POLITECNICO
MILANO 1863

DREAM

Data-driven Predictive Farming in Telengana

RASD

Requirement Analysis and Specification Document

Version 1.0 - 29/11/2021

Fateme Hajizadekiakalaye - 10831743
Reza Paki - 10832693

Table of Contents

1. Introduction

1.1. Purpose

One of the most important sectors in each countries' economy is agriculture. Thus, the governments should keep it alive. On the other hand, many issues such as global warming, population increase and COVID-19 pandemic may have negative impacts on this vital sector. Scientists have predicted a significant loss in food supply by the end of century.

It was like a warning to the Telengana's government to come up with the idea of "DREAM". This idea is about designing and implementing a system which can prevent the mentioned disaster with the help of stakeholders, policy makers, farmers, market analysts, agronomists and even normal citizens.

First, in order to achieve the goals of the system, some specific data about Telengana's state have been collected. For example, meteorological forecasts, humidity of the soil, amount of water which use for irrigation, type of products and amount of products which produced by farmers. Then, with respect to this data, the DREAM system should allow policy makers to identify farmers with good performance and poor performance. Also the system should allow farmers to access to collected data and use them to improve their performance. The farmers should be allowed to share their problems with others and request for help.

This document focuses on ***Requirements Analysis and Specification Document (RASD)*** of the system and describes the main goals, the domain assumptions, the scenarios which may happen, the uses cases, the list of functional and non-functional requirements which system should fulfill and finally the diagrams to visualize the interactions between components and performance of the system.

1.1.1. Goals

Goals	Description
G1	Allow policy makers to identify farmers who are performing well.
G2	Allow policy makers to identify farmers who need help.
G3	Allow policy makers to see the result of the steering initiatives.
G4	Allow farmers to see weather forecast.
G5	Allow farmers to see humidity of soil.
G6	Allow farmers to see suggestions relating to specific crop to plan or specific fertilizer to use.
G7	Allow farmers to insert their type of products and produced amount per product.
G8	Allow farmers to insert their problems.
G9	Allow farmers to request for help and suggestion.
G10	Allow farmers to create discussion forums.

1.2. Scope

To manage farmers and help them this application provided and contains 3 main parts:

- Farmers login in the application and then insert their information such as location, amount of production, type of production, and so on. By inserting this information, they could get guides from policy makers and other farmers for improving the quality of the products.
- Policy makers use this application to identify the good and bad farmers by their performance, then they help farmers by giving solutions and guides.
- Accessing information collected by sensors, water irrigation systems, and governmental agronomists and allowing the farmers to use this information.

Farmers use collected information to improve their product quality and then they insert their information such as the amount of production, quality of production. Then policy makers could identify the farmers that worked well or worse and send some solutions and guides to them to improve their production. As well as, farmers get a chance to create forums and discuss problems and get solutions.

1.2.1. World Phenomena

World Phenomena	Description
WP1	Farmer plans crops.
WP2	Farmer irrigates crops.
WP3	Farmer uses fertilizers.
WP4	Sensors measure the humidity of soil.
WP5	Irrigation system measures the amount of water used by each farmer.
WP6	Meteorological adverse events such as flood, storm, lightening fire, etc. happen.
WP7	Farmer faces problems.

1.2.2. Shared Phenomena

Shared Phenomena	Description	Control
SP1	Farmer selects to see weather forecast.	World
SP2	System shows weather forecast to farmer.	System
SP3	Farmer selects to see humidity of soil.	World
SP4	System shows humidity of soil to farmer.	System
SP5	Farmers selects to see suggestions relating to specific crop to plan or specific fertilizer to use.	World
SP6	System shows suggestions relating to specific crop to plan or specific fertilizer to use.	System
SP7	Farmer inserts his/her type of products.	World
SP8	Farmer inserts his/her produced amount per product.	World
SP9	Policy maker selects to see a farmer’s detailed info.	World

SP10	System shows farmer's detailed info to policy maker.	System
SP11	Policy maker identifies that the performance of farmer is good or not, based on produced amount, humidity of soil, water consumption.	World
SP12	Farmer inserts his/her problem.	World
SP13	Farmer requests for help or suggestion.	World
SP14	Farmer creates discussion forum.	World
SP15	Policy maker selects to see the result of the steering initiatives.	World
SP16	System shows the result of the steering initiatives.	System

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.
Notification	A message shown to the user by system when he/she must be notified about something (ex: getting new message in forum).

1.3.2. Acronyms

Acronyms	Description
DREAM	Data-dRiven prEdictive fArMing in Telengana
RASD	Requirement Analysis and Specification Document
GPS	Global Positioning System
API	Application Programming Interface

1.3.3. Abbreviations

Abbreviations	Description
G	Goal
WP	World Phenomena
SP	Shared Phenomena
D	Domain Assumption
R	Requirement

1.4. Revision History

Version	Date	Modification
1.0	29/11/2021	First version

2.0	21/12/2021	• Acronyms added • Software interfaces added • Communication interfaces added • Software system attributes added • References added • Effort spent updated

1.5. Reference Documents

- Specification Document: “01. Assignment RDD AY 2021-2022.pdf”
- Course slides
- IEEE/ISO/IEC 29148-2018 - ISO/IEC/IEEE International Standard - Systems and software engineering - Life cycle processes - Requirements engineering

1.6. Document Structure

- **Section1**
  Overview of the purpose of the project and defining the scope of the system. Describe the specifications such as the definitions, acronyms, abbreviations, revision history, and references. As well as introducing the goals, world and share phenomena of the software.

- **Section2**
  Defining the main scenarios and then explaining the main features in the software by class diagram and statecharts. In user characteristics, the types of actors that use the application are explained. The product function subsection defined the functionalities of the application. In the end, the domain assumptions are defined.

- **Section3**
  The main part of the project which introduces interface requirements such as user interface, hardware interface, software interface, and communication interfaces. Presenting the functional requirements that are shown by use case diagrams and sequence diagrams. Then requirements are mapped to use cases.

- **Section4**
  Using Alloy language for analyzing the system and brief comments for clarifying the Alloy codes.

- **Section 5**
  Shows how much time spent by each member of group.

- **Section 6**
  Contains the references.

2. Overall Description

2.1. Product Perspective

2.1.1. Scenarios

- **Identify farmers' performance**

  Mario is a policy maker. In order to improve the agriculture in his region, he has to check the performance of farmers every now and then. He already registered himself in DREAM application before. Opening the application, logs in as a policy maker and selects to see the list of farmers. He clicks on each farmer name to see the details of their work. Then he calculates the performance of each farmer based on their produced amount, water consumption, humidity of soil of their farm and their resilience to meteorological adverse events. Finally, if the farmer's performance is well, he clicks on green button, if not, he clicks on red button and asks an agronomist to gives some notes and suggestions to the farmer to improve his/her work. As a result, the application sends a notification to the farmer, containing congratulation message or the suggestions.

- **View steering initiatives' result**

  Jenifer is a policy maker. Last month, she used the DREAM application to identify some farmers as good performance ones and some as poor performance ones. Now she wants to check whether the guidance which agronomists give to farmers has worked. So she opens the application and logs in as a policy maker and selects to view steering initiatives' result. The application shows her some diagrams. Each diagram illustrates a specific factor (produced amount, humidity of soil, variety of products, water consumption, etc.) over time. If the results improved, she wouldn't change anything. Otherwise, she asks agronomists to update the steering initiatives and publishes them. Thus, the application sends a notification to every farmer, containing new updates.

- **Insert products**

  Mike is a farmer. He heard about the DREAM application from his colleague. He registered himself in the app as a farmer and the position of his farm was inferred by the GPS. Now he wants to insert the detail of his work into application. So he opens the app and logs in as a farmer and selects to insert new product. First, the system asks him type of his product. He enters "potato". Second, the system asks him produced amount of "potato". He enters 1000 kg/month. Then, the system shows a bar from 0% to 100% and asks him how much of his "potatoes" were lost by the recent flood. He marks 10% and confirms to publish. Now, everyone can see his detail of work in DREAM application.

- **Request for help**

  Julia is one of the farmers in the DREAM application. She has a small farm and plans cucumber, tomatoes and celery. However, she faces a problem. Her harvested crops in the recent month are less than in previous months. So she decides to discuss her problem into app. She opens the app and logs in as a farmer and selects to insert a problem. The system asks her to discuss

the problem. She explains the type of her products, the amount of water which use for irrigation, the type of fertilizers which use and any detail about the problem and confirms. Then the system asks her to select one or both of the following options: 1. Request for help from other farmers, 2. Request for help from agronomists. She checks both options and confirms. Finally, the system publishes her problem and now everyone can see her problem in DREAM application.

- **View suggestions**
  Paulo is a farmer who has been labeled as “poor performance” by a policy maker recently. The system has sent him a notification containing some suggestions from policy maker. On the other hand, he has inserted his problem in app a few days ago and has requested for help from agronomists and other farmers. Now he wants to check his suggestions to improve his performance. He opens the app and logs in as farmer and selects to view suggestions. The system shows list of suggestions and he selects each to read.

- **View weather forecast and humidity of soil**
  Sarah is a new farmer and she wants to plan her first crops. Before planning, she should check the weather and soil moisture. So she opens the app and clicks on the map icon. The systems shows map of all regions. Also the system show two button. One for weather forecast and one for humidity of soil. Sarah selects her region on the map. Then the system magnifies her region. Now, she can easily switch between weather forecast and humidity of soil buttons to see whatever she wants.

- **Create discussion forum**
  Ali is a farmer. He just runs out of fertilizer. So he goes shopping and buys a new high quality fertilizer which has a low cost. Now, he’s very excited and want to share his experience with his colleagues. He opens the DREAM app and logs in as farmer and selects to create a discussion forum. The system asks him to choose a title for discussion. He enters “The best and cheapest fertilizer” and confirms. The system asks him to write a message to start the discussion. He explains his experience by detail and sends it. Now, other farmers can see his message and reply to him.

2.1.2. Class Diagram

Additional notes on the class diagram:

- As a design choice, a “Farmer” can owns only a “Farm”.
- There is an extra relationship between “Suggestion” and “Agronomist”. But, we didn’t consider “Agronomist”, because it is beyond the scope of our team for this project.

