POLITECNICO
MILANO 1863

Software Engineering 2 project

Academic Year 2021-2022

DREAM - Data-dRiven PrEdictive FArMing in Telengana

Design Document

Version 0.1 – 06/01/2022

1.INTRODUCTION

1.1 Purpose

Agriculture plays a vital role in India's economy. However, with food demand increasing, climate change and COVID-19 pandemic, agriculture in Telengana India is facing severe challenges, like unstable food supply chains, vulnerabilities of marginalized communities and smallholders.

Telengana's government wants to build anticipatory governance models for food systems using digital public goods and community-centric approaches to strengthen data-driven policy making in Telengana.

This document contains an explanation of the design decisions that we have made for the whole system, going from the general architecture design to the specific components, their interfaces, their interactions and their physical deployment, along with a presentation of some graphical user interface mockups. Moreover, this document also contains a discussion on the implementation and testing plan in order to give the developers a general roadmap.

1.2 Scope

The Dream System is an easy-to-understand interface which aims to helping the user to complete their own tasks on the process of Farming.

The Dream System allows the Policy Makers to identify those farmers who are performing well or badly and understand whether the steering initiatives carried out by agronomists.

The Dream System allows farmers to visualize data relevant to them, provide information about their production, ask for help and suggestion by agronomists and other farmers and create discussion forums with the other farmers.

The Dream System also allows agronomists to receive requests for help and answer to these requests, visualize data concerning weather forecasts in the responsible area and the best performing farmers in the responsible area, create and modify a daily plan to visit farms in the responsible area and confirm the execution of the daily plan at the end of each day or specify the deviations from the plan.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- DREAM System (or “The System”): refers to the whole system to be developed.

1.3.2 Acronyms

- API: Application Programming Interface
- RASD: Requirement Analysis and Specification Document.
- UML: Unified Modelling Language.
- GPS: Global Positioning System.

1.3.3 Abbreviations

- C.i: i-th component

1.4 Revision history

Version	Date	Authors	Summary
0.1	06/01/2022	Rui Zhang	Update section1 and section 3

1.5 Reference Documents

- Specification document: Project Assignment A.Y. 2021-2022.pdf
- RASD of DREAM
- Software Engineering 2 course slides
- IEEE Standard on Requirement Engineering (ISO/IEC/IEEE 29148)

1.6 Document Structure

This document is structured as follows:

1. ***Introduction*** – A general introduction of the system-to-be, which aims at giving general information about what this document is going to explain.
2. ***Architectural Design*** – An overview of the high-level components and their interactions, with a focus on both static and dynamic view, with the help of diagrams.
3. ***User Interface Design*** – A representation of how the User Interface will look like.
4. ***Requirements Traceability*** – An explanation about how the requirements defined in the RASD map to the design elements defined in this document.
5. ***Implementation, Integration and Test Plan*** – Identification of the order in which the sub-components of the system should be implemented, integrated and tested.
6. ***Effort spent –*** Effort spent by all team members shown as the list of all the activities done during the realization of this document
7. ***References*** – References to documents that this project was developed upon.

2. ARCHITECTURAL DESIGN

2.1 Overview

2.2 Component view

2.2.1 Web Application Component

It is the application dedicated to three roles, including farmer, agronomist and policy maker, which allows them to keep access data under control and possibly modify the store information.

2.2.2 Web Server Component

A *Web Server* is required to provide *Web Application* for all users. This component receives **HTTPS** requests from users' browsers, forwards them to the *Application Server*, and generates the dynamic web pages based on the response from the *Application Server*.

2.2.3 Application Server Component

2.2.4 Data Components

2.2.5 External System

2.3 Deployment view

2.4 Runtime view

2.5 Component interfaces

This section provides a high-level description of the methods offered by the interfaces that the various system component expose.

2.6 Selected architectural styles and patterns

// Please explain which styles/patterns you used, why, and how

2.7 Other design decisions

3. USER INTERFACE DESIGN

All the interfaces have already been presented in the RASD. Therefore, in the next sections, flow graphs of the application are provided.

In the following graphs, rectangles represent the screens, referenced by figure number in the RASD, diamonds represent conditions, and arrows represent buttons or decisions. And the information in the “[ ]” reprensent the operations.

