POLITECNICO
MILANO 1863

Requirement Analysis and Specification Document

Authors: Pietro Valente
Andrea Seghetto
Professor: Damian Andrew Tamburri

Version: 2.0
Date: 2/1/2022
Repository GitHub: `https://github.com/pietrovalente/DREAM-software-engineering-2`

Contents

1 Introduction **1**
- 1.1 Purpose 1
- 1.2 Scope 2
  - 1.2.1 World Phenomena 3
  - 1.2.2 Shared Phenomena 3
  - 1.2.3 Goals 4
- 1.3 Definitions, Acronyms 4
  - 1.3.1 Definitions 4
  - 1.3.2 Acronyms 5
- 1.4 Revision history 5
- 1.5 Document Structure 6

2 Overall Description **7**
- 2.1 Product perspective 7
  - 2.1.1 Scenarios 7
  - 2.1.2 UML Description 8
  - 2.1.3 State Charts 10
- 2.2 Product functions 12
- 2.3 User characteristics 13
- 2.4 Assumptions, dependencies and constraints 14
  - 2.4.1 Domain Assumption 14
  - 2.4.2 Future Developments 14

3 Specific Requirements **15**
- 3.1 External Interface Requirements 15
  - 3.1.1 User Interface 15
  - 3.1.2 Hardware Interfaces 17
  - 3.1.3 Software Interfaces 17
  - 3.1.4 Communication Interfaces 17
- 3.2 Functional Requirements 18
  - 3.2.1 List of Requirements 18
  - 3.2.2 Mapping 19
  - 3.2.3 Use Cases 24
  - 3.2.4 Use Cases Diagram 33
  - 3.2.5 Sequence Diagrams 34
- 3.3 Performance Requirements 40
- 3.4 Design Constraints 40
  - 3.4.1 Standards compliance 40
  - 3.4.2 Hardware limitations 40
- 3.5 Software System Attributes 41
  - 3.5.1 Reliability 41
  - 3.5.2 Availability 41
  - 3.5.3 Security 41
  - 3.5.4 Maintainability 41
  - 3.5.5 Portability 41
  - 3.5.6 Scalability 41

4 Formal Analysis Using Alloy **42**
- 4.1 Alloy Code 42

4.2 Metamodels general 50
4.3 Metamodels goals 52
4.4 Result of Assertions 54
4.5 Result of Predicates 54

5 Effort Spent 55

1 Introduction

1.1 Purpose

Climate change is drastically changing our lives and is having a disastrous effect on many human activities, including agriculture.

The latter arises both as a cause and as an effect of climate change, in fact, if on the one hand it often involves highly polluting carbon emissions, on the other hand it is affected by those emissions as the increase in global temperature they cause is related to the sharp decline of agricultural productivity that is occurring in many parts of the planet. In India the primary sector is a strategic sector in which the government invests approximately 47 billion USD per year and which supports the 55 %[6] of the rural population; it is therefore an indispensable economic activity threatened by the effect of climate change to the point that precipitations and extreme temperatures are projected to cause a 4 % - 17 % decline in crop yield as well as 4 % - 26 % loss in net farm income towards the end of the century.

What has been said above highlights the importance of acting promptly to make agricultural activity less polluting and at the same time make decisions to try to safeguard productivity from the disastrous effects of climate change. A further particularly significant problem regarding agriculture lies in the availability and use of some resources, including the amount of water used for irrigation. In India it is estimated that 50 % of area under cultivation is not served by the hydric system and where available, water is used in a non-optimal way since the country uses about 2-3 times the amount of water spent by major agricultural countries to produce the same amount of 1 tonne of food. It is therefore evident the need to make decisions also to optimize the use of these resources as well as their fair distribution among farmers.

Finally, in the context of the technological project that will be described below a further problem arises, namely that of digital education that is particularly relevant in rural India and which is determined by the lack of knowledge by many citizens of the notions necessary to use technological devices and services.

We will deal with the design and implementation of DREAM, which is a data-driven predictive and also distributed system designed for governmental use and realized through the collaboration between the government of Telengana and some IT providers which must interface with multiple stakeholders. The main goal of the whole system is to monitor and optimize the farming production of the Indian state Telengana on which the food needs of a large part of the population depend, making it resilient to the many difficulties that threaten it, including phenomena related to climate change.

This document is a *Requirements Analysis and Specification Document (RASD)* containing the description of some scenarios, their related use cases, the models that describe the requirements of the system and its specification for the problem considered. In particular this document addresses the description of the main goals of the system, of its domain and of its representation through some models. An appropriate mix of natural language, UML and Alloy has been used for this purpose.

The document also includes the research conducted on the interfaces, on functional and non-functional requirements and also on the attributes that distinguish the quality of the system.

Finally, to understand better the development of the document; we have included its history which reports how the document is made, which references are used and the description of its structure.

At the end of the document you can find an account of the hours of work necessary for the realization of this RASD.

The reading of this document is intended for all those who are interested in understanding the operation of the system, whether for use or for its development (including future improvements).

1.2 Scope

As a distributed system we foresee the existence of a central server that include the front-end part, that is a user interface for webapp and application, with a simple design, to cope with the poor digital education of farmers and also back-end part that is a Database Management System (DBMS) which will take care of the storage and management of data.

The system will use some external tools/APIs (i.e.TSDPS, Google Maps) to make available to its users some important features. One of this features lies in showing predictions about short or long-term climatic events in order to anticipate and govern them in the best possible way.

The resources used by each farm are a very important issue, but for this first development it was decided only to monitor them and in case to report with a notification the excessive consumption of water by a farm, if it exceeds a certain threshold.

We will consider three types of stakeholders:

- **Policy makers:** the system must guarantee them the ability to monitor the evolution of the entire national production process, distinguishing the farmers who have been more successful from those who needs help and allowing them to analyze whether the advice given by agronomists leads to significant results. Policy makers have the opportunity to see the ranking of the year with the efficiency scores of the farms in real time, they reward the best ones only once a year, after which the ranking is reset.
- **Agronomists:** the system must assure them to write reports on the farms they visit containing the advice they have given with the help provided by DREAM AI; namely an AI algorithm of the system which proposes possible general suggestions on the production of each farm that the agronomist can choose to use or not. Only agronomists are allowed to give advice because they have the skills to do so, DREAM AI is just a tool used to help agronomists improving their advices which works by analysing the evolution of the farm's production overtime and comparing it to the data related to other farms. The agronomists selected by the government are trained to give, in particular, effective advice against climate change. Each agronomist is associated with an area, which corresponds to a province of Telengana.
- **Farmers:** the system must allow them to view relevant information such as weather predictions but also to take advantage of the personalized suggestions provided by agronomists after each visit to the farm. Furthermore, it will be possible for each farmer to interface with others by asking them for help or by creating a new discussion on the related forum. The advice given in this area remains informal and its effectiveness is not monitored.

For each of these figures, a different account is provided, based on their needs.

It was decided to entrust the updating of data to farmers to make them more involved in the system.

To reward the best farmers, an annual ranking is drawn up in real time until the deadline. Each farm has its own score which is made up of:

$$score = \frac{\sum_{i=1}^{n} (\frac{c_i}{s_i} * F_i)}{n} * Z$$

where c_i is the collected product ($gr.$), s_i is the size of the land where the product was grown (m^2), n is the total number of lands in the farm, F_i is a coefficient related to the type of food, Z is a coefficient related to the zone of the farm.

This score metric could benefit small farms, which have more time and effort for each field. This is not a problem because large farms do not need government incentives as much as small ones, having more

products on which make profits. Furthermore, the aim of the project is to show the best management of any land that a farmer has, regardless of the number. The 10 farms with the best score will be awarded, in case of a tie between them, all farms will be rewarded.

1.2.1 World Phenomena

WP1	Weather conditions
WP2	Farm location
WP3	Resources available in the farm warehouse
WP4	Number of people working in the farm
WP5	Total size of arable land
WP6	Water amount in irrigation system measured by a counter
WP7	State of the soil
WP8	Time required for crop to grow
WP9	Total collected product
WP10	Number of lands owned

1.2.2 Shared Phenomena

SP1	View meteorological short-term and long-term forecasts	World controlled
SP2	Receive information about the farm (position, size, labor, resource, owner)	World controlled
SP3	Receive information about farmers production	World controlled
SP4	Receive information about sensors deployed on the farmers territory and measuring the humidity of soil	World controlled
SP5	Receive information by the agronomists who periodically visit the farms	World controlled
SP6	Send a notification to farmer and his agronomist if the water amount exceed the threshold	Machine controlled
SP7	Send a notification to agronomists when their intervention is required	Machine controlled
SP8	Send a notification to policy makers when the annual ranking is finished	Machine controlled
SP9	Propose through the use of DREAM AI some personalized advises that can help agronomist in their job	Machine controlled

1.2.3 Goals

G1	Allow policy makers to identify the farmers to be rewarded and those performing bad
G2	Allow farmers to ask the help they need
G3	Allow policy makers to evaluate if the advise given by agronomists works
G4	Being able to anticipate climatic phenomena
G5	Allow farmers to visualize relevant data based on their location and type of production
G6	Allow farmers to keep track of their production data
G7	Allow farmers to create discussion forums with the other farmers
G8	Encourage the best farmers to share their experience on the forum
G9	Allows agronomists to send production advice to farmers and write reports on their visits
G10	The system must distinguish users in 3 categories: farmers, policy makers and agronomists
G11	The system must give advice to agronomists (DREAM AI)

1.3 Definitions, Acronyms

1.3.1 Definitions

Validation code	These codes are used to identify users for who they really are. Each user will receive an email invitation to register on the platform, with a personal code for registration. Validation verifies the correctness of the code.
Validation data	Each time data is entered by the user, a validation process verifies that the characters do not exceed the maximum number and that there are no harmful strings (i.e. SQL injection).
Query	Request made to the system in order to find information.
Webapp	Application software that runs on a web server.
Zone	A province of Telengana to which an agronomist is associated.

1.3.2 Acronyms

RASD	Requirements Analysis and Specification Document
DREAM	Data-dRiven PrEdictive FArMing in Telengana
DREAM AI	Data-dRiven PrEdictive FArMing in Telengana Artificial intelligence
TSDPS	Telangana State Development Planning Society
API	Application Programming Interface

1.4 Revision history

For the history of the creation and modification of the document refer to the link: `https://github.com/pietrovalente/ValenteSeghetto/commits/main`

Date	Modifications
5/12	First Version without Alloy chapter
8/12	• Changes in UML description • Changes in Product Functions • Update in Domain Assumption • Update in List of Requirements • Update Mapping
15/12	• Changes in Purpose • Update in Scope • Add tables to see better the mapping

22/12	• Adding Alloy part • Changing in UML chart • Final small changes in UML description, state chart, software system attributes
2/1	• Effort part updated for the last hours of work (forgotten) • Ranking update changed: not every time the score changes but every minute (more efficient). Changes in R22 and in Performance Requirements.

1.5 Document Structure

- **Chapter 1** gives an introduction about the purpose of the document and the development of the application, with its corresponding specifications such as the definitions, acronyms, abbreviation, revision history of the document and the references. Besides, are specified the main goals, world and shared phenomena of the software.
- **Chapter 2** contains the overall description of the project. In the product perspective are included some scenarios useful to identify specific cases in which the application can be utilised, the statecharts of the major function of the application and also the model description of the system through a Class diagram. In user characteristic are explained the types of actors that can use the application. Moreover, the product function clarified the functionalities of the application. Finally, are included the domain assumption that can be deducted from the assignment.
- **Chapter 3** presents the interface requirement including: user, hardware, software and communication interfaces. This section contains the core of the document, the specification of functional and non-functional requirements. Functional requirements are submitted with a list of use cases with their corresponding sequence diagrams. Non-functional requirements included: performance, design and the software systems attributes.
- **Chapter 4** includes the alloy code and the corresponding metamodels generated from it, with a brief introduction about the main purpose of the alloy model.
- **Chapter 5** shows the effort spent for each member of the group.

Finally, a section dedicated to all the references used.

2 Overall Description

2.1 Product perspective

2.1.1 Scenarios

1.Farmer registration
Alisha is a farmer who owns a farm in the province of Nizamabad, she decided to take part in the government project DREAM. Alisha decides to download and install the app related to the project on her smartphone; then, launch it for the first time and select sign in on the start screen to be able to register in the system. Alisha then fills in the personal data requested and also some other entries related to the characteristics of her farm. Finally, Alisha confirms the data entered and agrees by ticking the appropriate item to the processing of her data by other users registered in the system.

2.Water consumption
Ravi is a farmer who owns a farm outside Warangal, is a member of DREAM and uses it to optimize the productivity of his tea plantations. Shaila is the agronomist registered in the system responsible for the area where Ravi 's farm and all the neighboring farms are located. Irrigation of Ravi 's plantations is involving the use of an excessive amount of water exceeding the permitted threshold, DREAM promptly detects the problem and consequently both Ravi and Shaila receive a notification on their smartphone which shows to both of them the same content, ie "excessive water consumption detected". Finally, Shaila meets Ravi and give him advice to reduce consumption.

3.Identification of the best farmer
Bipin is an Indian politician from Hyderabad and the current government's minister of agriculture. He joined the system because he believes in the initiative and because he believes that DREAM can help him make more effective decisions regarding agriculture and economic development. Bipin met with his colleagues to celebrate New Year's Eve together, just after midnight the system forwards a notification to Bipin containing the following content "Annual ranking available". Once the celebrations are over, the next day Bipin accesses the platform from his office desktop and after inspecting the ranking, he contacts the ten farmers with the best result to congratulate them and proceed to their awarding.

