



POLITECNICO

MILANO 1863

SOFTWARE ENGINEERING 2 PROJECT
ACADEMIC YEAR 2021 - 2022

Requirements Analysis and Specifications Document

Valeria DETOMAS Sofia MARTELLOZZO

Professor

Elisabetta DI NITTO

Version 1
December 12, 2021

Contents

1	Introduction	3
1.1	Purpose	3
1.1.1	Goals	3
1.2	Scope	3
1.2.1	world phenomena	3
1.2.2	shared phenomena	4
1.3	Glossary	4
1.3.1	Definitions	4
1.3.2	Acronyms	4
1.3.3	Abbreviations	4
1.4	Document Structure	4
2	Overall Description	6
2.1	Product Perspective	6
2.1.1	Class Diagram	6
2.1.2	State Diagrams	8
2.2	Product Functions	10
2.2.1	Farmers insert data	10
2.2.2	Farmers visualize data	10
2.2.3	Identify how farmers are performing	11
2.2.4	Interaction between farmers	11
2.3	User Characteristics	11
2.3.1	Farmer	11
2.3.2	Policy Maker	12
2.4	Domain Assumptions, Dependencies and Constraints	12
2.4.1	Domain Assumptions	12
2.4.2	Constraints	13
3	Specific Requirements	14
3.0.1	External Interface Requirements	14
3.0.2	User Interfaces	14
3.0.3	Hardware Interfaces	26
3.0.4	Software Interfaces	26
3.0.5	Communication Interfaces	26
3.1	Functional Requirements	27
3.1.1	Use Case Diagram	27
3.1.2	Use Cases Description ad Sequence Diagram	28
3.1.3	Scenarios	50
3.1.4	Requirements	50
3.1.5	Goals	51
3.1.6	Traceability Matrix	57
3.2	Performance Requirements	57
3.3	Design Constraints	57
3.3.1	Standards Compliance	57

3.3.2	Any Other Constraints	57
3.4	Software System Attributes	57
3.4.1	Reliability	57
3.4.2	Availability	58
3.4.3	Security	58
3.4.4	Maintanability	58
3.4.5	Portability	58
4	Formal Analysis using Alloy	59
5	Effort Spent	59

1 Introduction

1.1 Purpose

The purpose of this document is to thoroughly describe Data-dRiver PrEdic-tive FArMing in Telengana(DREAM). It presents functional and non functional requirements of the system and its components. Moreover it provides use cases and scenarios for the users involved.

This document is meant as a contractual basis for the customer and the developer.

1.1.1 Goals

G1	allow policy makers to retrieve information from farmers and to evaluate their performance
G2	allow farmers to communicate with each other
G3	allow farmers to insert data, questions, problems
G4	allow farmers to retrieve information relevant for their activity (meteo, humidity..)

1.2 Scope

The aim of the system is to acquire and combine data and information of farmers in Telengana. The system will also provide support both to Telengana's policy makers and farmers thanks to new innovative technologies.

(non so se vogliamo mettere questo : The system consists in a back-end server application and in a web application front-end)

Thanks to the system policy makers are able to get a complete picture of the agriculture status in the whole state. In order to do this, Dreams provides information that makes them able to give incentives to those farmers who are performing well and keep track of those who needs help. The farmers have access to a forum on which they are able to communicate with other farmers, a forum to spread useful suggestions and to request for help to those who are having a harder time. The application provides a personalized page for each farmer in which they can find, based on his location and type of production, specific advice, meteorological forecasts and the condition of the soil. This information are already provided by Telengana's government. In this page they can also find several buttons, one that allows them to specify any problem that they face, another one to update their production trend. There is also another button to let those who are recognized as good farmers send advice to the system, so that everyone can improve their knowledge about the local farming ... Data concerning weather are already provided by Telengana's government,

1.2.1 world phenomena

- farmers decide type of production

- weather conditions influence production
- agronomists visit periodically farmers
- agronomists respond to help requests from farmers
- farmers can be identified as those who are performing well or not.
- farmers receive some type of advantage if they are the best one in their production activity

1.2.2 shared phenomena

- humidity of soil is measured by sensors
- amount of water used by each farmer is retrieved by water irrigation system
- Telengana's governments collects data concerning weather forecast
- farmers insert data about their production in the system
- farmers can insert problems they face into the system
- farmers can answer to requests for help from other farmers
- farmers can discuss with each other through the system
- the system identifies the farmers who are performing well

1.3 Glossary

1.3.1 Definitions

1.3.2 Acronyms

1.3.3 Abbreviations

1.4 Document Structure

1. Introduction

This section offers an introduction and a brief overview of the system that is presented in the document. It highlights the purpose of the system and the goals that are meant to be achieved with it. At the end there is also a glossary that contains a list of definitions, acronyms and abbreviations.

2. Overall Description

This section starts with a product perspective that contains a description of the system's domain through a class diagram. It includes also state diagrams which are used to give more details about the behavior of some objects in the model. The section contains also a clear description of the features offered by the system, it identifies the actors involved and it

describes their characteristics. At the end there are domain assumptions and general constraints.

3. Specif Requirements

This section enters into the details on how the system interacts with the external world. It describes the interfaces that are required and offered through several visual mockups. Moreover the section provides functional and nonfunctional requirements. Functional requirements are additionally described by use cases, sequence diagrams and scenarios. At the end the section focuses on nonfunctional requirements and various limitations that the system might face.

4. Formal Analysis using Alloy

This section provides the model described through Alloy language.

5. Effor Spent

This section has a record of the hours spent to complete this document.

2 Overall Description

2.1 Product Perspective

2.1.1 Class Diagram

The class diagram represented in Figure 1 represents the domain of the system. The main elements in the class diagram are:

- **User:** identifies two types of users who can access the application. Farmer and policy makers are the two categories of users that are present in the system. The distinction has to be applied in order to have different permissions.
- **Farm:**
- **Evaluation:**
- **Map:**
- **Weather Info:**
- **Production:**
- **Form:**
-

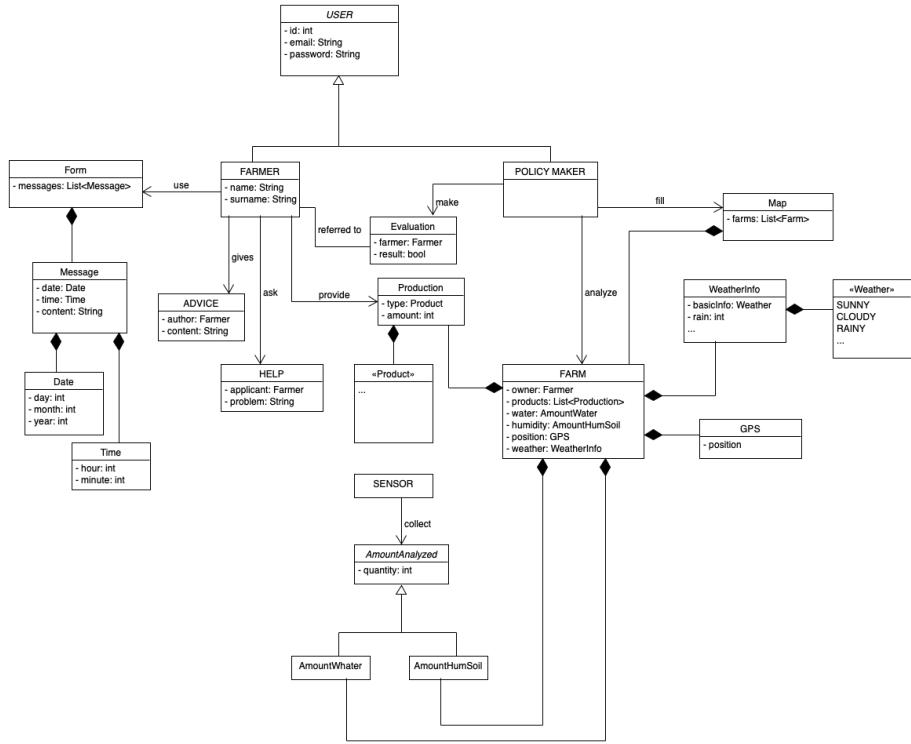


Figure 1: UML diagram.

2.1.2 State Diagrams

The following state diagrams describe the behavior of the main objects of the system's domain previously described(Figure 1). They consider all the potential states that the element examined can have while a certain event takes place.

1. **Update of the Farmer Page** The state diagram represented in figure 2 represents the events that might occur whenever the farmer page is updated. The three main circumstances when the page is updated are:

- Farmer inserts new production data
- Every day the weather forecast is updated with the new data available on it
- Data collected from sensors are always up to date

The page is so update after one of this events occurs.

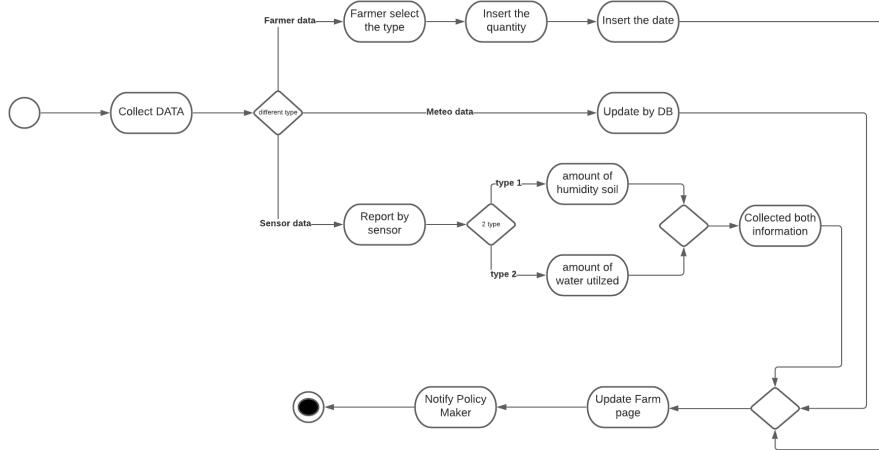


Figure 2: Update Farmer page.

2. **Analysis of different farmers** The state diagram in figure 3 represents the analysis that is performed by the policy makers once a month. As a matter of fact the policy maker starts the analysis after selecting the farm, he visualizes then the farmer's farm page and analyzes the information that is present in it. After the analysis the map is updated and the system sends a notification, that is then different if the farmer's performance is either good or bad.