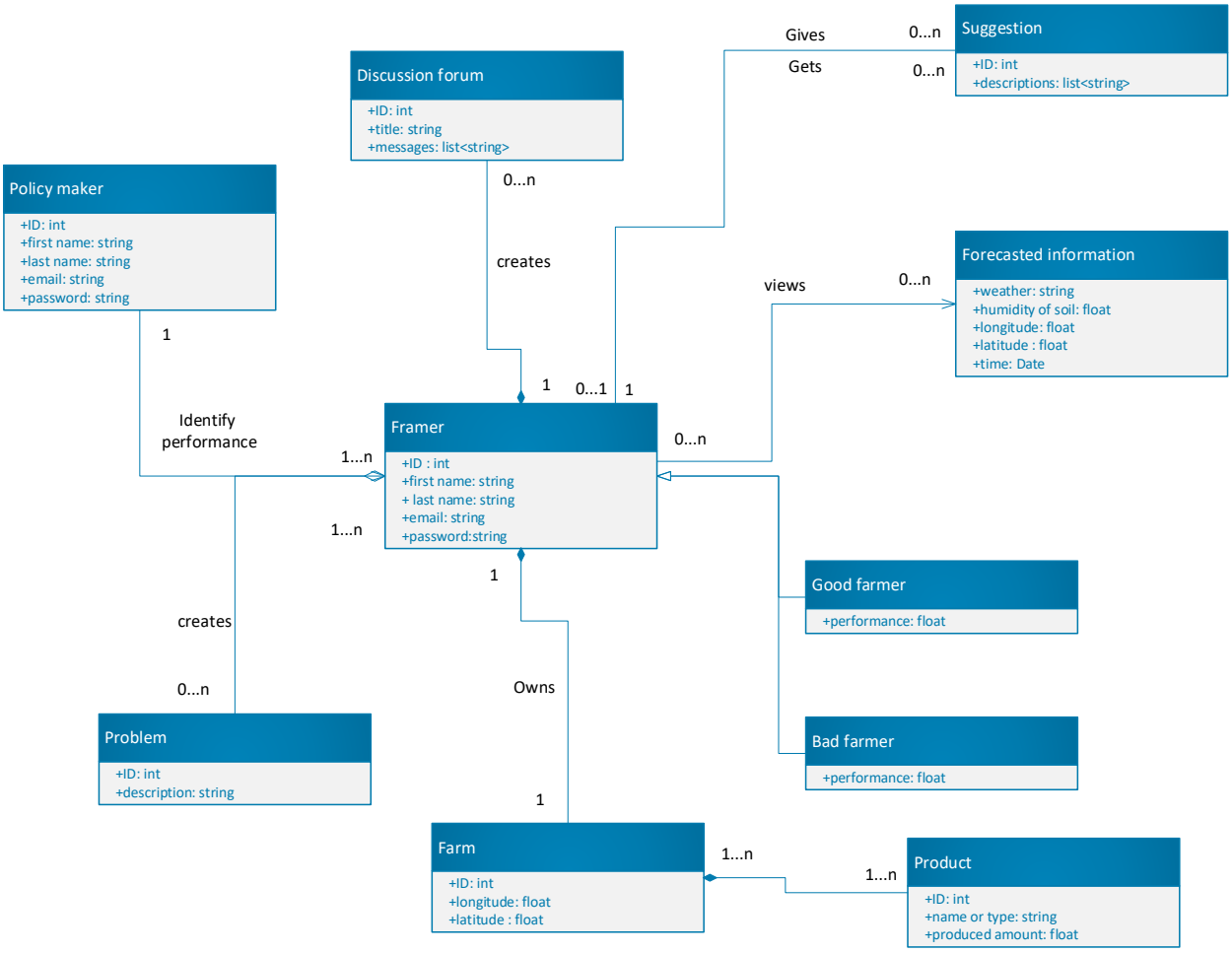


Figure 1 - Class diagram

2.1.3. Statecharts

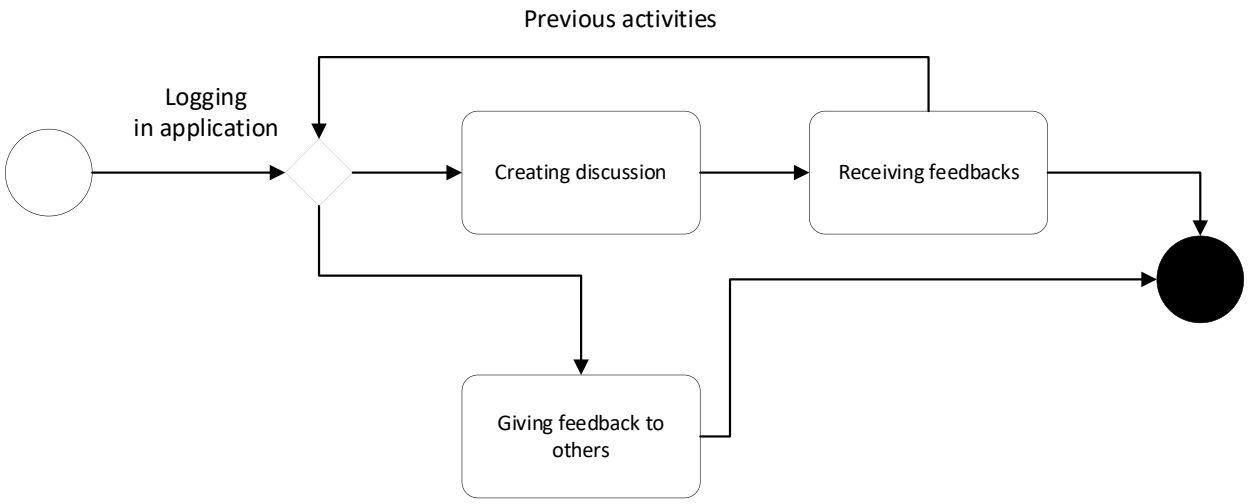


Figure 2 - Statechart 1

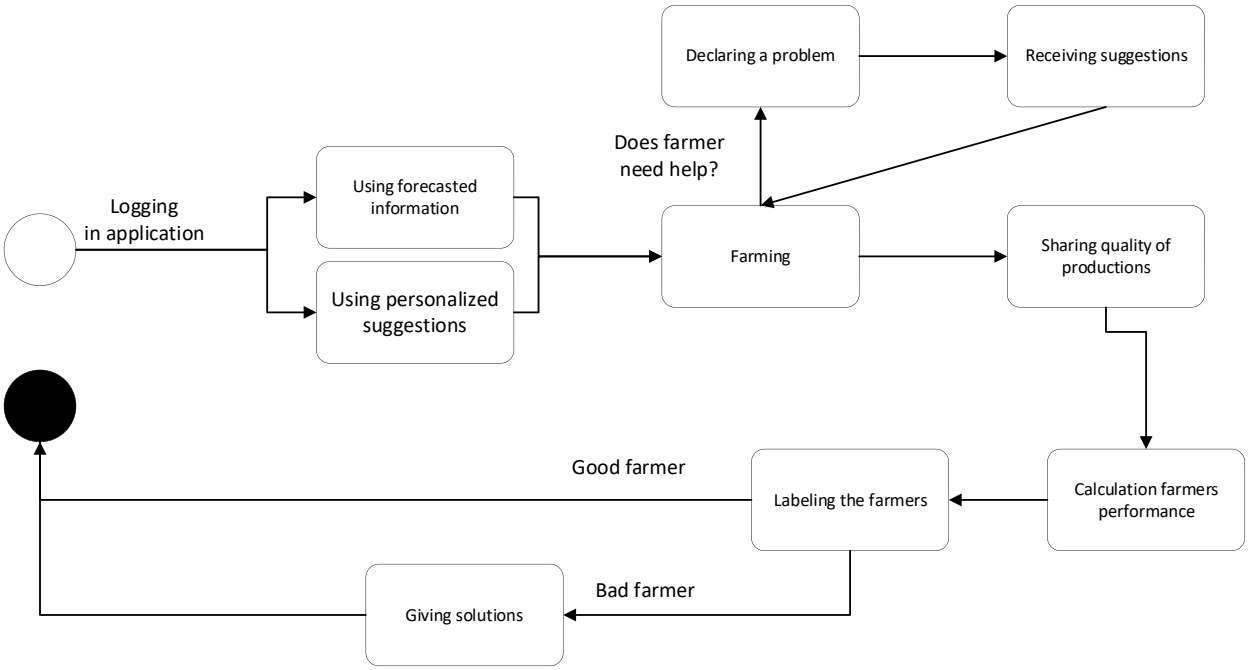


Figure 3 - Statechart 2

2.2. Product Functions

Here are described the majority of functions of the system. The less important ones are mentioned in other sections as well.

- **Request for help**
  The most important functionality of the DREAM application for farmers is requesting for help. Farmers can ask for help from other farmers or even agronomists whenever they face a new problem in their work. They should enter the detailed information about their work and the specific problem which happened. Then others can see the request and give any relevant suggestion if they have any experience in it.

- **Identify performance**
  The most important functionality of the DREAM application for policy makers is identifying farmers' performance. Policy makers should frequently monitor farmers' performance, in order to improve their policies. They can view the list of farmers and their detailed information by using this application. Then calculate their performance and label them as good or bad ones.

- **Create discussion forum**
  Another important functionality that DREAM application offers to farmers is to create a discussion forum. Farmers can start a new conversation about a specific topic with other farmers by selecting this option. They should enter a title and start messaging. Then other farmers can see and reply.

2.3. User Characteristics

Generally, the actors of the system can be divided into two groups:

- Farmer: a person who registers into DREAM application as a farmer and owes a farm. They can insert their products and their problems. Both they can request for help and give suggestion to others. Also they can view forecasted information and create discussion forums.
- Policy maker: a person who registers into DREAM application as a policy maker. They should monitor farmers' work and calculate their performance. Also they should check whether steering initiatives have had positive result.

2.4. Assumptions, Dependencies and Constraints

Domain Assumption	Description
D1	Position of farm which is indicated by farmer during the registration process is observed.
D2	Type of product which is inserted by farmer is observed.
D3	Produced amount which is inserted by farmer is observed.
D4	Farmers' performance which is calculated by policy maker is correct.
D5	Humidity of soil which is measured by sensors is accurate.
D6	Weather which is forecasted by IT providers is correct.
D7	Amount of water used by each farmer which is obtained by water irrigation system is accurate.
D8	Problem which is inserted by farmer is observed.
D9	Suggestion which is given by farmers or agronomists is related to problem.

