POLITECNICO
MILANO 1863

DREAM

Data-driven Predictive Farming in Telengana

DD

Design Document

Version 1.0 - 21/12/2021

Fateme Hajizadekiakalaye - 10831743
Reza Paki - 10832693

Table of Contents

1. Introduction **4**

1.1. Purpose 4

1.2. Scope 4

1.3. Definitions, Acronyms and Abbreviations 4

1.3.1. Definitions 4

1.3.2. Acronyms 5

1.3.3. Abbreviations 5

1.4. Revision History 5

1.5. Reference Documents 5

1.6. Document Structure 5

2. Architectural Design **6**

2.1. Overview 6

2.1.1. High-level Architecture 7

2.2. Component View 8

2.3. Deployment View

2.4. Runtime View

2.5. Component Interfaces

2.6. Selected Architectural Styles and Patterns

2.7. Other Design Decisions

3. User Interface Design

4. Requirements Traceability

5. Implementation, Integration and Test Plan

6. Effort Spent

7. References

1. Introduction

1.1. Purpose

The main goal of this document is to provide more technical and detailed information about the DREAM application discussed in the RASD document. In the fact, programmers who develop the application could use this document as a strong guide. DD represents deep detail on application design, hardware and software architecture of the system in terms of components and interactions among those components. It also gives a detailed presentation of the implementation plan, integration plan, and testing plan. In general, the main different features listed in this document are:

- The high-level architecture
- Main components of the system
- Interfaces provided by the components
- Design patterns adopted
- Implementation, integration and testing plan

1.2. Scope

DREAM is an application that is provided for both farmers and policymakers. Policymakers could manage farmers by this app and farmers could use published data by the application for farming purposes. DREAM uses some specific data about meteorological forecasts, the humidity of the soil, amount of water used for irrigation and then allows farmers to use this information for farming purposes. DREAM also provides a forum for sharing experiences or problems with other farmers. When a farmer creates a problem, he/she is allowed to choose who could answer the problem (farmers or agronomists, or both of those). Also, farmers could create discussions and share experiences with other farmers. Farmers insert their information such as production amount, amount of usage water, quality of production. Then Policymakers observe the farmers' activities and label them as good farmers or bad farmers. If a farmer labeled as a bad farmer gets technical guides from agronomists. This way policymakers allow to manage farmers and by giving special guides help to farmers. More detailed information can be found on the RASD document.

1.3. Definitions, Acronyms and Abbreviations

1.3.1. Definitions

Definition	Description
Steering initiatives	The provided solutions for farmers by agronomists.
Discussion forum	A meeting at which farmers can exchange ideas and opinions about a specific topic.

1.3.2. Acronyms

Acronyms	Description
DREAM	Data-dRiven prEdictive fArMing in Telengana
DD	Design Document
RASD	Requirement Analysis and Specification Document
DBMS	Data-Base Management System
API	Application Programming Interface

1.3.3. Abbreviations

Abbreviations	Description

1.4. Revision History

Version	Date	Modification
1.0	21/12/2021	First version

1.5. Reference Documents

- Specification Document: “01. Assignment RDD AY 2021-2022.pdf”
- UML diagrams: https://www.uml---diagrams.org/
- Slides of the lectures
- IEEE/ISO/IEC 29148-2018 - ISO/IEC/IEEE International Standard - Systems and software engineering - Life cycle processes - Requirements engineering

1.6. Document Structure

- **Section1**

  Brief description about DD and introducing purpose and scope. Also, including definitions, acronyms and abbreviations

- **Section2**

  The main part of DD is this section that contains architectural design choice, includes all the components, the interfaces used for the development of the application. Also, it contains deployment view, runtime view. In the end, is explained the architectural patterns chosen with the other design decisions.

- **Section3**
  Contains how should be the user interface on mobile applications.

- **Section4**
  Contains the traceability matrix that shows which components satisfy certain requirements.

- **Section 5**
  Including the implementation plan, integration plan, and testing plan, and shows how these plans are executed.

- **Section 6**
  Shows how much time is spent by each member of the group.

- **Section 7**
  Includes the reference documents.

2. Architectural Design

2.1. Overview

The architecture of this application is structured by 3 logic layers:

- **Presentation level (P):** This level manages the user communication with the system.
- **Application layer (A):** This layer handles the business logic of the application, its functionalities and moving data between the other two layers.
- **Data access layer (D):** This layer manages the information, with the corresponding access to the databases and prepares useful information for the users in the database and passes them along with the other layers.

Each user interfaces (farmer or policymaker) by calling methods to provide any functionalities starts the process. Callable methods are like using weather forecasts, creating discussion, sharing the problem, labeling the farmers, and so on.