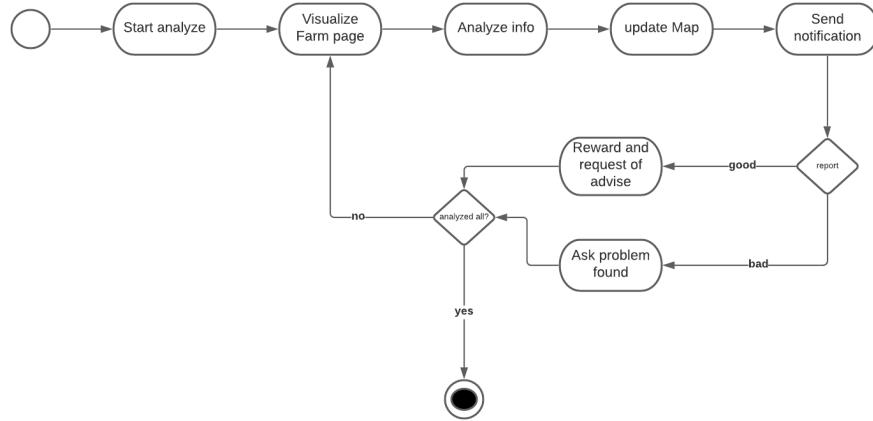


Figure 3: Analysis of farmers.

3. Request of help from a farmer

4. The diagram represented in Figure 4 shows the events that takes place when a farmer sends a request of help. It can happen by writing a message in the forum or by sending a request through a form in the homepage.

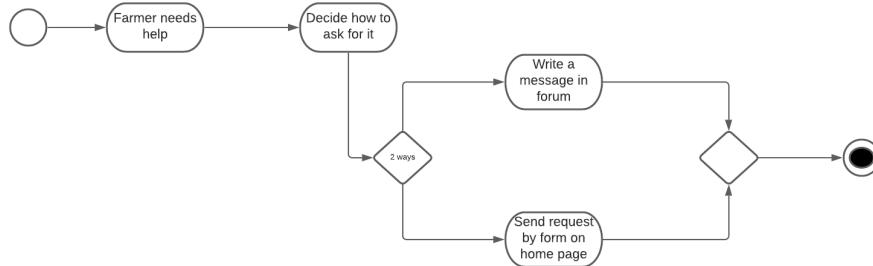


Figure 4: Request of help.

2.2 Product Functions

This section provides a summary of the main features and functions offered by the software regarding the goals already described in section 1.1.1

In the following description it is important to highlight that both the policy makers and the farmers must be logged in.

2.2.1 Farmers insert data

This functionality is accessible to all farmers. The application provides a form in which the farmer can easily insert data of his/her production. The form is easy to fill in, in order to complete it the farmer need to indicate:

- the **type of product**
- the **amount** produced of the selected type
- the **date** relative to the date of the production

If the farmer needs to add more than one type of product, he/she can fill in the form multiple times. After completing the form the user is redirected to the homepage and the policy makers can see the updated data. This functionality can be done more than once a day since the farmer can select the date, so it is possible for him to insert data of past days too. This operation can be repeated more than once a day since the farmer can select the date, so it is possible for him/her to insert data of past days too.

2.2.2 Farmers visualize data

This functionality lets the farmer visualize all the data acquired from the system. The farmer can visualize all data on his homepage.

The application shows:

- meteorological short-term and long-term forecast
- amount of water used by the farmer
- humidity of soil
- personalized suggestions concerning specific crops to plant or specific fertilizers to use – based on their location and type of production

Should I say why he needs to visualize this data or how he uses it

This functionality is always up to date, and does not need any input from the farmer. It is used by the farmer only to have a general view of his farm and on how he could improve the productivity of his farm.

2.2.3 Identify how farmers are performing

The main features of the policy makers is to evaluate the work of each Farmer. In order to do that, they periodically analyze each farm page: the system allows them to visualize all the data in those pages (but not to modify it). The analysis takes place twice a month. With this analysis they classify the workers in two different way:

- GOOD farmer : those who have been able to produce a significant amount of product with low resources, despite bad weather in that period.
- BAD farmer : those who did not produce much.

Policy makers inform each farmer the result they have achieved with a notification:

- GOOD farmers receive a special incentive, and also a request to submit from their personal web page some advice that could be useful to the others.
- To BAD farmers is asked to submit an explicit request of help specifying the problems they had.

The system has a specific web page that allows all the users to look at a map of the area in which are specified all the farms. It is also shown if the farm's owner has performed a good job in that period. At the end of each analysis the policy makers update the map (they are the only ones that are able to modify it).

2.2.4 Interaction between farmers

This functionality permits the farmers to communicate with each other. The application has a specific web page where the farmers can send messages whenever they want. If a farmer has an issue, before submitting a formal request of help to the policy makers by their home page, he can ask informally an advice by sending a message in the Forum. It is not necessary to be a good farmer in order to answer someone else's message. All the messages are visible to everyone and 24h. Since it is an online application an internet connection is required to read or write on this page.

2.3 User Characteristics

Dream has two different customers that need to be distinguished in order to provide the various features specified in subsection 2.2.

2.3.1 Farmer

- Can register on Dream in order to be recognized as farmer
- Must log in on the website to use the services offered

- Can discuss with other farmers
- Is able to insert data about their daily production
- Can ask for help to other farmers or to policy makers
- Can retrieve data regarding weather forecast, water irrigation system or humidity of soil
- Can look for advice of several products
- Can check whether their performance is identified as good or bad

2.3.2 Policy Maker

- Already has the credential to access to the system
- Must login to Dream to benefit of its services
- Can look all farms' pages
- Can update the map
- Can send suggestions to whom explicitly notices a problem
- Can send notifications to the farmers
- Decides the value of the incentive for the good farmers
- Evaluates the performance of the farmers

2.4 Domain Assumptions, Dependencies and Constraints

This subsection focuses on what it is assumed in order for our system to offer the services as expected. Moreover it focuses on the limitations that the system could face.

2.4.1 Domain Assumptions

DA1 In order to access the system users need to have Internet connection.

DA2 Farmers always insert correct data on their production activity.

DA3 Data from sensors is always correct.

DA4 Date and Time on the system are always correct.

DA5 The position of the Farm is always correct.

DA6 Internet connection works always without errors.

DA7 Meteorological data is accurate.

DA8 Every farm has a different position.

DA9 Each farm belongs to exactly one farmer.

DA10 Discussion on the forum are related only to the farm activity.

DA11 Formal request of help must be related to a farmer's own production.

DA12 Advice on a product must be given by a farmer that produces the same

type.

DA13 Performances of farmers are always identified correctly.

DA14 Farmers can insert data more than once a day.

DA15 The email and farm's name must be unique.

DA16 Policy Makers have a given code and password to access the system.

DA17 All farmers that sign up own a real farm in Telengana.

2.4.2 Constraints

3 Specific Requirements

3.0.1 External Interface Requirements

This section of the document gives a clear description of all the requirements that the system needs in order to perform all the functionalities described in section 2.2.

3.0.2 User Interfaces

The following mockups of the web application provide a first idea on how the features of the system are offered to the users. Moreover it gives a real perspective of how the users interact with the system. The Design Document (reference) provides a thorough and detailed description of the features represented in each mockup.