3. Specific Requirements

3.1. External Interface Requirements

3.1.1. User Interfaces

The following mockups illustrate the most important sections of the interactions between DREAM application and users:
(Rest of them will be mentioned in the Design Document)

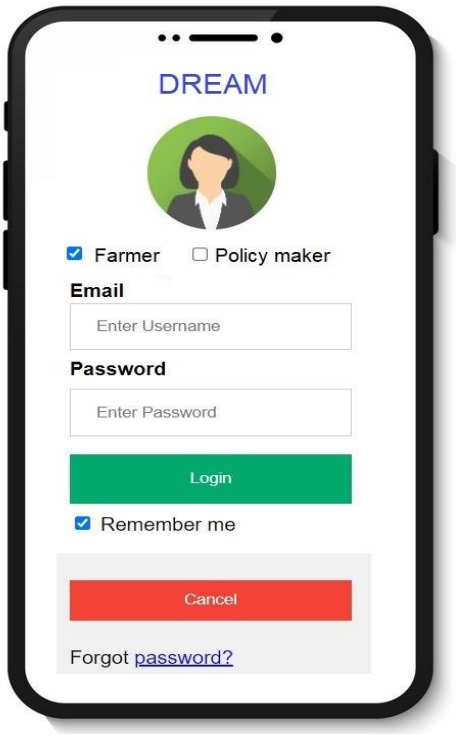


Figure 4 - Mockup: Login

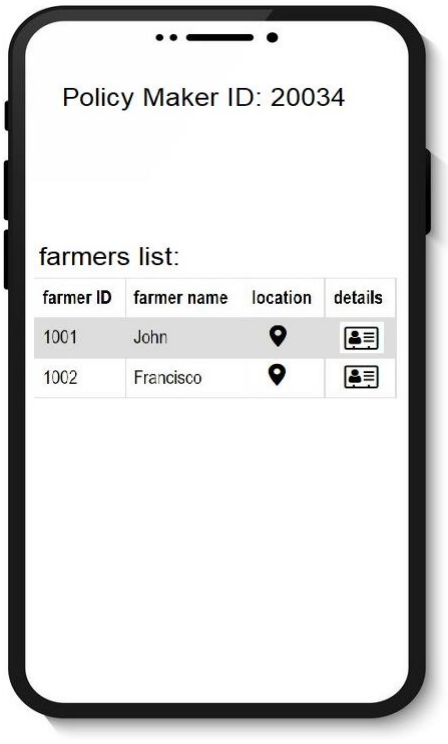


Figure 5 - Mockup: List of Farmers

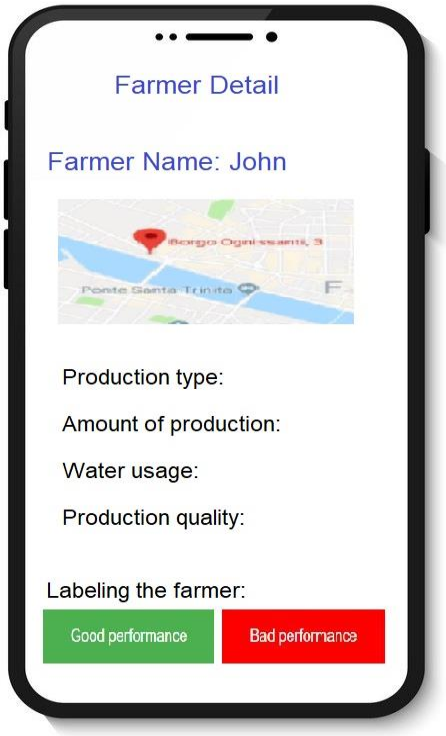


Figure 6 - Mockup: Farmer's Detail

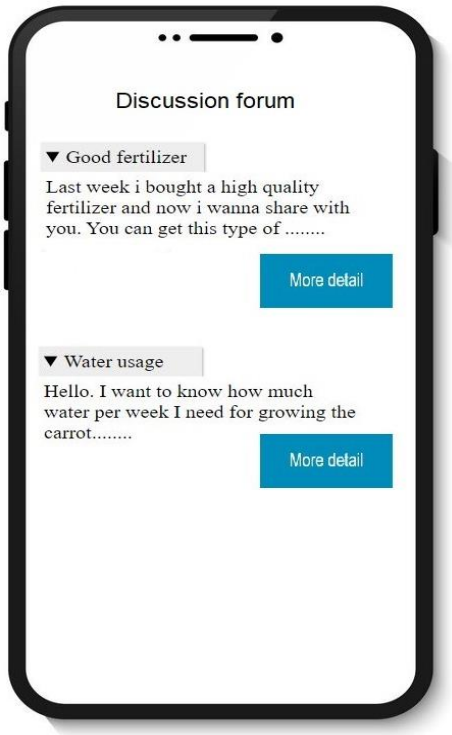


Figure 7 - Mockup: Discussion Forum

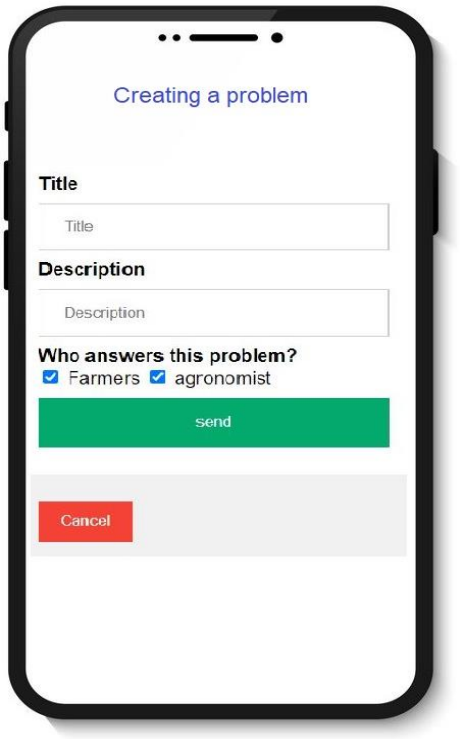


Figure 8 - Mockup: Creating a Problem

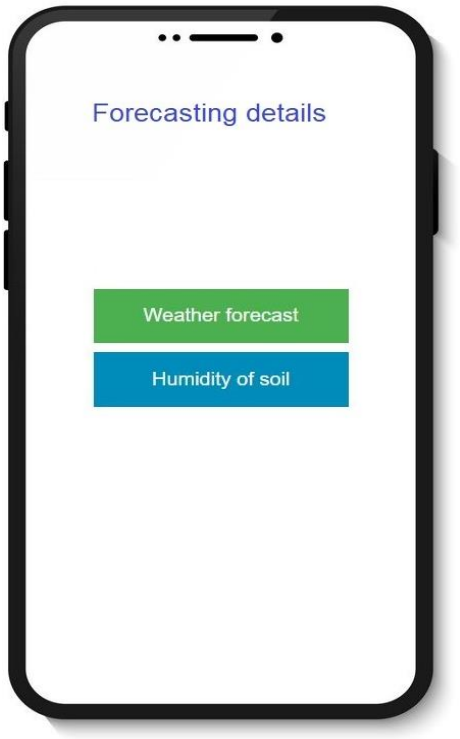


Figure 9 - Mockup: Forecasted Details

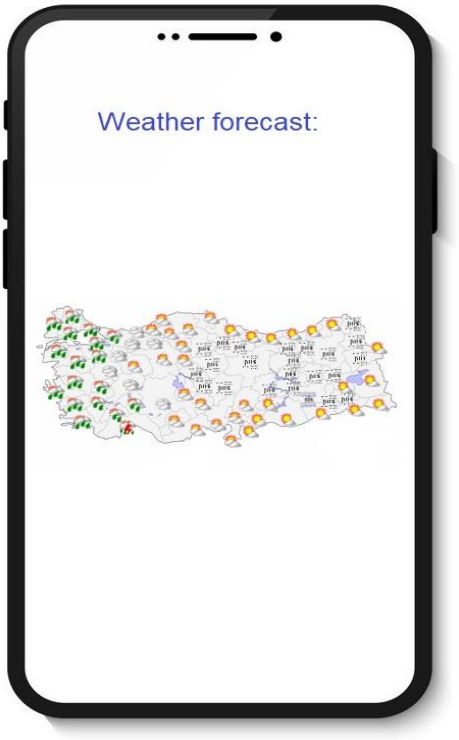


Figure 10 - Mockup: Weather

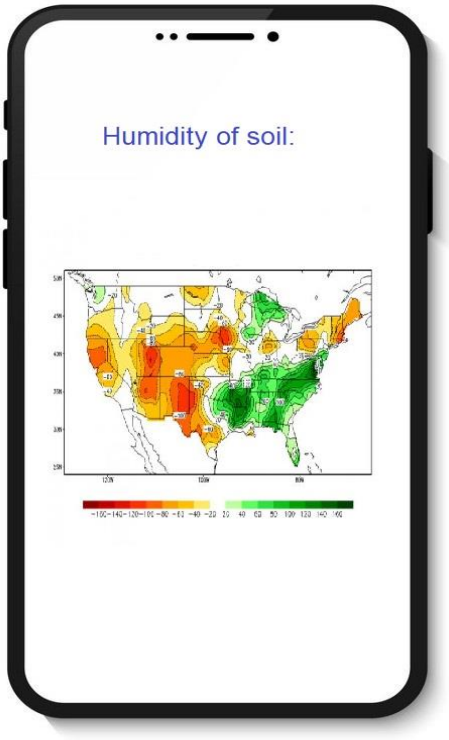


Figure 11 - Mockup: Humidity of Soil

3.1.2. Hardware Interfaces

All the hardware that the DREAM application requires is:

- Users have to use a web browser or a smartphone which can connect to the Internet and can use GPS services.
- In order to measure the humidity of soil, this app uses many sensors which already have been deployed on the territory.

3.1.3. Software Interfaces

The application uses an external interface:

- Map: The system provides public API for user to choose his/her farm position.

3.1.4. Communication Interfaces

- Communication with water irrigation system: The system obtains information related to water consumption from water irrigation system.
- Communication with Telengana: The system uses collected information by Telengana such as weather forecasting.

3.2. Functional Requirements

Requirement	Description
R1	The system must allow an unregistered policy maker to register.
R2	After a policy maker fills all the blanks in registration page correctly, the system must send an email to him/her, in order to confirm his/her email.
R3	The system must allow a logged out policy maker to login.
R4	The system must allow a policy maker to view list of farmers to identify their performance.
R5	During identifying performance process, the system must allow a policy maker to select a farmer among the list and view his/her detailed information.
R6	During identifying performance process, the system must allow a policy maker to click on “poor performance” button or “good performance” button.
R7	During identifying performance process, if a policy maker clicks on the “poor performance” button, the system must allow him/her to ask an agronomist to make a few suggestions to the farmer.
R8	After identifying performance process, the system must send a notification to the farmer containing congratulation message, if he/she was labeled as “good performance”.
R9	After identifying performance process, the system must send a notification to the farmer containing notes and suggestions, if he/she was labeled as “poor performance”.
R10	During viewing steering initiatives process, the system must allow a policy maker to view list of steering initiatives diagrams.

R11	During viewing steering initiatives process, if the results didn't improve, the system must allow a policy maker to ask agronomists to update the steering initiatives.
R12	After viewing steering initiatives process, if the results didn't improve, the system must send a notification to every farmer, containing new updates.
R13	The system must allow an unregistered farmer to register.
R14	After a farmer fills all the blanks in registration page correctly, the system must send an email to him/her, in order to confirm his/her email.
R15	After a farmer confirms his/her email, the system must allow him/her to identify his/her farm's position.
R16	The system must allow a logged out farmer to login.
R17	During inserting products process, the system must allow a farmer to insert his/her type of product.
R18	During inserting products process, the system must allow a farmer to insert his/her produced amount.
R19	During inserting products process, the system must allow a farmer to insert his/her crop loss by meteorological adverse events.
R20	After a farmer inserted all the information related to inserting products process, the system must allow him/her to confirm.
R21	After a farmer confirmed all the information related to inserting products process, the system must publish them and allow everyone to see them.
R22	During requesting for help process, the system must allow a farmer to insert his/her problem.
R23	During requesting for help process, the system must allow a farmer to select who he/she wants help from (agronomists, farmers, both).
R24	After a farmer inserted all the information related to requesting for help process, the system must allow him/her to confirm.
R25	After a farmer confirmed all the information related to requesting for help process, the system must publish them and allow the group which was selected by farmer to see them.
R26	After the system published all the information related to requesting for help process, the system must allow the group which was selected by farmer to give suggestions.
R27	After a suggestion is given to a farmer, the system must send a notification to him/her to inform him/her.
R28	The system must allow a farmer to view his/her suggestions
R29	During creating a discussion forum, the system must allow a farmer to insert the title of discussion.
R30	During creating a discussion forum, the system must allow a farmer to insert his/her message.

