Department of Computer Engineering

CENG350 Software Engineering

Software Requirements Specification for FarmBot

Group 52

By

Emre Çam

Tufan Özkan

Monday 18th March, 2024

Contents

List of Figures **iii**

List of Tables **iv**

1 Introduction **1**

1.1 Purpose of the System 1

1.2 Scope 2

1.3 System Overview 3

1.3.1 System Perspective 3

1.3.1.1 System Interfaces 4

1.3.1.2 User Interfaces 4

1.3.1.3 Hardware Interfaces 4

1.3.1.4 Software Interfaces 4

1.3.1.5 Communication Interfaces 4

1.3.1.6 Memory Constraints 5

1.3.1.7 Operations 5

1.3.2 System Functions 5

1.3.3 Stakeholder Characteristics 5

1.3.4 Limitations 6

1.4 Definitions 6

2 References **7**

3 Specific Requirements 8

3.1 External Interfaces 8

3.2 Functions 8

3.3 Logical Database Requirements 8

3.4 Design Constraints 9

3.5 System Quality Attributes 9

3.6 Supporting Information 9

4 Suggestions to Improve The Existing System 10

4.1 System Perspective 10

4.2 External Interfaces 10

4.3 Functions 10

4.4 Logical Database Requirements 11

4.5 Design Constraints 11

4.6 System Quality Attributes 11

4.7 Supporting Information 11

List of Figures

1.1 Context Diagram of FarmBot Express 3

1.2 Example 5

List of Tables

1 Table Example v

Revision History

(Clause 9.2.1)

You can use a table to show your reports' versions and their date.

To create tables, you can use https://www.latex-tables.com/; the Table 1 is generated by it.

A	B	C	D

Table 1: Table Example

1. Introduction

This document provides the Software Specification Requirement (SRS) of an open-source precision agriculture CNC farming project that lets the user grow food from anywhere with a web app. The website for this system is `https://farm.bot`

1.1 Purpose of the System

The purpose of the FarmBot system is to revolutionize agricultural practices through automation. FarmBot aims to regularize farming methods, enhance crop yields and minimize resource waste. Additionally; FarmBot provide a platform to manage and monitor their crops and agricultural operations efficiently. Through automation and monitoring, the system aims to:

- Automate planting, watering, weeding and harvesting processes to reduce human work and improve efficiency.
- Provide precise control over the environmental factors such as soil moisture, harmful weeds.
- Facilitate real-time monitoring from web application to enhance crop health.
- Provide an environment to design a layout for crops and implement that design in real world.

1.2 Scope

In the scope of this system, Farmbot is humanity's open-source CNC farming machine that automatically grows food right in your backyard keeping you in complete control, as opposed to the current food production system which no longer has control over how the food is produced.

- Farmbot is provided with 95 pre-assembled elements to be set up quickly and easily and perform the basic functions needed to grow a garden.
- Raspberry Pi computers operate the system with its hardware components (webcam, Arduino firmware microcontroller combined with powerful stepper motors and dynamic devices, ph sensors) and user application requests.
- Farmduino microcontrollers allow positioning of the tool head with millimetre accuracy for sowing, and watering plants in any pattern and frequency with given built-in features of plants based on their type, age, and soil conditions. The sensors also send the condition of soil and weather data of pH, temperature, and moisture.
- The Onboard camera system allows user to monitor their garden, and capture images, and the farmbot can take action by detecting the weeds utilising advanced computer vision.
- The web and mobile application allows users to control the garden remotely, and configure or update the software of Raspberry Pi computers without any need for hardware change.
- The farm designer and sequence editors allow users to lay out their plants with built-in data for plants, creating optimal layouts each season, and taking care of their plants in the way they want without requiring any coding.
- The farmbot software development allows professionals to modify and use it for

any purpose (e.g. education, business) in hardware tools or develop completely new versions of the farmbot based on the existing library of components.

1.3 System Overview

This section of the document will provide detailed information about the system including all components.