4.Agronomist helping farmer in difficulty
Gulab is an Indian farmer, his potato crop is not going as he hoped and needs help. He then decides to connect to webapp, logs in, press the help button and compiles the brief form necessary to forward the request in the system. Pavan is the agronomist who takes care of the Gulab area and receives a notification regarding the help request. Pavan connects from the app and displays the farm data, after which he calls the farm to arrange a visit. After visiting, he writes a report into the system on the potato field and advises Gulab to change fertilizer.

5.Farmers help each other on the forum
Shail is a farmer who has a quick doubt: when to plant tomatoes? So he decides to connect to DREAM and ask for it on the forum. Surya is another farmer who is watching the forum that morning, reads Shail's question and decides to answer him. He comments on the discussion by explaining that tomatoes are transplanted between March and May, while postponing to June is fine anyway even if you risk shortening the harvesting period. Shail reads the answer and comments again thanking him.

2.1.2 UML Description

The UML[3] below describes at high-level the model of the system to be developed but does not include every class that will be necessary to define the complete architecture of the system. Here we can identify the main aspects related to DREAM:

- Each farmer, in the registration phase, enters the data relating to himself and his farm, also specifying the description of the fields he has and the resources of products (seeds) he has in stock. He can change these data at any time and it is assumed that those relating to production are updated daily, if the user has not updated his data by 9 pm the system automatically sends a notification urging him to log in and proceed with the update. Linked to this data is the ranking.

- The ranking is an annual ranking as it lasts one year, when it expires the top ten best farmers are awarded and the ranking is reset.

- Based on the area to which the farm belongs, this is associated with an agronomist. He has the task of monitoring farmers in his zone by making periodic visits or responding to their request for help. After each of these visits he writes a report on the situation of the farm and with the advices he has given with the help of weather forecasts site and the suggestions of DREAM AI.

- Farmers can use the forum to create a new discussion, view an old useful discussion, comment a discussion (new or old).

- Policy makers can view ranking and individual farm data, understand if an agronomist is doing a good job and reward a farmer in case of high ranking.

- The "crop" association between land and product wants to keep track of previous crops, this information is provided to us by the date and quantity, provided as attributes of the association.

- In the report, in the "description" section an analysis of the farm status is reported, while in "advise" the suggestions given to improve production are reported.

- The forum is used only by farmers, this is the reason why they have the attribute "username".

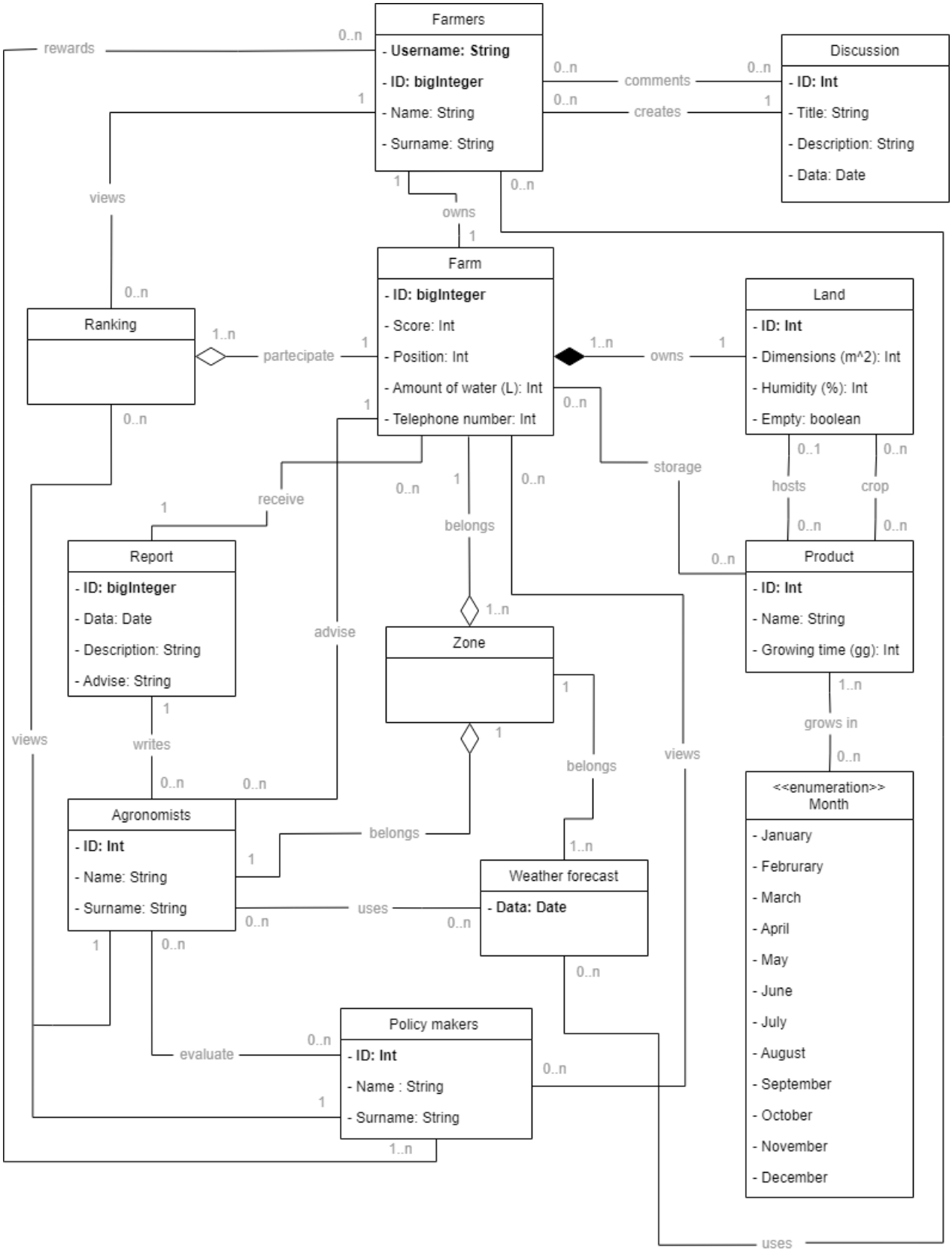


Figure 1: Class Diagram using UML

2.1.3 State Charts

This section analyzes some aspects of the system. Relative system states have been modeled over time with appropriate state diagrams as shown below.

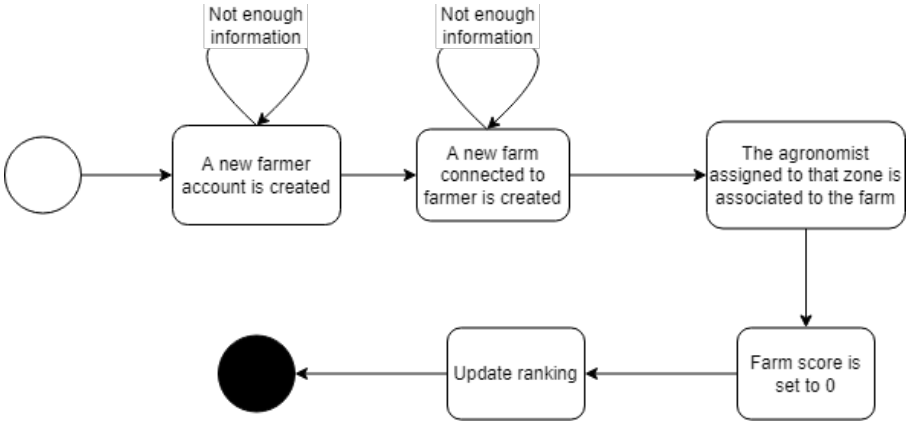


Figure 2: State diagram 1 - Farmer registration

In the first state diagram (Fig. 2) statuses are shown when registering a farmer to the system. If the data relating to the farmer or farm are not sufficient, the system does not allow you to proceed to the next stage.

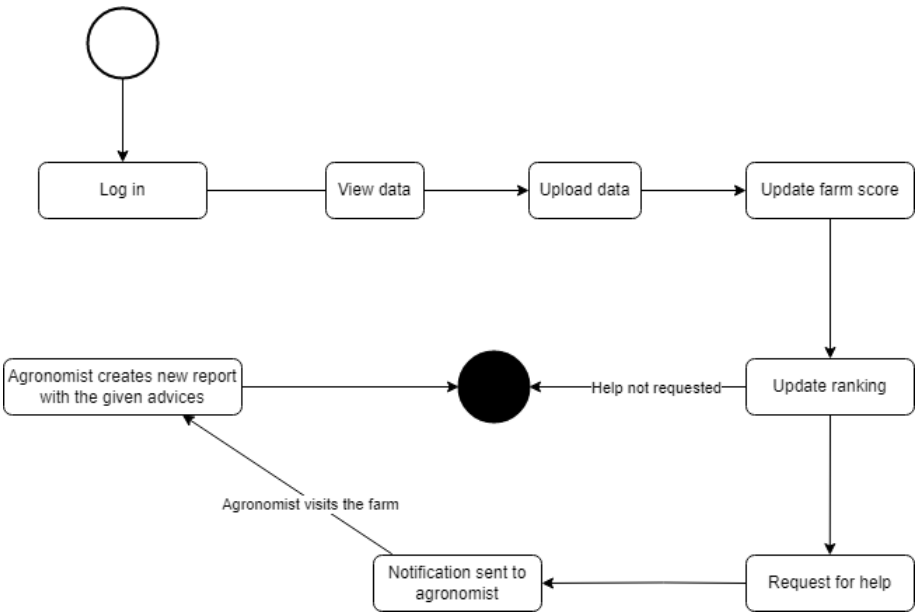


Figure 3: State diagram 2 - Farmer in difficulty

In the second state diagram (Fig. 3) system states are shown when a user, having a complete view on the productivity of his farm, decides whether to ask for help or not.

It is important that during uploading data process, the farmer must necessarily pass through view data screen, where the information about his farm, up to that moment, is present. From here he can press the "edit data" button to switch into edit mode and be able to update them.

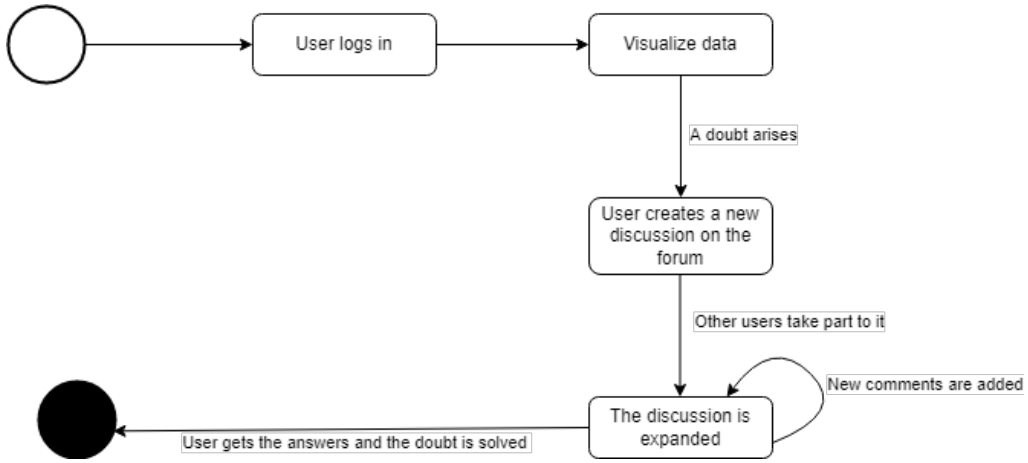


Figure 4: State diagram 3 - Discussion on the forum

In the third state diagram (Fig. 4) the statuses related to an interaction of a user who has a doubt on the forum are shown.

2.2 Product functions

Here are described some functions of the software. Be aware that some products functions are described also in other parts of this documents.

Keep track of farmers' production and identify the best among them
One of the main functionalists offered by the system lies in the possibility for agronomists and politicians to inspect the production data of each farmer. The system, starting from the production data entered and updated daily by each farmer, associates a score to the corresponding farm, from which a ranking of the productivity of all the various farms registered in the system is built. If a farmer forgets to update his production data at the end of the day, the system automatically forwards a notification to the farmer at a fixed time, i.e. 9pm as a reminder of the obligation to which he must fulfill. The ranking during the year is constantly updated by the system in line with the evolution of production data that has occurred. At the end of the year, the system allows politicians to identify and reward the most virtuous farmers, namely, the ones associated with the top ten farms of the ranking with the best production data. The score mainly considers the relationship between what has been planted and what is collected by each farm during the year. The goal of the ranking is to estimate the production efficiency of each farm, in the case of farms with multiple plantations the estimated overall efficiency is computed as the result of the average of their single efficiencies. Obviously, however, the ranking also takes into account other factors such as the location of the farm and the type of products grown, that greatly affect production.

Ask for help
A further functionality of the system lies in the possibility for each farmer to request the intervention of the agronomist associated with his area when in difficulty. Instead of waiting for the agronomist to visit periodically him, the farmer can decide to connect to the platform and ask directly for an interview to be able to solve the problems affecting his farm. The agronomist to the extent of his commitments will agree on an appointment with the farmer and will then be able to provide him with the help he needs. To complete and send the request through the system, the farmer will have to fill in a form with a brief description of the reasons for the request containing the main problems affecting his farm's production.

