GTA-Group

WaterMe

Software Requirements Specification

Version 1.1.4

Revision History

Date	Version	Description	Author
19/Oct/16	1	First version	Paul Giesa Olga Akymenko Chris Todt
24/Oct/16	1.1	Roughly filled	Paul Giesa Olga Akymenko Chris Todt
1/Nov/16	1.1.1	Updated links to UC Change Settings & See LEDs	Chris Todt
28/Nov/16	1.1.2	Updated UC-Diagram Summarized Use Cases	Paul Giesa, Chris Todt
7/Dez/16	1.1.3	Updated links to Use Cases	Chris Todt
1/Mai/17	1.1.4	Added FP-Calculation/Complexity	Olga Akymenko

Table of Contents

GTA-Group

Software Requirements Specification

1. Introduction

1.1 Purpose

This SRS provides documentation for WaterMe. WaterMe is a system based on a Raspberry Pi and sensors. These sensors measure the humidity of the soil. If the soil is too dry the system will notify you with a tone in regular cycles that the user has to water the plant.
The system is able to communicates with an Android application on your mobile Android device via Bluetooth, thus giving you additional notifications on your device.
The Android App will be usable with multiple systems.

1.2 Scope

This document is designed for internal use only and will outline the development of the project

1.3 Definitions, Acronyms, and Abbreviations

Term/ Acronym/ Abbreviation	Definition
SRS	Software Requirement Specification
IDE	Integrated Development Environment
Pi	Raspberry Pi
UML	Unified Modeling Language

1.4 References

Blog	https://gtaproject.wordpress.com/2016/10/12/first-blog-post/
GitHub	https://github.com/AkymenkoOlga/WaterMe
Use Case Diagram:	https://github.com/AkymenkoOlga/WaterMe/blob/master/UC/Overall%20UC%20diagram.png