3.1.1 Web Application Flow Graph

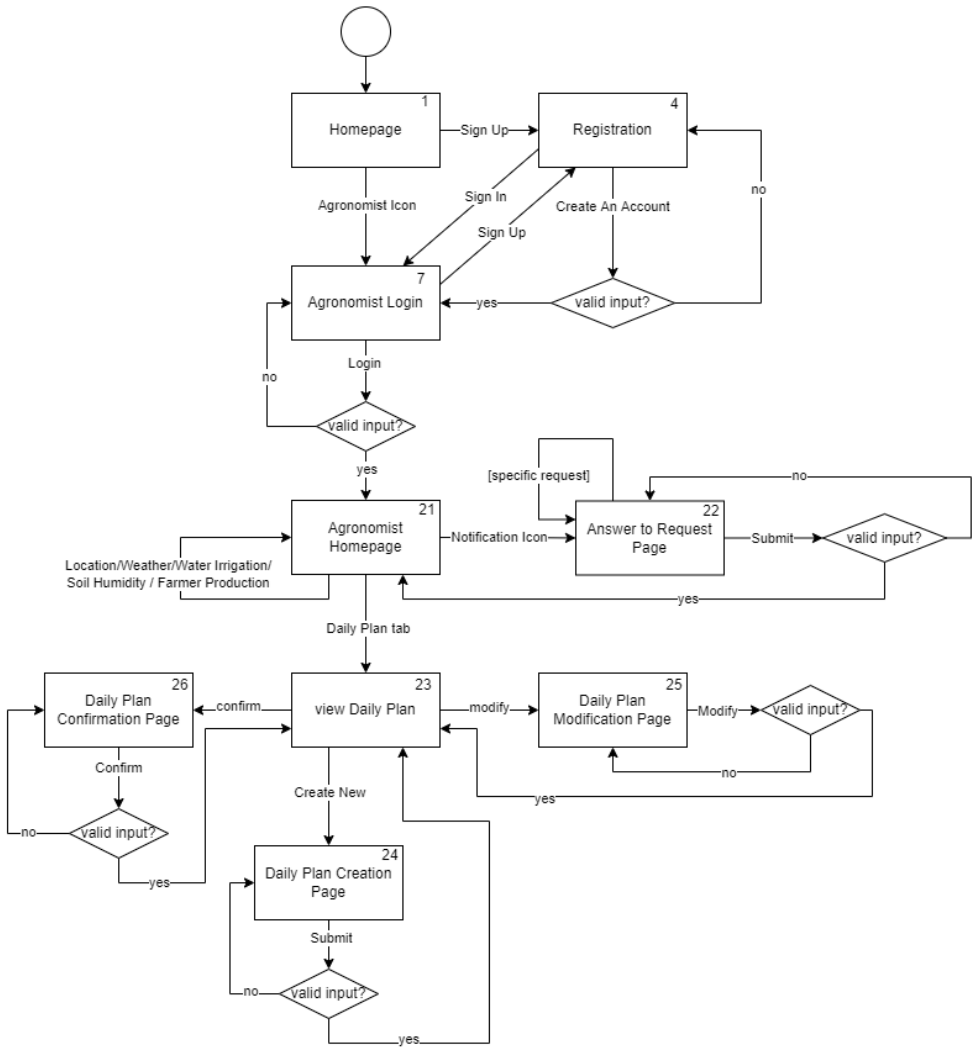

Diagram 1: Web application flow for Agronomist

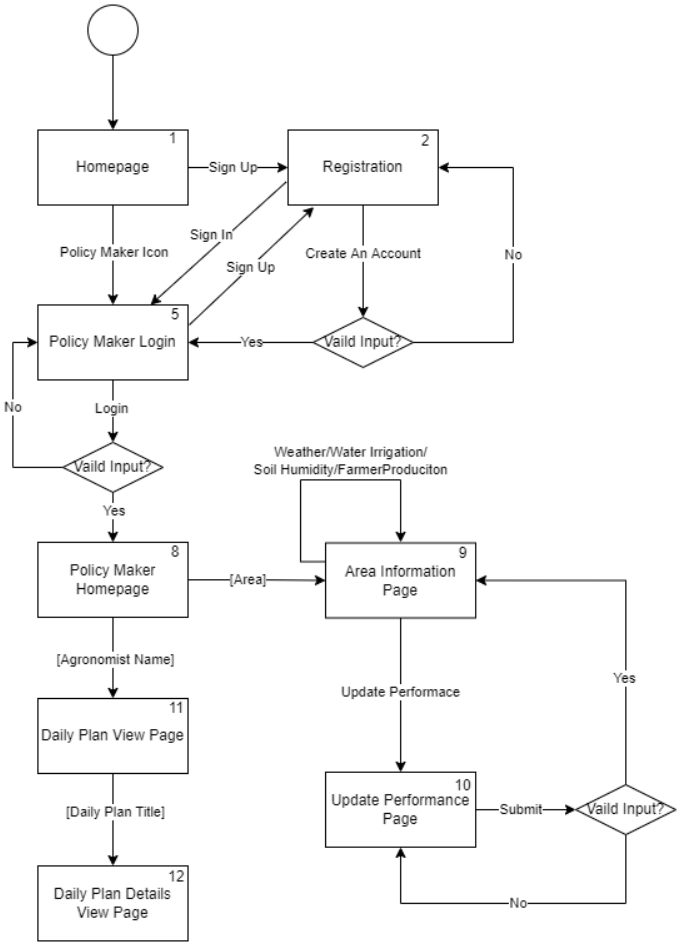

Diagram 2: Web application flow for Policy Maker

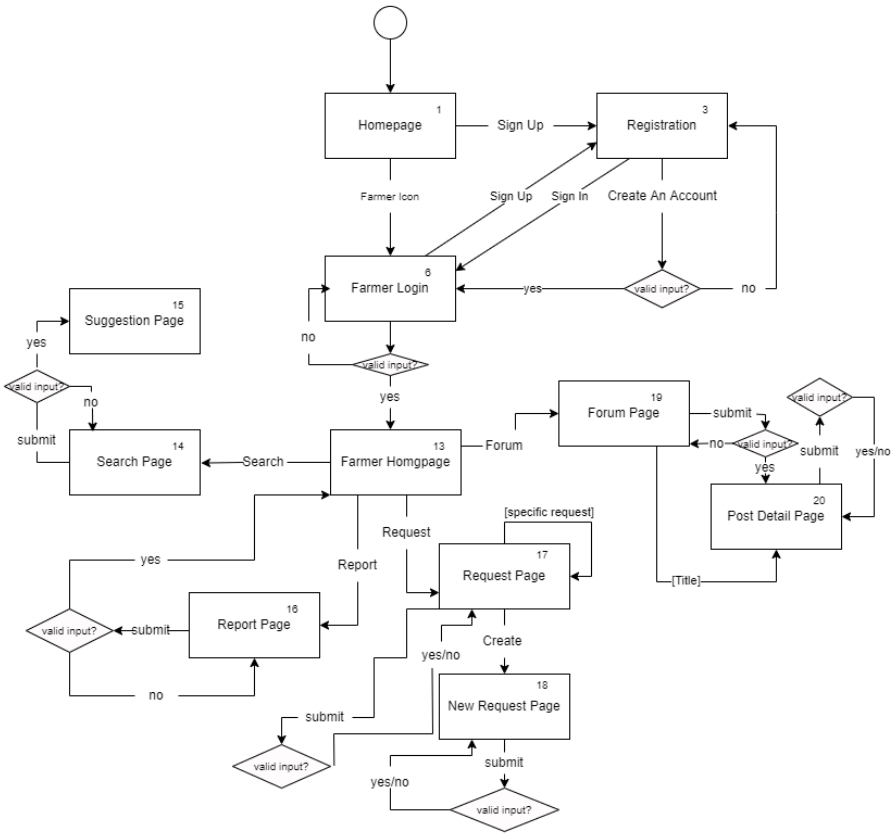

Diagram 3: Web application flow for Farmer

4. REQUIREMENTS TRACEABILITY

5. IMPLEMENTATION, INTEGRATION AND TEST PLAN

// Identify here the order in which you plan to implement the subcomponents of your system and the order in which you plan to integrate such subcomponents and test the integration.

5.1 Development Process

// dependency diagram: https://www.uml-diagrams.org/dependency.html

5.2 Implementation Plan

//

5.3 Integration Sequence

5.4 System Testing

Once all system components have been integrated, System Testing begins. This process aims at verifying functional and non-functional requirements and must take place in a testing environment which is as close as possible to the production environment. Specifically, DREAM System will be subjected to the following tests:

- Functional testing.
- Performance testing.
- Load testing.
- Stress testing.

6. EFFORT SPENT

// In this section you will include information about the number of hours each group member has worked for this document.

TimeLine	Comment	Xu	Zhang	Hu
27/12/2021	Overview and arragement	3h		
28/12/2021	section 1	0.5h		
30/12/2021	arrangement section2	1h	1h	2h
31/12/2021-05/01/2022	Section2	8h	12h	4h
06/01/2022	Section2 review	3h	3h	3h

7. REFERENCES

- E. Di Nitto. Lecture Slides. Politecnico di Milano
- E. Di Nitto. Project Assignment AY 2021-2022. Politecnico di Milano
-