R31	After a farmer inserted all the information related to creating a discussion forum process, the system must allow him/her to confirm.
R32	After a farmer confirmed all the information related to creating a discussion forum process, the system must publish them and allow every farmer to see them.
R33	After the system published all the information related to creating a discussion forum process, the system must allow every farmer to reply.
R34	After a farmer replied in a discussion forum, the system must send a notification to creator of discussion forum to inform him/her.
R35	During viewing forecasted information process, the system must allow a farmer to select “weather forecast” or “humidity of soil”.
R36	During viewing forecasted information process, the system must allow a farmer to see his/her selected forecasted information.

3.2.1. Use Case Diagrams

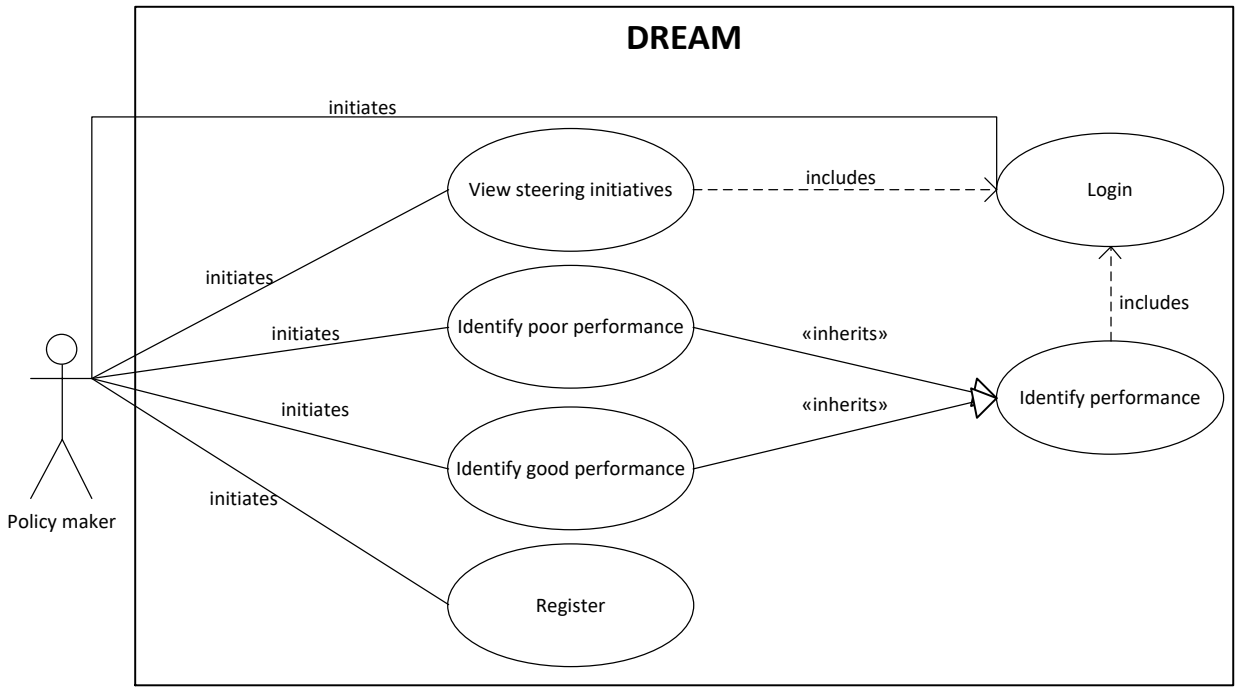


Figure 12 - Use case diagram: policy maker

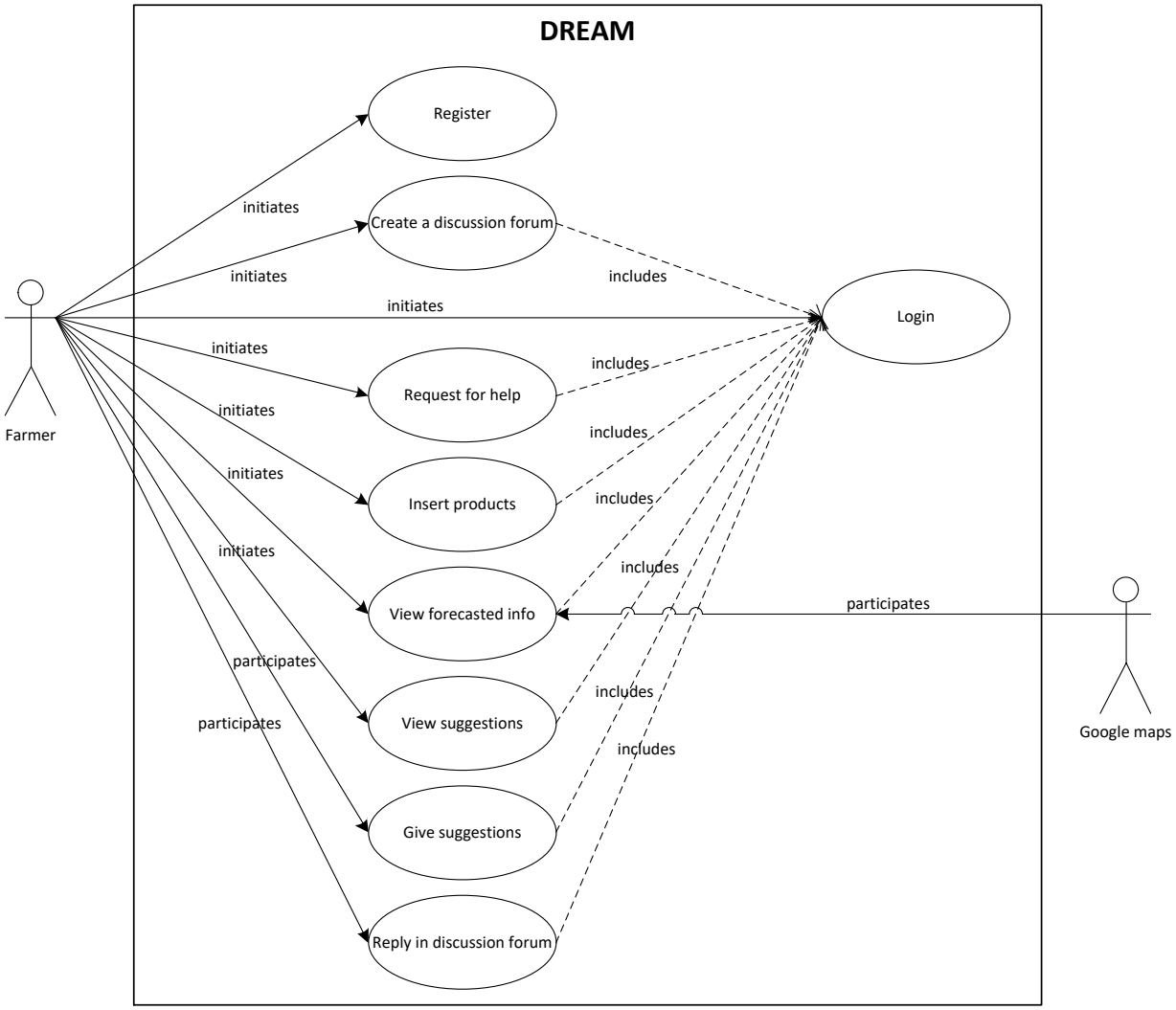


Figure 13 - Use case diagram: farmer

3.2.1. Use Cases

Name	Register
Actor	Policy maker
Entry conditions	A policy maker opens the application.
Events flow	1. The policy maker selects to register. 2. The policy maker checks the “policy maker” checkbox. 3. The policy maker fills all the blank fields. 4. The policy maker clicks on register button. 5. The system sends an email to him/her. 6. The policy maker confirms his/her email. 7. The system allows him/her to login.
Exit conditions	The policy maker is registered in the app.
Exceptions	1. The policy maker doesn’t check any checkboxes. In this case, the system warns him/her to check a checkbox.

	2. The policy maker doesn't fill all the blank fields. In this case, the system warns him/her to fill all the blank fields. 3. The inserted email has been already registered. In this case, the system warns him/her to login instead of register. 4. The inserted password is too weak. In this case, the system warns him/her to choose a stronger password.

Name	Login
Actor	Policy maker
Entry conditions	1. A policy maker opens the application. 2. He/she has already registered.
Events flow	1. The policy maker selects to login. 2. The policy maker checks the "policy maker" checkbox. 3. The policy maker enters email and password. 4. The policy maker clicks on login button.
Exit conditions	The system allows the policy maker to login.
Exceptions	1. The policy maker checks the "farmer" checkbox. 2. The policy maker enters the wrong email. 3. The policy maker enters the wrong password. In all cases, the system warns him/her to enter the correct data.

Name	Identify good performance
Actor	Policy maker
Entry conditions	1. A policy maker has already logged in successfully. 2. He/she is on the main view of application.
Events flow	1. The policy maker selects to view list of farmers. 2. The system shows list of farmers. 3. The policy maker clicks on a farmer. 4. The system show the farmer's details, a "good performance" button and a "poor performance" button. 5. The policy maker calculates performance of the farmer based on produced amount, water consumption, farm's humidity of soil and resilience to meteorological adverse events. 6. The policy maker clicks on "good performance" button. 7. The system sends a congratulation notification to the farmer.
Exit conditions	The policy maker labeled the farmer as "good performance".
Exceptions	1. The policy maker clicks on "poor performance" button. In this case, the system performs step 7 in the "identify poor performance" use case. (next use case table)

Name	Identify poor performance
Actor	Policy maker
Entry conditions	1. A policy maker has already logged in successfully. 2. He/she is on the main view of application.
Events flow	1. The policy maker selects to view list of farmers. 2. The system shows list of farmers. 3. The policy maker clicks on a farmer. 4. The system show the farmer's details, a "good performance" button and a "poor performance" button. 5. The policy maker calculates performance of the farmer based on produced amount, water consumption, farm's humidity of soil and resilience to meteorological adverse events. 6. The policy maker clicks on "poor performance" button. 7. The system allows the policy maker to ask agronomists to give suggestions to the farmer. 8. The policy maker asks an agronomist and confirms. 9. The system sends a notification to the farmer containing the suggestions.
Exit conditions	The policy maker labeled the farmer as "poor performance".
Exceptions	1. The policy maker clicks on "good performance" button. In this case, the system performs step 7 in the "identify good performance" use case. (previous use case table) 2. During step 8, the policy maker confirms without asking an agronomist. In this case, the system warns him/her to ask an agronomist.