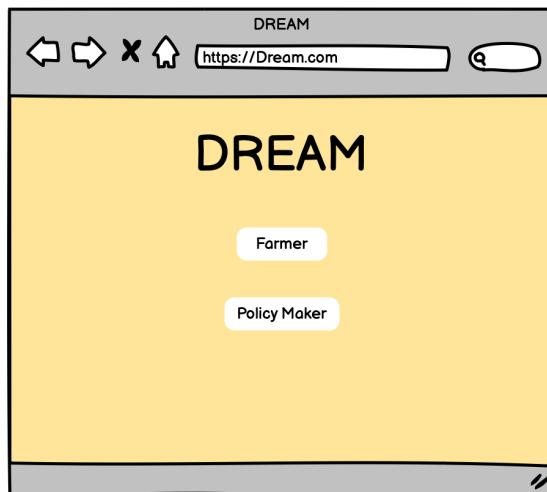


Figure 5: *DREAM* Web page

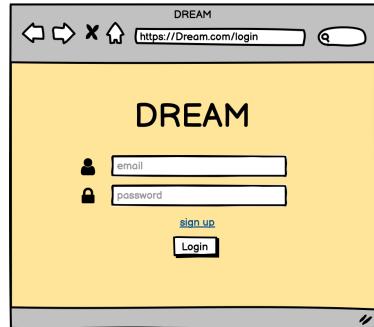


Figure 6: *Farmer Log in* Web page



Figure 7: *Farmer Sign up* Web page

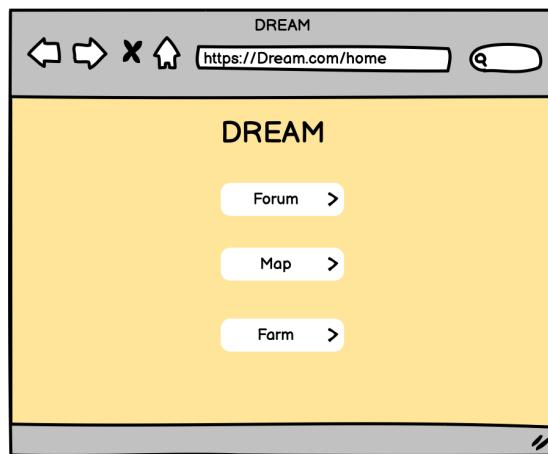


Figure 8: *Farmer Home* page

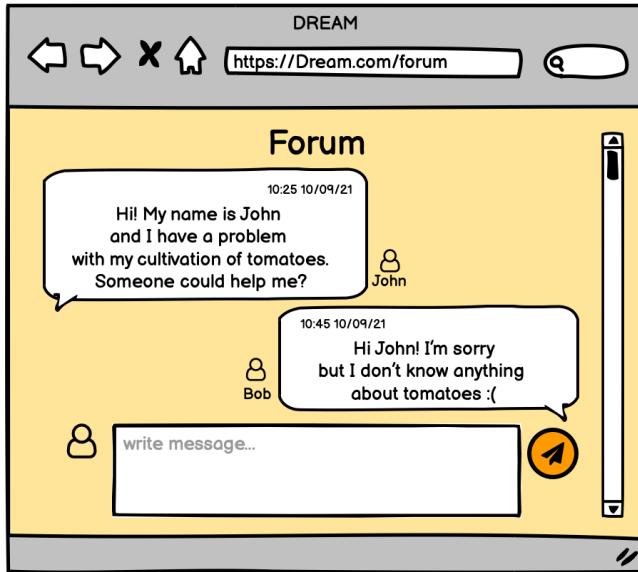


Figure 9: *Forum* Web page

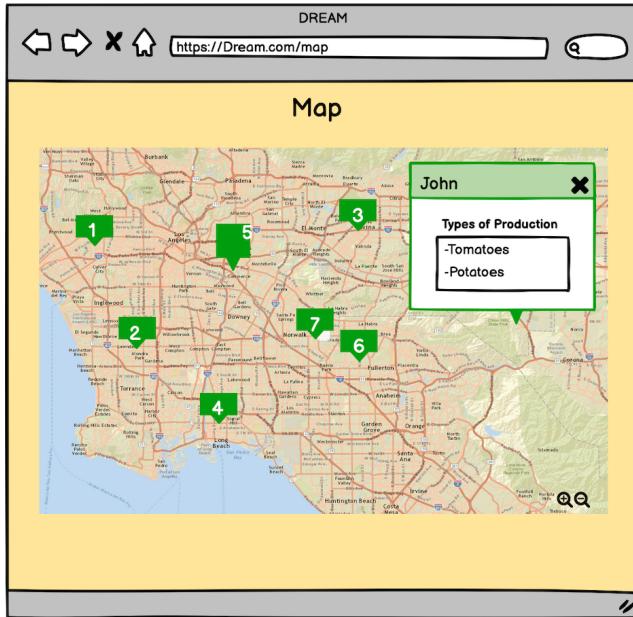


Figure 10: *Farmer* Map page

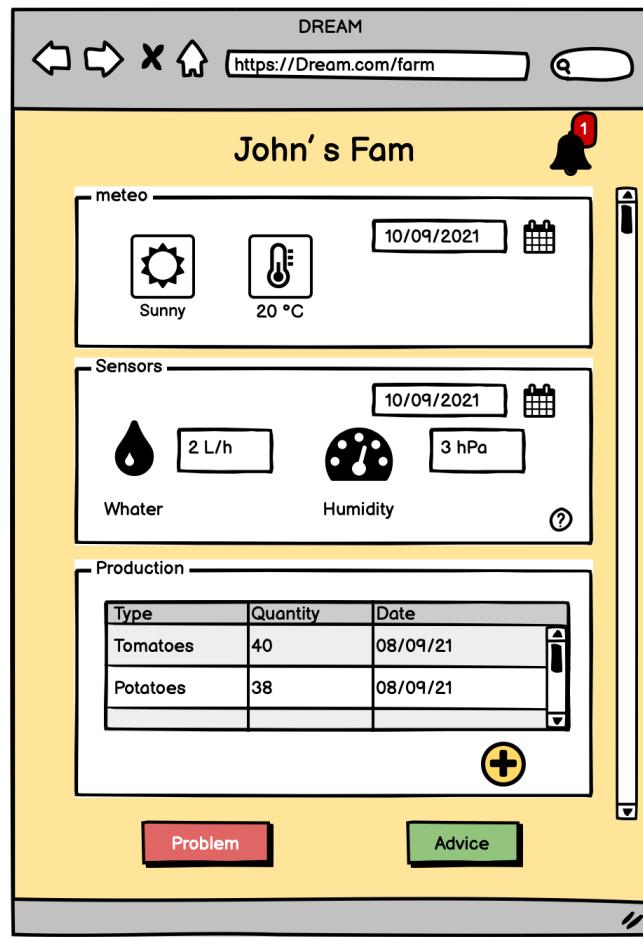


Figure 11: *Farmer's Farm* page

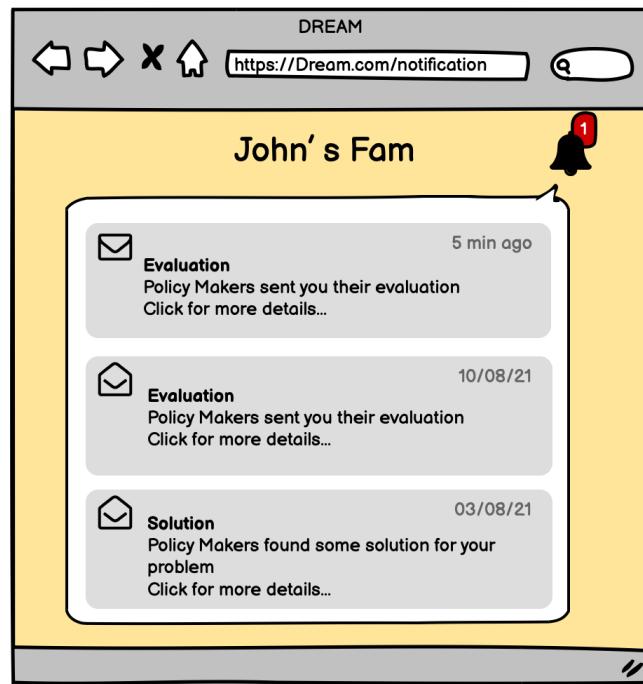


Figure 12: *Farmer's notifications list*

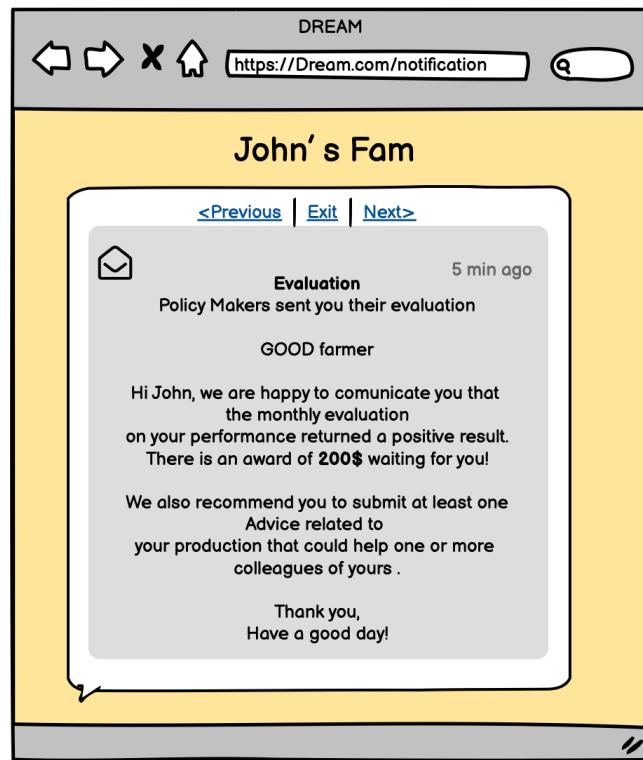


Figure 13: *Farmer's single notification*

DREAM

https://Dream.com/farm

John's Fam

Production

Type	Quantity	Date
Tomatoes	40	08/09/21
Potatoes	38	08/09/21

Type of production

Tomatoes
Potatoes
Else...

Quantity

Date

Add

The screenshot shows a web application interface for managing farm production. At the top, there's a header with navigation icons (back, forward, search) and a URL bar showing 'https://Dream.com/farm'. Below the header, the title 'John's Fam' is displayed. A section titled 'Production' contains a table with two rows: one for Tomatoes (40 units, 08/09/21) and one for Potatoes (38 units, 08/09/21). Below this, a dropdown menu labeled 'Type of production' lists 'Tomatoes', 'Potatoes', and 'Else...'. There are also input fields for 'Quantity' (set to 0) and 'Date' (showing '/ /'). A large orange 'Add' button is positioned at the bottom of the form.

Figure 14: Farmer add data Web page

DREAM

https://Dream.com/problem

Request of help

Type of production

Tomatoes
Potatoes
Else...

Problem

write here the message...

Send

A screenshot of a web browser window titled "DREAM". The address bar shows the URL "https://Dream.com/problem". The main content area has a yellow background and features a heading "Request of help". On the left, there is a label "Type of production" followed by a dropdown menu containing the options "Tomatoes", "Potatoes", and "Else...". To the right of the dropdown is a text input field with the placeholder text "write here the message...". At the bottom of the form is a red rectangular button with the word "Send" in white.

Figure 15: *Farmer send request of help* Web page

DREAM

https://Dream.com/advice

Advice

Type of production

Tomatoes
Potatoes
Else...

Advice

write here the message...

Send

A screenshot of a web browser window titled "DREAM". The address bar shows the URL "https://Dream.com/advice". The main content area has a yellow background and features a heading "Advice". On the left, there is a label "Type of production" followed by a dropdown menu containing the options "Tomatoes", "Potatoes", and "Else...". To the right of the dropdown is a text input field with the placeholder text "write here the message...". At the bottom of the form is a red rectangular button with the word "Send" in white.

