**Software Requirements Specification**

**For**

**<Smart Water Level Detection System>**

**Software Engineering Project**

**Universiti Teknologi Malaysia**

**The members for our group are:**

* Farah Abdul Muttalib
* Zulfadli Kamaruzzaman
* Syahmi Khairudin
* Amirul Fahmi Abdullah
* Shahnizam Abd Saini
* Zulzhafri Zuki
* Wan Zahiruddin Wan Abd Kadir
* Atif Abd Razak

**1. Introduction**

* 1. **Purpose**

The purpose of this documentation is to specify the objectives, aim and requirement for the Smart Water Level Detection System. In this documentation, it also provide the introduction and some of the analysis that can be carried out by the Smart Water Level Detection System. This project implement the concept of IoT that uses the benefits of internet.

Smart Water Level Detection System consist of 3 main parts that’s are web monitoring (Django), smartphone application and Smart Water Level Detection System Device (Raspberry Pi.and water sensor).

* Smart Water Level Detection System Device – sensor and controller that bused to measure the water level at any environment it being installed
* Web monitoring – act as the database and monitoring the data that being send by the device. It also functions to create all the analysis for client to use for further actions. The web page can be access via any browser through internet. <link to the page>
* Smartphone apps – act as monitoring application that use data from the database to provide latest data/reading from the device. <name for the apps>

Smart Water Level Detection System Device consist of controller and sensors that will record the manipulated data. The controller that being used is Raspberry Pi and sensor that being used is water sensor. Raspberry Pi being programmed by using Python language. The controller contained build in Wi-Fi modules that enable the controller to connect to the internet. The device offers the function to record all the data and determine the output. The data is then being sent to the web server (Django) via internet. User can install this device at any place that need to be measured the water level. Some of the most suitable place to install the device are places with heavy rain (flood detection), dam, beaches and etc.

Web monitoring is used as the database for the project and also used to monitor/analysis the data received by the device. It receive the data from the device periodically and stored it inside its database. The data also being used to generate graph and provides information and alerts user if the water level reach critical condition. Users can monitor the data real-time anywhere by accessing the web page. Multi device can be integrated within the same web page that enable users to install multiple device at multiple places and monitor all the data at single web page.

Smartphone apps gives users the same advantage to monitor the data by using their smartphones. The apps can gives the latest data that being requested from the database and visualize it on the smartphones. It becomes more convenient for users that used their smartphones daily.

All the component mentioned above used the concept of IoT that enable them to send/receive data from each other using the internet. The data stored at the database also can be downloaded into the computer for further analysis.

* 1. **Document Convention**

NA

* 1. **Intended Audience and Reading Suggestion**

This document is intended for any individual user, developer, tester, project manager or documentation writer that needs to understand the basic system architecture and its specifications. Here are the potential uses for each one of the reader types:

* Developer: The developer who wants to read, change, modify or add new requirements into the existing program, must firstly consult this document and update the requirements with appropriate manner so as to not destroy the actual meaning of them and pass the information correctly to the next phases of the development process.
* User: The user of this program reviews the diagrams and the specifications presented in this document and determines if the software has all the suitable requirements and if the software developer has implemented all of them.
* Tester: The tester needs this document to validate that the initial requirements of this programs actually corresponds to the executable program correctly.

For each one of the reader types to better understand this document, here is a suggestion of the chapters to read in this document:

Developer:(1.1 ,1.3 ,2.2 ,2.3 ,2.5 ,2.7 ,3 ,4 ,5 and rest)

User :( 1, 2.1, 2.2, 2.3, 4.1, 5.5)

Tester:(1.1 ,2.1 ,2.4 ,2.5 ,2.7 ,3 ,4 ,5)

This document contains the necessary requirement and some aspects of the analysis of the requirements and is organized based on the IEEE Standard for Software Requirements Specification (IEEE 830-1993).

**Overview**

1. **Introduction:** Provide an overview of the application, describe the document structure and point the individual objectives.

2. **Overall Description:** Provide the specification of the system model, the classes’ model, the main constraints and the list any assumed factors that used within this document.

3. **System Features:** Provide the analysis of the requirements by feature.

4. **External Interface Requirements:** Provide the visualization of the program and the requirements that are related with hardware, software and networking.

5. **Other Nonfunctional Requirements:** Provide some other constraints that apply to factors such as performance, safety and security.