Name	View steering initiatives
Actor	Policy maker
Entry conditions	1. A policy maker has already logged in successfully. 2. He/she is on the main view of application.
Events flow	1. The policy maker selects to view steering initiatives. 2. The system shows him/her some diagrams some diagrams which illustrate specific factors (produced amount, humidity of soil, variety of products, water consumption, etc.) over time.
Exit conditions	The system illustrated the information which the policy maker requested for.
Exceptions	-

Name	Register
Actor	Farmer
Entry conditions	A farmer opens the application.
Events flow	1. The farmer selects to register. 2. The farmer checks the “farmer” checkbox. 3. The farmer fills all the blank fields. 4. The farmer clicks on register button. 5. The system sends an email to him/her. 6. The farmer confirms his/her email. 7. The system asks him/her to identify his/her farm’s position. 8. The farmer identifies his/her farm’s position. 9. The system allows him/her to login.
Exit conditions	The farmer is registered in the app.
Exceptions	1. The farmer doesn’t check any checkboxes. In this case, the system warns him/her to check a checkbox. 2. The farmer doesn’t fill all the blank fields. In this case, the system warns him/her to fill all the blank fields. 3. The inserted email has been already registered. In this case, the system warns him/her to login instead of register. 4. The inserted password is too weak. In this case, the system warns him/her to choose a stronger password.

Name	Login
Actor	Farmer
Entry conditions	1. A farmer opens the application. 2. He/she has already registered.
Events flow	1. The farmer selects to login. 2. The farmer checks the “farmer” checkbox. 3. The farmer enters email and password. 4. The farmer clicks on login button.
Exit conditions	The system allows the farmer to login.
Exceptions	1. The farmer checks the “policy maker” checkbox. 2. The farmer enters the wrong email. 3. The farmer enters the wrong password. In all cases, the system warns him/her to enter the correct data.

Name	Insert products
Actor	Farmer
Entry conditions	1. A farmer has already logged in successfully. 2. He/she is on the main view of application.

Events flow	1. The farmer selects to insert a new product. 2. The system asks him/her to enter type of product and produced amount. 3. The farmer inserts type of product and produced amount. 4. The system shows a bar from 0% to 100% and asks the farmer how much of his/her product were lost by the meteorological adverse events. 5. The farmer identifies percentages of his/her crop loss and confirms.
Exit conditions	The system publishes farmer's new product in the app.
Exceptions	1. The farmer confirms without inserting type of product or produced amount or crop loss percentage. In this case, the system warns him/her to insert mentioned data. 2. The farmer cancels the operation. In this case, the system returns to the main page.

Name	Request for help
Actor	Farmer
Entry conditions	1. A farmer has already logged in successfully. 2. He/she faces a problem. 3. He/she is on the main view of application.
Events flow	1. The farmer selects to insert a problem. 2. The system asks him/her to insert the problem. 3. The farmer inserts the problem and confirms. 4. The system shows the following options and asks him/her to check at least one: 1. Request for help from other farmers, 2. Request for help from agronomists. 5. The farmer checks and confirms.
Exit conditions	The system publishes farmer's problem in the app.
Exceptions	1. The farmer confirms without inserting the problem. In this case, the system warns him/her to insert the problem. 2. The farmer doesn't check any checkbox. In this case, the system warns him/her to check at least one checkbox.

Name	Create a discussion forum
Actor	Farmer
Entry conditions	1. A farmer has already logged in successfully. 2. He/she is on the main view of application.
Events flow	1. The farmer selects to create a discussion forum.

	2. The system asks him/her to insert a title. 3. The farmer inserts title and confirms. 4. The system asks him/her to write a message. 5. The farmer write a message and confirms.
Exit conditions	The system creates the discussion forum and every farmer can see and reply it.
Exceptions	1. The farmer confirms without inserting a title. In this case, the system warns him/her to insert a title. 2. The farmer confirms without writing a message. In this case, the system warns him/her to write a message.

Name	Reply in discussion forum
Actor	Farmer
Entry conditions	1. A farmer has already logged in successfully. 2. He/she is on a discussion forum. 3. He/she wants to reply to a message
Events flow	1. The farmer selects to reply. 2. The system asks him/her to write a message. 3. The farmer writes a message and confirms.
Exit conditions	The system sends farmer's message in the discussion forum.
Exceptions	1. The farmer confirms without writing a message. In this case the system asks him/her to write a message.

Name	View forecasted information
Actor	Farmer
Entry conditions	1. A farmer has already logged in successfully. 2. He/she is on the main view of application.
Events flow	1. The farmer selects the map icon to view forecasted information. 2. The systems shows two option and asks him/her to select one: weather forecast and humidity of soil. 3. The farmer selects one of the items. 4. The systems shows map of selected option.
Exit conditions	The system showed the information which the farmer requested for.
Exceptions	-

Name	View suggestions
Actor	Farmer
Entry conditions	1. A farmer has already logged in successfully. 2. He/she is on the main view of application.

Events flow	1. The farmer selects to view suggestion. 2. The system shows his/her list of suggestions. 3. The farmer clicks on a suggestion to read it. 4. The system shows the selected suggestion.
Exit conditions	The system showed the information which the farmer requested for.
Exceptions	-

Name	Give suggestions
Actor	Farmer
Entry conditions	1. A farmer has already logged in successfully. 2. He/she is viewing a problem of another farmer. 3. He/she has a suggestion for the problem.
Events flow	1. The farmer selects to give a suggestion. 2. The system asks him/her to write the suggestion. 3. The farmer writes the suggestion and confirms.
Exit conditions	The system sends the suggestion to the other farmer.
Exceptions	1. The farmer confirms without writing the suggestion. In this case, the system warns him/her to write the suggestion.

3.2.3. Sequence Diagrams

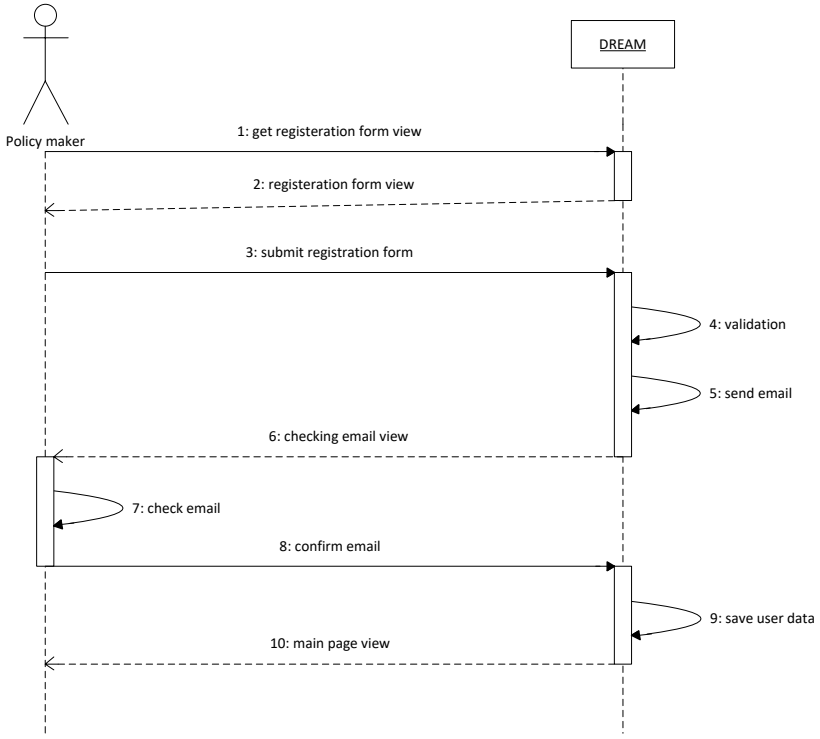


Figure 14 - Sequence diagram: policy maker registration

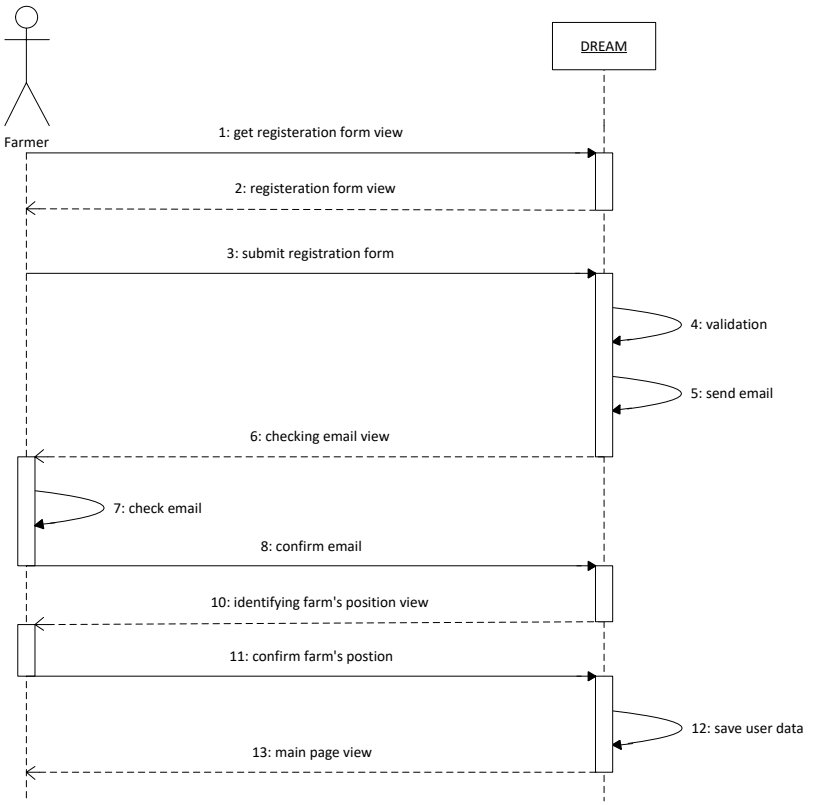


Figure 15 - Sequence diagram: farmer registration

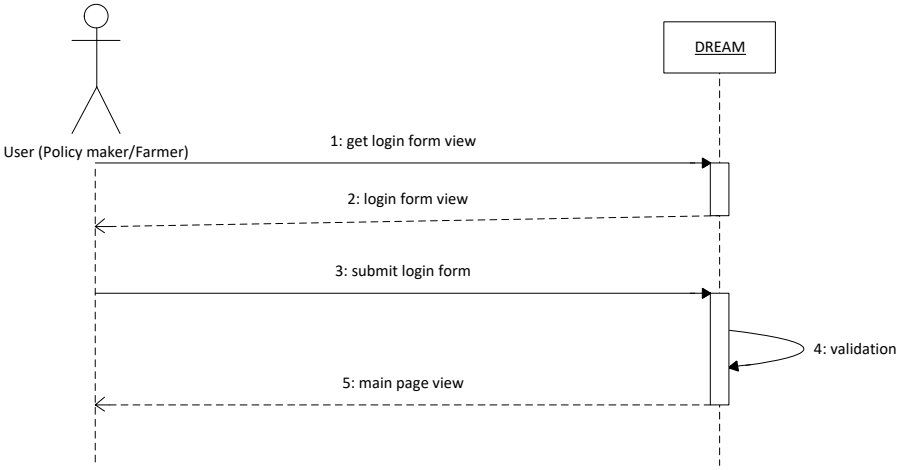


Figure 16 - Sequence diagram: login

In Figure 17, there is also an agronomist actor implicitly, but implementing agronomist is outside the scope of our team.