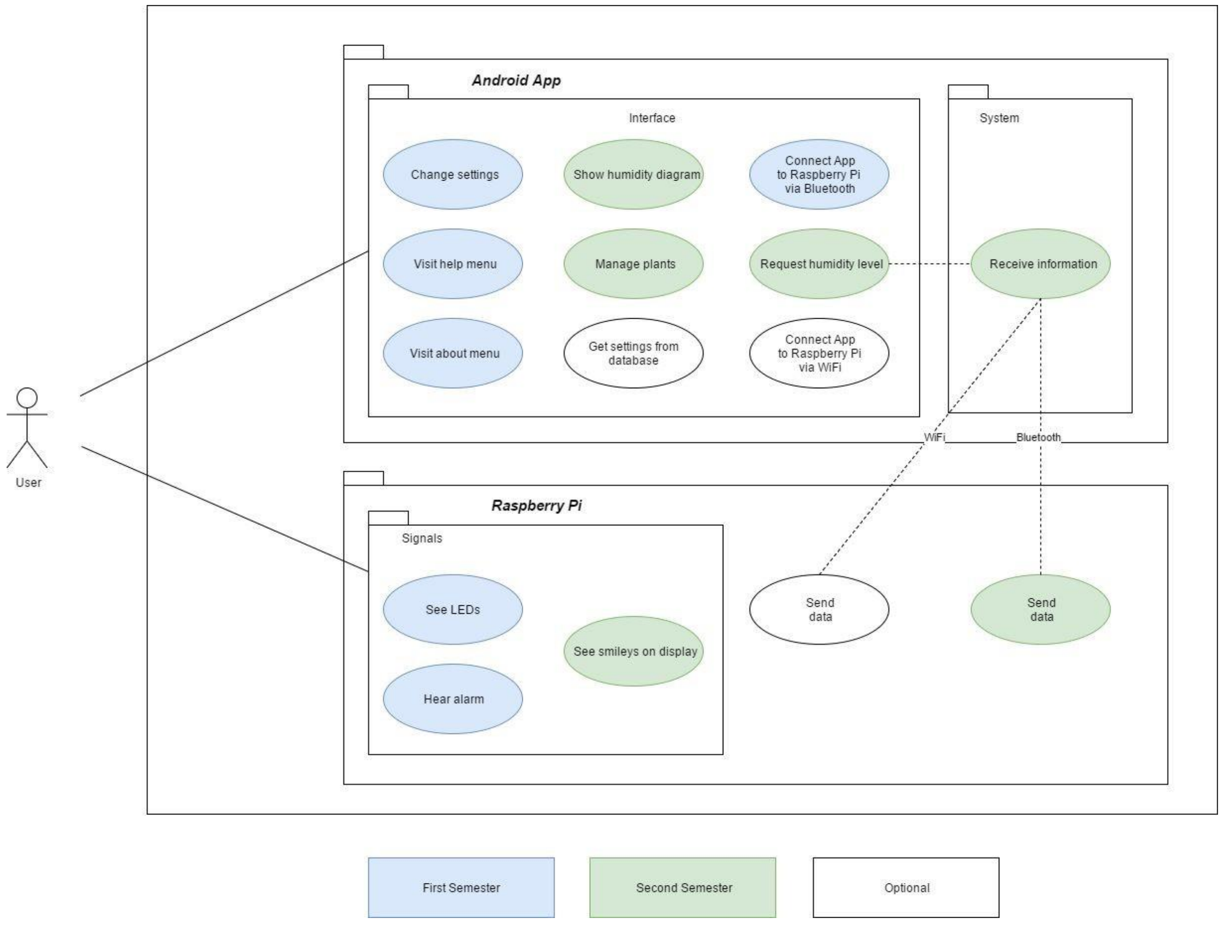
1.5 Overview

The next chapters give an overall description about our project and requirements such as functionality, usability, reliability, and performance.

2. Overall Description

The user will start WaterMe after after setting the sensor in plant and adding information about the humidity level . After adding the information sensor would send the notifications about humidity level to let user know, that he should water the flower.

2.1 Product perspective



Our overall Use Case UML diagramm

2.2 Product functions

1st semester:

- Pi:
    - Receive and process data from humidity sensor
    - LEDs
    - Alarm signal
- Application:
    - Enter Help Menu
    - Visit Settings
    - Connect via Bluetooth to Pi

2nd semester:

- Pi:
    - Transmit data to App
    - Display Smileys

- Application:
    - Request and the plant’s status
    - Humidity diagram
    - Add new plants
    - Add picture
    - Add information
    - Request Humidity level

2.3 User characteristics

The actor should have plants at home. In addition he can, but not has to, be forgetful.

2.4 Constraints

to be determined

2.5 Assumptions and dependencies

- IDE: Android Studio
- Version-Control: GitHub
- Programming language: Java, Python
- Tools: ObjectAid (Eclipse), Espresso, Jira, Gantter, Wordpress, GIMP, Visio, Trello

3. Specific Requirements

3.1 Functionality

3.1.1 Request humidity level

Within the App the user is able to request the current humidity level of the plant.

3.1.2 Manage plants

The User is able to take a picture of the plant and set it as an avatar.

The User is able to add other plants for monitor.

The User can add some information about the plants (e.g. name).

UC Manage plants

3.1.3 Humidity diagram

The User can observe the humidity process in a diagram over time

3.1.4 Settings

In the settings menu the user can adapt notification, sound, LED and connection settings.

UC Change Settings

3.1.5 Connect App to Raspberry Pi via Bluetooth

The User can connect the application to the Pi via Bluetooth to receive notifications about the plant and to change the settings of the Pi.

UC Connect App to Raspberry Pi via Bluetooth

3.1.6 Help

In the help menu we provide information how to use the app properly.

UC Visit help menu

3.1.7 About

In the about menu we provide information about the idea of WaterMe, how it works and about us.

UC Visit about menu

3.1.8 LEDs

The LEDs indicate the humidity level.

UC See LEDs

3.1.9 Alarm

An alarm signal indicates when humidity level is too low.

3.1.10 Smileys

On a small display smileys indicate the humidity level.

3.2 Usability

3.2.1 Using a Smartphone

The user should know how to handle an android smartphone (e.g. how to install and start an applications).

3.2.2 Installing further sensors

In case of adding further plants the user needs to know how to install and connect an additional sensor to the Pi.

3.2.3 Water plants

In order to provide the full experience it is recommended to water the plants according to the notifications from the app.

3.3 Reliability

3.3.1 Availability

The user should be able to request the humidity level all day so the Pi has to run 24 hours per day.

3.3.2 Mean Time Between Failures (MTBF)

This allegation depends mostly on the durability of the humidity sensor. Maybe a few months up to a year.

3.3.3 Mean Time To Repair (MTTR)

The loss of a sensor is not that dramatic if the user reminds watering the plant until a new sensor arrives. The delivery will normally take 2-3 days and the installation can be done in minutes.

3.3.4 Accuracy

Since the plant will not die immediately after getting notified by the app it is alright if the sensor has a deviation of 10%.

3.3.5 Maximum Bug / Defect Rate

to be determined

3.3.6 Bugs / Defect Rate

to be determined

3.4 Performance

3.4.1 Response time

to be determined

3.4.2 *Throughput*

to be determined

3.4.3 Capacity

WaterMe is planned to serve as measurement device for one plant.

The Application will be usable with multiple measurement devices.

3.4.4 Degradation modes

The refresh rate between Pi and App can be lowered.

3.4.5 Resource utilization

WaterMe requires a smartphone with Bluetooth and Android KitKat 4.3 or higher.

3.5 Supportability

3.5.1 Coding standards

3.5.2 Naming conventions

3.5.3 Class libraries

3.5.4 Maintenance access

3.5.5 Maintenance utilities

3.6 Design Constraints

The backend of this software should be written in Java.