Facilitate the work of agronomists and report their activity
In addition to notifying agronomists about new urgent interventions required, the system also facilitates their work because through the intelligent algorithm it is equipped with (DREAM AI), it provides them with data, projections and possible personalized suggestions to improve the productivity of each farm. It will then be up to the agronomists to choose whether to follow these suggestions or opt for other solutions. In any case, the system allows to keep track of each visit made by an agronomist to a farm and of the advice given by compiling and uploading a report at the end of the visit. In this way, by keeping track of the agronomists' activity and analyzing the progress of farm production from the graphs, politicians will be able to verify the effectiveness of the various advice provided from time to time and the changes they have brought about.

Allow the sharing of ideas and experiences through discussions in the forum
The system guarantees registered farmers the opportunity to exchange views through the forum. Specifically, farmers who need help or opinions on certain topics have the possibility, after accessing, to create discussions in the forum where other farmers can comment on the topic of discussion by offering their personal vision on what is debated. On the other hand, the forum, in addition to collecting doubts and curiosities from farmers, also serves to tell their stories, especially if they are successful. In fact, the best farmers who top the annual government rankings are encouraged to share their experience in the forum so that it can also serve as an example to other registered users, the system will notify them for this purpose.

2.3 User characteristics

The actors of the system are the following:

- **Policy maker user:** these users are those in the government who are responsible for agriculture (minister of agriculture and his collaborators). They can monitor the data of each farm at any time and they also dispose of data related to the global agricultural trend in Telegana. They can also view the leaderboard with farm scores and understand which are doing well and which are in trouble. For each farm they can see a graph that represents the evolution of their productivity as well as the reports made by an agronomist every time he has given some advices visiting a farm. This also allows them to evaluate the effectiveness of agronomists' advice.
- **Agronomist user:** these users represent those who have been mandated by the government to follow a certain area. They must monitor the farmers in their area by making them periodic visits and responding to their requests for help. After each visit they are asked to write a report on the farm visited, reporting the various advice given in the system.
- **Farmer user:** these users are all farmers who have decided to participate in the DREAM project. They have downloaded the application or otherwise use accessed through the webapp keeping their farm data up-to-date. They can take advantage of all the services that the system provides (forums, help from agronomists, etc.) and they can compete for government incentives for the most productive farms, seeing their ranking score and position in real time.

2.4 Assumptions, dependencies and constraints

2.4.1 Domain Assumption

D1	Meteorological forecasts are always available
D2	The information inserted about the farm are reliable
D3	The information inserted about the farm are updated daily
D4	The sensors deployed on the farmers territory and measuring the humidity of soil work correctly
D5	The reports are uploaded regularly in the system
D6	Each farm has internet connection and a device to access the platform (smartphone or pc)
D7	Each user has the necessary skills to navigate the designed user interface and use its features
D8	Farmers use the counter to measure the amount of water used and it works correctly
D9	The agronomist is available during working days
D10	The agronomist responds promptly to requests for help
D11	The intelligent algorithm works correctly by helping agronomists
D12	The user picks up the right category

2.4.2 Future Developments

- The system could interface with another component that manages the subdivision of resources (i.e. water, seeds, fertilizers) in order to avoid waste and make them available to those who do not have it.
- The management of reservations for agronomists' visits to farmers, which in the current system takes place via telephone agreements.
- Adjust the scoring formula in case a study demonstrates possible problems.
- The system could be expanded to other regions of the world.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interface

The webapp and the app are two access points to the system, but allow access to the same data, which is the reason why they are expected to have a similar interface. This solution allows a farmer to avoid buying a smartphone and be forced to learn how to use it, as through an old computer it can perform the same functions. The interface designed is very simple to deal with the scarce digital knowledge widespread in this part of the population.

This section shows some screenshots[7] relating to a possible interface of the farmer, the one who is expected to use the system the most. The user interfaces of agronomist and policy makers are similar and have been left out.

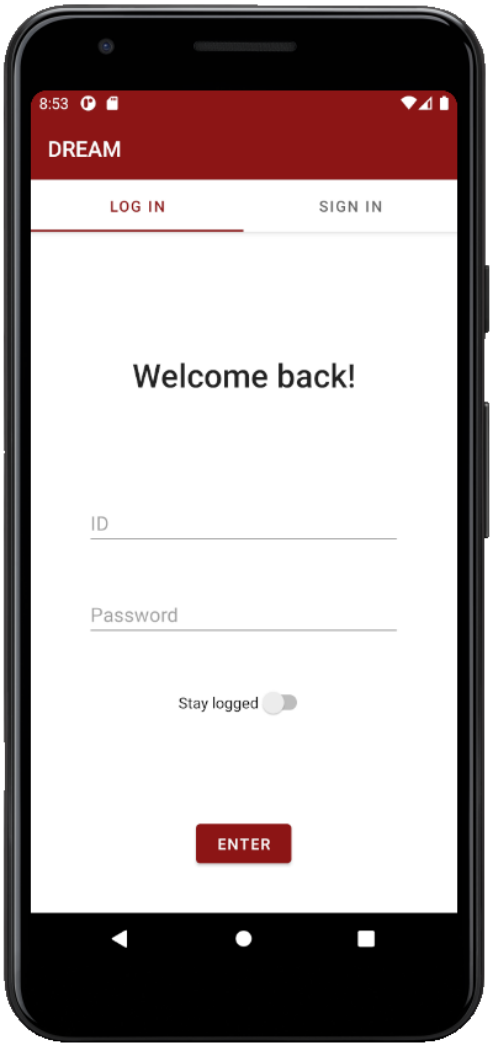


Figure 5: App - Login screen

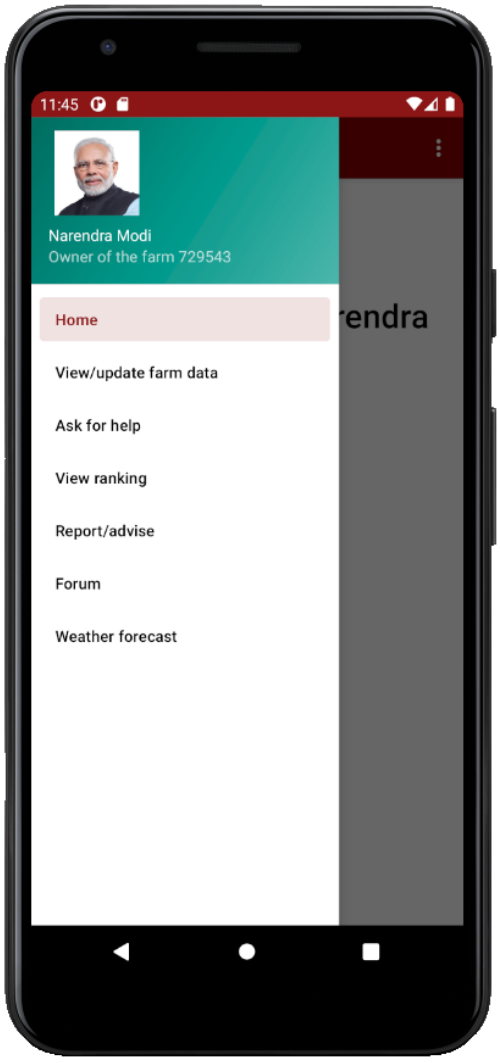


Figure 6: App - Menu

These screens are related to the app. A login screen (Fig. 5), where you can also register or stay connected and the menu (Fig. 6) with which a farmer can access the various sections.

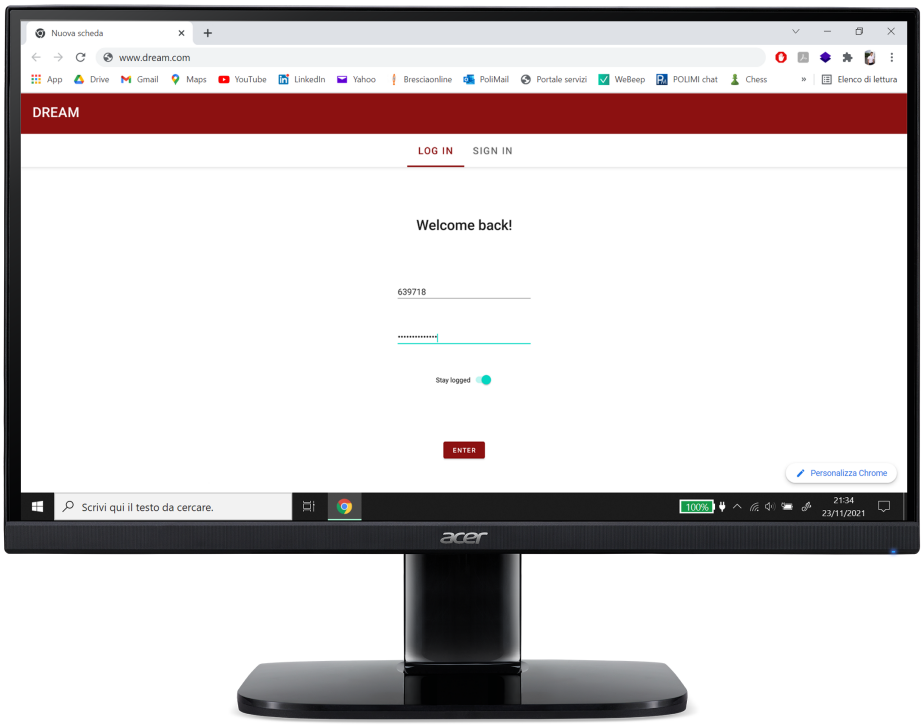


Figure 7: PC - Login screen

These screenshots shows the login part (Fig. 7) and the home screen (Fig. 8) on the PC side.

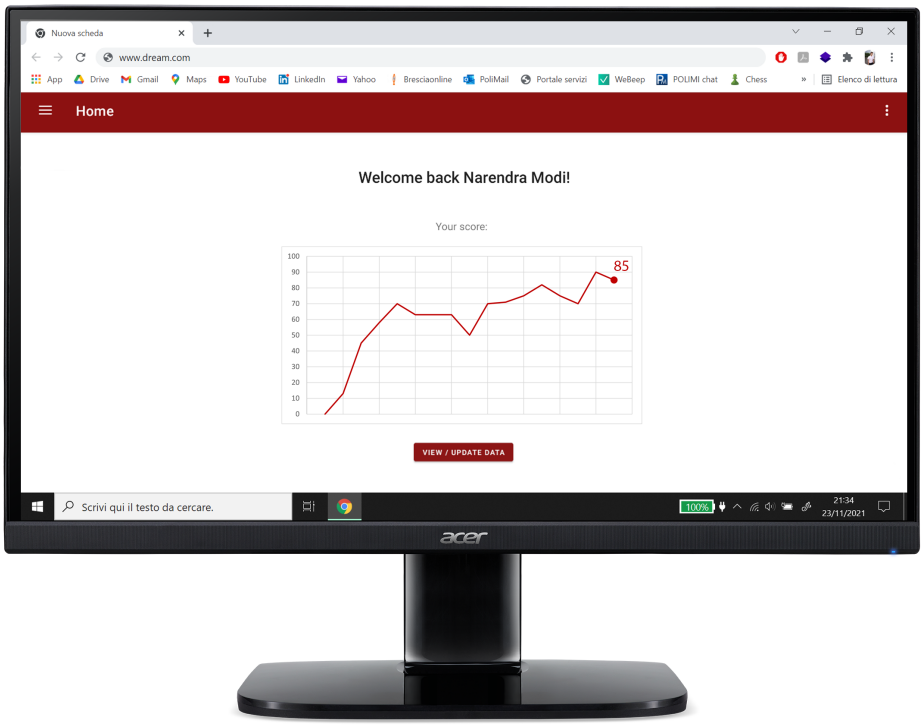


Figure 8: PC - Home screen

3.1.2 Hardware Interfaces

In order to use the functions offered by the system, its users must have:

- A pc if they want to access from the webapp, this access mode is especially designed for politicians and agronomists who want to consult the data reported on the platform directly from their offices using their desktop or for those farmers who do not have a smartphone available to download the application
- A smartphone if they want to access through the application in the store, this mode is designed instead to facilitate access to the platform in conditions of mobility in a simple and fast way

3.1.3 Software Interfaces

The system relies on an external API namely that of TSDPS[4] (Telengana State Development Planning Society) to provide information about weather. Specifically, the system provides, through the appropriate item on the screen displayed immediately after access, the possibility for each farmer to consult weather forecasts by redirecting him to the site linked below. Then, users will be able to extrapolate all the desired information from the site and use it for their advantage.

The system also uses an additional API, namely that of Google Maps,[5] it is a tool that facilitates the visualization of each area into which the state is divided and of the farms contained through their representation on a map. It is made available to agronomists to see where each farm is located in their area of competence and to politicians because it is useful to associate the areas with the corresponding agronomists.

3.1.4 Communication Interfaces

The devices connect to DREAM via Internet connection that we take for granted is an available resource. As already anticipated, the designed system is a distributed system having a client-server architecture, such that the individual devices interface with a central server and interrogate it to view some of the data stored. After viewing data, they also have the possibility of overwriting them by adding further information to the database, this is what occurs for example when the production data is updated daily by each farmer or when agronomists upload their reports.