Services for farmers:

- Farmer wants to use forecasted information: server analysis the queries and provides forecasted information from the database for that region and gives those to farmers.
- Farmer wants to create a discussion forum: farmer writes own messages and sends them. The server receives information and saves those in the database as well as replied messages.

- Farmer wants to create a problem: farmer writes own problem, the information about created problem will save in database as well as replied answers.
- Farmer wants to see the suggestions which has received from others: server analysis the query and provides the suggestions from database and gives them to the farmer.
- Each farmer inserts their production details in profile: thus, the server receives a query that contains these details and will send them to the database.

Services for policymakers:

- A policymaker could see farmers' performance and information and label them. The server will receive labels for each farmer and update farmer status in the database.
- A policymaker could see steering initiatives progress. The server receives the query and will provide data from the database and gives them to policy maker.

2.1.1. High-level Components

The chosen hardware architecture for this application is **Three-Tier.** The architecture contains three application layers. A tier to interface with the client, the second tier for the application level, and another server for the database management. This division is efficient because the server tier can make a permanent connection with the DBMS, which is less expensive. As well as, having a middle tier between user and server can guarantee more security to the access control of the database from the users.

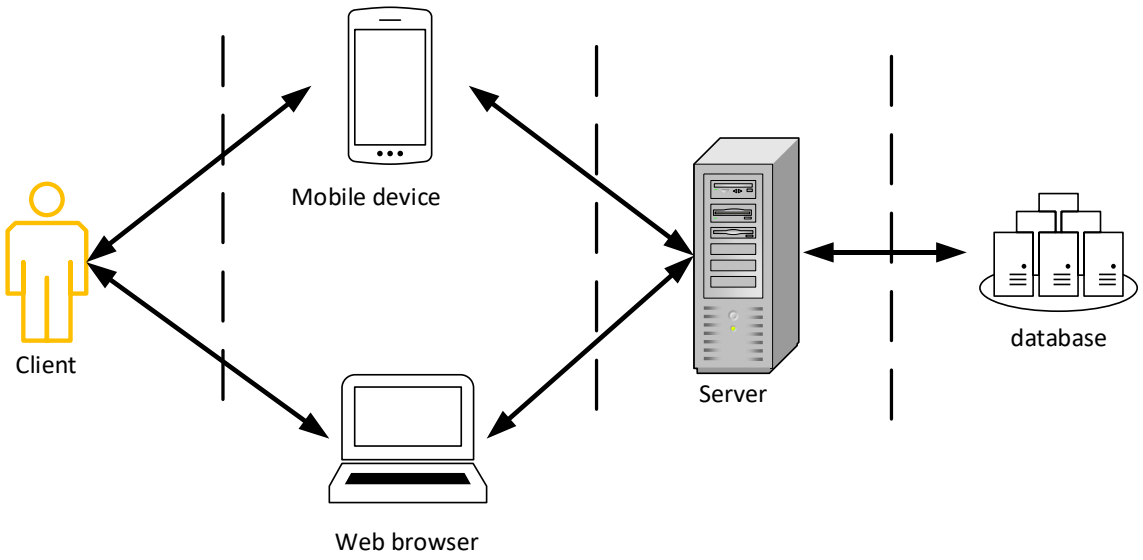

Figure 1 - Tree tier architecture

The following figure represents our system architecture, its high-level components and their interaction. The client could use this application by a smartphone or a computer. Then, user interactions will be processed by the application server. As well as, application server connected

to the database. Finally, the application server is allowed to use APIs for Google Maps and forecasted information.

These figures do not include all details about the architecture of the application, and in the next section, more details are represented.