1.3.1 System Perspective

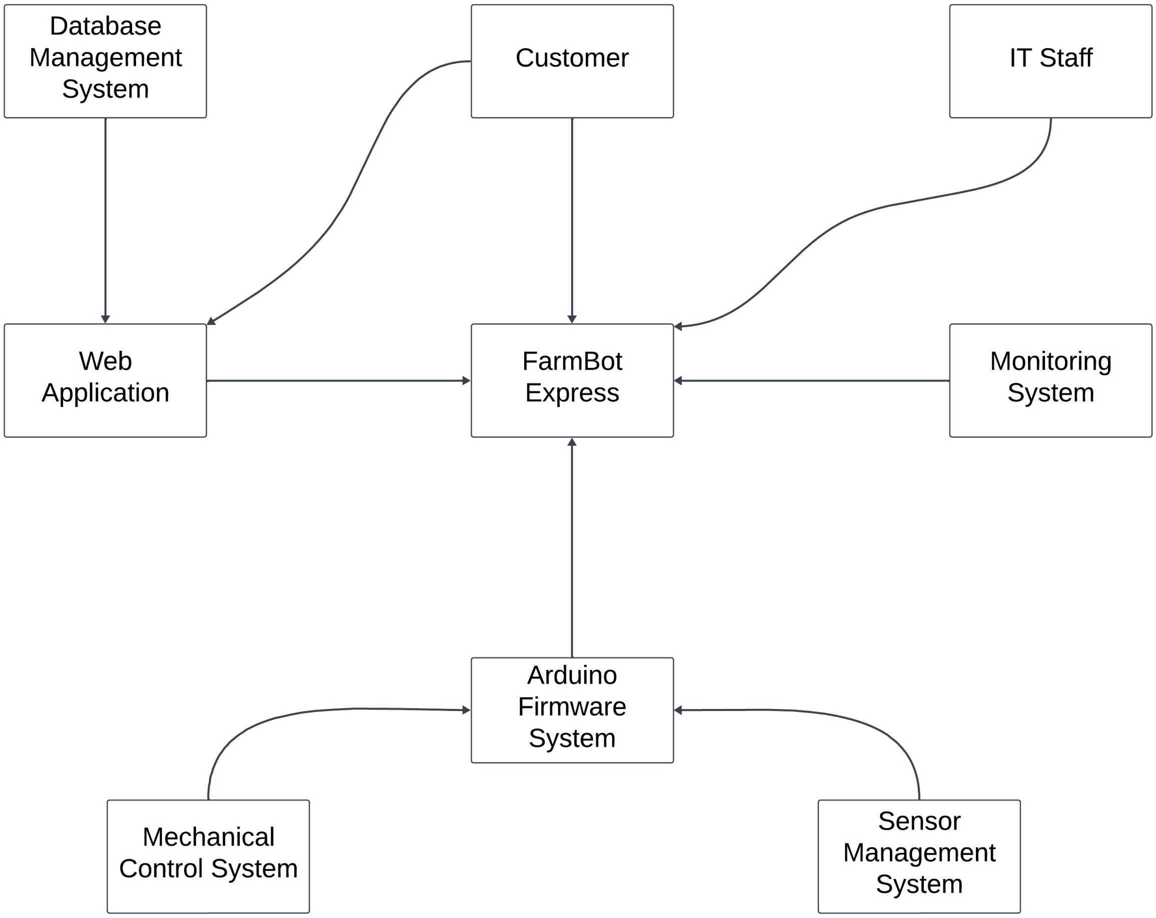

Figure 1.1: Context Diagram of FarmBot Express

FarmBot is not part of a large system nor necessarily requires the download of an application. Hence customers shall go into the web browser of the app and configure their FarmBot using their laptop, tablet, or smartphone. The web application features real-time manual controls and logging, a sequence builder for custom routines for FarmBot to execute.The web application keeps the data and receives the built-in data for plants from the database management system. The machine maintains a connection and synchronises with the web application via the message broker. Besides that, Farmbot communicates with the Arduino firmware system over a serial connection to send commands for motor and tool positioning and receive collected from sensors and rotary encoders. Moreover, the FarmBot is connected to a monitoring system through a webcam to provide real-time view, image-capturing, and detecting weeds, taking actions according to it. Furthermore, the system provides an IT staff interface in case of loggings, errors and customary configurations to help customers.