3.2 Functional Requirements

3.2.1 List of Requirements

R1	All users are notified when the annual ranking is completed
R2	The system must allow policy makers to view the data needed to contact farmers
R3	The best farmers have to be notified asking them to share their story in the forum
R4	Policy makers must be able to access sensitive data of other users
R5	Policy makers must be able to view farm production data
R6	Agronomists must be notified if their help has been required
R7	The system must allow agronomists to view the data needed to contact farmers
R8	Agronomists must be able to access to sensitive data of other users
R9	Politicians can view the graph of each farm's score trend
R10	The system must show the reports written by agronomists with the advice given
R11	The system must show the ranking of the farms
R12	The system must show the map of the farms contained in each zone
R13	The system must show the agronomist associated to each zone
R14	The system must allow users to be redirected to TSDPS meteo
R15	DREAM AI must process suggestions based on the evolution overtime of a farm's production and on other farms results
R16	The system must store the data entered by the farmer in chronological order from the most recent to the least
R17	The system must display the data entered by the farmer in chronological order from the most recent to the least
R18	Farmers must be able to view their farms' production data
R19	Farmers must be able to view their score
R20	Farmers can update their farms' production data
R21	The system must update the score associated to a farm every time its data are updated
R22	The system must update the ranking of the farms

R23	DREAM AI has to reprocess the projections and suggestions associated to each farm when its data are updated
R24	The system must allow farmers to create a new discussion on the forum
R25	A farmer must be able to comment on a forum discussion
R26	A comment from a user must be able to tag another user
R27	The system must send a notification to a user when when his discussion is commented on
R28	The system must send a notification to a user when tagged in a comment
R29	The forum must store the discussions/comments in chronological order
R30	The forum must display the discussions/comments in chronological order
R31	The system must send a notification to the farmers (at 21.00) to remind them to update the daily data
R32	Agronomists must be able to view farm production data
R33	Agronomists must be able to write reports in the system
R34	DREAM AI must propose the advices computed to agronomists
R35	The system must show a specific different form to compile to each user, based on the category he selects, during registration
R36	The system must ask each user to fill in the form to complete the registration
R37	Each user can view only the features specific to his category

3.2.2 Mapping

Goals	Domain Assumptions	Requirements
G1	D2,D3,D7	R1,R2,R4,R5,R11,R21,R22
G2	D6,D7,D9,D10	R6,R7,R8
G3	D2,D3,D5,D7	R4,R5,R9,R10,R11,R12,R13,R21,R22
G4	D1,D6,D7	R14
G5	D1,D2,D3,D6,D7	R11,R17,R18,R19
G6	D2,D3,D4,D6,D7,D8	R16,R20,R21,R22,R23,R31
G7	D6,D7	R24,R25,R26,R27,R28,R29,R30
G8	D6,D7	R3,R21,R22
G9	D1,D2,D3,D6,D7,D9	R8,R21,R22,R32,R33
G10	D6,D7,D12	R35,R36,R37
G11	D2,D3,D7,D9,D11	R15,R23,R34

G1	**Allow policy makers to identify the farmers to be rewarded and those performing bad**
D2	The information inserted about the farm are reliable
D3	The information inserted about the farm are updated daily
D7	Each user has the necessary skills to navigate the designed user interface and use its features
R1	All users are notified when the annual ranking is completed
R2	The system must allow policy makers to view the data needed to contact farmers
R4	Policy makers must be able to access sensitive data of other users
R5	Policy makers must be able to view farm production data
R11	The system must show the ranking of the farms
R21	The system must update the score associated to a farm every time its data are updated
R22	The system must update the ranking of the farms

G2	**Allow farmers to ask the help they need**
D6	Each farm has internet connection and a device to access the platform (smartphone or pc)
D7	Each user has the necessary skills to navigate the designed user interface and use its features
D9	The agronomist is available during working days
D10	The agronomist responds promptly to requests for help
R6	Agronomists must be notified if their help has been required
R7	The system must allow agronomists to view the data needed to contact farmers
R8	Agronomists must be able to access to sensitive data of other users

G3	**Allow policy makers to evaluate if the advise given by agronomists works**
D2	The information inserted about the farm are reliable
D3	The information inserted about the farm are updated daily
D5	The reports are uploaded regularly in the system
D7	Each user has the necessary skills to navigate the designed user interface and use its features
R4	Policy makers must be able to access sensitive data of other users
R5	Policy makers must be able to view farm production data
R9	Politicians can view the graph of each farm's score trend
R10	The system must show the reports written by agronomists with the advice given
R11	The system must show the ranking of the farms
R12	The system must show the map of the farms contained in each zone
R13	The system must show the agronomist associated to each zone
R21	The system must update the score associated to a farm every time its data are updated
R22	The system must update the ranking of the farms

G4	Being able to anticipate climatic phenomena
D1	Meteorological forecasts are always available
D6	Each farm has internet connection and a device to access the platform (smartphone or pc)
D7	Each user has the necessary skills to navigate the designed user interface and use its features
R14	The system must allow users to be redirected to TSDPS meteo

G5	Allow farmers to visualize relevant data based on their location and type of production
D1	Meteorological forecasts are always available
D2	The information inserted about the farm are reliable
D3	The information inserted about the farm are updated daily
D6	Each farm has internet connection and a device to access the platform (smartphone or pc)
D7	Each user has the necessary skills to navigate the designed user interface and use its features
R11	The system must show the ranking of the farms
R17	The system must display the data entered by the farmer in chronological order from the most recent to the least
R18	Farmers must be able to view their farms' production data
R19	Farmers must be able to view their score

G6	Allow farmers to keep track of their production data
D2	The information inserted about the farm are reliable
D3	The information inserted about the farm are updated daily
D4	The sensors deployed on the farmers territory and measuring the humidity of soil work correctly
D6	Each farm has internet connection and a device to access the platform (smartphone or pc)
D7	Each user has the necessary skills to navigate the designed user interface and use its features
D8	Farmers use the counter to measure the amount of water used and it works correctly
R16	The system must store the data entered by the farmer in chronological order from the most recent to the least
R20	Farmers can update their farms' production data
R21	The system must update the score associated to a farm every time its data are updated
R22	The system must update the ranking of the farms
R23	DREAM AI has to reprocess the projections and suggestions associated to each farm when its data are updated
R31	The system must send a notification to the farmers (at 21.00) to remind them to update the daily data

G7	Allow farmers to create discussion forums with the other farmers
D6	Each farm has internet connection and a device to access the platform (smartphone or pc)
D7	Each user has the necessary skills to navigate the designed user interface and use its features
R24	The system must allow farmers to create a new discussion on the forum
R25	A farmer must be able to comment on a forum discussion
R26	A comment from a user must be able to tag another user
R27	The system must send a notification to a user when when his discussion is commented on
R28	The system must send a notification to a user when tagged in a comment
R29	The forum must store the discussions/comments in chronological order
R30	The forum must display the discussions/comments in chronological order

G8	Encourage the best farmers to share their experience on the forum
D6	Each farm has internet connection and a device to access the platform (smartphone or pc)
D7	Each user has the necessary skills to navigate the designed user interface and use its features
R3	The best farmers have to be notified asking them to share their story in the forum
R21	The system must update the score associated to a farm every time its data are updated
R22	The system must update the ranking of the farms

G9	Allows agronomists to send production advice to farmers and write reports on their visits
D1	Meteorological forecasts are always available
D2	The information inserted about the farm are reliable
D3	The information inserted about the farm are updated daily
D6	Each farm has internet connection and a device to access the platform (smartphone or pc)
D7	Each user has the necessary skills to navigate the designed user interface and use its features
D9	The agronomist is available during working days
R8	Agronomists must be able to access to sensitive data of other users
R21	The system must update the score associated to a farm every time its data are updated
R22	The system must update the ranking of the farms
R32	Agronomists must be able to view farm production data
R33	Agronomists must be able to write reports in the system

G10	**The system must distinguish users in 3 categories: farmers, policy makers and agronomists**
D6	Each farm has internet connection and a device to access the platform (smartphone or pc)
D7	Each user has the necessary skills to navigate the designed user interface and use its features
D12	The user picks up the right category
R35	The system must show a specific different form to compile to each user, based on the category he selects, during registration
R36	The system must ask each user to fill in the form to complete the registration
R37	Each user can view only the features specific to his category

G11	**The system must give advice to agronomists (DREAM AI)**
D2	The information inserted about the farm are reliable
D3	The information inserted about the farm are updated daily
D7	Each user has the necessary skills to navigate the designed user interface and use its features
D9	The agronomist is available during working days
D11	The intelligent algorithm works correctly by helping agronomists
R15	DREAM AI must process suggestions based on the evolution overtime of a farm's production and on other farms results
R23	DREAM AI has to reprocess the projections and suggestions associated to each farm when its data are updated
R34	DREAM AI must propose the advices computed to agronomists

3.2.3 Use Cases

1.User registration

Name	User registration
Actors	Farmer, Agronomist, Policy maker
Entry Condition	1. User has the internet connection available 2. The user has already opened the application on his device or accessed the webapp from his pc
Event Flow	1. User clicks on "Sign up" button 2. User select what type of user is (Farmer, Agronomist, Policy maker) 3. User compiles all the mandatory personal fields of the form 4. User press "continue" bottom 5. If user is a Farmer compiles all the mandatory farm fields of the farm form 6. The system validates the user code/farm code 7. The system confirms the registration of the user (and his farm) 8. The system saves the information
Exit Conditions	User is successfully registered to the system and can take advantage of all its features
Exception	1. User is already in the system 2. User code is not valid 3. User did not fill up all mandatory fields with valid data If one of these situations occurs, the system returns an error screen message and returns to the registration form page

2.User logs in

Name	User logs in
Actors	Farmer, Agronomist, Policy maker
Entry Condition	1. User has the internet connection available 2. The user is registered in the system 3. The user has already opened the application on his device or accessed the webapp from his pc
Event Flow	1. The user selects the “Log in” option 2. The user enters username and password in the respective fields 3. The user chooses the confirmation option
Exit Conditions	The user is logged in and the system allows him to visualize his data and use the specific features of his category
Exception	1. The user enters the wrong username 2. The user enters the wrong password In both cases, the system warns the user and tells him which field is wrong, suggesting to correct it

3.Start a discussion

Name	Start a discussion
Actors	Farmer
Entry Condition	1. Farmer has already logged in the system 2. Farmer is in the forum section screen
Event Flow	1. The farmer presses the "plus" button 2. The farmer insert title and body of the discussion 3. The farmer presses "publish" button 4. System saves the information

Exit Conditions	A new thread has been created successfully
Exception	1. Does not insert the title or body of the discussion 2. Some words entered are not allowed by the system (swear words, etc.) 3. The number of characters entered exceeds the maximum expected If one of these situations occurs, the system returns an error screen message and returns to the forum home page

4.Comment a discussion

Name	Comment a discussion
Actors	Farmer
Entry Condition	1. Farmer has already logged in the system 2. Farmer is in the forum section screen
Event Flow	1. Farmer presses a discussion to to view it 2. The system shows the selected discussion and its comments 3. The farmer presses "comment" button 4. The farmer enters the comment text 5. The farmer presses "publish" button 6. System saves the information
Exit Conditions	A new comment is added to a selected discussion
Exception	1. Does not insert the text of the comment 2. Some words entered are not allowed by the system (swear words, etc.) 3. The number of characters entered exceeds the maximum expected If one of these situations occurs, the system returns an error screen message and returns to the forum home page

5.Update data

Name	Update data
Actors	Farmer
Entry Condition	1. Farmer has already logged in the system 2. Farmer is in the home page screen
Event Flow	1. Farmer can press the "update data" button on the home-page, or open the drop-down menu and select "Update data" option 2. The system shows the main parameters of the farm (eg water consumption) and the situation for each field on the farm 3. The farmer presses "edit data" button 4. The farmer changes the data according to the daily situation 5. The farmer presses "confirm" button 6. System saves the changes (if a farmer changes the data several times during a day, each change overwrites the previous one in the daily tracking)
Exit Conditions	The daily data is updated
Exception	1. If the data is not updated, the same condition as the previous day is recorded 2. If the parameters are not changed before pressing "confirm" an alert message warns the user before saving the data

6.User views weather forecasts

Name	User views weather forecasts
Actors	Farmer, Agronomist, Policy maker
Entry Condition	1. The user has successfully logged in 2. User is in the home page screen

Event Flow	1. The user open the drop-down menu and select "Weather forecast" option 2. The system redirects the user to the external API of TS-DPS 3. The forecasts desired are shown to the user
Exit Conditions	The user is able to view the forecasts and to extrapolate the needed information
Exception	1. If an error occurs while redirecting the user to the API, the system will show an error message and the user will be redirected to his home page

7.Farmer asks for help

Name	Farmer asks for help
Actors	Farmer, Agronomist
Entry Condition	1. The actors involved are registered in the system 2. The farmer has successfully logged in 3. The agronomist is assigned the zone to which the farmer belongs
Event Flow	1. The farmer open the drop-down menu and select “Ask for help” option 2. The farmer compiles a brief form describing his main problems 3. The farmer chooses the confirmation option 4. The agronomist receives a notification informing him of the new intervention requested 5. The agronomist logs in the system 6. The agronomist by selecting the appropriate option displays the data of the farmer and of his farm, including the telephone number entered 7. The agronomist contacts the farmer via the telephone number found and together they arrange a meeting

Exit Conditions	The agronomist and the farmer have agreed on a meeting through which the farmer will be able to express and resolve his problems
Exception	1. If the farmer does not fill in the form correctly or leaves blank entries, the system warns the farmer and tells him which field is wrong, suggesting to correct it 2. If an error occurs while compiling or actually loading the help request into the system, an error message is displayed and the farmer is redirected to the home page