* 1. **Project Scope**

Smart Water Level Detection System provide easy and efficient way to monitor and analysis the water level at places of these device being installed. The data can be monitored at either on any web browsers by accessing the web page or by using the smartphone apps. The water levels that being recorded by the device can be classified to several classes that represent different situation. The data also can be used to predict incoming flood or dam malfunction that will alert users so that safety measures can be taken. Compared to the traditional way, users usually monitored by using estimation or on-site monitoring. These traditional data sometimes not accurate and wrong prediction might being made. The other advantage of using this system is that user can monitor and analyze the data from the device real-time anywhere there is internet connection. The device also will determine the critical water level and send alert to users either on the web page or on the smartphones apps. The data that being sent to the database (web page) will be stored and can be used to generate graph or data tabulation to prevent any related catastrophe. Another feature of the system is that the data inside the database can be downloaded into the computer for storage/analysis purpose.

* 1. **References**

This citation is used as a model of reference:

[IEEE Std 830-1998]

**2. Overall Description**

* 1. **Product Perspective**

For the web page development, the framework used is Django. Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. This framework was built by experienced developers, it takes care of much of the hassle of Web development. It’s free and open source. Django used Python language as it programming code. Python is a programming language that will produce maintainable code for improvement unlike other programming language. The web page can be easily modified by improving the code. The device also used Python as the programming language because Raspberry Pi operate in Linux that uses Python as its main language.

For the smartphone apps, the apps being develop by using Android Platform that using Java as its programming language. Java also good to construct maintainable and organize scripts and algorithm.

* The web page that being developed using Django is the databased for the system. The data from the Raspberry Pi is sent to the database. The data was received periodically and the web also can receive from several device. Users can monitor multiple devices that being installed at different places to increase the product capabilities.
* Smartphone apps is used to monitor real-time data and received alerts if the water level reach critical level. The data displayed on the apps is upon request by the apps from the database. If the Smart Water Level System lost its internet connection, there are no data sent to the web page (database). This is the disadvantage of implementing IoT system. The project rely on internet to execute its function.

The significant of this project are this project offers a complete maintainable, improvable, user-friendly and multiple device hook onto one web server. The data obtained also real-time based analyzed automatically unlike other product in the market that still using traditional/manual analyze method.

* 1. **Product Feature**

The main feature the product offers are as following:

• Cross platform support: Offers operating support for most of the known and commercial operating systems

• Language support: Offers multiple language support for global use.

• Download: Offers download option. The data can be downloaded into computers.

* 1. **User Classes and Characteristic**

**Physical Actors:**

* Buyers: Buyers will buy the product and install the system at places where they need to measure the water level for monitoring purpose. Buyers also is the person that have the need to install the system in places that they think might need to be monitored to prevent accident and for analysis purpose. Multiple device can be connected to the same web page and share the same database.
* Users: Users are the person that use the web page to monitor the data from the Smart Water Detection Device. Users can monitor the data by visiting the web page. In the web page, several analysis can be visualize to make the data can be understand easily and user’s friendly.
* Smartphone’s user: Smartphone’s user is the one that will use the apps to monitor the water level stage. The advantage of this apps is that users can install the apps inside the smartphones and starts to monitor the data through their phones. Mobile phones plays an important roles in our life and most of people uses mobile phones every day.

**System Actors:**

* Client: The client is the system that connects to the server and handles the tests based on the session and finally submits the information back to the server.
* Server: Server is the system that is the internet that will connect all module in the system so that it can send and receive data from each other.
* Database: The database is where all the data from the device is stored. The data can be retrieved easily by visiting and downloaded the data wanted.

The primary actor for this project is the server that will connect the controller to the database. The data receive by the database is stored. The users act as person that uses the service that being provided by the system that enable users to monitor water level condition through the web page or smartphone apps. The buyers is the person that bought and install the system at places where the system can monitor the water level such as river, heavy rain region, dam, river and etc.