Figure 16: *Farmer submit advice* Web page

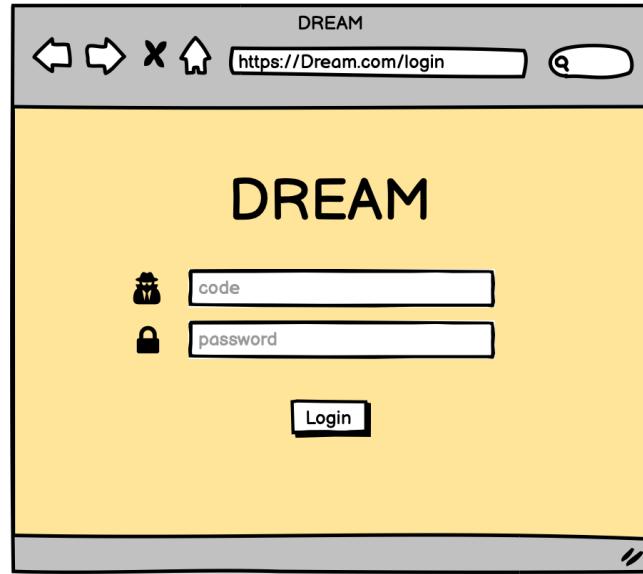


Figure 17: *Policy Maker Log in* Web page

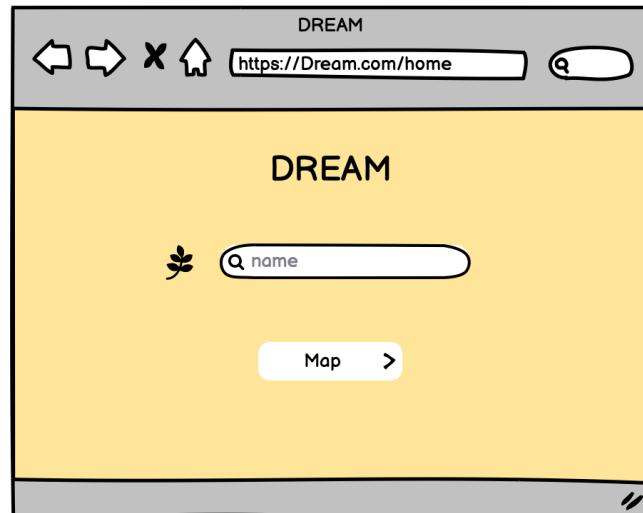


Figure 18: *Policy Maker Home* page

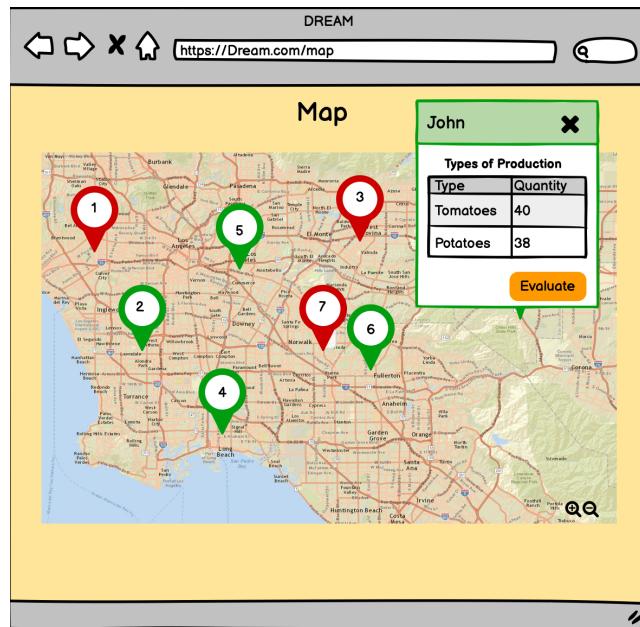


Figure 19: *Policy Maker Map* page

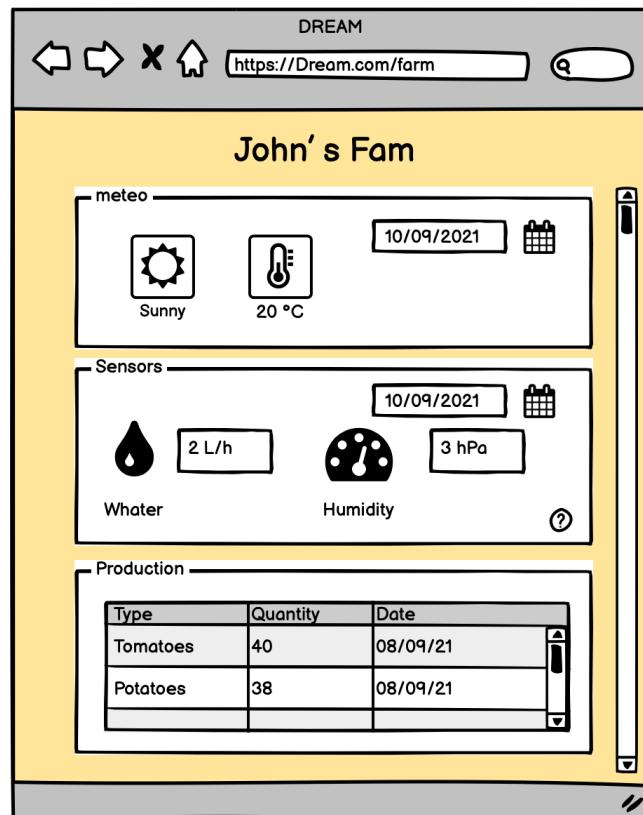


Figure 20: *Policy Maker* Farm's page visualization

DREAM

https://Dream.com/evaluation

Evaluation

Form name

Result

John's
Bob's
Else...

Good
Bad

Body

write here the message...

Send

This screenshot shows the 'Evaluation' page of the Policy Maker application. It features two dropdown menus: one for 'Form name' containing 'John's', 'Bob's', and 'Else...', and another for 'Result' containing 'Good' and 'Bad'. Below these is a text area labeled 'Body' with the placeholder 'write here the message...'. A red 'Send' button is located at the bottom of the page.

Figure 21: *Policy Maker* evaluation page

DREAM

https://Dream.com/solution

Solution of a problem

Form name

John's
Bob's
Else...

Body

write here the message...

Send

This screenshot shows the 'Solution of a problem' page of the Policy Maker application. It has a single dropdown menu for 'Form name' containing 'John's', 'Bob's', and 'Else...'. Below it is a text area labeled 'Body' with the placeholder 'write here the message...'. A red 'Send' button is at the bottom of the page.

Figure 22: *Policy Maker* send solution page

3.0.3 Hardware Interfaces

The application does not need any specific hardware requirements.

3.0.4 Software Interfaces

The web app requires a computer with a web browser installed and connected to The system has to rely on a DBMS API. It allows the management of all the data the system needs in order to provide its functionalities, described in subsection 2.2.

3.0.5 Communication Interfaces

All the communications between users and Dream website are made via HTTPS.

3.1 Functional Requirements

3.1.1 Use Case Diagram

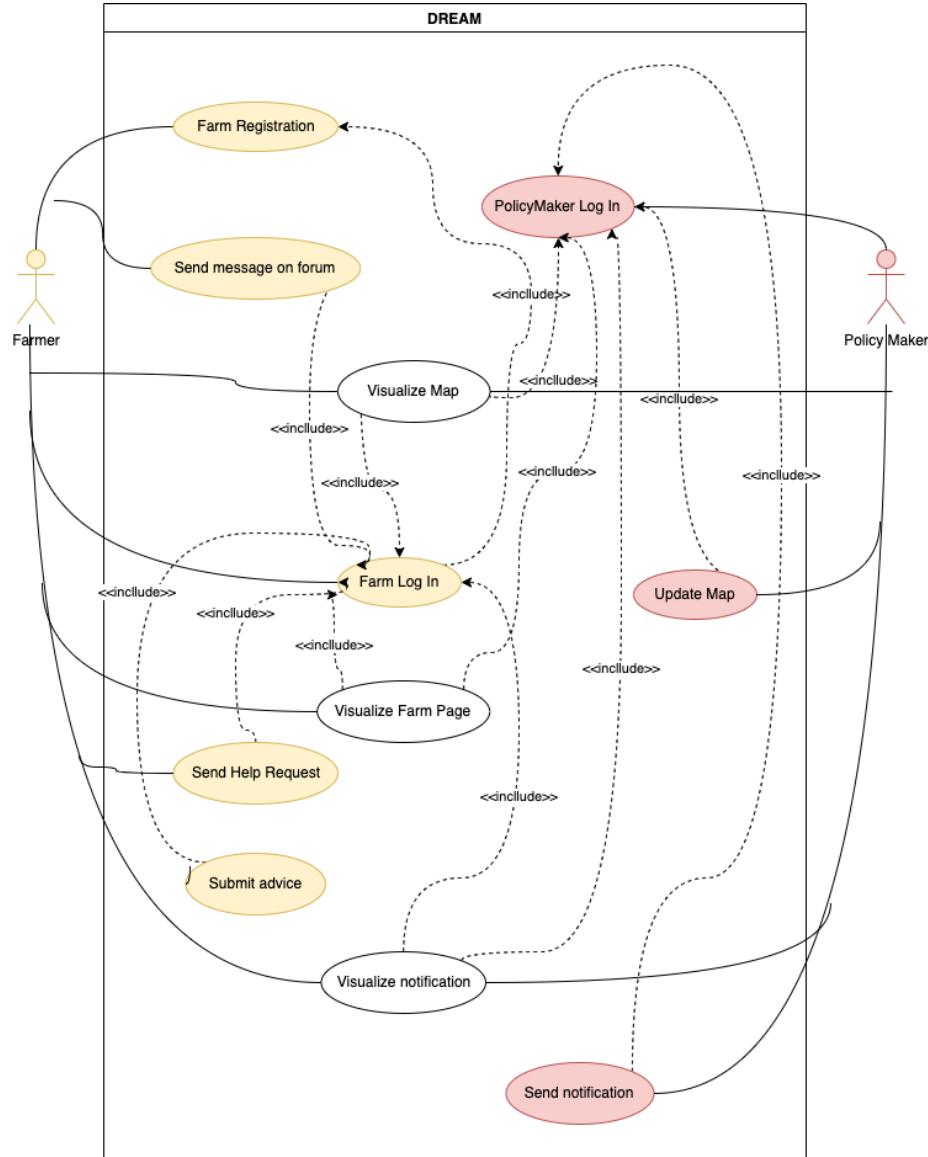


Figure 23: Use Case diagram.

