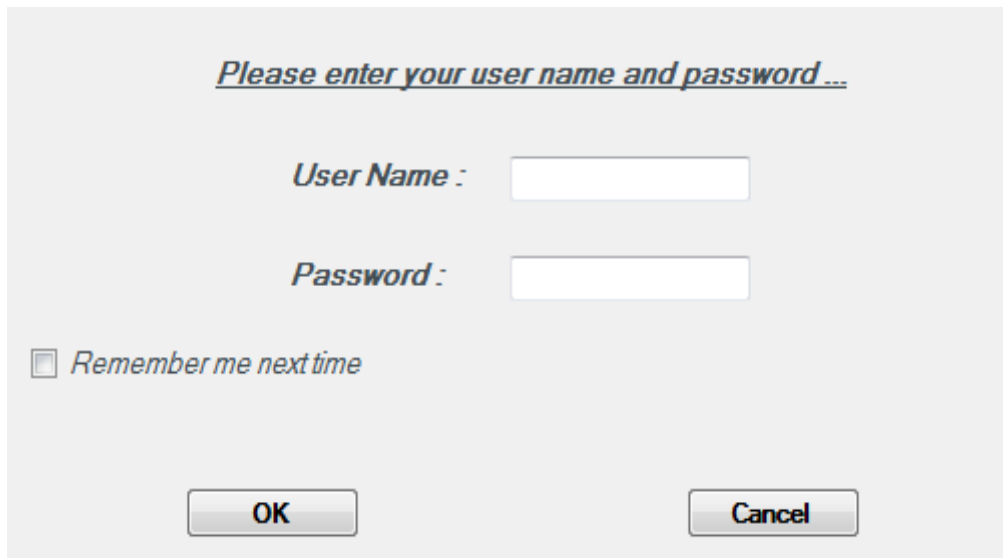
At the beginning of the application, the system authorization is required for the users. This login process is same for all users. They will submit valid authorized username and password.

After the user succeeds in logging in the system, he will be redirected to a new page to select a device to be able to manipulate it. After the device selection process, there will be created a panel to make the related adjustments according to the device type.

6.2. Screenshots

The example screenshots are shown in the following subsections.

6.2.1 Sign In



When the Smart Home web page is opened, the user will face with the login screen. Previously authenticated users will login the system with their exact usernames and passwords.

By this means, different users can be associated with different home appliances. However, this association can only be achieved by the system admin.

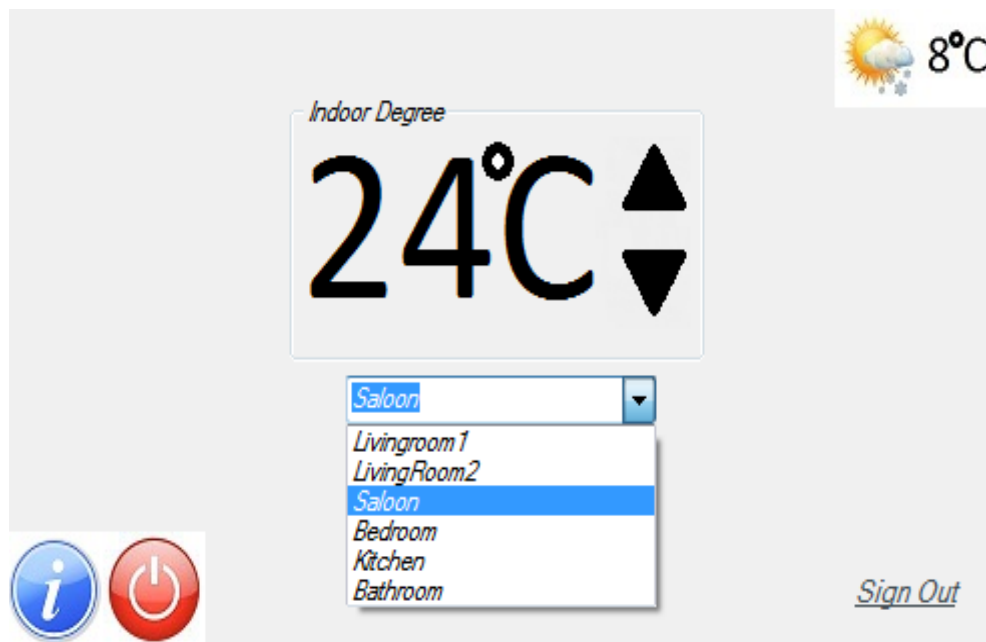
6.2.2 Select Category



After the authentication process explained above, the user is waited for category selection. If the user clicks on one of the links below the icons, the user will be redirected to a new page related to the clicked category adjustments. There is also a sign out preference on the screen, which allows the user to log out from the system.