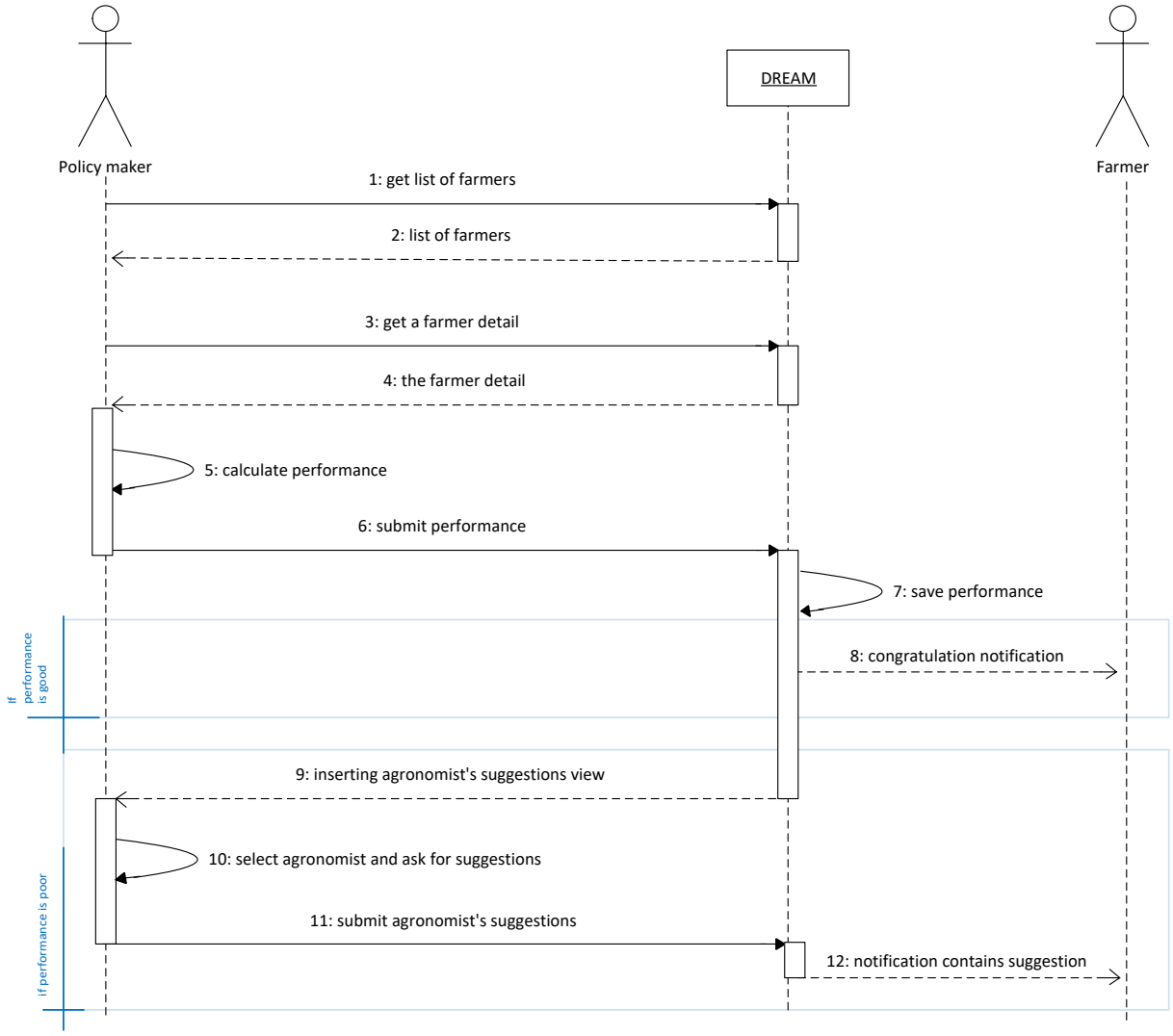


Figure 17 - Sequence diagram: identify performance

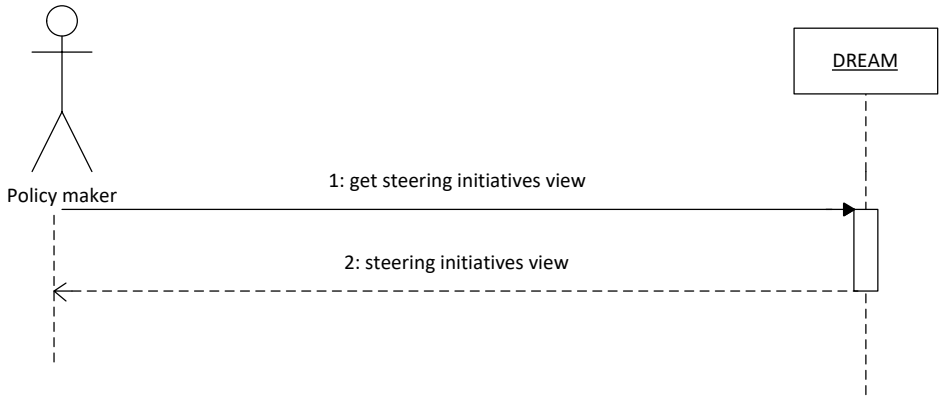


Figure 18 - Sequence diagram: view steering initiatives

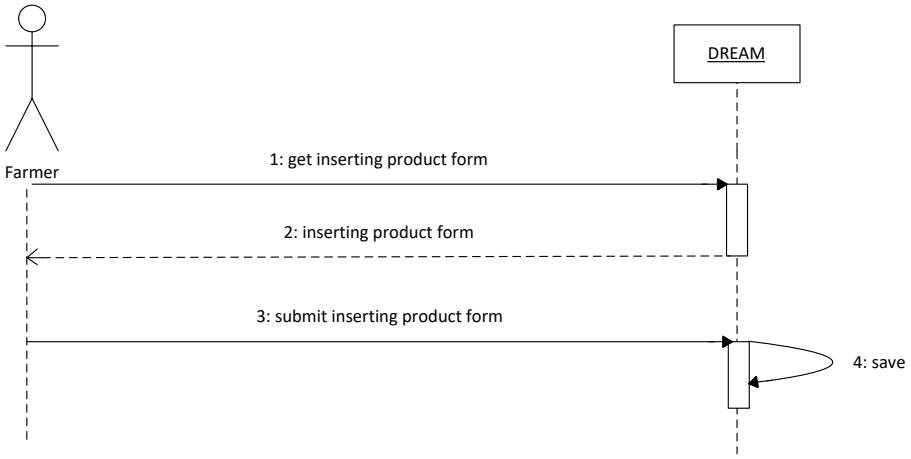


Figure 19 - Sequence diagram: insert product

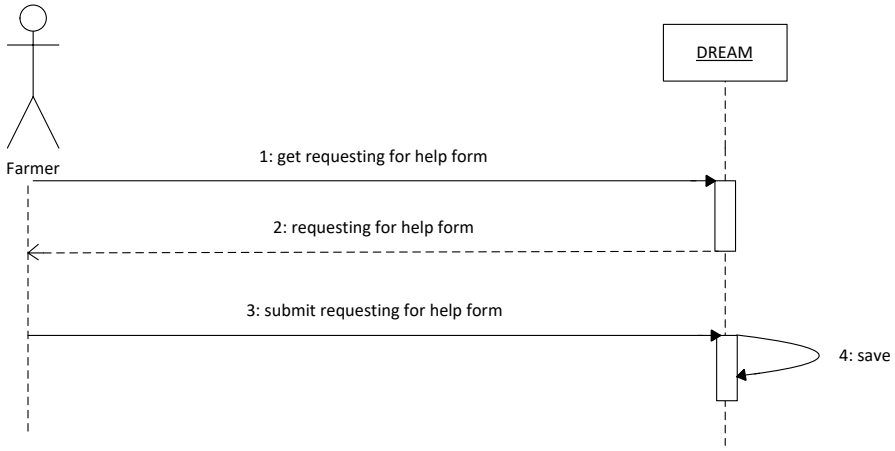


Figure 20 - Sequence diagram: request for help

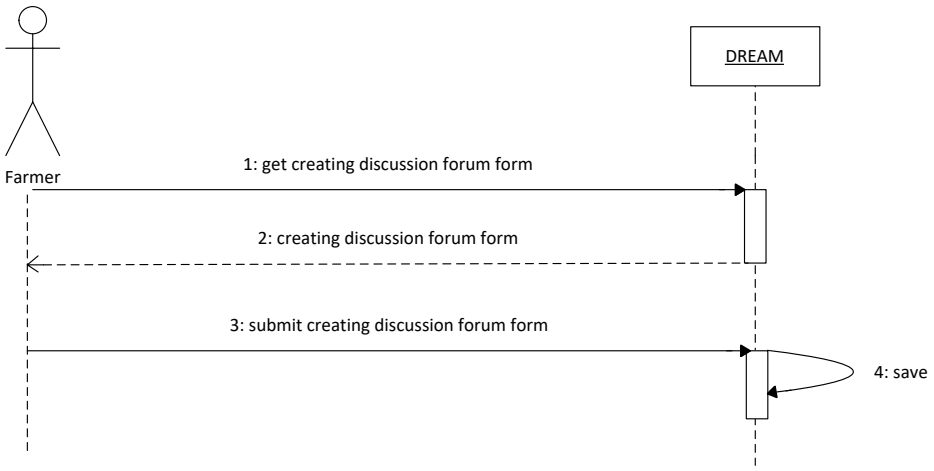


Figure 21 - Sequence diagram: create discussion forum

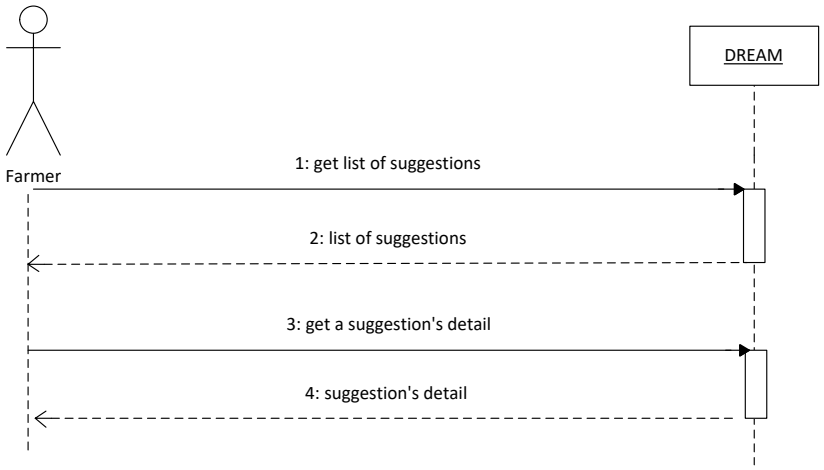


Figure 22 - Sequence diagram: view suggestions

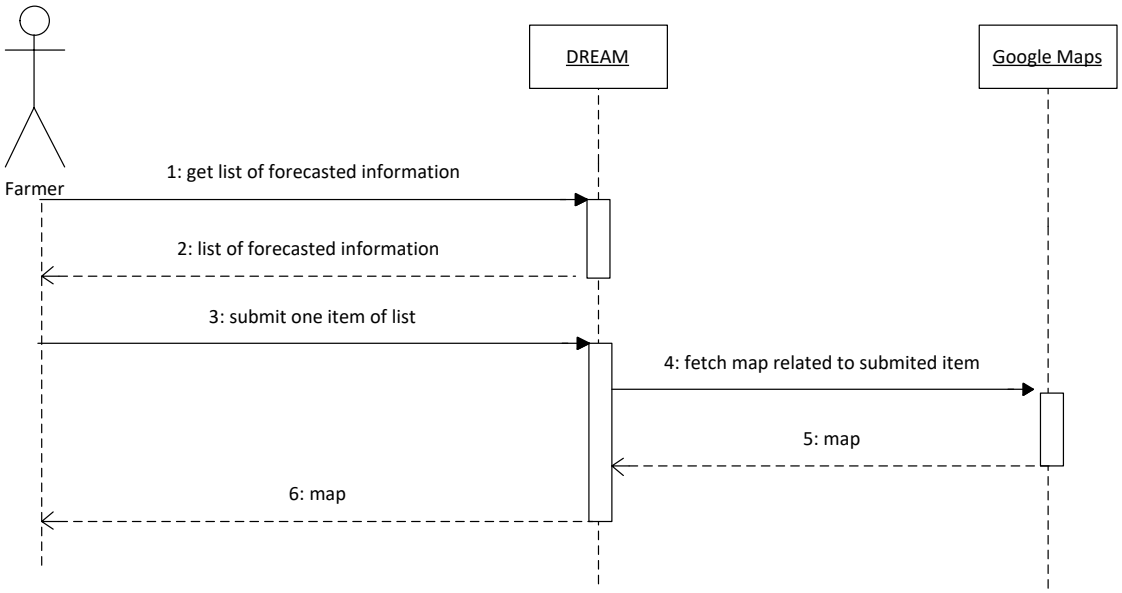


Figure 23 - Sequence diagram: view forecasted information

3.2.4. Mapping on Requirements

G1	**Allow policy makers to identify farmers who are performing well.**
D3	Produced amount which is inserted by farmer is observed.
D4	Farmers' performance which is calculated by policy maker is correct.
D5	Humidity of soil which is measured by sensors is accurate.
D7	Amount of water used by each farmer which is obtained by water irrigation system is accurate.
R3	The system must allow a logged out policy maker to login.

R4	The system must allow a policy maker to view list of farmers to identify their performance.
R5	During identifying performance process, the system must allow a policy maker to select a farmer among the list and view his/her detailed information.
R6	During identifying performance process, the system must allow a policy maker to click on “poor performance” button or “good performance” button.
R8	After identifying performance process, the system must send a notification to the farmer containing congratulation message, if he/she was labeled as “good performance”.

G2	**Allow policy makers to identify farmers who need help.**
D3	Produced amount which is inserted by farmer is observed.
D4	Farmers' performance which is calculated by policy maker is correct.
D5	Humidity of soil which is measured by sensors is accurate.
D7	Amount of water used by each farmer which is obtained by water irrigation system is accurate.
D9	Suggestion which is given by farmers or agronomists is related to problem.
R3	The system must allow a logged out policy maker to login.
R4	The system must allow a policy maker to view list of farmers to identify their performance.
R5	During identifying performance process, the system must allow a policy maker to select a farmer among the list and view his/her detailed information.
R6	During identifying performance process, the system must allow a policy maker to click on “poor performance” button or “good performance” button.
R7	During identifying performance process, if a policy maker clicks on the “poor performance” button, the system must allow him/her to ask an agronomist to make a few suggestions to the farmer.
R9	After identifying performance process, the system must send a notification to the farmer containing notes and suggestions, if he/she was labeled as “poor performance”.

G3	**Allow policy makers to see the result of the steering initiatives.**
D3	Produced amount which is inserted by farmer is observed.
D5	Humidity of soil which is measured by sensors is accurate.
D7	Amount of water used by each farmer which is obtained by water irrigation system is accurate.
R10	During viewing steering initiatives process, the system must allow a policy maker to view list of steering initiatives diagrams.

R11	During viewing steering initiatives process, if the results didn't improve, the system must allow a policy maker to ask agronomists to update the steering initiatives.
R12	After viewing steering initiatives process, if the results didn't improve, the system must send a notification to every farmer, containing new updates.

G4	**Allow farmers to see weather forecast.**
D6	Weather which is forecasted by IT providers is correct.
R35	During viewing forecasted information process, the system must allow a farmer to select "weather forecast" or "humidity of soil".
R36	During viewing forecasted information process, the system must allow a farmer to see his/her selected forecasted information.

G5	**Allow farmers to see humidity of soil.**
D5	Humidity of soil which is measured by sensors is accurate.
R35	During viewing forecasted information process, the system must allow a farmer to select "weather forecast" or "humidity of soil".
R36	During viewing forecasted information process, the system must allow a farmer to see his/her selected forecasted information.

G6	**Allow farmers to see suggestions relating to specific crop to plan or specific fertilizer to use.**
D8	Problem which is inserted by farmer is observed.
D9	Suggestion which is given by farmers or agronomists is related to problem.
R28	The system must allow a farmer to view his/her suggestions

G7	**Allow farmers to insert their type of products and produced amount per product.**
D2	Type of product which is inserted by farmer is observed.
D3	Produced amount which is inserted by farmer is observed.
R16	The system must allow a logged out farmer to login.