Below are the main goals of each of the actor list:

|  |  |
| --- | --- |
| **Actor** | **Buyers** |
| Want to installs automatic system that can record water level at any place that need to be monitor to prevent flood or any related disaster. | |

|  |  |
| --- | --- |
| **Actor** | **Users** |
| Want to monitor the water level condition anywhere without presence at the actual site.  Want to analyze the data from raining seasons.  Want to record water level at dam/river and view it anywhere and anytime.  Also want to download the data recorded for further studies. | |

|  |  |
| --- | --- |
| **Actor** | **Smartphone’s user** |
| Need to view the data by using smartphone.  Want to receive alerts through phone if the water level reached critical level. | |

|  |  |
| --- | --- |
| **Actor** | **Database** |
| Must connect to the internet and ready to receive data from the Raspberry Pi device.  Able to store data inside the database and visualize it in form of graphs and tables. | |

|  |  |
| --- | --- |
| **Actor** | **Server** |
| Able to connect all three sub device together so that communication between these modules are uninterrupted.  Need to send the data periodically and avoid data losses. | |

|  |  |
| --- | --- |
| **Actor** | **Web page (Django)** |
| Offers users and buyers analytical data in human readable form.  Need to be maintainable and user friendly so that buyers can made changes to the web if needed.  Provide download option for users to download the data.  Alert users and buyers if the water level reach dangerous level. | |

* 1. **Operating Environment**

This project will operate in the following operating environment for the device (controller), web page (Django) and smartphone apps.

GUI:

Device (controller)

* Linux/Unix (Primary code in Python language)

Web page (Django)

* Any browsers (Universal with specific web address/link)

Smartphone Apps

* Android Operating System
  1. **Design and Implementation Constraints**

This project is created by using Python and Java programming language and uses Raspberry Pi as the controller for the main device and Django framework for the web page development. Both of the modules uses Python as the programming language. For the smartphone apps, it was developed using Android Platform software and uses Java as the programming language. The web page can be access by using any browsers from any device that can surf the internet. The database created by the Django for the web page can stored about 100mb data that used to stored received data in the cloud. The web page was design based on the aim so that users finds it simple to use and monitors. The product also easy to maintain as the code is organized and maintainable.

The constraints for the system is that whenever there is no internet connections for the controller device, the data will not be sent to the server. It consider as data loss thus making the system less efficient. IoT concept required internet so that it can full fill its actual function.

* 1. **User Documentation**

Here are the official links of the project where you can retrieve more information about it and download the latest version:

Online Documentation

<https://github.com/SoftwareEgineeringWaterDetectionSystem/Smart-Water-Detection-System>

Official Webpages

-NA

-NA

* 1. **Assumption and Dependencies**

For the alert mechanism, buyers can set manually the water level condition that will trigger alerts that being sent to the web page and the smartphone apps. Django framework also provide open source for users that interested to develop their own web page. For users that only want to use the built-in web page provided by the product, we assume that they not interested in the step how to create their web page and the program coded in the product.

**3.0** System Features

Using the actor list we begin the analysis of the client-Rpi features that takes place in Section A and then for each modules that is being reported

**Section A**

**Raspberry Pi**

**3.1 System Feature Execute SensorModule::REQ1**

3.1.1 Description and Priority

A User executes the SensorModule and the startup setup script.

Priority level=High

3.1.2 Stimulus/Response Sequences

Preconditions: None

1. User execute auto.csh on Rpi terminal.

2. SensorModule setup script shows up.

Postconditions: auto.csh will run all the require setup script to run SensorModule

3.1.3 Functional Requirements

NA

**3.2 System Feature Connect to a Database (online mode)::REQ2**

3.2.1 Description and Priority

User sets Database name and port and connects to the server.

Priority level=High

3.2.2 Stimulus/Response Sequences

Preconditions: User access makeNet.py. User knows Database name and port. Rpi is online.

1. User replace variable <database> to the desired database name.

2. User replace variable <port number> to desired port.

Postconditions: User is connected to the database after auto.csh run. The database information is displayed.

Abnormal Paths: If the servers name and port is incorrect error will be print out to the terminal to inform the User that which one of the two is incorrect. auto.csh will halt

3.2.3 Functional Requirements

NA

**3.3 System Feature Setting Sensor Config::REQ3**

3.3.1 Description and Priority

User set the sensor configuration.

Priority level=High

3.3.2 Stimulus/Response Sequences

Preconditions: User access makeConfig.py. User knows detail of sensor mode.

1. User replace variable <mode> to desired mode.

2. User replace variable <unit> to change the SI unit of sensor reading.

Postconditions: Sensor mode change successfully and changes will appear on database upon running auto.csh.

Abnormal Path: If the mode and unit is incorrect error will be print out to the terminal to inform the User that which one of the two is incorrect. auto.csh will halt.