Refer to *(Clause 9.6.4, 9.5.4.1)*

1.3.1.1 System Interfaces

Refer to *(Clause 9.6.4.1)*

1.3.1.2 User Interfaces

Refer to *(Clause 9.6.4.2)*

1.3.1.3 Hardware Interfaces

Refer to *(Clause 9.6.4.3)*

1.3.1.4 Software Interfaces

Refer to *(Clause 9.6.4.4)*

1.3.1.5 Communication Interfaces

Refer to *(Clause 9.6.4.5)*

1.3.1.6 Memory Constraints

Refer to *(Clause 9.6.4.6)*

1.3.1.7 Operations

Refer to *(Clause 9.6.4.7)*

1.3.2 System Functions

Refer to *(Clause 9.6.5, 9.5.4.2)*. If you want to add any figure or diagram, you can use a figure environment.In Figure 1.2

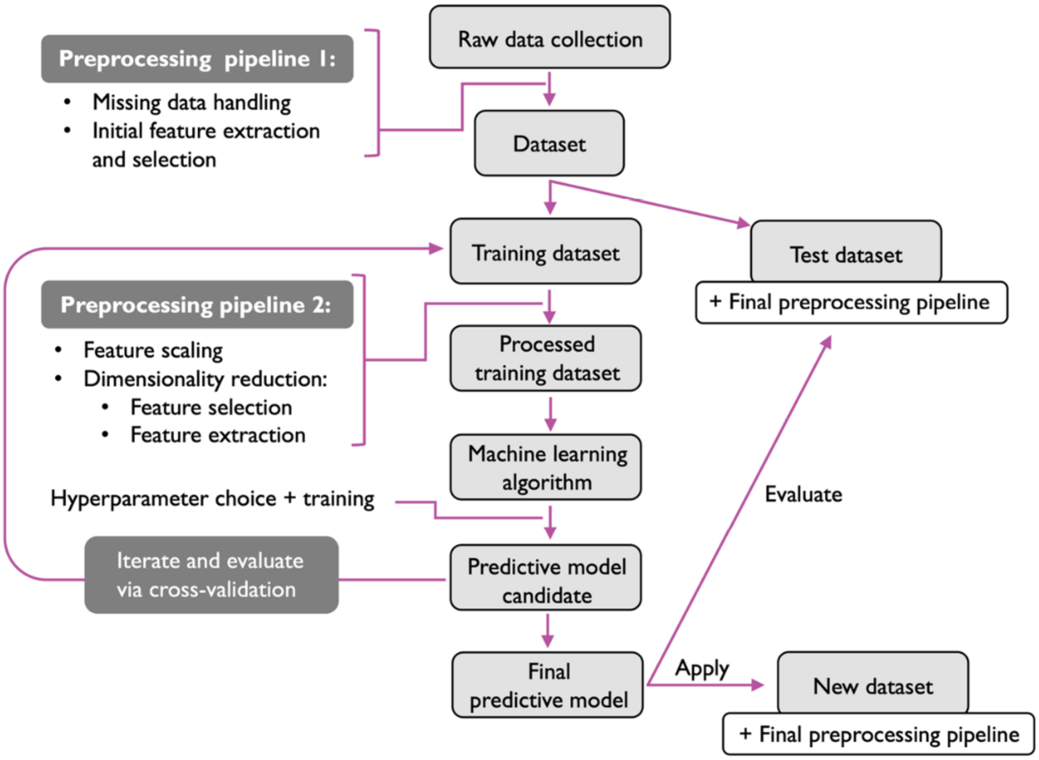
Preprocessing pipeline 1:
• Missing data handling
• Initial feature extraction and selection
Raw data collection
Dataset
Training dataset
Test dataset
+ Final preprocessing pipeline
Preprocessing pipeline 2:
• Feature scaling
• Dimensionality reduction:
• Feature selection
• Feature extraction
Processed training dataset
Machine learning algorithm
Hyperparameter choice + training
Evaluate
Iterate and evaluate via cross-validation
Predictive model candidate
Final predictive model
Apply
New dataset
+ Final preprocessing pipeline

Figure 1.2: Example

1.3.3 Stakeholder Characteristics

Refer to *(Clause 9.6.6, 9.5.4.3, 9.4.5)*

1.3.4 Limitations

Refer to *(Clause 9.6.7)*

1.4 Definitions

You should add acronyms and abbreviations here. Refer to *(Clause 9.6.7)*

For any citation, refer to it as [1].