3.1.2 Use Cases Description ad Sequence Diagram

1. Farmer Registration

Name	Farmer Registration
Actors	Farmer
Entry conditions	The web application has started
Flow of events	
	<ul style="list-style-type: none"> (a) The farmer wants to sign up (b) The farmer inserts email, name, password, farm's name and farm's position (c) The farmer clicks submit (d) The system checks if email is unique and if all the form is correctly fill up (e) The system inserts the information in the data base
Exit conditions	The farmer is signed up
Exceptions	
	<ul style="list-style-type: none"> • If the farmer did not insert data correctly the system will send an alert and let the user do that again • If the email is already present in the database the system will send an alert saying that the email already exists

Table 1: *Farmer Registration* use case description

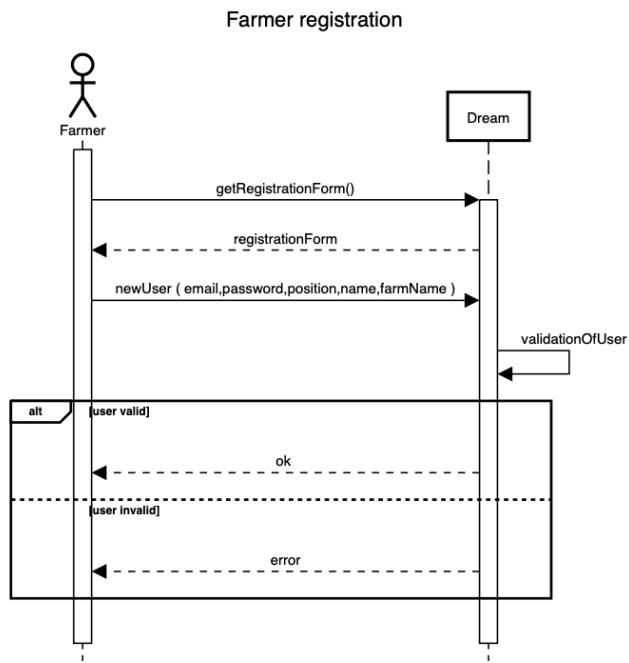


Figure 24: *Farmer Registration* sequence diagram

2. Farmer Login

Name		Farmer Login		
Actors	Farmer			
Entry conditions	The web application has started			
Flow of events				
<ul style="list-style-type: none"> (a) The farmer wants to log in (b) The farmer inserts email and password (c) The farmer clicks submit (d) The system checks if the credentials are correct (e) The system notifies the farmer about the correct login 				
Exit conditions	The farmer has logged in			

Exceptions

- If the system does not recognize the email it will send and alert to the farmer saying that the email inserted is wrong
- If the password is not correct the system will notify the farmer

Table 2: *Farmer Login* use case description

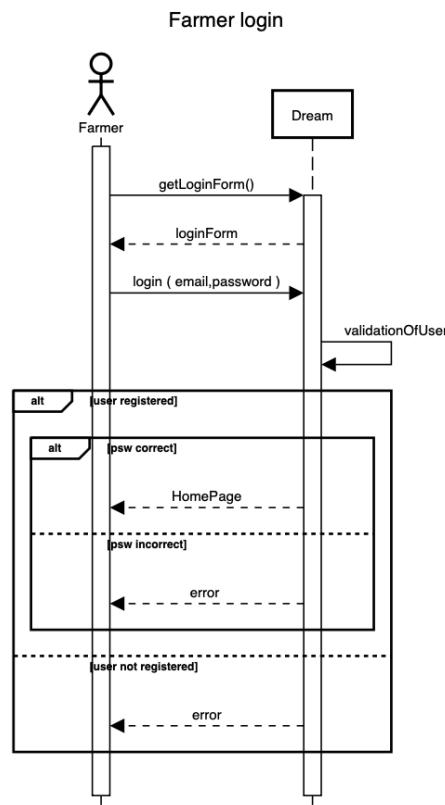


Figure 25: *Farmer Login* sequence diagram

3. Farmer send a message on the Forum

Name	Farmer sends a message on the forum
------	-------------------------------------

Actors	Farmer
Entry conditions	The farmer has logged in
Flow of events	<p>4. (a) The farmer wants to send a message (b) The farmer clicks on forum button (c) The system send the ser to the forum page (d) The farmer inserts the message (e) The farmer clicks on send message (f) The system inserts the message into the database</p>
Exit conditions	The farmer's message is published
Exceptions	<ul style="list-style-type: none"> If the message body is empty the system shows an error alert

Table 3: *Farmer message* use case description

Farmer send message on Forum

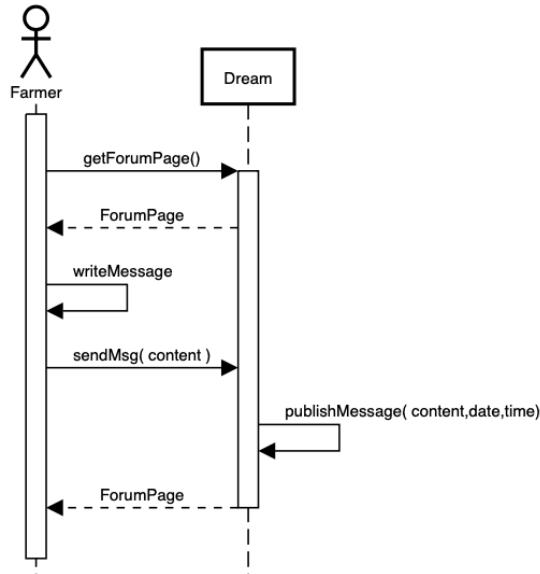


Figure 26: *Farmer message* sequence diagram

5. Find a farmer on the Map

Name	Farmer visualizes the map
Actors	Farmer
Entry conditions	The farmer has logged in
Flow of events	
	<ul style="list-style-type: none"> (a) The farmer wants to visualize the map (b) The farmer clicks on map button (c) The system send the farmer to the map page (d) The farmer visualizes the map and selects a farm (e) The system retrieves the information about the farm and shows them to the farmer (f) The farmer visualizes the data about the selected farm
Exit conditions	The farmer visualized the map
Exceptions	

Table 4: *Farms' map visualization* use case description

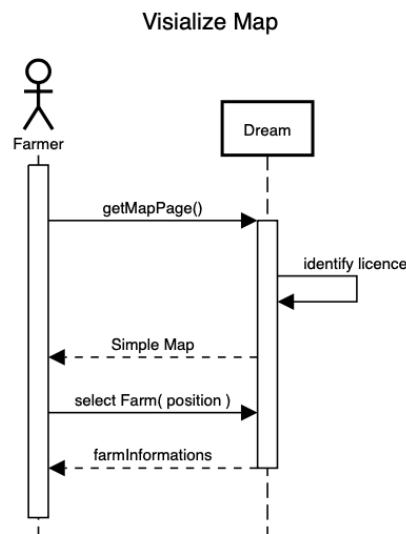


Figure 27: *Farms' map visualization* sequence diagram

6. Find farm's information

Name	Farmer visualizes their own data
Actors	Farmer
Entry conditions	The farmer has logged in
Flow of events	
	<ul style="list-style-type: none"> (a) The farmer wants to visualize his own page (b) The farmer clicks on the profile page button (c) The system retrieves the information about the farmer and redirect him to the farmer's farm page (d) The farmer visualizes his own page
Exit conditions	The farmer visualizes his own data
Exceptions	<ul style="list-style-type: none"> (a)

Table 5: *Farm information visualization* use case description

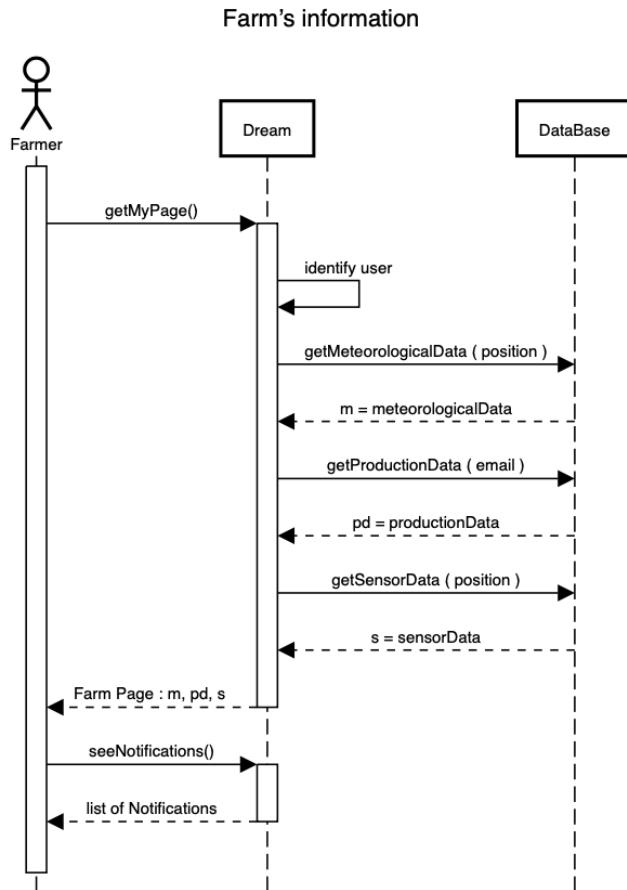


Figure 28: *Farm information visualization* sequence diagram

7. Submit a request of help

I put only the case of a formal request to the Policy Maker (button on Farm Page) if you want we can create 2 "secion" (alt) one with this formal request and one sending a message on Forum (but is yet specified in 3)

Name	Farmer submit a request of help
Actors	Farmer
Entry conditions	The farmer has logged in

Flow of events	<ul style="list-style-type: none"> (a) The farmer wants to ask for help (b) The farmer clicks on his own page button (c) The farmer clicks on Help button (d) The farmer chooses the type of production on which he had problems (e) The farmer fills the form specifying the problem (f) The farmer clicks on the submit button and sends the message (g) The system sends the notification to the Policy Makers (h) The system notifies the farmer that the operation went successfully (i) The system returns a list of advices retrieved from the database about the type of production selected
Exit conditions	The help request is sent to the Policy Makers
Exceptions	<ul style="list-style-type: none"> • If no type of production is selected the system notifies the farmer and waits for him to insert it • If the request body is empty the system notifies the farmer and waits for him to fill it in

Table 6: *Farmer ask for help* use case description

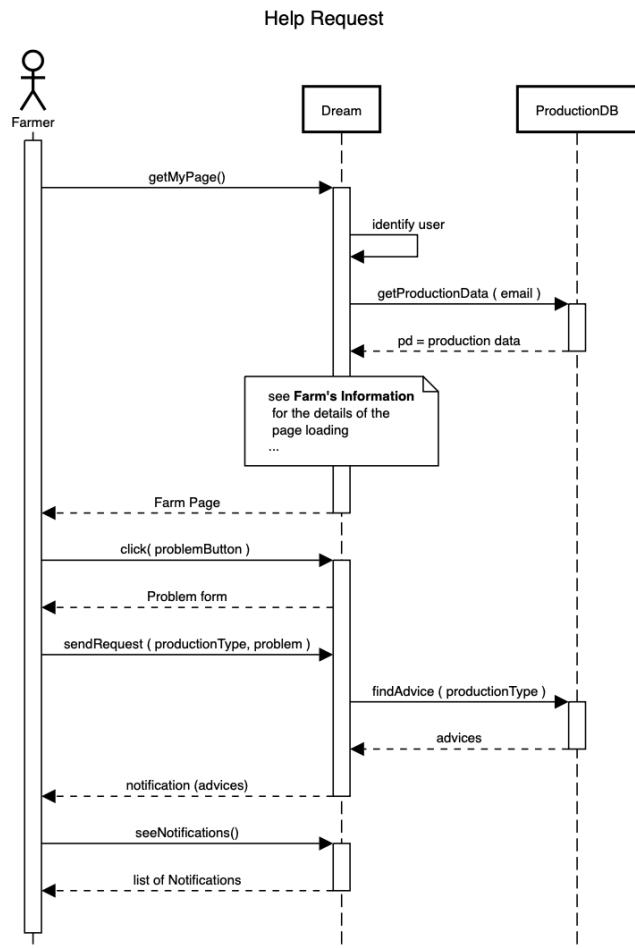


Figure 29: *Farmer ask for help* sequence diagram

8. Submit an advice

Name	Farmer submits an advice
Actors	Farmer
Entry conditions	The farmer has logged in