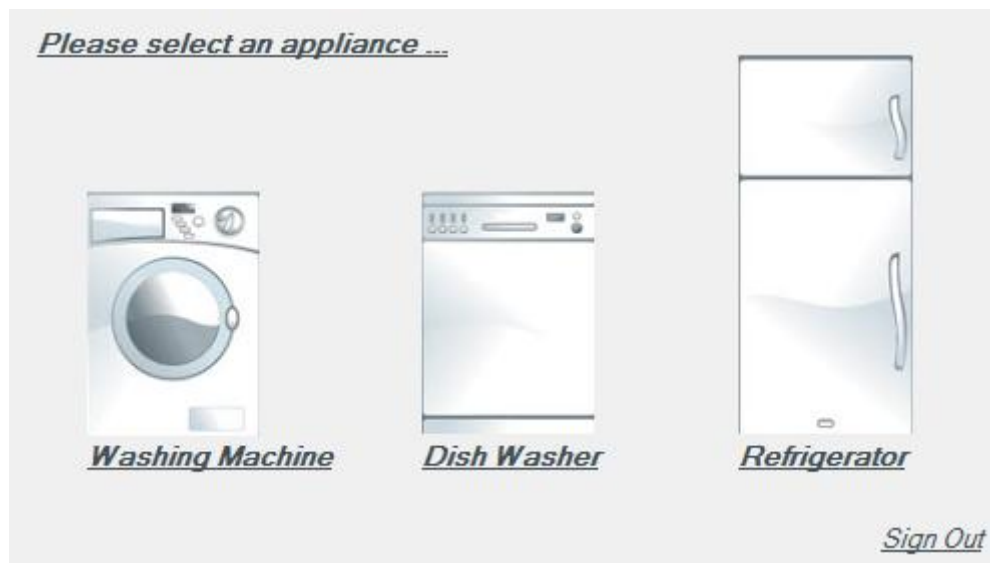
6.2.3 Adjust Configurations

6.2.3.1 Thermostat



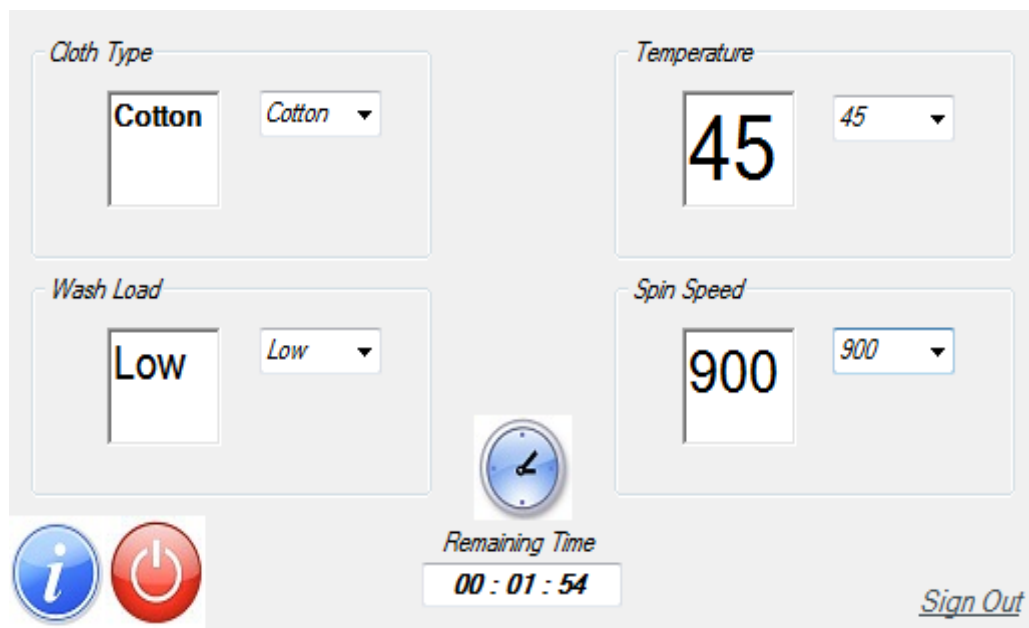
If the user selects the thermostat category, the user should be redirected to this page. In this page there are two buttons on the bottom left side, one of them is to display all degree information gathered from the rooms. Second button stands for shutting down the whole heating system. In the above screenshot, indoor degree is shown as 24 Celcius in the livingroom2. The up and down arrows are used for temperature adjustments. For instance, if the user clicks on the down button, the system starts to cool down. This screenshot is taken just before clicking on the saloon choice. Moreover, on the top right corner, the user can see the outdoor degree information.