3.7 On-line User Documentation and Help System Requirements

The Application will have a simple and intuitive design. In case of questions, it will be built in the opportunity to ask us questions personally or to go to the help menu

3.8 Purchased Components

- JOY-iT Explorer 700 Expansion Board
- Allnet Starterset ArdDevKIT1 ATMega328
- Bluetooth 4.0 Stick
- Samsung Galaxy S5 mini
- Pi 2 Model B
- MCP3008 - I/P (analog/digital converter)

3.9 Interfaces

3.9.1 User Interfaces

When user starts the application, he/she sees a main page
with a list of plants, connected to the coordinator box, with few limited details about their status of humidity.
From this list, user can select the plant. After a plant is selected, user can see or change the information about the plant through application interface. After a reasonable time for data sending/receiving protocols, the device to be controlled will respond to these changes and sensor information will be updated.

3.9.2 Hardware Interfaces

GPIO Pins (Humidity sensor).

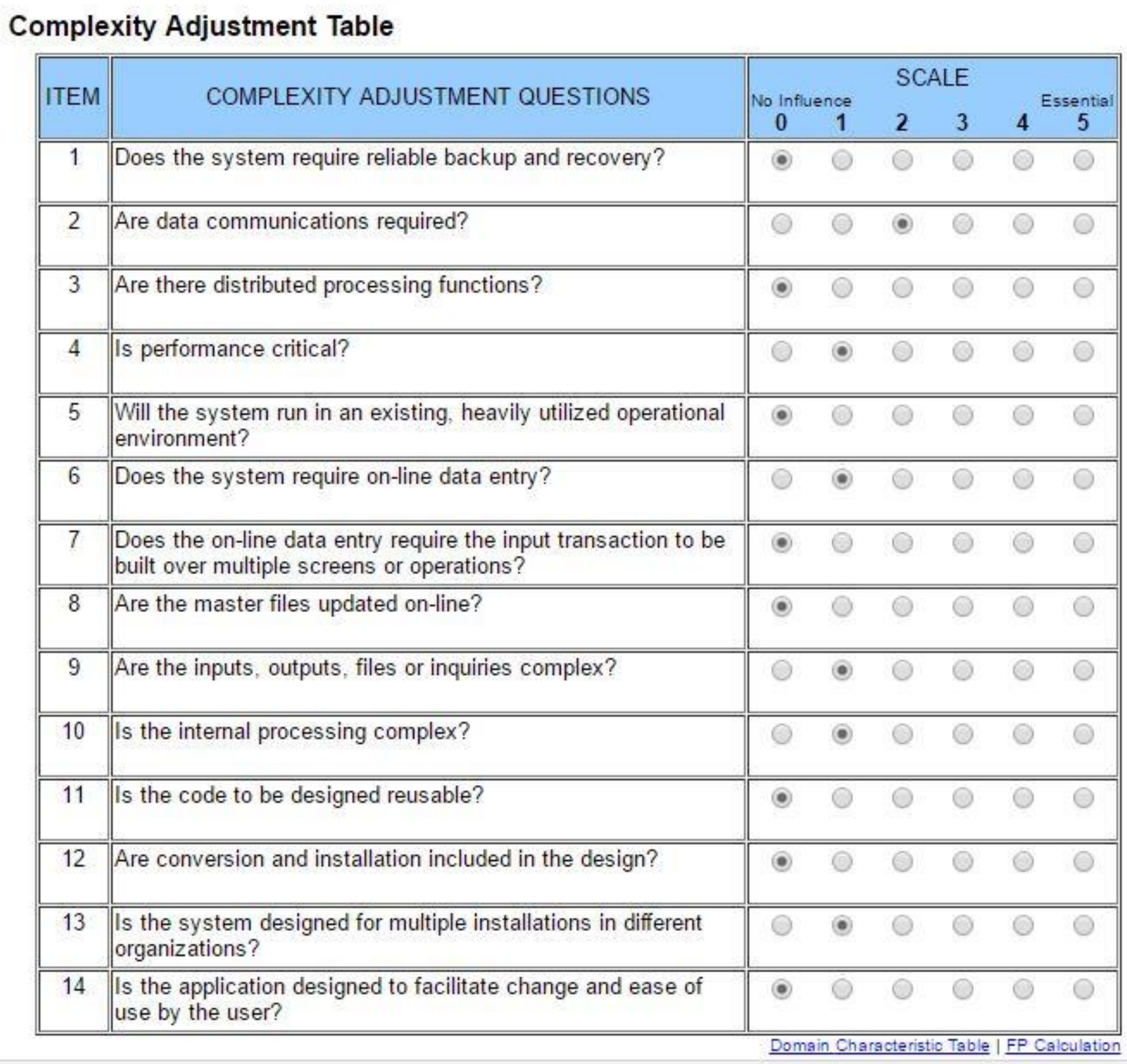
3.9.3 Software Interfaces

to be determined

3.9.4 Communications Interfaces

Bluetooth/WiFi (Data transfer between Pi and app)