Flow of events	<ul style="list-style-type: none"> (a) The farmer has to submit an advice (b) The farmer clicks on his own page button (c) The farmer clicks on Advice button (d) The farmer chooses the type of production on which he wants to give advices (e) The farmer fills the form writing his advice (f) The farmer clicks submit button and sends the message (g) The system saves the advice in the database (h) The system notifies the farmer that the operation went successfully
Exit conditions	The farmer submit an advice
Exceptions	<ul style="list-style-type: none"> • If the farmer is not a "good" one the system doesn't save the advice and notifies the farmer with an error message

Table 7: *Farmer send advice* use case description

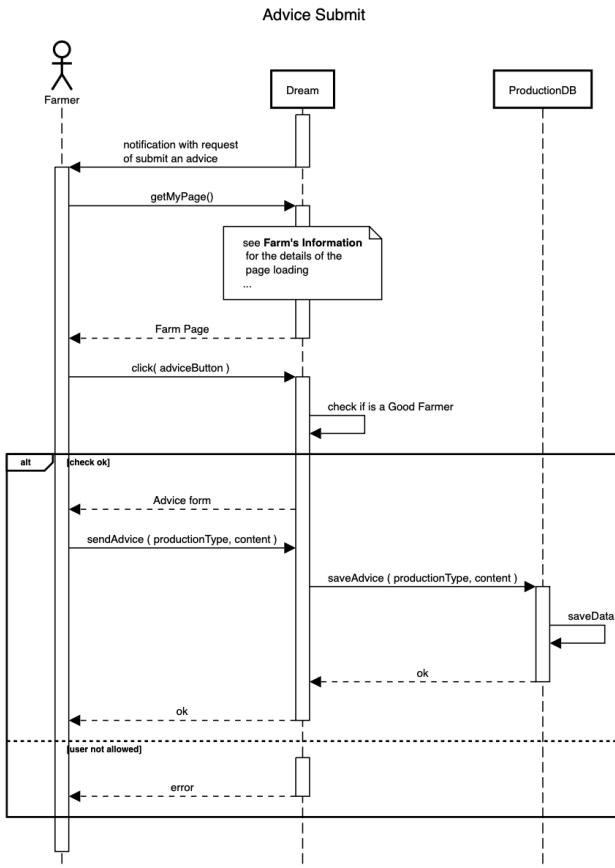


Figure 30: *Farmer send advice* sequence diagram

9. Visualize notifications

Name	Farmer visualizes his notifications
Actors	Farmer
Entry conditions	The farmer has logged in

Flow of events	<ul style="list-style-type: none"> (a) The farmer wants to read his notifications (b) The farmer clicks on his own page button (c) The farmer clicks on Notification button (d) The system returns a list of messages (e) The farmer selects one message between the ones in the list (f) The system returns the body of the message
Exit conditions	The farmer visualizes his notifications
Exceptions	<ul style="list-style-type: none"> •

Table 8: *Farmer visualize notifications* use case description

See Notifications

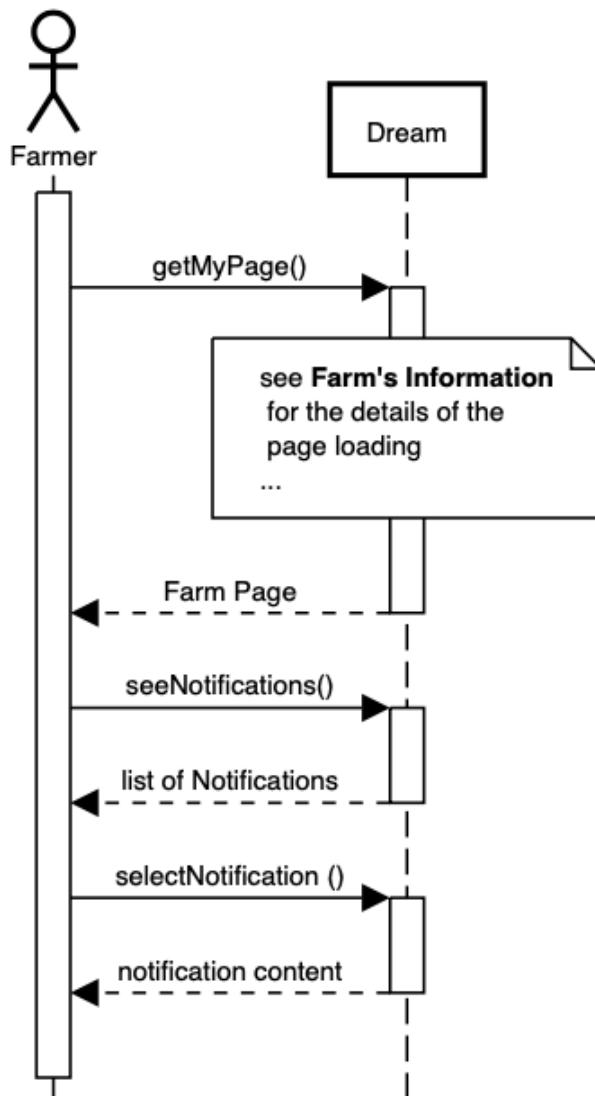


Figure 31: *Farmer visualize notifications* sequence diagram

10. Policy Maker login

Name	Farmer visualizes their own data
------	----------------------------------

Actors	Policy Maker
Entry conditions	The web application has started
Flow of events	<p>(a) The Policy Maker wants to log in (b) The Policy Maker inserts code and password (c) The Policy Maker clicks the submit button (d) The system checks if the credentials are correct (e) The system notifies the Policy Maker about the correct login</p>
Exit conditions	The Policy Maker has logged in
Exceptions	<ul style="list-style-type: none"> • If the system does not recognize the code it will send an alert to the Policy Maker saying that the code inserted is wrong • If the password is not correct the system will notify the Policy Maker

Table 9: *Policy Maker login* use case description

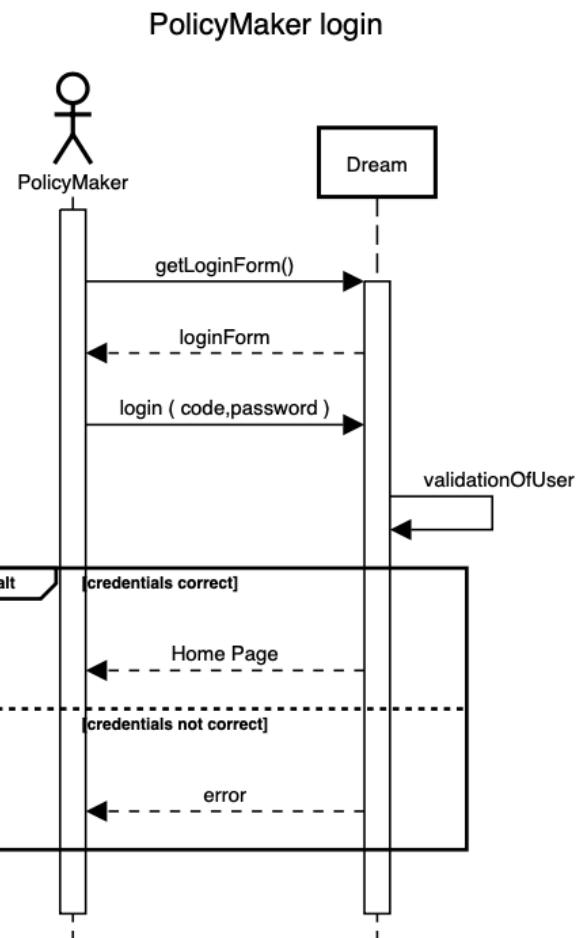


Figure 32: *Policy Maker login* sequence diagram

11. Find a farm's page

Name	Policy Maker visualizes a farm's page
Actors	Policy Maker
Entry conditions	The Policy Maker has logged in

Flow of events	<ul style="list-style-type: none"> (a) The Policy Maker wants to visualize a farm's page (b) The Policy Maker types the name of the farm on the search form (c) The Policy Maker clicks search button (d) The system redirects the Policy Maker to the farm's page
Exit conditions	The Policy Maker visualizes the farm's page
Exceptions	<ul style="list-style-type: none"> • If no name is inserted in the search form the system notifies the Policy Maker about the missing data • If the name inserted is not present in the system's database, the system sends an error message

Table 10: *Farm's page visualization* use case description

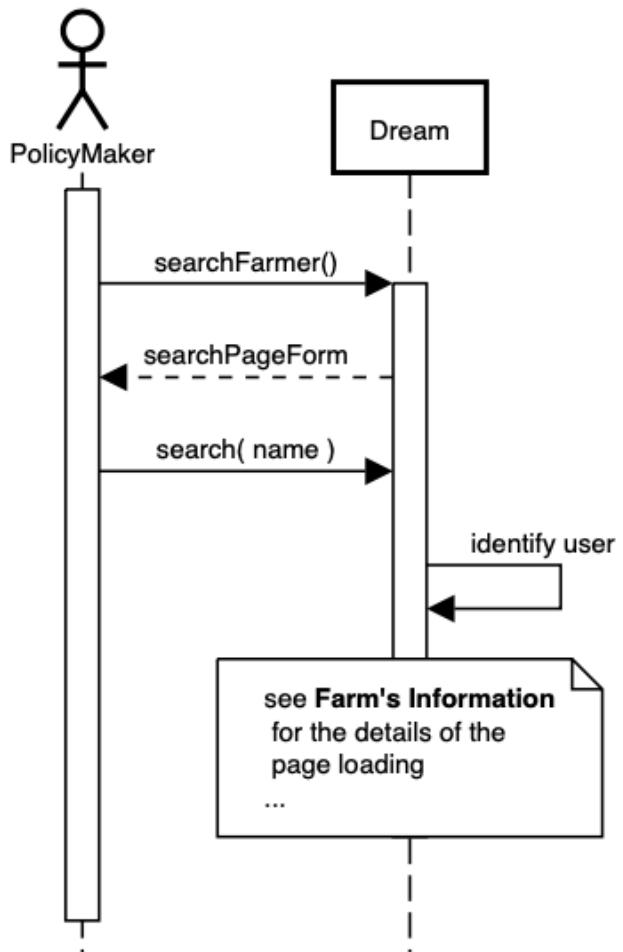


Figure 33: *Farm's page visualization* sequence diagram

12. Find a farmer on the Map

Name	Policy Maker visualizes a farm's page
Actors	Policy Maker
Entry conditions	The Policy Maker has logged in