8.User views the ranking

Name	User views the ranking
Actors	Farmer, Agronomist, Policy maker
Entry Condition	1. The actor has successfully logged in 2. Actor is in the home page screen
Event Flow	1. The actor open the drop-down menu and select “View ranking” option 2. The system shows the ranking of all the farms registered, ordered from the first farm with the highest score to the one with the least 3. If the user who is viewing the ranking is a farmer, the system automatically highlights the item in the ranking corresponding to his farm 4. If the user who is viewing the ranking is an agronomist, the system automatically highlights all the items in the ranking corresponding to the farms belonging to his zone 5. If the user who is viewing the ranking is a policy maker, the system automatically highlights the top 10 items in the ranking
Exit Conditions	The ranking has been displayed satisfying the request received

Exception	1. If a certain user requests the ranking to be displayed while another user is updating his farm data, the system, having not yet processed the new information entered, will show the user the last computed ranking 2. If an error occurs while displaying the ranking, the system will show an error message and the user will be redirected to his home page

9.View farm data

Name	View farm data
Actors	Farmer, Agronomist, Policy maker
Entry Condition	1. The actor has successfully logged in 2. Actor is in the home page screen
Event Flow	1. The user open the drop-down menu and selects the "View farm data" option 2. If the user trying to view the data is a farmer, the system displays the data entered by him in chronological order from the most recent to the least 3. If the user trying to view the data is an agronomist, the system shows him the list of all the farms under his competence. Then, he will be able to view the data, sorted in chronological order, of any farm of the list by selecting the corresponding item 4. If the user trying to view the data is a policy maker, the system shows him the list of all the farms registered in the system. Then, he will be able to view the data, sorted in chronological order, of any farm of the list by selecting the corresponding item
Exit Conditions	The user is able to view the searched data
Exception	1. If an error occurs while displaying the data, the system will show an error message and the user will be redirected to his home page 2. If an agronomist or a policy maker requests the data of a certain farm to be displayed while his owner is updating them, the system will show the last inserted data

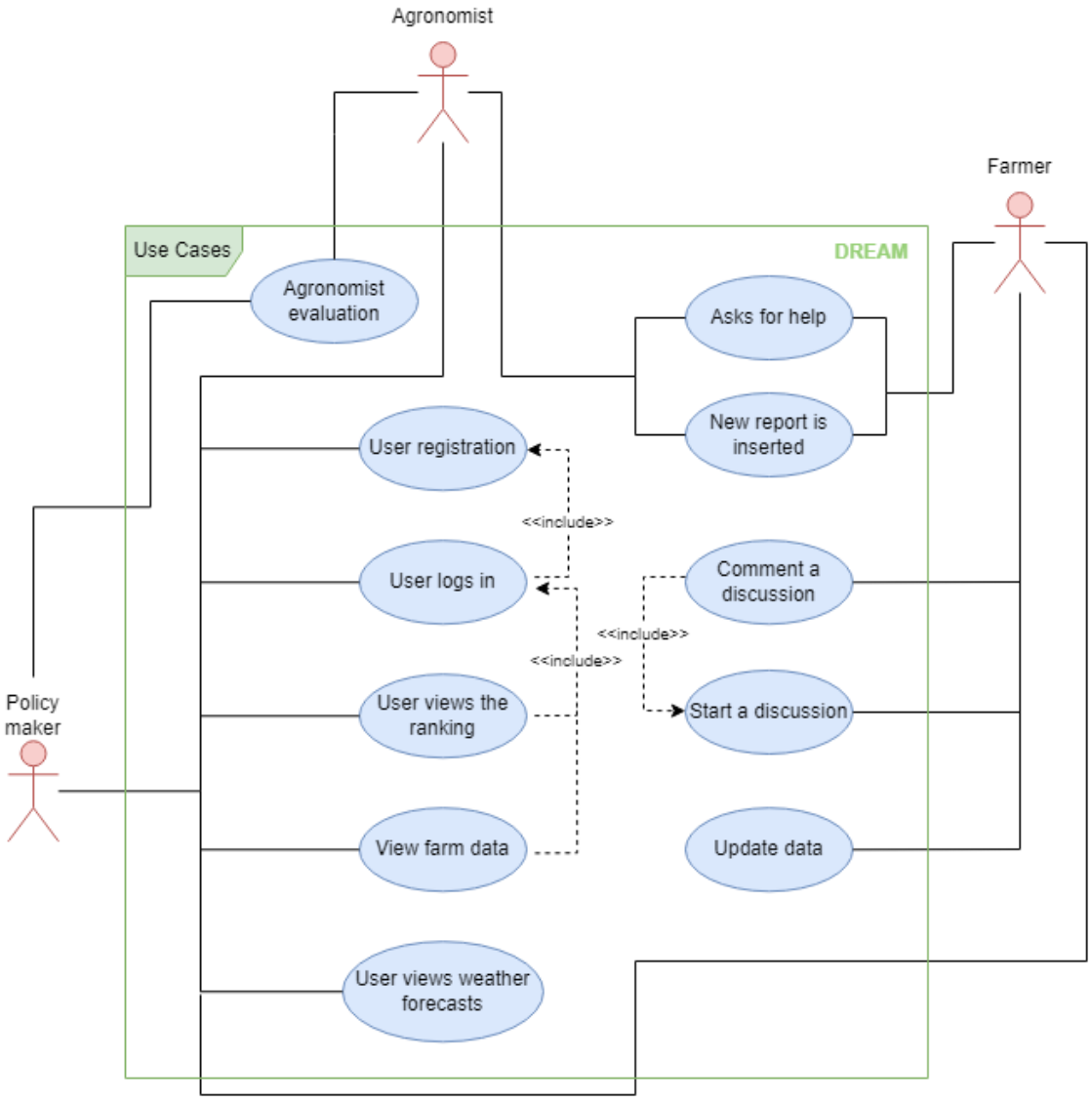
10.Policy maker evaluates an agronomist

Name	Policy maker evaluates an agronomist
Actors	Agronomist, Policy maker
Entry Condition	1. Both the agronomist and the policy maker are registered in the system 2. The policy maker has successfully logged in 3. The policy maker is in the home page screen
Event Flow	1. The policy maker open the drop-down menu and selects the “View map” option 2. The system refers to the external API of Google Maps to display a map of the state containing all the farms registered in the system divided into zones represented with distinct colors, each associated to an agronomist 3. The policy maker selects a zone of the map corresponding to an agronomist to evaluate, then for each farm within that zone he chooses to view the data of the farm by selecting the placeholder representing it on the map 4. The system shows him the graph of the farm’s score trend and all the reports written by the agronomist on that farm 5. The policy maker reads all the advice reported in the reports 6. The policy maker analyzes the graph to see when advises were given and how they have changed the farm’s production over time
Exit Conditions	Analyzing the impact of the advice on the productivity of the farm and repeating this analysis for each farm in a specific area of the map, the policy maker can evaluate the activity of the agronomist assigned to that area
Exception	1. If an error occurs while displaying the data of a farm, the system will show an error message and the policy maker will be redirected to his home page 2. If an error occurs while connecting or using the API, the map will be closed and the user redirected to his home page

11.New report is inserted

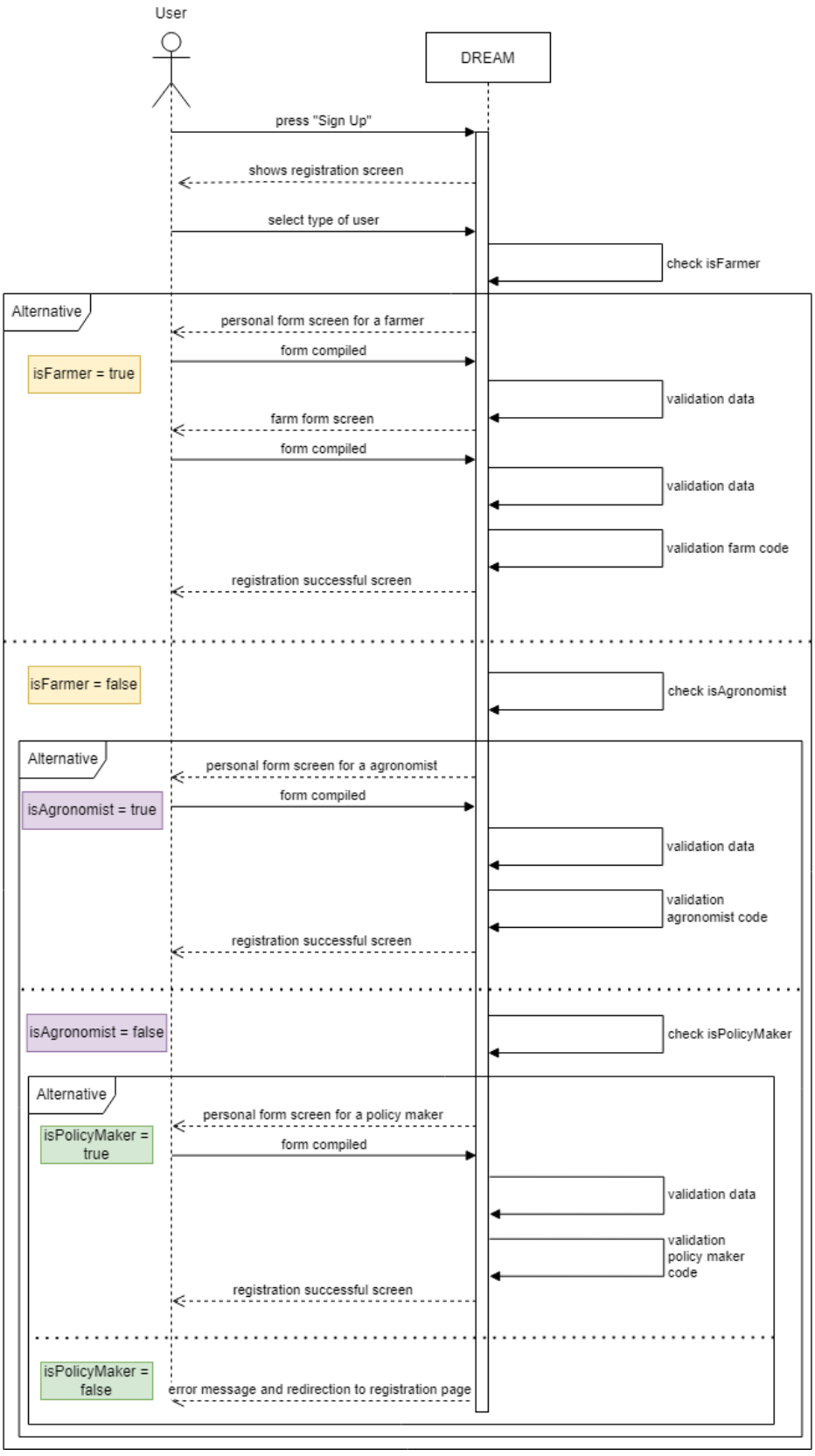
Name	New report is inserted
Actors	Agronomist, Farmer
Entry Condition	1. The agronomist has successfully logged in 2. The farmer has successfully logged in
Event Flow	1. Agronomist can press the "write report" button on the homepage, or open the drop-down menu and select "Write report" option 2. The system shows the google map with all the farms that the agronomist takes care of 3. The agronomist select the farm to report on 4. The agronomist press "continue" bottom 5. DREAM AI shows some possible suggestions for advice to give 6. Agronomist fills in the form relating to the visit carried out and fills in the part relating to the advice 7. The agronomist press "finish" bottom 8. The system save the data 9. Farmer can open the drop-down menu and select "Advise" option to see the report and advice given
Exit Conditions	A report was successfully inserted into the system
Exception	1. If an error occurs while connecting or using the API, the map will be closed and the agronomist redirected to report initial page 2. If agronomist did not fill up all mandatory fields with valid data during writing the report, a box with an error message appears and the agronomist redirected to report initial page