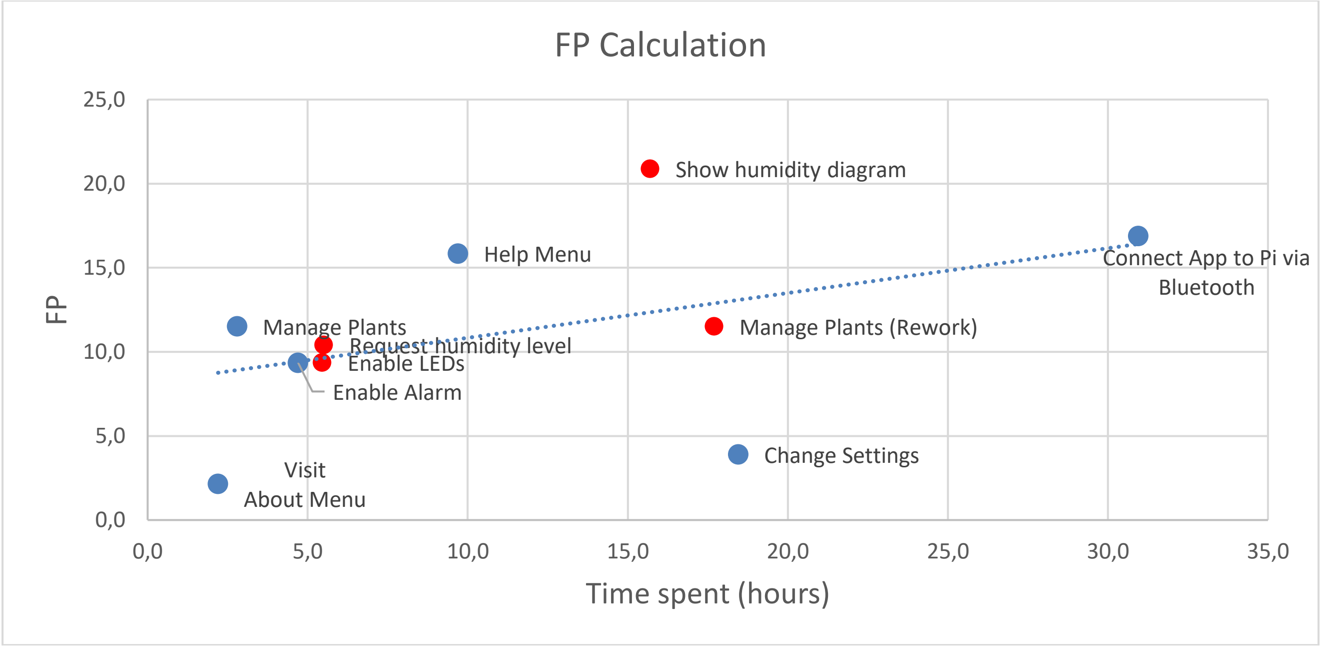
3.10 Complexity



Complexity Adjustment Table

ITEM	COMPLEXITY ADJUSTMENT QUESTIONS	SCALE No Influence 0	1	2	3	4	Essential 5
1	Does the system require reliable backup and recovery?	●	○	○	○	○	○
2	Are data communications required?	○	○	●	○	○	○
3	Are there distributed processing functions?	●	○	○	○	○	○
4	Is performance critical?	○	●	○	○	○	○
5	Will the system run in an existing, heavily utilized operational environment?	●	○	○	○	○	○
6	Does the system require on-line data entry?	○	●	○	○	○	○
7	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	●	○	○	○	○	○
8	Are the master files updated on-line?	●	○	○	○	○	○
9	Are the inputs, outputs, files or inquiries complex?	○	●	○	○	○	○
10	Is the internal processing complex?	○	●	○	○	○	○
11	Is the code to be designed reusable?	●	○	○	○	○	○
12	Are conversion and installation included in the design?	●	○	○	○	○	○
13	Is the system designed for multiple installations in different organizations?	○	●	○	○	○	○
14	Is the application designed to facilitate change and ease of use by the user?	●	○	○	○	○	○

Domain Characteristic Table | FP Calculation



FP-Calculations for other USE Cases:

UC Change Settings
UC ConnectAppToRaspberryPiViaBluetooth
UC Enable LED
UC Manage Plants
UC About Menu
UC Help Menu

3.11 Licensing Requirements

to be determined

3.12 Legal, Copyright, and Other Notices

to be determine

3.13 Applicable Standards

to be determined

4. Supporting Information

to be determined