6.2.3.2 Home Appliances

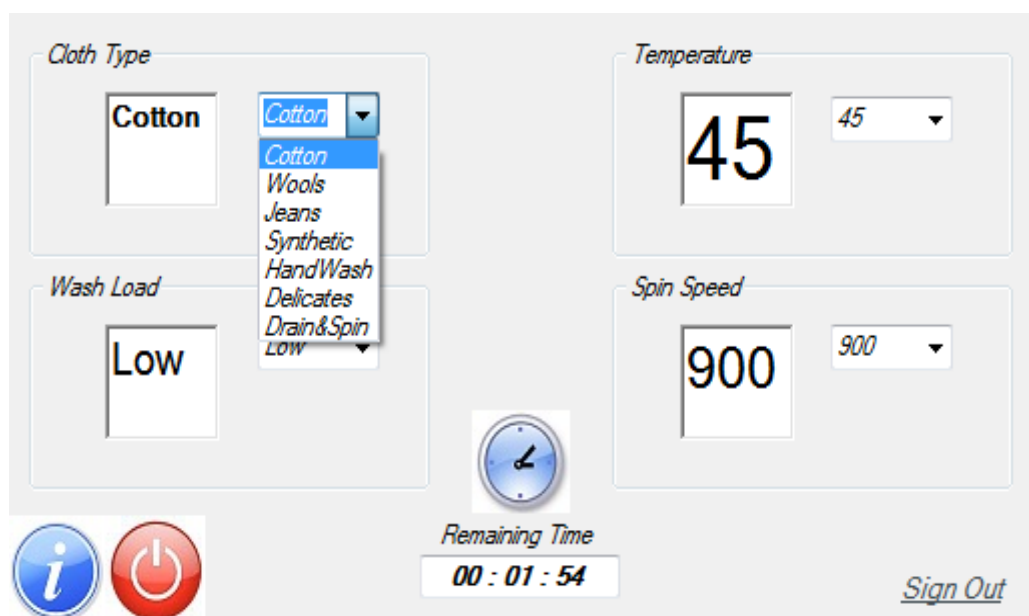


If the user selects the home appliances category, s/he will be redirected to a new page showing some home appliances connected to the system. By clicking the links under the images, the user can pass to the next step to make adjustments or see current situation of the selected home appliance.

6.2.3.2.1 Washing Machine



After the user selects the washing machine home appliance, s/he will be redirected to a page whose screenshots shown on above and below pictures. There are four preference type on the interface: cloth type, temperature, wash load, spin speed. The user can select the cloth type from the menu and it is shown on the left box of the cloth type menu. This is valid for other preference types. If the user does not specify the temperature and spin speed, they will be set automatically according to cloth type.



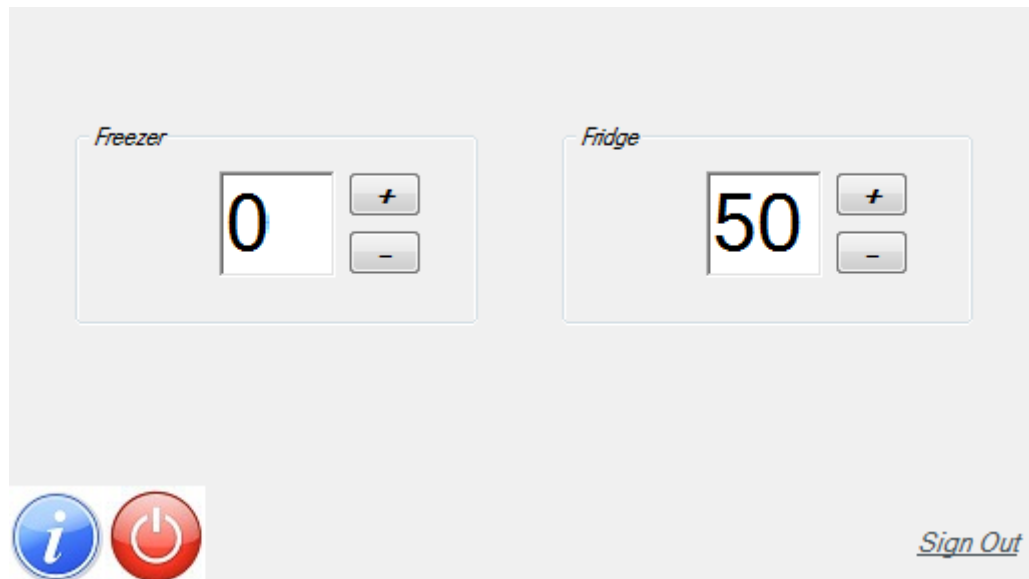
The purpose of the above screenshot is to introduce cloth type menu.