R17	During inserting products process, the system must allow a farmer to insert his/her type of product.
R18	During inserting products process, the system must allow a farmer to insert his/her produced amount.
R19	During inserting products process, the system must allow a farmer to insert his/her crop loss by meteorological adverse events.

R20	After a farmer inserted all the information related to inserting products process, the system must allow him/her to confirm.
R21	After a farmer confirmed all the information related to inserting products process, the system must publish them and allow everyone to see them.

G8	**Allow farmers to insert their problems.**
D8	Problem which is inserted by farmer is observed.
R22	During requesting for help process, the system must allow a farmer to insert his/her problem.

G9	**Allow farmers to request for help and suggestion.**
D8	Problem which is inserted by farmer is observed.
R23	During requesting for help process, the system must allow a farmer to select who he/she wants help from (agronomists, farmers, both).
R24	After a farmer inserted all the information related to requesting for help process, the system must allow him/her to confirm.
R25	After a farmer confirmed all the information related to requesting for help process, the system must publish them and allow the group which was selected by farmer to see them.
R26	After the system published all the information related to requesting for help process, the system must allow the group which was selected by farmer to give suggestions.
R27	After a suggestion is given to a farmer, the system must send a notification to him/her to inform him/her.

G10	**Allow farmers to create discussion forums.**
R29	During creating a discussion forum, the system must allow a farmer to insert the title of discussion.
R30	During creating a discussion forum, the system must allow a farmer to insert his/her message.
R31	After a farmer inserted all the information related to creating a discussion forum process, the system must allow him/her to confirm.
R32	After a farmer confirmed all the information related to creating a discussion forum process, the system must publish them and allow every farmer to see them.
R33	After the system published all the information related to creating a discussion forum process, the system must allow every farmer to reply.

R34	After a farmer replied in a discussion forum, the system must send a notification to creator of discussion forum to inform him/her.

3.3. Performance Requirements

- The system must be able to serve a great number of users simultaneously.
- The system must guarantee correct responses.
- The system must be able to send a response to a query less than 5 seconds since it has been received.
- The system must be available 99% of the time.

3.4. Design Constraints

3.4.1. Standards Compliance

- The system requires farmer's permission to retrieve their farm's position.
- The system maintain the data retrieved from the users with respect to the privacy laws.

3.4.2. Hardware Limitations

- The web browser or the smartphone which user is using must has the ability of connecting to the internet and using GPS services.
- The sensors which have been already deployed on the territory must have a measurement error of less than 0.01%.

3.4.3. Any other Constraint

- The information which provided by IT providers related to weather forecast must be accurate 99% of the time.
- The information which has been obtained by water irrigation system must be accurate 99% of the time.

3.5. Software System Attributes

3.5.1. Reliability

The system must be able to run continuously without any interrupts. Reliability of the system depends on the services of the system, and should be up for a 99% of time. This means the MTTR or downtime should be 3.65 days per years. In order to guarantee this time of downtime the system must have an appropriate infrastructure with a fully backup system located in different office that replicates the core services for covering general failure of the main system.

3.5.2. Availability

The system does not relate to the emergency; thus, we don't need high availability. Moreover, the service is not fully automated because has to rely on the policymakers that follow office hours.

3.5.3. Security

The provided information by policymakers or farmers is not sensitive, thus we don't need high security for the software. But for preventing some problems, we need to encrypt stored data and also the password of the users hashed before stored.

3.5.4. Maintainability

The software must be written in Java and codes must be written with good standards. The occurred problems must be detected easily and solved simply. The problem in one component must have not interrupted other components.

3.5.5. Portability

The software must be designed simply and implemented on different platforms. The software Run in different platforms must support Android and iOS operating systems for mobile devices, as well as a Web application for use simply in other platforms.