3.2.4 Use Cases Diagram

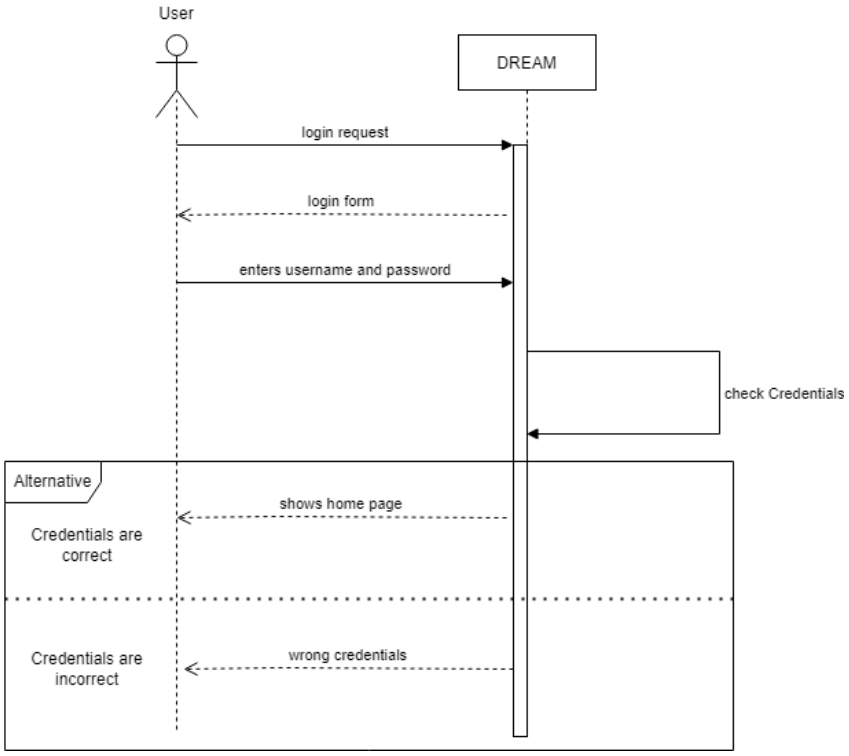


3.2.5 Sequence Diagrams

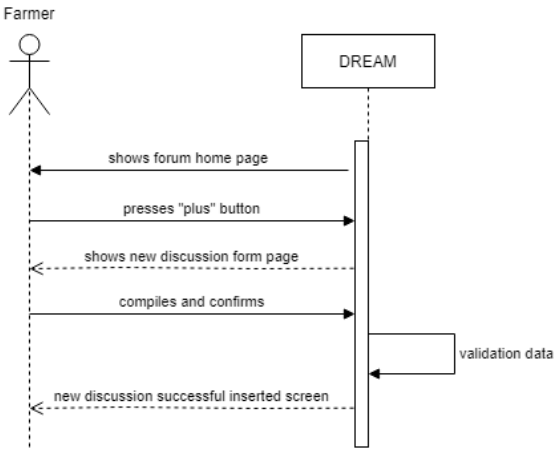
1.User registration



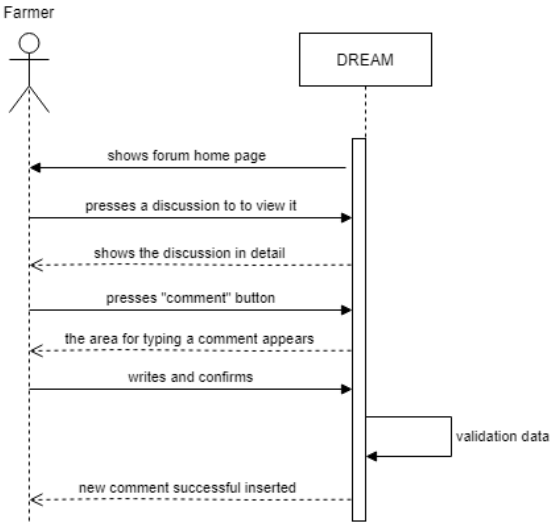
2.User logs in



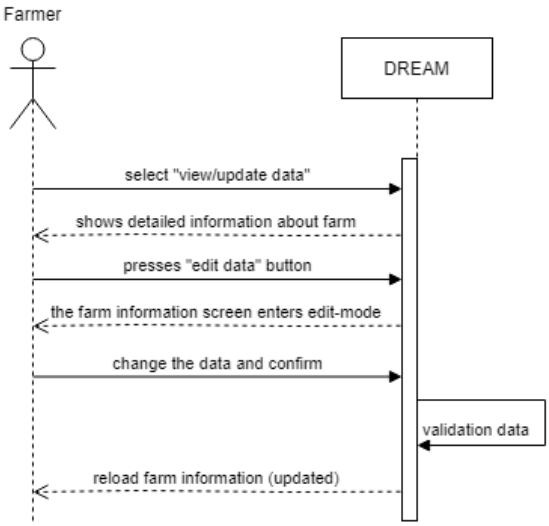
3.Start a discussion



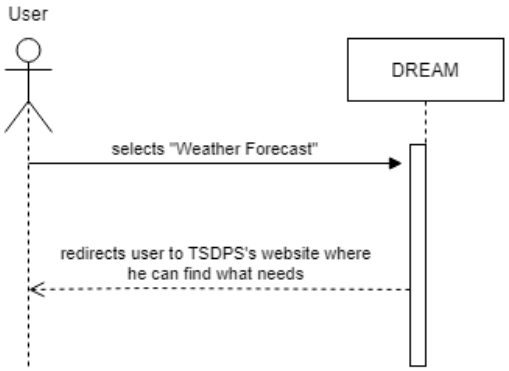
4.Comment a discussion



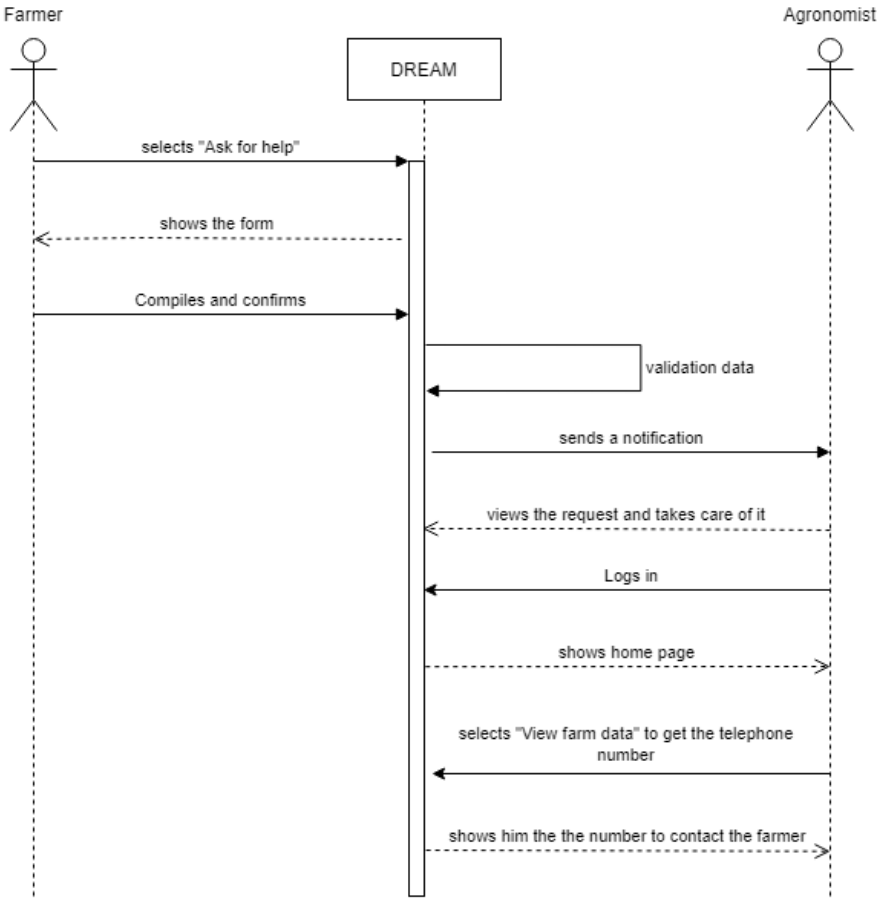
5.Update data



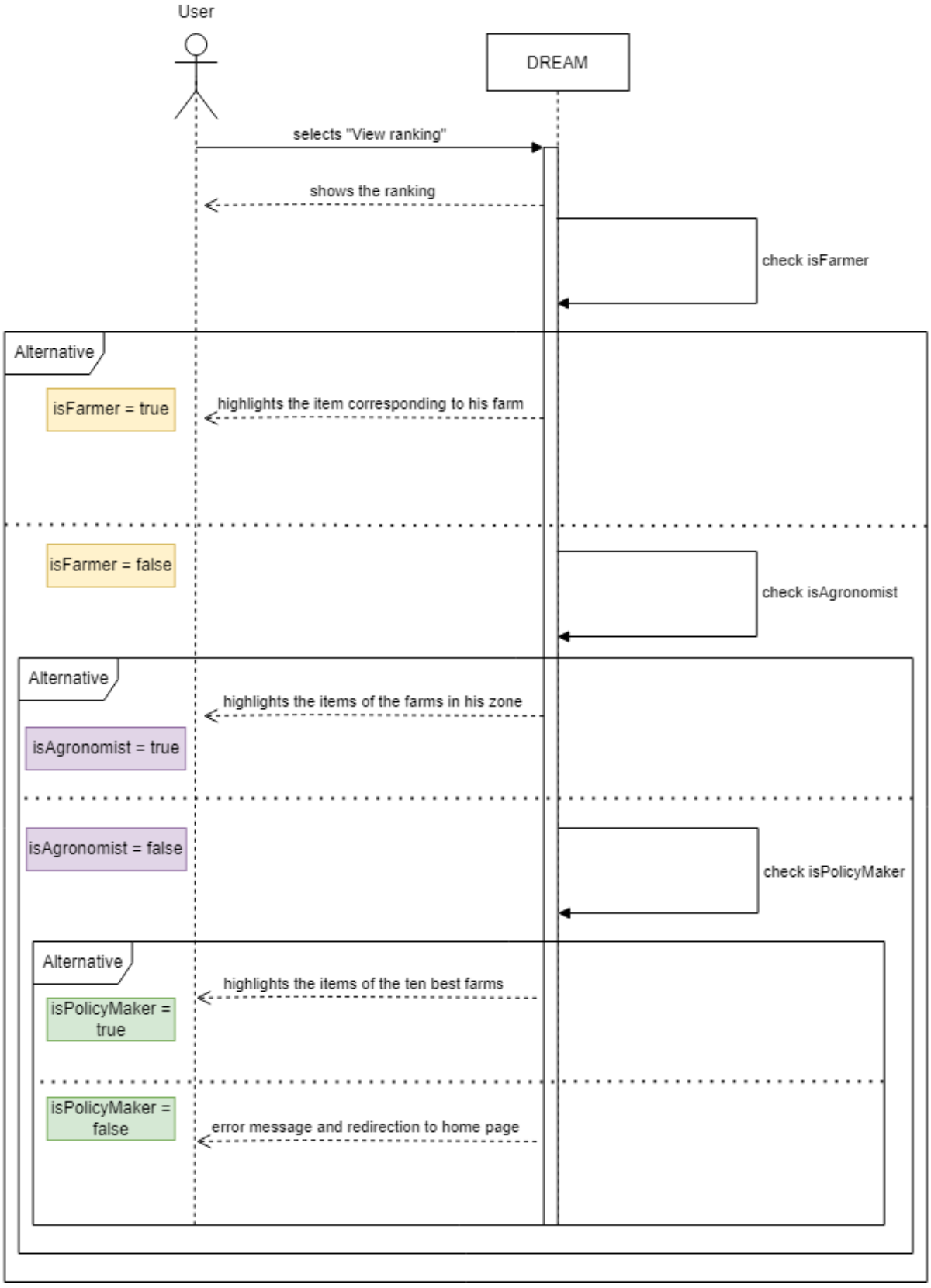
6.User views weather forecasts



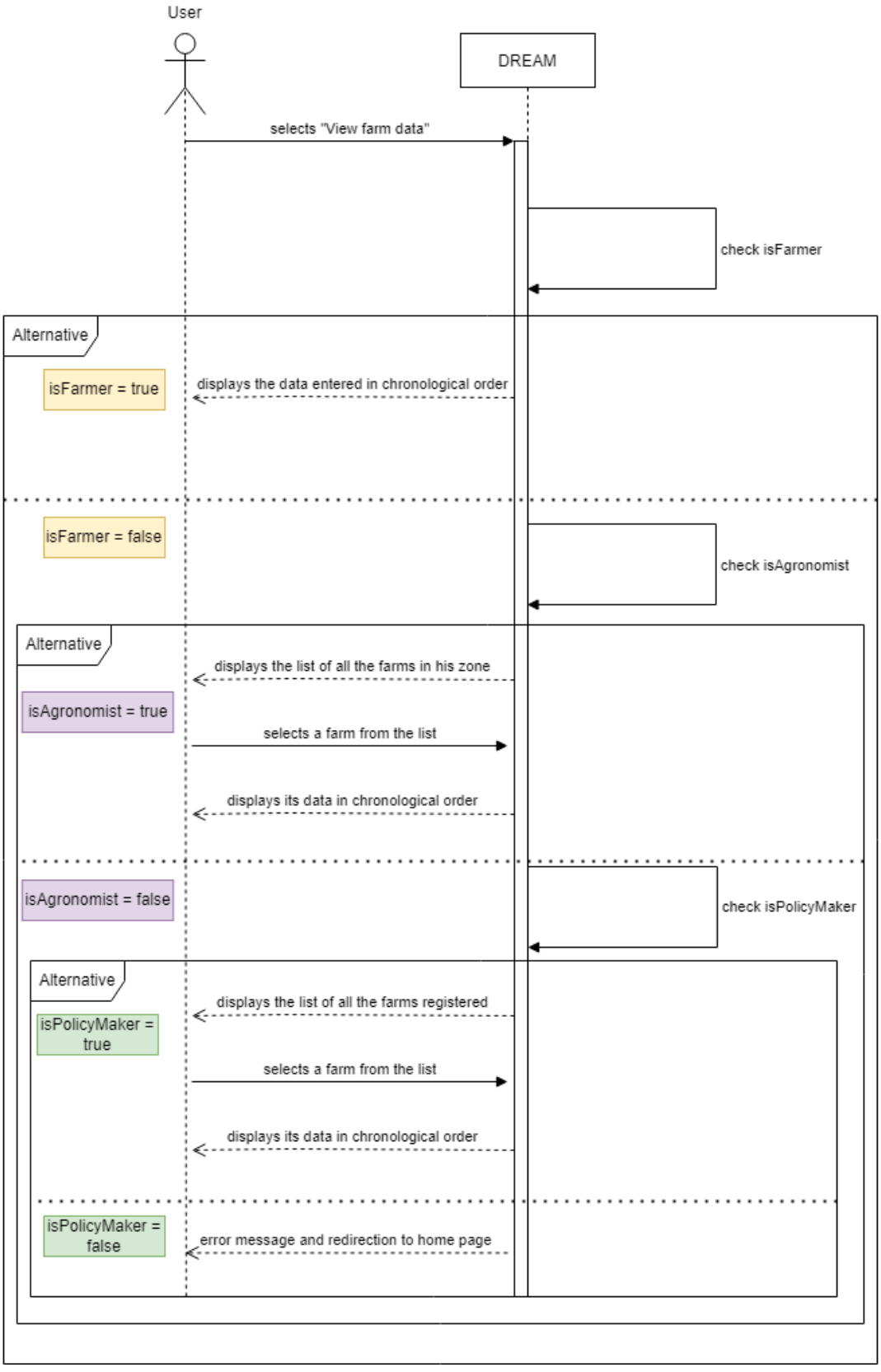
7.Farmer asks for help



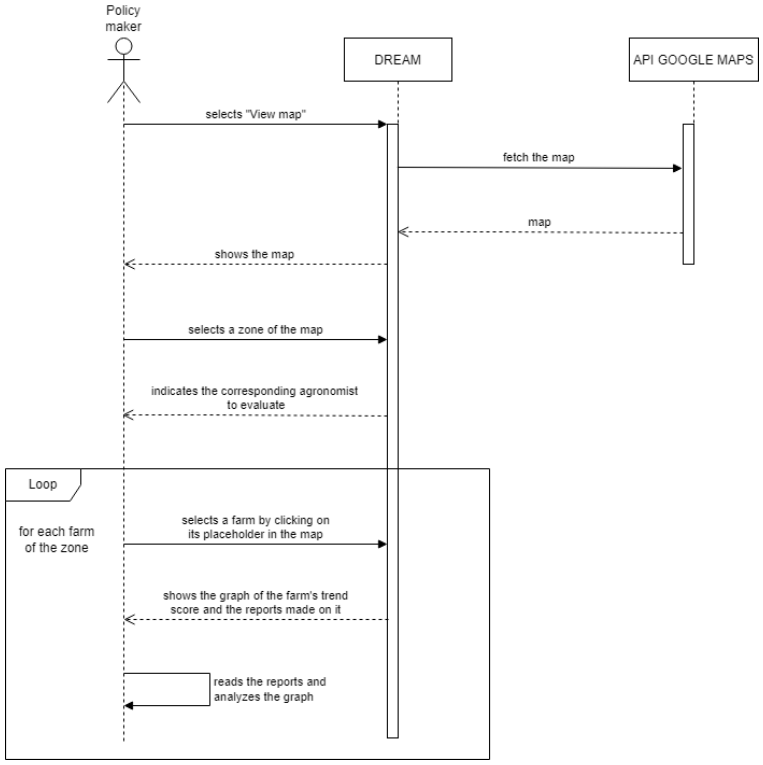
8.User views the ranking



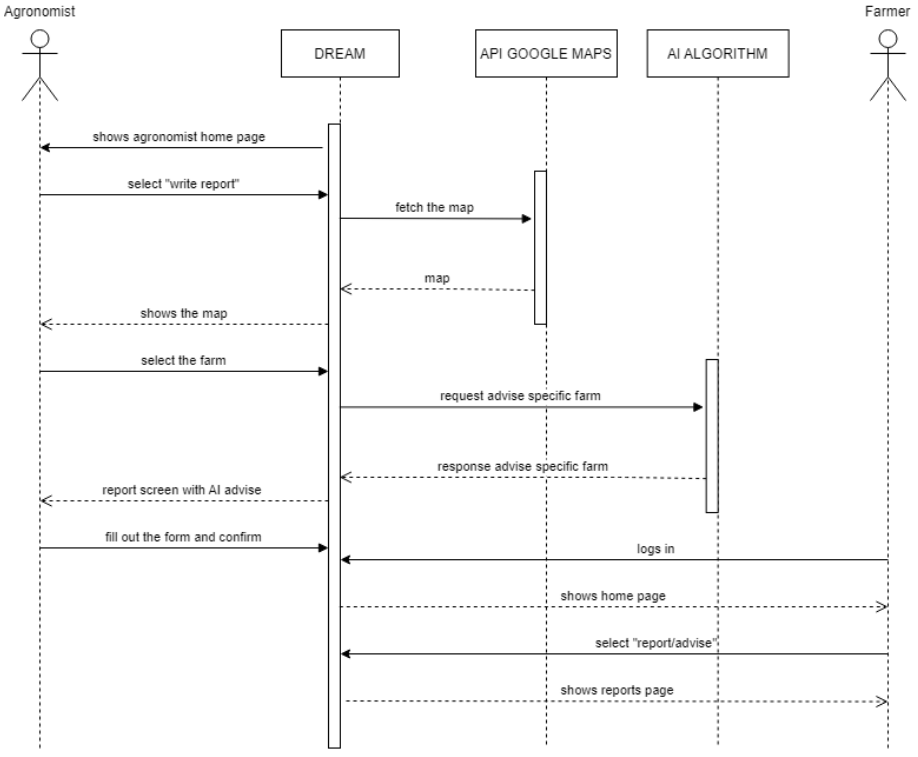
9.View farm data



10.Policy maker evaluates an agronomist



11.New report is inserted



3.3 Performance Requirements

System cannot go down for more than 8.76 hours a year.

If the system goes down it must recover in no more than 15 minutes and that when it resumes the system must notify users with a notification.

System must be able to guarantee the simultaneous connection of 25 millions individuals (as 55% of the Telegana population lives on agriculture).[6]

The system has to update the ranking once a minute.

Finally, it should be able to send a response to a query and run its algorithm on metadata in less than 5 seconds.

3.4 Design Constraints

3.4.1 Standards compliance

The code should follow the requirements contained in this document. Furthermore, its comments should be clear and focused.

3.4.2 Hardware limitations

The system requires an internet connection.

Furthermore, all users, to access the system, must have either a device to access the internet, or a smartphone on which to download the app. In case of the app, the smartphone must have android 5.0/iOS 8.0 (2014) or later versions.

3.5 Software System Attributes

3.5.1 Reliability

The system must be fault tolerant, for this to be guaranteed it must be created a backup system that allows the replication of the data contained in the central server but also of the running processes which provide the services of the system. In the occurrence of an error, the system must guarantee a reduced recovery time of no more than 15 minutes. Once the system is back to work properly, the resumption must be notified to all its users through an appropriate notification.

3.5.2 Availability

The system must guarantee a good availability while not being critical; in fact, it manages important information and guarantees some functions such as the possibility to ask for help that must be available when requested. The system should be available 99.9% of the time, that means that the average time between the occurrence of a fault and service recovery (MTTR) should be roughly 8.76 hours per year.

3.5.3 Security

The data entered and updated by users in the system contain sensitive information which must therefore be protected. To this end, encryption is applied to both the storage of production data and the login credentials of each user, in this way the system is able to prevent unauthorized accesses to confidential information. This context also includes the procedure of validation of the data entered carried out by the system during the registration phase as well as during other processes. Firewalls are used to better manage security in the system. In general, the central database must be protected to prevent any possible external and internal attack and to manage any hardware malfunctions.

3.5.4 Maintainability

The development of the system must be characterized by the application of a good standard in writing its code by using the appropriate design patterns, such as to make the code reusable and easily modified in case of malfunctions. Furthermore, the code must be written in Java, be extensively commented to make it easy to read and be test driven. Also, from the hardware point of view, the system must be constantly monitored and maintained thanks to the deployment of network engineers who have the necessary skills to manage it.

3.5.5 Portability

The system if used through the application must be compatible with different platforms like Android and iOS operating system for mobile devices. While its desktop version must be supported by Windows, Linux and Mac operating system.

3.5.6 Scalability