6.2.3.2.2 Dish Washer

The screenshot shows a web-based control interface for a dishwasher. It features several sections: a 'Temperature' section with a large digital display showing '60'; an 'Extras' section with two checked checkboxes labeled 'Soil' and 'Polisher'; a 'Program Type' section with three checkboxes labeled 'Economic', 'Daily' (which is checked), and 'Dense'; a 'Remaining Time' section with a clock icon and a digital display showing '00 : 00 : 47'; and a 'Sign Out' link in the bottom right corner. At the bottom left, there are two circular icons: a blue one with a white 'i' and a red one with a white power symbol.

If the user selects the dishwasher home appliance, s/he will be redirected to the page whose screenshot shown on the above picture. There are 3 types of dish washing programs. The user can click the checkbox of one of them to select the washing type. Temperature will be automatically assigned to its field on the top left part according to the program. There are also extra choices that can be applied at the time of washing according to the user wishes.

6.2.3.2.3 Refrigerator



If the user selects the refrigerator, s/he will be redirected to the page whose screenshot shown on the picture. There two types of adjustments seen on the interface. One of them is the freezer temperature in Fahrenheits and the other one is the fridge temperature in Fahrenheits too. The user can adjust them by clicking on the plus and minus buttons as increasing and decreasing functionalities relatively.

6.2.3 Sign Out



If the user clicks on the sign out button seen on every page after signing in, s/he will be redirected to the page whose screenshot is shown on the above picture. There are two choices left after signing out: OK and Home Page. If the user clicks on the OK button, the window will be closed. If the user clicks on Home Page, the user should be redirected to login page previously described.

6.2. Screen Objects and Actions

Smart Home System is interacted with the user via a browser. Consequently screen objects will be mainly in a form of web application interface objects.

- Text Box: It is used for input fields, such as user name and password.
- Check Box: It is used for approving a preference. For instance, soil and polisher preferences approval can be done via checkboxes.
- Button: It is used for the use of functionality defined in the button event.
- Combo Box: It is used for menu displaying. The user can make a choice between multiple choices included in the drop menu.
- Picture Box: It is used for helping the user easily understand the meanings of the functionalities by supporting them with icons placed into the picture boxes.
- Link Labels: This component is used at making the user redirected to a new page.

7. Dictionary of User Initiated Events

User initiated events (commands) are generally specified in the table below.

<i>Events</i>	<i>Description</i>
Choose a Device	The user choose a device from the context (Refrigerator, washing machine..etc)
Turn On	Enable the user to turn on the chosen device from GUI.
Turn Off	Enable the user to turn off the chosen device from GUI.
Get Sensor Info	Enable the user to learn info about the current status of the device(s).
Adjust	Adjustment panels on the interface will be provided for each device included in the system. This makes the adjustments according to the device capabilities. For example, turning on/off the device, setting the temperature in a refrigerator, changing program on a washing machine.
Show Statistics	Show statistics about the power consumption of device(s) over some period of time.

Table 1 - User Initiated Events

8. Traceability Matrix

Requirements Traceability Matrix	REQ UC 3.2.1	REQ UC 3.2.2	REQ UC 3.2.3	REQ UC 3.2.4	REQ UC 3.2.5	REQ UC 3.2.6
Sign In window	X					
Select Category window	X					
Adjust Configurations window				X		
Sign Out window						X
Devices Class Diagram	X					
Sensor Data Class Diagram			X		X	

Table 2 - Traceability Matrix (between requirements and design)