4. Formal Analysis Using Alloy

4.1. Code

```
////////// SIGNATURES //////////

sig Email {}

sig Password {}

sig Firstname {}

sig Lastname {}

abstract sig User {
    email: one Email,
    password: one Password,
    firstname: one Firstname,
    lastname: one Lastname
}
```

```
sig PolicyMaker extends User {
	labels: some Farmer
}

sig Farmer extends User {
	performance: one Int,
	owes: one Farm,
	faces: set Problem,
	views: set Map
}{
	performance >= 0
	performance =< 5
}
/* we defined the performance range from 0 to 5, for simplicity!
because alloy Int range is from -8 to 7
*/


sig Farm {
	locatedOn: one Location,
	products: some Product
}

sig Location {
	latitude: one Int,
	longitude: one Int
}{
	latitude >= -5 and latitude =< 5
	longitude >= -5 and longitude =< 5
}
/* we defined the latitude and longitude range from -5 to 5, for simplicity!
because alloy Int range is from -8 to 7
in fact, latitude range is from -90 to 90 and longitude range is from -180 to 180
*/
```

```
sig Product {
    type: one Type,
    producedAmount: one Int
}{
    producedAmount > 0
}

sig DiscussionForum {
    creator: one Farmer,
    title: one Title,
    messages: some Message
}

sig Suggestion {
    sender: one Farmer,
    receiver: one Farmer,
    solves: one Problem
}


sig Type {}

sig Problem {}

sig Message {}

sig Title {}

sig Map {}

////////// FACTS //////////

// each email always must be associated with only one user
fact EmailAssociationUser {
    all e: Email | one u: User | e in u.email
}

// each password always must be associated with at least one user
fact PasswordAssociationUser {
    all p: Password | some u: User | p in u.password
}

// each firstname always must be associated with at least one user
fact FirstnameAssociationUser {
    all fn: Firstname | some u: User | fn in u.firstname
}

// each lastname always must be associated with at least one user
fact LastnameAssociationUser {
    all ln: Lastname | some u: User | ln in u.lastname
}
```

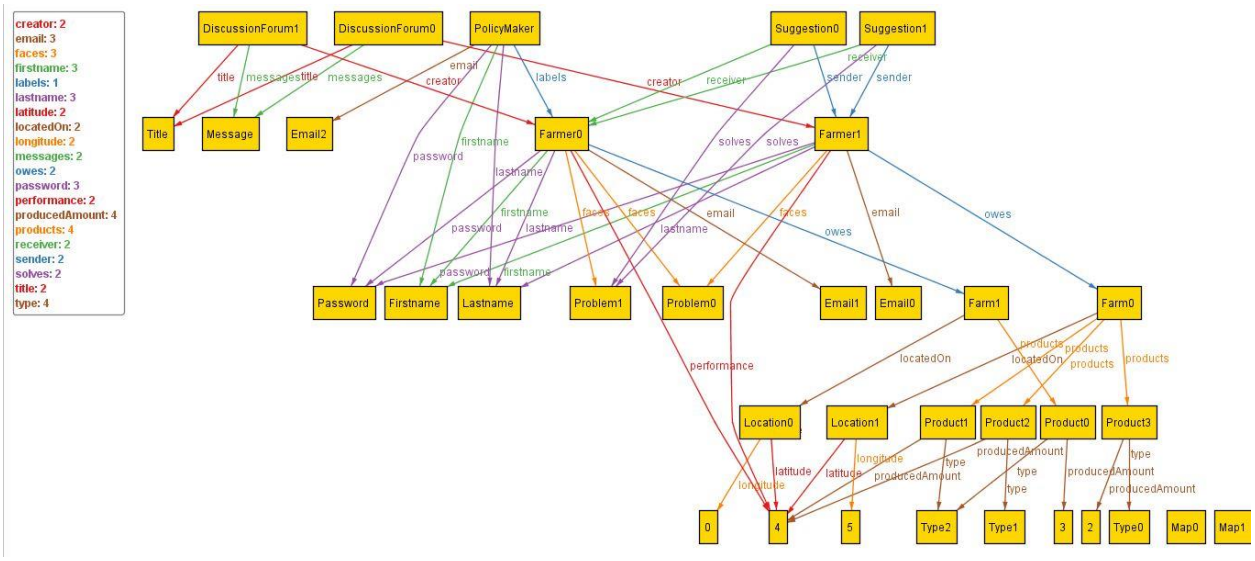
```
// each farm always must have only one farmer as owener
fact FarmOwenershipFarmer {
    all farm: Farm | one farmer: Farmer | farm in farmer.owes
}

// each location always must be associated with only one farm
fact LocationAssociationFarm {
    all l: Location | one f: Farm | l in f.locatedOn
}

// each location must refer to a different place
fact {
    all disj l1, l2: Location |
    l1.latitude = l2.latitude => l1.longitude != l2.longitude
}


// each product always must be associated with at least a farm
fact ProductAssociationFarm {
    all p: Product | some f: Farm | p in f.products
}

// each type of product always must be associated with at least one product
fact TypeAssociationProduct {
    all t: Type | some p: Product | t in p.type
}

// A farm shouln't have two product with same type
fact DifferentProductType {
    all f: Farm | all disj p1, p2: Product |
    p1 in f.products and p2 in f.products => p1.type != p2.type
}
```

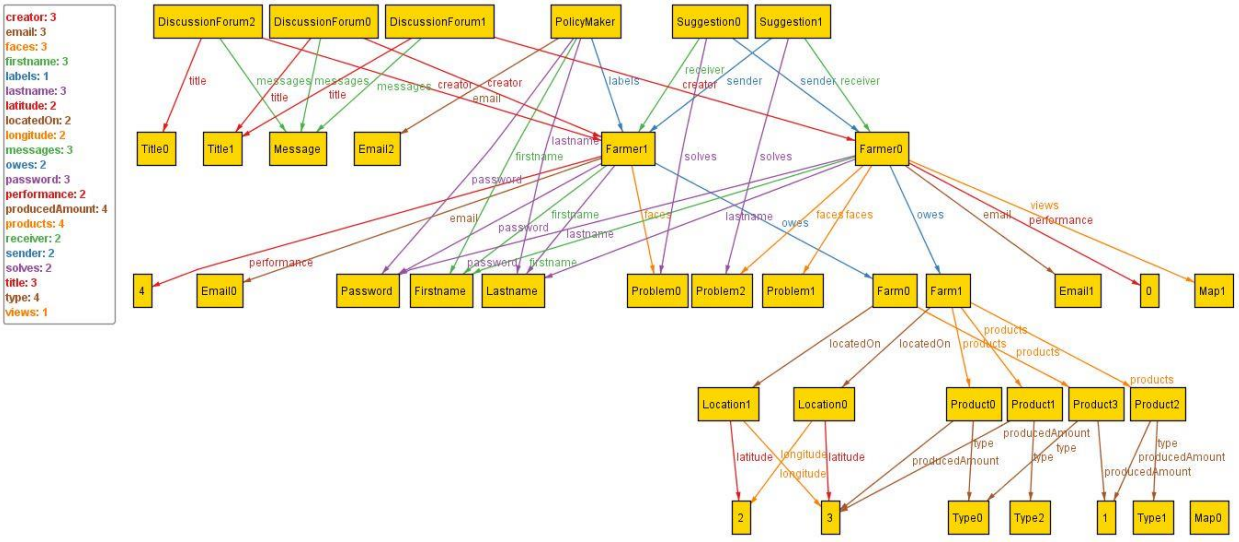
```
// each problem always must be associated with at least one farmer
fact ProblemAssociationFarmer {
    all p: Problem | some f: Farmer | p in f.faces
}

// sender and receiver of a suggestion must be different
fact DistinctSenderReceiver {
    all s: Suggestion | s.sender != s.receiver
}

// sender of a suggestion shouldn't be someone who faces the same problem
fact SuggestionSenderLimitation {
    all f: Farmer, s: Suggestion |
    s.solves in f.faces => f != s.sender
}

// receiver of a suggestion must be someone who faces the problem
fact SuggestionReceiverLimitation {
    all f: Farmer, s: Suggestion |
    f = s.receiver => s.solves in f.faces
}

// each message always must be associated with at least one discussion forum
fact MessageAssociationDiscussionForum {
    all m: Message | some df: DiscussionForum | m in df.messages
}

// each title always must be associated with at least one discussion forum
fact TitleAssociationDiscussionForum {
    all t: Title | some df: DiscussionForum | t in df.title
}

// a farmer shouldn't create two discussion forum with same title
fact DifferentDiscussionForumTitle {
    all f: Farmer | all disj df1, df2: DiscussionForum |
    f = df1.creator and f = df2.creator => df1.title != df2.title
}

pred show {
    #Farmer > 1
    #PolicyMaker > 0
    #Problem > 1
    #Suggestion > 1
    #DiscussionForum > 1
    #Map = 2
    #Product > 3
}

run show for 10
```

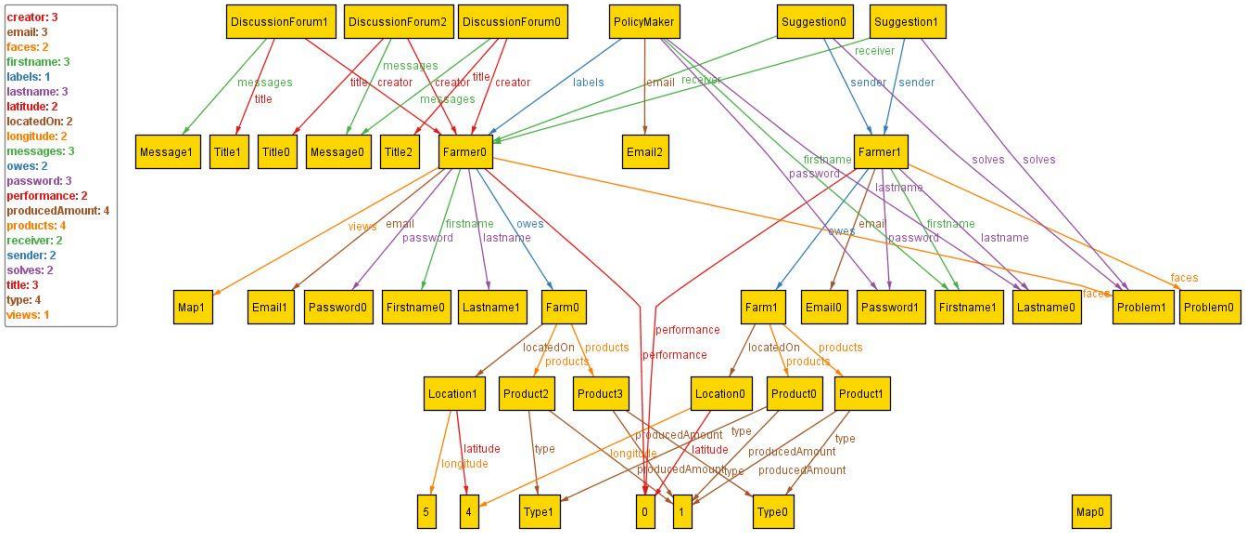
4.2. Results

```
Executing "Run show for 10"
  Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20
  49472 vars. 3300 primary vars. 86828 clauses. 2608ms.
  [redacted] found. Predicate is consistent. 1694ms.
```

4.3. Generated Instances







5. Effort Spent

- **Student 1:**

Topics	Hours
General reasoning	5h
Goals	2h
World phenomena and Shared phenomena	2h
Preparing Introduction	0.5h
Class Diagram	3h
Statecharts	0h
Domain Assumptions	1.5h
Functional Requirements	4h
Use cases	3.5h
Use case Diagrams	1.5h
Sequence Diagrams	2h
Performance Requirements and Design Constraints	1h
Software System Attributes	1h
Alloy	8.5h

- **Student 2:**

Topics	Hours
General reasoning	5h
Goals	2h
World phenomena and Shared phenomena	2h
Preparing Introduction	2.5h
Class Diagram	0h
Statecharts	2.5h

Domain Assumptions	1.5h
Functional Requirements	4h
Use cases	3.5
Use case Diagrams	2h
Sequence Diagrams	0.5h
Performance Requirements and Design Constraints	2h
Software System Attributes	0h
Alloy	6h

6. References

- Specification Document: “01. Assignment RDD AY 2021-2022.pdf”
- Course slides
- IEEE/ISO/IEC 29148-2018 - ISO/IEC/IEEE International Standard - Systems and software engineering - Life cycle processes - Requirements engineering