The system must be scalable to adapt its multiple services to the great variety of users connected, guaranteeing all of them the possibility of using its functions and promptly receiving answers as previously established. This is the main property that the system must guarantee, since it is foreseen, for future developments, the possibility of extending the use of the platform even outside the state of Telengana to create a global network.

4 Formal Analysis Using Alloy

This section is dedicated to the Alloy[8] model of the DREAM software, included all the functions of the application (Section 2.2) and their most important constraints. In this model, we want to prove that the links and cardinalities in the UML chart are correct, in particular we want to prove:

1. Check that the ranking is working correctly so that you can identify the best farms.
2. Verify that farmer requests for help are being handled correctly.
3. Reports are properly linked to other entities.
4. The forum works correctly and allows farmers to share ideas.

Time references have also been included in this model, which will not be shown in the graphs to avoid complicating them too much.

The cardinality of all the strings has been set to "lone", this is because the alloy software is not able to manage them (and this is the reason why these fields do not appear in the graphs). In some cases, however, the field would be with cardinality "one", as it is a mandatory field, as for example in Report.desciption.

4.1 Alloy Code

```
//VARIABLES
//Boolean variables
abstract sig Bool{}
one sig True extends Bool{}
one sig False extends Bool{}

//Temporal variables
sig Date{
    number: one Int,
    month: one Int,
    year: one Int
    } {
    number>0
    month>0
    year>0
}
sig Time{
    hours: one Int,
    minutes: one Int,
    seconds: one Int
    } {
    hours>=0
    minutes>=0
    seconds>=0
}
```

```
//Users
abstract sig User
{
    id: Int,
    name: lone String,
    surname: lone String,
    views: one Ranking
    }{
    id>0
}
sig PolicyMaker extends User{}
sig Farmer extends User
{
    username: lone String,
    owns: one Farm,
    askHelp: lone Agronomist
}
sig Agronomist extends User
{
    manages: one Province
}

//Area
sig Province{}

//Farm data
sig Farm
{
    id: Int,
    score: Int,
    positionRanking: Int,
    waterAmount: Int,
    telephoneNumber: Int,
    assigned: one Agronomist,
    partecipate: one Ranking,
    located: one Province
    }{
    id>0
    score>0
    positionRanking>0
    waterAmount>=0
    telephoneNumber>911099999999 //numbers in india with area code

}
sig Land
{
    id: Int,
    belongs: one Farm,
    dimensions: Int,
    humidity: Int,
    empty:one Bool,
```

```
    host: lone Product,
    crop: set Product
    }{
    id>0
    dimensions>0
    humidity>0
}
sig Product
{
    id: Int,
    name: lone String,
    storage: one Farm,
    growingTime: Int
    }{
    id>0
    growingTime>1 //at least a month to grow
}

//Report
sig Report
{
    id: Int,
    desciption: lone String,
    advise: lone String,
    author: one Agronomist,
    farm: one Farm,
    date: one Date,
    time: one Time,
    }{
    id>0
}

//Forum
sig Discussion
{
    creator: one Farmer,
    id: Int,
    title: lone String,
    description: lone String,
    date: one Date,
    time: one Time,
    }{
    id>0
}
sig Comment
{
    id:Int,
    text: lone String,
    author: one Farmer,
    topic: one Discussion,
    date: one Date,
```

```
    time: one Time,
    }{
    id>0
}

//Ranking
one sig Ranking{}

//CONSTRAINT

//unique id constraint
fact uniqueIDDiscussion{
all disj d, d': Discussion | d.id != d'.id
}
fact uniqueIDComment{
all disj c, c': Comment | c.id != c'.id
}
fact uniqueIDUser{
all disj u, u': User | u.id != u'.id
}
fact uniqueIDFarm{
all disj f, f': Farm | f.id != f'.id
}
fact uniqueIDReport{
all disj r, r': Report | r.id != r'.id
}
fact uniqueIDLand{
all disj l, l': Land | l.id != l'.id
}
fact uniqueIDProduct{
all disj p, p': Product | p.id != p'.id
}

//Ranking
//two farms cannot be in the same position
fact farmRankingPosition{
all disj f, f': Farm | f.positionRanking != f'.positionRanking
}

//one farm is above another if it has a higher score
fact farmRankingPositionScore{
all disj f, f': Farm | (f.positionRanking < f'.positionRanking) iff (f.score
>= f'.score)
}

//Farm
//a farm for every farmer
fact oneFarmerFarm{
all f:Farm | one f.~owns
}
```