Flow of events	<ul style="list-style-type: none"> (a) The Policy Maker wants to visualize the map (b) The Policy Maker clicks on map button (c) The system redirects the Policy Maker to the map page (d) The Policy Maker visualizes the map and selects a farm (e) The system retrieves the information about the farm (also the last evaluation) and shows them to the Policy Maker (f) The Policy Maker visualizes the data about the selected farm
Exit conditions	The Policy Maker visualizes his own data
Exceptions	•

Table 11: *Farm visualization on map* use case description

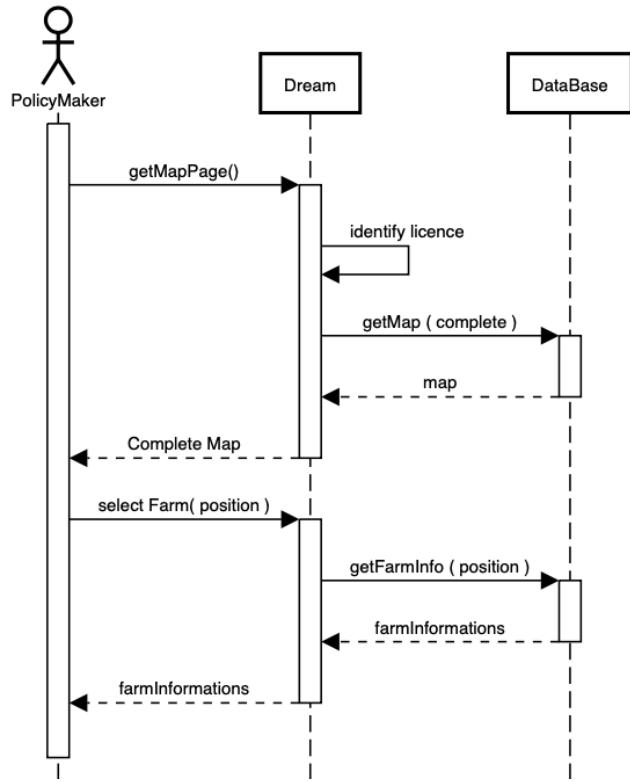


Figure 34: *Farm visualization on map* sequence diagram

13. Update the Map

Name	Policy Maker update the map
Actors	Policy Maker
Entry conditions	The Policy Maker has logged in

Flow of events	<ul style="list-style-type: none"> (a) The Policy Maker wants to report an evaluation (b) The Policy Maker clicks on the map button (c) The system redirects the Policy Maker to the map page (d) The Policy Maker visualizes the map and selects a farm (e) The system retrieves the information about the farm (also the last evaluation) and shows them to the Policy Maker (f) The Policy Maker clicks update button (g) The Policy Maker select the evaluation <ul style="list-style-type: none"> • The Policy Maker selects "good", decides an amount of money for the award and writes the body of the message • The Policy Maker selects "bad" and writes the body of the message (h) The Policy Maker clicks submit button and sends the message (i) The system saves the evaluation on the map (j) The system sends the message to the Farm's owner (k) The system notifies the Policy Maker that the operation went successfully
Exit conditions	The Policy Maker visualizes the map updated
Exceptions	<ul style="list-style-type: none"> •

Table 12: *Map updating* use case description

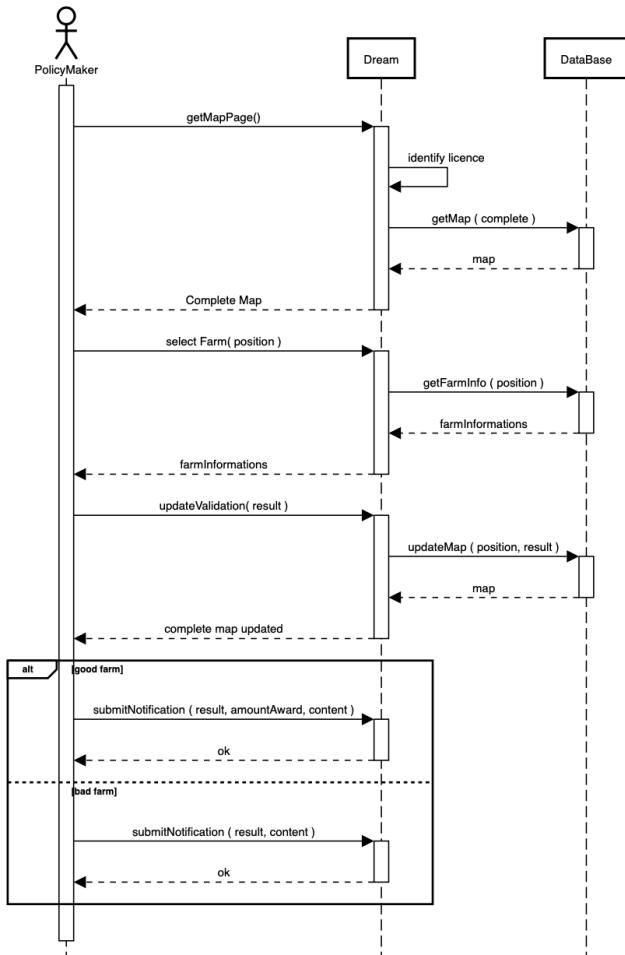


Figure 35: *Map updating* sequence diagram

14. Reply to a request of help

Name	Policy Maker send anvice
Actors	Policy Maker
Entry conditions	The Policy Maker has logged in

Flow of events	<ul style="list-style-type: none"> (a) The Policy Maker wants to send specific advices to a farmer (b) The Policy Maker wants to see the information about a farm (c) The Policy Maker types the name of the farm on search form (d) The Policy Maker clicks search button (e) The Policy Maker wants to see the advices stored in the database on the type of production the farmers asked for help on (f) The Policy Maker analyzes the data (g) The Policy Maker selects the farmer (name) and the type of production and writes the solution he found (h) The Policy Maker clicks send button (i) The system sends the message to the farmer (j) The system notifies the Policy Maker that the operation was successful
Exit conditions	The Policy Maker send solutions
Exceptions	•

Table 13: *Problem resolution* use case description

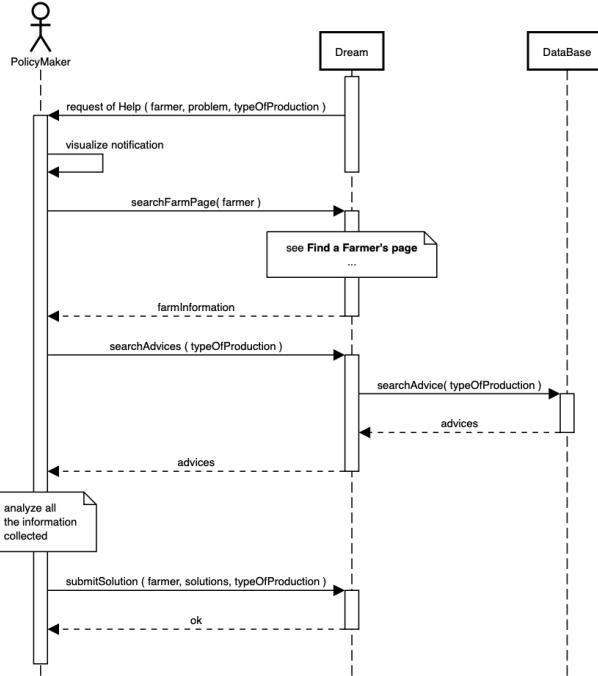


Figure 36: *Problem resolution* sequence diagram

3.1.3 Scenarios

1. **Scenario 1** Dayanita is a young woman that lives in Singaram, a small town in the state of Telengana in India. She lives with her parents in their family's farm.
- 2.

3.1.4 Requirements

- R1** the system must allow farmers to register
- R2** the system must allow farmers to log in
- R3** the system must save the farmers registration data
- R4** the system must guarantee that each email address is unique
- R5** the system must verify that the email address is valid (the type!)
- R6** the system must save the farmers information about their production submitted
- R7** the system must allow farmers to insert the type of production
- R8** the system must allow farmers to insert the amount of production type
- R9** the system must allow farmers to specify a problem they faced to the Policy Makers

- R10** the system must allow farmers to select the type of production on which they had troubles
- R11** the system must save the advice submitted by the farmers
- R12** the system must allow farmers to select the type of product in their suggestion
- R13** the system must be able to show to the farmers advices send by the Policy Makers (as a notification)
- R14** the system must be able to show the meteorological data of the Farm's position
- R15** the system must be able to show the farm's sensor data
- R16** the system must allow farmers to send messages on the forum
- R17** the system must register date and time of a message in the forum
- R18** the system must be able to show all the messages on the forum
- R19** the system must be able to show the map of the zone
- R20** the system must be able to show the farms position on the map
- R21** the system must be able to show on the map if a farm is performing well or not
- R22** the system must allow farmers to visualise notification send by Policy Makers
- R23** the system does not allow Policy Makers to register
- R24** the system must allow Policy Makers to log in
- R25** the system must allow Policy Makers to search a farm by name (OK?)
- R26** the system must allow Policy Makers to see all farms' pages
- R27** the system must not allow Policy Makers to modify any farm's page
- R28** the system must allow Policy Makers to update the performance of a farmer
- R29** the system must allow Policy Makers to send notification to the farmers
- R30** the system must allow Policy Makers to receive request of help by the farmers