3.3.3 Functional Requirements

NA

**3.4 System Feature Connect to Devices::REQ4**

3.4.1 Description and Priority

User sets devices MAC address or IP to connect with RPI.

Priority level=High

3.4.2 Stimulus/Response Sequences

Preconditions: User access makeDevice.py. User knows MAC address or IP of the device. Device is online.

1. User replace variable <macaddress> to the device MAC address.

2. User replace variable <ip> to the device ip address.

Postconditions: User is connected to the device after auto.csh run. The device information is displayed.

Abnormal Paths: If the device MAC address or IP address is not available, error will be print out to the terminal to inform the User that which one of the two is incorrect. auto.csh will halt

3.4.3 Functional Requirements

NA

**3.5 System Feature Next Run Automation::REQ5**

3.5.1 Description and Priority

User can make auto.csh to run automatically on the next RPI boot.

Priority level=High

3.3.2 Stimulus/Response Sequences

Preconditions: User access auto.csh. User confirm that current configuration is the same on next run.

1. User execute auto.csh with variable <1/2>

Note: 1 is automate on the next boot

2. Do not automate

Postconditions: On the next Rpi boot, all the setup script will run by itself. If there is an error, a message will be display during Rpi boot.

3.5.3 Functional Requirements

NA

**DJANGO**

**3.6 System Feature Connect to a server (online)::REQ6**

3.6.1 Description and Priority

The user sets Server name based on the place that user want to monitor the water level.

Priority level=low

* + 1. Stimulus/Response Sequences

Preconditions: User knows servers name and the place to monitor.

1. User enters Servers name.

2. User enters Place Name.

3. User elects to Connect to the server.

4. User elects to view the water level information.

Post conditions: User is connected to the server. The water level information is displayed.

Abnormal Paths: If the servers name and place is incorrect SmartCity informs the user that which one of the two is incorrect.

* + 1. Functional Requirements

Specializes: NA

Specialized by: NA

**3.7 System Feature Connect to a server (offline)::REQ7**

3.7.1 Description and Priority

Server will get the data from rpie and store data inside Django.

Priority level=low

* + 1. Stimulus/Response Sequences

Preconditions: Rpie send data to server.

1. Rpie search server IP

2. Server receive data from rpie.

3. Server store data based on rpie place.

Post conditions: The information store in server and ready to display in SmartCity website.

Abnormal Paths: If the server no receive data from rpie, server will warning to technical engineer.

* + 1. Functional Requirements

Specializes: NA

Specialized by: NA

**3.8 System Feature User enters Place::REQ8**

3.8.1 Description and Priority

The user is obliged to choose a place after connected to the server and clicked next button.

Priority level=medium

* + 1. Stimulus/Response Sequences

Preconditions: Users clicked the next button from SmartCity startup screen.

1. User enters the place.

2. User clicks the start button.

Post condition: screen will show the information of water level based on place registered.

Abnormal Paths: if the place not registered in server, the screen will displayed “NO DATA” and user need to reenter the corrected place name.

* + 1. Functional Requirements

Specializes: NA

Specialized by: NA

**3.9 System Feature Display Graph::REQ9**

3.9.1 Description and Priority

The user choose a graph button and graph are displayed.

Priority level=low

* + 1. Stimulus/Response Sequences

Preconditions: Users clicked the graph button from screen.

1. User click the graph button.

2. Type of graph will appears.

3. User need to choose type of graph.

Post condition: screen will show the information with graph that user choose.

* + 1. Functional Requirements

Specializes: NA

Specialized by: NA

**3.10 System Feature View Information::REQ10**

3.10.1 Description and Priority

The result information are being forwarded to a printer

Priority level=medium

* + 1. Stimulus/Response Sequences

Preconditions: Users viewing the information of water level.

1. User enter the print button.

Post condition: Test information is printed using the default printer settings.