9. Time Planning

9.1. Term 1 Gantt Chart

	Task	Assigned To	Start	End	Dur	%	2012			2013
							Oct	Nov	Dec	Jan
	SmartHome Project (Term 1)		4/10/12	4/1/13	67					
1	Topic Selection		4/10/12	8/10/12	3					
2	Literature Research and Brainstorming		8/10/12	16/10/12	7					
3	Preparing Project Proposal		16/10/12	23/10/12	6					
4	Requirement Analysis & Project Start-up		23/10/12	5/11/12	10					
5	Testing and Programming Zigbee Devices		5/11/12	30/11/12	20					
6	Finishing SRS Document		5/11/12	12/11/12	6					
7	Initial Design Report		13/11/12	1/12/12	14					
8	Handle PIC-Zigbee communication		19/11/12	4/12/12	12					
9	Client side of web application		19/11/12	4/12/12	12					
10	Last Preparations for Prototype Demo		5/12/12	11/12/12	5					
11	Prototype Demo		11/12/12	4/1/13	19					

9.2. Term 2 Gantt Chart

	Task	Assigned To	Start	End	Dur	%	2012				2013			
							Q4		Q1	Q2	Q3	Q4		
	SmartHome Project (Term 2)		14/2/13	20/6/13	173									
1	System Integration		14/2/13	1/3/13	12									
2	Implementing User Interface		2/3/13	13/3/13	8									
3	Solving Communication Problems		20/2/13	10/3/13	6									
4	Test All Together (Including BeagleBoard & Sensors)		11/3/13	24/3/13	10									
5	Further Implementations		25/3/13	9/4/13	12									
6	Debug		10/4/13	23/4/13	10									
7	Software Test Document		23/4/13	9/5/13	13									
8	Unit Testing		9/5/13	16/5/13	6									
9	Integration Testing		16/5/13	23/5/13	6									
10	Final Implementations and Getting Feedbacks		23/5/13	31/5/13	7									
11	Presentation of the System		31/5/13	20/6/13	15									

10. Libraries and Tools

10.1 MPLAB X IDE

MPLAB Integrated Development Environment (IDE) is a free, integrated toolset for the development of embedded applications employing Microchip's PIC and dsPIC microcontrollers. MPLAB IDE runs as a 32-bit application on MS Windows, is easy to use and includes a host of free software components for fast application development and super-charged debugging. MPLAB IDE also serves as a single, unified graphical user interface for additional Microchip and third party software and hardware development tools. Moving between tools is a snap, and upgrading from the free software simulator to hardware debug and programming tools is done in a flash because MPLAB IDE has the same user interface for all tools. [2]

10.2 NetBeans IDE

The NetBeans IDE is an open-source Integrated Development Environment, a software tool that you can download and use for free to develop applications. The version that contains C/C++ support is used in this project. [3]

10.3 PHP

PHP is a powerful tool for making dynamic and interactive Web pages. It is free and is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. [4]

10.4 HTML5

HTML or Hypertext Markup Language is a formatting language that programmers and developers use to create documents on the Web. The latest edition HTML5 has enhanced features for programmers such as video, audio and canvas elements.[5]

10.5 JavaScript

JavaScript is scripting language used for client side scripting. JavaScript developed by Netscape in 1995 as a method for validating forms and providing interactive content to web site. Microsoft and Netscape introduced JavaScript support in their browsers. [6]

10.6 CSS

CSS is an abbreviation for Cascading Style Sheets. Style sheets are simply text files (.css), composed of lines of code that tell browsers how to display an HTML page. They give the designer more control over the appearance of a webpage by allowing to specifically define styles for elements, such as fonts, on the page. By using CSS one could separate HTML content from its appearance, distinguishing style from structure. [7]

10.7 XAMPP

XAMPP is a very easy to install Apache Distribution for Linux, Solaris, Windows and Mac OS X. The package includes the Apache web server, MySQL, PHP, Perl, a FTP server and phpMyAdmin. [8]

10.8 MySQL

MySQL is a database system used on the web. Basically, a MySQL database allows you to create a relational database structure on a web-server somewhere in order to store data or automate procedures. [9]

10.9 StarUML

StarUML is an open source project to develop fast, flexible, extensible, featureful, and freely-available UML/MDA platform running on Win32 platform. The goal of the StarUML project is to build a software modeling tool and also platform that is a compelling replacement of commercial UML tools such as Rational Rose, Together and so on. [10]

10.10 SmartDraw

SmartDraw is a visual processor used to create flowcharts, organization charts, mind maps, project charts, and other visuals. [11]

11. Conclusion

In this design document, we provided initial details about software architecture and design of the project SMARTHOME, as well as technical details regarding nature of data and system components in various aspects. This document, along with the Software Requirements Specification Document, is intended to describe and give an overview of functionalities provided by project and corresponding mechanical design entities for fulfilling those functionalities properly.