3.1.5 Goals

Table 14: default

Goals	Domain Assumptions	Requirements
G1	1.08	-0.49
G2	-1.13	-0.42
G3	0.02	-0.54
G4	-0.54	0.97
G5	-0.29	-0.53

- **G1** Allow Policy Makers to retrieve information about a farm and to evaluate their performance

- **R14** The system must be able to show the meteorological data of the Farm's position
- **R15** The system must be able to show the farm's sensor data
- **R23** The system does not allow Policy Makers to register
- **R24** The system must allow Policy Makers to log in
 - The system must verify that the code is valid
- **R25** The system must allow Policy Makers to search a farm by name
- **R26** The system must allow Policy Makers to see all farms' pages
- **R27** The system must not allow Policy Makers to modify any farm's page
- **R28** The system must allow Policy Makers to update the performance of a farmer
- **DA1** In order to access the system users need to have Internet connection
- **DA2** Farmers always insert correct data on their production activity
- **DA3** Data from sensors is always correct
- **DA4** Date and Time on the system are always correct
- **DA5** The position of the Farm is always correct
- **DA6** Internet connection works always without errors
- **DA7** Meteorological data is accurate
- **DA8** Every farm has a different position
- **DA9** Each farm belongs to exactly one farmer
- **DA13** Performances of farmers are always identified correctly
- **DA16** Policy Makers have a given code and password to access the system
- **DA13** Performances of farmers are always identified correctly
- **G2** Allow farmer to communicate with each other
 - **R1** The system must allow farmers to register
 - **R2** The system must allow farmers to log in
 - **R3** The system must save the farmers registration data
 - **R4** The system must guarantee that each email address is unique
 - **R5** The system must verify that the email address is valid
 - **R16** The system must allow farmers to send messages on the forum
 - **R17** The system must register date and time of a message in the forum
 - **R18** The system must be able to show all messages on the forum

- **DA1** In order to access the system users need to have Internet connection
 - **DA4** Date and Time on the system are always correct
 - **DA6** Internet connection works always without errors
 - **DA10** Discussion on the forum are related only to the farm activity
 - **DA15** The email and farm's name must be unique
 - **DA17** All farmers that sign up own a real farm in Telengana
- **G3** Allow farmer to insert data and advices on his production
 - **R1** The system must allow farmers to register
 - **R2** The system must allow farmers to log in
 - **R3** The system must save the farmers registration data
 - **R4** The system must guarantee that each email address is unique
 - **R5** The system must verify that the email address is valid
 - **R6** The system must save the farmers information about their production submitted
 - **R7** The system must allow farmers to insert the type of production
 - **R8** The system must allow farmers to insert the amount of production type
 - **R11** The system must save the advice submitted by the farmers
 - **R12** The system must allow farmers to select the type of product in their suggestion
 - **DA1** In order to access the system users need to have Internet connection
 - **DA2** Farmers always insert correct data on their production activity
 - **DA6** Internet connection works always without errors
 - **DA9** Each farm belongs to exactly one farmer
 - **DA12** Advice on a product must be given by a farmer that produces the same type
 - **DA14** Farmers can insert data more than once a day
 - **DA15** The email and farm's name must be unique
 - **DA17** All farmers that sign up own a real farm in Telengana
- **G4** Allow farmer to send request of help to Policy Makers
 - **R1** The system must allow farmers to register
 - **R2** The system must allow farmers to log in
 - **R3** The system must save the farmers registration data

- **R4** The system must guarantee that each email address is unique
- **R5** The system must verify that the email address is valid
- **R9** The system must allow farmers to specify a problem they faced to the Policy Makers
- **R10** The system must allow farmers to select the type of production on which they had troubles
- **R13** The system must be able to show to the farmers advices send by the Policy Makers
- **R22** The system must allow farmers to visualise notification send by Policy Makers
- **R29** The system must allow Policy Makers to send notification to the farmers
- **R30** The system must allow Policy Makers to receive request of help by the farmers
- **DA1** In order to access the system users need to have Internet connection
- **DA6** Internet connection works always without errors
- **DA9** Each farm belongs to exactly one farmer
- **DA11** Formal request of help must be related to a farmer's own production
- **DA15** The email and farm's name must be unique
- **DA17** All farmers that sign up own a real farm in Telengana
- **G5** Allow farmers to retrieve information relevant for their activity
 - **R1** The system must allow farmers to register
 - **R2** The system must allow farmers to log in
 - **R3** The system must save the farmers registration data
 - **R4** The system must guarantee that each email address is unique
 - **R5** The system must verify that the email address is valid
 - **R14** The system must be able to show the meteorological data of the Farm's position
 - **R15** The system must be able to show the farm's sensor data
 - The system must allow farmers to their own Farm's page
 - **R22** The system must allow farmers to visualise notification send by Policy Makers
 - **DA1** In order to access the system users need to have Internet connection
 - **DA3** Data from sensors is always correct

- **DA4** Date and Time on the system are always correct
 - **DA5** The position of the Farm is always correct
 - **DA6** Internet connection works always without errors
 - **DA7** Meteorological data is accurate
 - **DA8** Every farm has a different position
 - **DA9** Each farm belongs to exactly one farmer
 - **DA10** Discussion on the forum are related only to the farm activity
 - **DA13** Performances of farmers are always identified correctly
 - **DA15** The email and farm's name must be unique
 - **DA17** All farmers that sign up own a real farm in Telengana
- **G6** Policy Makers and farmers should be able to consult the map of the zone (and the informations stored in it) with different levels of visibility
 1. **G7A** Policy Makers and farmers should be able to see the position of the farms on the map
 - **R1** The system must allow farmers to register
 - **R2** The system must allow farmers to log in
 - **R3** The system must save the farmers registration data
 - **R4** The system must guarantee that each email address is unique
 - **R5** The system must verify that the email address is valid
 - **R19** The system must be able to show the map of the zone
 - **R20** The system must be able to show the farms position on the map
 - **R23** The system does not allow Policy Makers to register
 - **R24** The system must allow Policy Makers to log in
 - The system must verify that the code is valid
 - **DA1** In order to access the system users need to have Internet connection
 - **DA5** The position of the Farm is always correct
 - **DA6** Internet connection works always without errors
 - **DA8** Every farm has a different position
 - **DA9** Each farm belongs to exactly one farmer
 - **DA15** The email and farm's name must be unique
 - **DA16** Policy Makers have a given code and password to access the system
 - **DA17** All farmers that sign up own a real farm in Telengana
 2. **G7B** Policy Makers should be able to see both the information about the production and the evaluation of each farm
 - The system must be able to show the types of production cultivated in a farm

- The system must be able to show the quantity of product has been cultivated for each type and farm
 - **R19** The system must be able to show the map of the zone
 - **R20** The system must be able to show the farms position on the map
 - **R21** The system must be able to show on the map if a farm is performing well or not
 - **R23** The system does not allow Policy Makers to register
 - **R24** The system must allow Policy Makers to log in
 - The system must verify that the code is valid
 - **R28** The system must allow Policy Makers to update the performance of a farmer
 - **R29** The system must allow Policy Makers to send notification to the farmers
 - **DA1** In order to access the system users need to have Internet connection
 - **DA2** Farmers always insert correct data on their production activity
 - **DA5** The position of the Farm is always correct
 - **DA6** Internet connection works always without errors
 - **DA8** Every farm has a different position
 - **DA9** Each farm belongs to exactly one farmer
 - **DA13** Performances of farmers are always identified correctly
 - **DA15** The email and farm's name must be unique
 - **DA16** Policy Makers have a given code and password to access the system
 - **DA17** All farmers that sign up own a real farm in Telengana
3. **G7C** Farmers should be able to see only the type of production of a farmer by the map
- **R1** The system must allow farmers to register
 - **R2** The system must allow farmers to log in
 - **R3** The system must save the farmers registration data
 - **R4** The system must guarantee that each email address is unique
 - **R5** The system must verify that the email address is valid
 - The system must be able to show the types of production cultivated in a farm
 - **R19** The system must be able to show the map of the zone
 - **R20** The system must be able to show the farms position on the map
 - **DA1** In order to access the system users need to have Internet connection

- **DA2** Farmers always insert correct data on their production activity
- **DA5** The position of the Farm is always correct
- **DA6** Internet connection works always without errors
- **DA8** Every farm has a different position
- **DA9** Each farm belongs to exactly one farmer
- **DA15** The email and farm's name must be unique
- **DA16** Policy Makers have a given code and password to access the system
- **DA17** All farmers that sign up own a real farm in Telengana

3.1.6 Traceability Matrix

3.2 Performance Requirements

The system serves its user with a web application. All the computations will take place on the server side, thus the app is meant to be lightweighted. Moreover the load in the night is expected to be really low. There are no problems about reliability. The insertion of new data requires a quick response in order to store it in the system.

3.3 Design Constraints

3.3.1 Standards Compliance

The only standard that needs to be highlighted here is the interaction with the database. It is important that the information are stored in a standardized form. In such manner, it is easier to memorize and retrieve data.

3.3.2 Any Other Constraints

Interaction between Dream and users needs to consider also regulatory policies. As a matter of fact the application asks and retrieves data of each farmer. More information about security and privacy will be provided in the section 3.4.3

3.4 Software System Attributes

3.4.1 Reliability

The system must prevent any failure in order to guarantee continuity. Simultaneous accesses are expected to work, especially in the afternoon when users are expectd to insrt and retrieve more information.

3.4.2 Availability

It is expected that the system has the lowest downtime possible. The system is available with a minimum time of 96%, so that it about 14 days a year of downtime are allowed.

3.4.3 Security

Security of the data and of the communication user-system is a primary concern. Users credential are stored in a data base, so the system crypt the password data before store it. The system recognises the right type of user during the log in phase to ensure providing the correct level visibility of the data and the permission to update them or not. In this way farmer's privacy is guaranteed. As a matter of fact a farmer can't see another farmer's page.

3.4.4 Maintanability

The web application requires ordinary maintenance for improvements and in order to fix potential bugs. It is going to be scheduled at local night time, when user traffic is the lowest. Moreover the system must be designed in a way that allows future addition of features.

3.4.5 Portability

The system as a web application must run on different software system as Windows, Linux an macOS. On the server side is crucial focusing on the interaction between APIs and the data base to insert, update or read data.

4 Formal Analysis using Alloy

5 Effort Spent