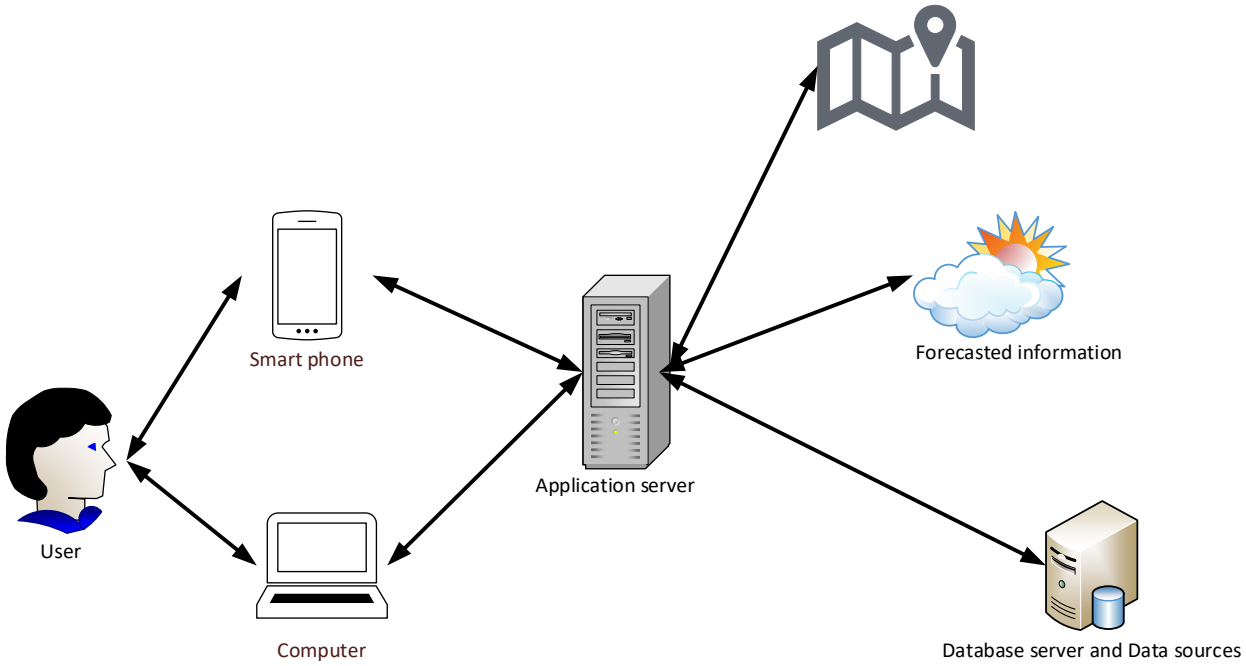

Figure 2 - System architecture

2.2. Component View

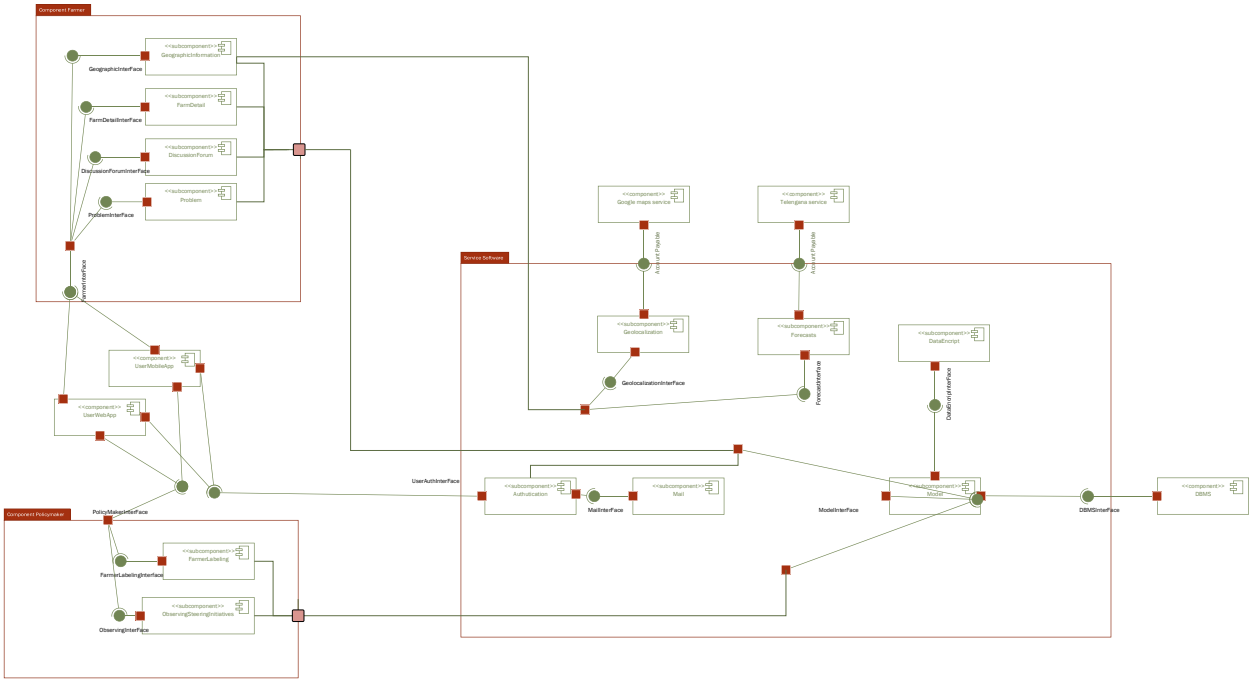

Figure 3 - Component View

2.3. Deployment View

2.4. Runtime View

6. Effort Spent

- **Student 1:**

Topics	Hours

- **Student 2:**

Topics	Hours

7. References