2. References

[1] M. C. Younis and H. Abuhammad, "A hybrid fusion framework to multi-modal biometric identification," *Multimedia Tools and Applications*, vol. 80, no. 17, pp. 25799–25822, 2021.

3. Specific Requirements

Refer to *(Clause 9.6.10)*.

3.1 External Interfaces

External Interfaces Class Diagram and its explanations go here. Plus, other content as appropriate

Refer to *(Clause 9.6.11, 9.5.8)*.

3.2 Functions

Use-case diagram goes here; **detailed use-case descriptions in a reasonable template** follow. You are expected to have **about 10 use cases covering major system functionality**. Have some associations in your use-case diagram, e.g. include, extend, specialization. Choose *three* most complicated use cases. Construct three diagrams **(one sequence diagram, one activity diagram, and one state diagram)** to elaborate on these three use cases. Plus, other content as appropriate.

Refer to *(Clause 9.6.1)*.

3.3 Logical Database Requirements

Key data objects (persistent or not) and their major attributes. Draw the **Class Diagram** with associations. A class dictionary can be omitted, provided that the naming is understandable.

Refer to *(Clause 9.6.15)*.

3.4 Design Constraints

Specify constraints on the system design imposed by external factors, such as official standards, regulatory requirements, or organizational/managerial limitations.
Refer to *(Clause 9.6.16)*.

3.5 System Quality Attributes

Important quality attributes (Usability *(Clause 9.6.13, 9.5.6)*, Performance *(Clause 9.6.14, 9.5.7)*, Dependability properties, Maintainability, and so on) in the order of priority with associated requirements.

Refer to *(Clause 9.6.18)*.

3.6 Supporting Information

Refer to *(Clause 9.6.20)*

4. Suggestions to Improve The Existing System

4.1 System Perspective

Context diagram and explanations of context diagram go here for suggestions to improve the existing system. Plus, other content as appropriate.

Refer to *(Clause 9.6.4, 9.5.4.1)*.

4.2 External Interfaces

External Interfaces Class Diagram and its explanations go here for suggestions to improve the existing system. Plus, other content as appropriate

Refer to *(Clause 9.6.11, 9.5.8)*.

4.3 Functions

Use-case diagram for suggestions to improve the existing system goes here; **detailed use-case descriptions in a reasonable template** follow. You are expected to have **about 4 use cases covering suggestions to improve the existing system**. Have some associations in your use-case diagram, e.g. include, extend, specialization. Choose three most complicated use cases. Construct three diagrams **(one sequence diagram, one activity diagram, and one state diagram)** to elaborate on these three use cases. Plus, other content as appropriate.

Refer to *(Clause 9.6.12, 9.5.5, 9.5.10)*.

4.4 Logical Database Requirements

Key data objects (persistent or not) and their major attributes for suggestions to improve the existing system. Draw the **Class Diagram** with associations. A class dictionary can be omitted, provided that the naming is understandable.

Refer to *(Clause 9.6.15)*.

4.5 Design Constraints

Specify constraints on the system design imposed by external factors, such as official standards, regulatory requirements, or organizational/managerial limitations for suggestions to improve the existing system.

Refer to *(Clause 9.6.16)*.

4.6 System Quality Attributes

Important quality attributes (Usability *(Clause 9.6.13, 9.5.6)*, Performance *(Clause 9.6.14, 9.5.7)*, Dependability properties, Maintainability, and so on) in the order of priority with associated requirements for the improved system.

Refer to *(Clause 9.6.18)*.

4.7 Supporting Information

Refer to *(Clause 9.6.20)*.