* 1. 3 Functional Requirements

Specializes: NA

Specialized by: NA

**Smartphone Apps**

**3.11 System feature install Smart Water Level Detection system.apk::REQ11**

3.11.1 Description and priority

User install the Smart Water Level Detection system.apk and the system wizard start up client screen appears

Priority level = high

3.11.2 Stimulus /response sequences

Precondition: User must have mobile internet connection

User already download Smart Water Level Detection system.apk

1. User mobile phone run on android 5.0 and above
2. User double click the Smart Water Level Detection system.apk
3. System wizard start up screen appear
4. User read and accept the term and condition
5. User close the system wizard

3.11.3 Functional requirement

  NA

**3.12 System feature Connect to a server (online Mode)::REQ12**

3.12.1 Description and Priority

User connect to Django server and allow data reading

3.12.2 Stimulus /response sequences

Preconditions: User must have mobile internet connection

1. User click on the installed application icon on the mobile home screen
2. System application user interface screen appear
3. "Connect to server" and "sign up" button appear
4. User click on the "connect to server" button
5. Include REQ4
6. System application home screen appear

Postconditions: None

**3.13 System feature Sign up (online Mode)::REQ13**

3.13.1 Description and Priority

User sign up in order to create the account before login to the system application

Priority level: medium

3.13.2 Stimulus /response sequences

Preconditions: User must have mobile internet connection

1. User click on "Sign up" button
2. User enter the email address
3. User enter password
4. User confirm the password
5. User click " Done" button
6. Connect to server user interface appear

Postconditions: None

**3.14 System feature login to the system (online Mode)::REQ14**

3.4.1 Description and Priority

User login to the system and get permission to access Django’s database

3.4.2 Stimulus /response sequences

Preconditions: None

1. User click "connect to server" button
2. User enter the email and password on the login section
3. User click "login" button
4. System application home screen appear

Postconditions: None

**3.15 System feature request data (online Mode)::REQ15**

3.5.1 Description and Priority

User request data from Django’s database and display on the application screen

3.5.2 Stimulus /response sequences

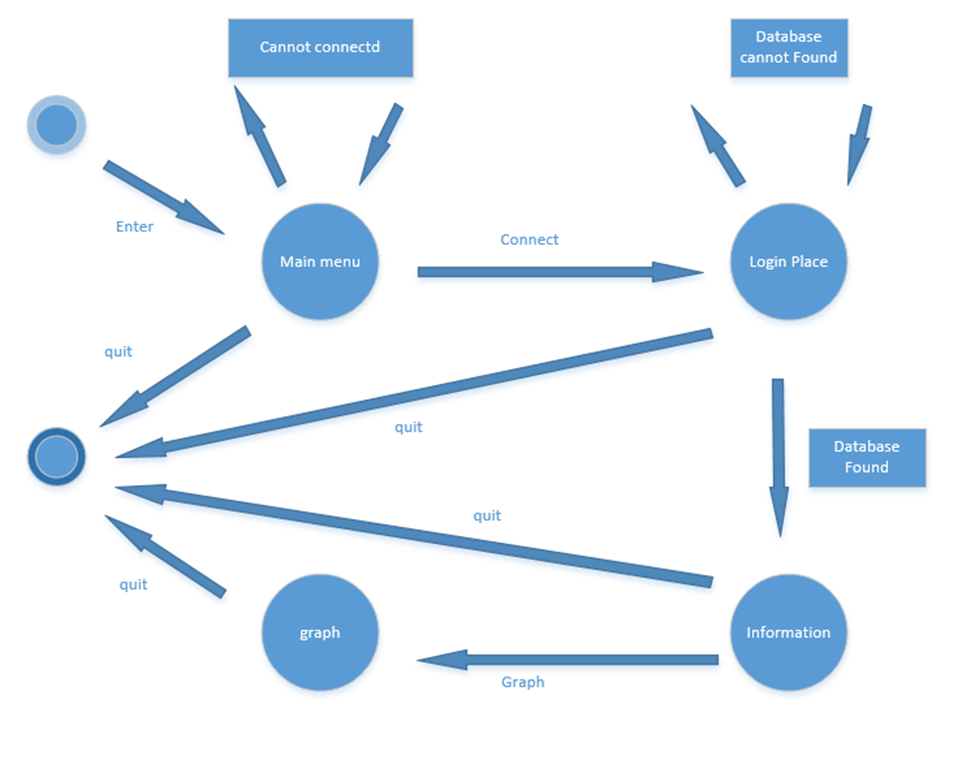
Preconditions: User already connect to Django’s server

1. User select " Location for System water level Detection "
   1. Location A
   2. Location B
   3. Location C

2. User select “Time detection"

1. User click " confirm' button
2. System application request data from server
3. Data from server appear on the user screen

**Section B**



State machine diagram of Smart Water Level Detection System