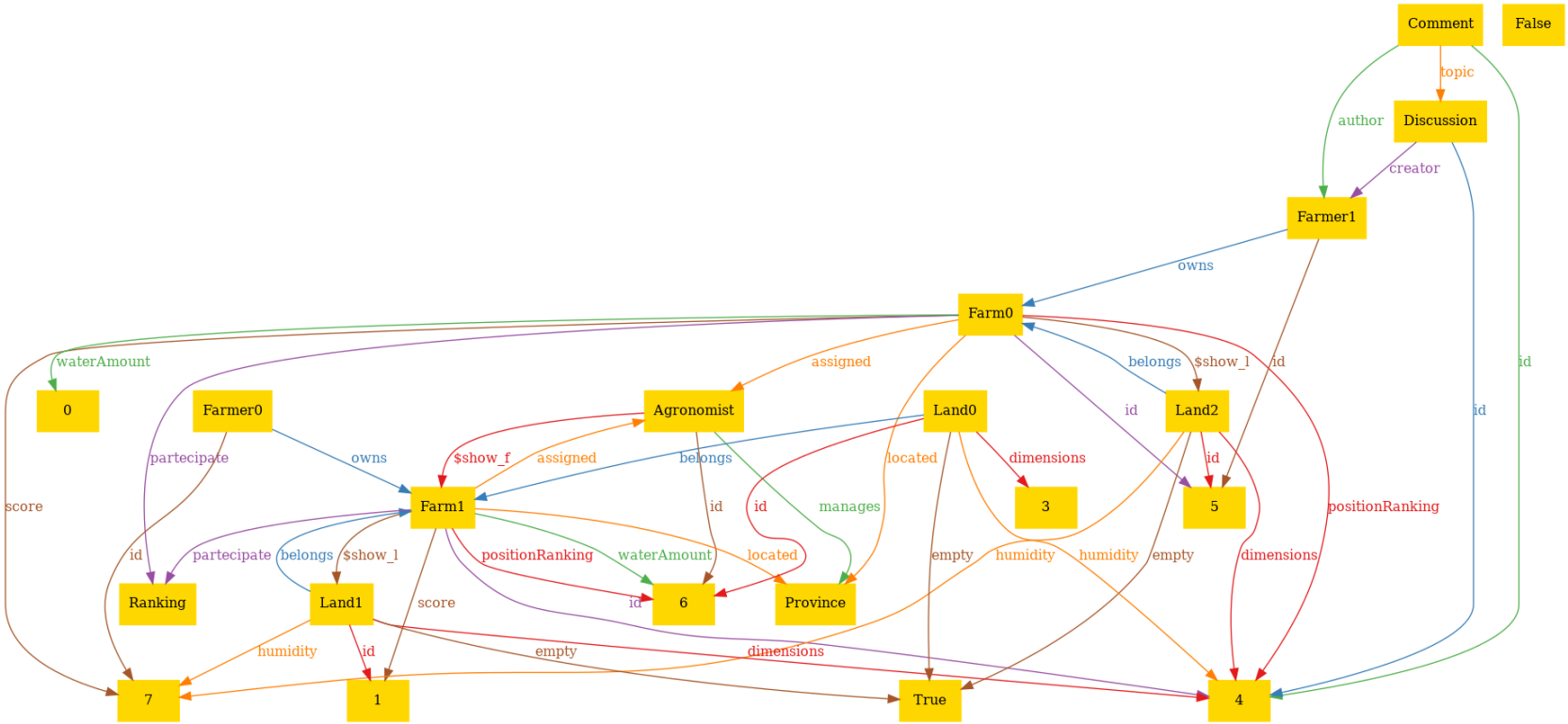
```
//waterAmount = 0 iff all lands are empty, only water consumption for
agriculture is considered
fact noWaterAmount{
all f:Farm, l:Land |( l.belongs = f and f.waterAmount = 0) implies l.empty =
True
}

//a field is empty if it is not "hosting" products
fact emptyLand{
all l:Land |
(l.empty = True implies no l.host)
and
(l.empty = False implies one l.host)
}

//each farm has at least 1 land
fact oneFarmLand{
all f: Farm| some l:Land | l.belongs = f
}

//Agronomist
//each agronomist manages a different province
fact oneProvinceAgronomist{
all disj a, a': Agronomist | a.manages != a'.manages
}

//each agronomist has at least 1 farm
fact oneFarmAgronomist{
all  a: Agronomist | some f: Farm | f.assigned = a
}

//if a report is linked to a farm and agronomist, the same agronomist is
assigned to the farm
fact reportConnection{
all r: Report, f:Farm, a: Agronomist | (r.author = a and r.farm = f) implies
(f.assigned = a)
}

//a farmer ask for help to the agronomist assigned to his farm
fact farmerAgronomistFarmConnection{
all f: Farmer, a: f.askHelp | f.owns.assigned = a
}

//for every request for help there is a report
fact askHelpReportConnection{
all a: Farmer.askHelp | some r: Report | r.author = a
}

//if a farm is located in a province and is linked to an agronomist, that
agronomist manages that province
fact farmConnection{
```

```
all p: Province, f:Farm, a: Agronomist | (f.assigned = a and f.located = p)
implies (a.manages = p)
}

//Province
//each province has 1 agronomist
fact oneFarmAgronomist{
all  p: Province | one a: Agronomist | a.manages = p
}

//ASSERTIONS

//Forum
//there are no two farmers who have created the same discussion
assert oneDiscussionCreator{
no disj f,f': Farmer, d: Discussion | d.creator = f and d.creator = f'
}
check oneDiscussionCreator

//there are no two farmers who have created the same comment
assert oneCommentAuthor{
no disj f,f': Farmer, c: Comment | c.author = f and c.author = f'
}
check oneCommentAuthor

//a comment cannot be related to two discussions
assert oneCommentDiscussion{
no disj d,d': Discussion, c: Comment | c.topic = d and  c.topic = d'
}
check oneCommentDiscussion

//Farm
//there are no two farmers who have the same farm
assert oneFarmFarmer{
no disj f,f': Farmer, fa: Farm | f.owns = fa and f'.owns = fa
}
check oneFarmFarmer

//there are no two farms owned by a farmer
assert oneFarmerFarm{
no disj f,f': Farm, fa: Farmer | fa.owns = f and fa.owns = f'
}
check oneFarmerFarm

//a farm cannot be located in two provinces
assert oneProvinceFarm{
no disj p,p': Province, f: Farm | f.located = p and f.located = p'
}
check oneProvinceFarm

//there are no two agronomists assigned to the same farm
```

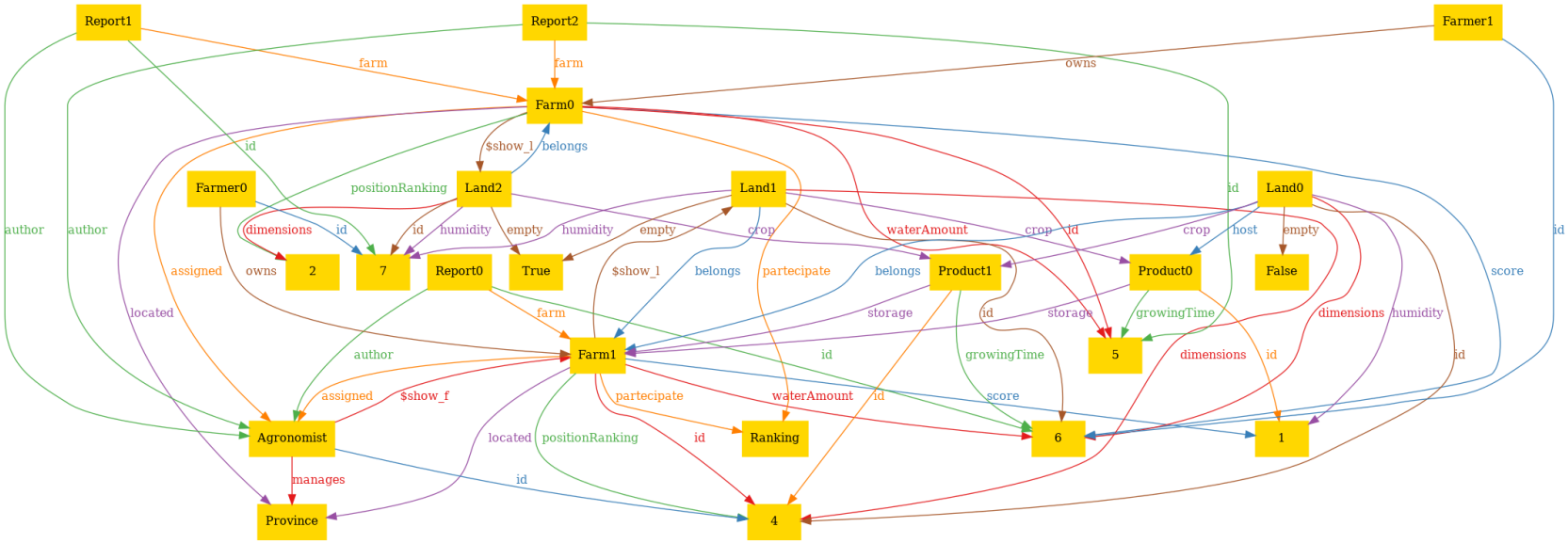
```
assert oneAgronomistFarm{
no disj a,a': Agronomist, f: Farm | f.assigned = a and f.assigned = a'
}
check oneAgronomistFarm

//there are no two farms with the same agronomist located in different
provinces
assert provinceFarmAgronomistConnection{
no disj  p,p': Province, f,f': Farm, a:Agronomist | f.assigned = a and
f'.assigned = a and f.located = p and f'.located = p'
}
check provinceFarmAgronomistConnection

//a land cannot belong to two different farms
assert oneFarmLand{
no disj f,f': Farm, l: Land | l.belongs = f and l.belongs = f'
}
check oneFarmLand

//Report
//there are no two reports written by different agronomists referring to the
same farm
assert agronomistReportFarmConnection{
no disj  a,a': Agronomist, r,r': Report, f:Farm | r.author = a and r'.author
= a' and r.farm = f and r'.farm = f
}
check agronomistReportFarmConnection

//there are no ignored askHelp
assert askHelpIgnored{
no a: Farmer.askHelp, r: Report | no r.author -> a
}
check askHelpIgnored

//Ranking
//there are no two farms that, despite having the same score, are in the same
position
assert oneRankingPosition{
no disj f,f': Farm | f.score = f'.score and f.positionRanking =
f'.positionRanking
}
check oneRankingPosition

//PREDICATES

pred showRanking
{
#Ranking=1
#Farm=2
#Farmer.askHelp=0
#Report=0
```

```
#Discussion=0
#Agronomist.views=1
}
run showRanking

pred showAskHelp
{
#Farmer=2
#Farmer.askHelp=1
#Discussion=0
#Report=3
}
run showAskHelp

pred showReports
{
#Report=3
#Discussion=0
}
run showReports

pred showForum
{
#Farmer=2
#Discussion=2
#Comment=3
#Report=0
}
run showForum

pred showGeneral1{
#Farm=2
#Land=3
#Discussion=1
}
run showGeneral1

pred showGeneral2{
#Farm=2
#Land=3
#Report=3
}
run showGeneral2
```

4.2 Metamodels general



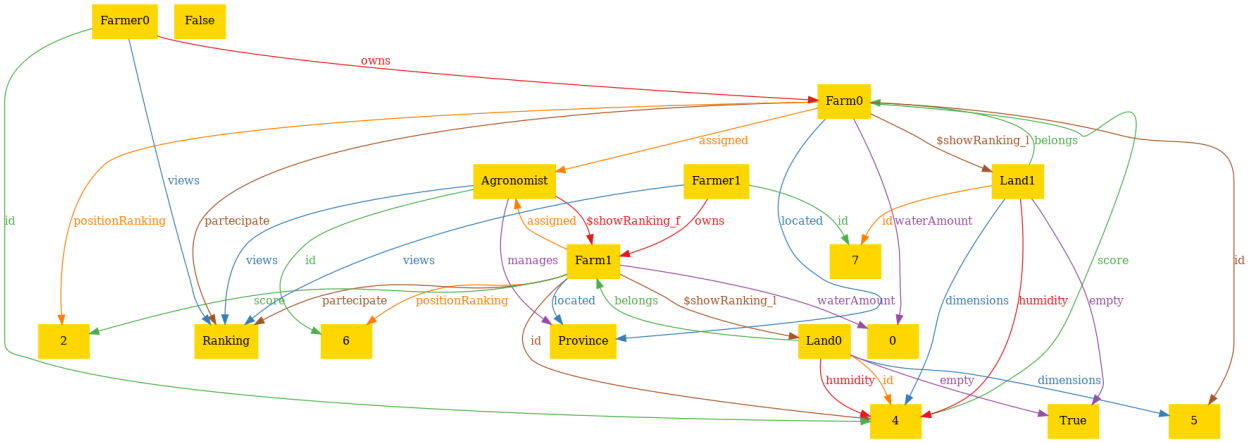
Zooming in you can better understand this general examples of the model.



In these graphs you can see how the system is very connected, many entities are necessarily linked to others and cannot exist independently. It is also important to highlight how the farm data collection process, which is the basis of all other functions, works correctly.

4.3 Metamodels goals

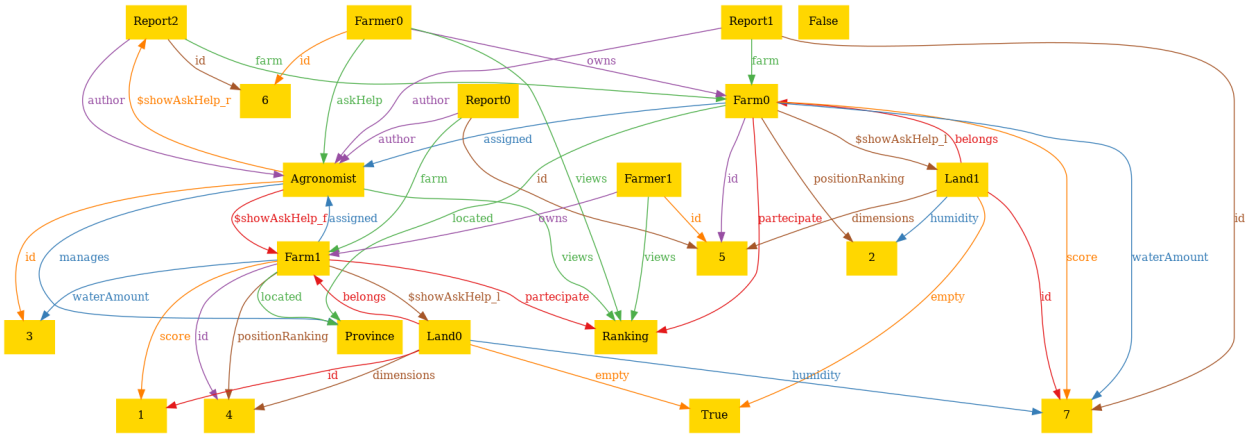
1.Check that the ranking is working correctly so that you can identify the best farms



This example shows us how two farms participate in the ranking. As you can see, Farm0 has a score of 4, while Farm1 has a score of 2. For this reason Farm0 is in position 2, while Farm1 is in position 6.

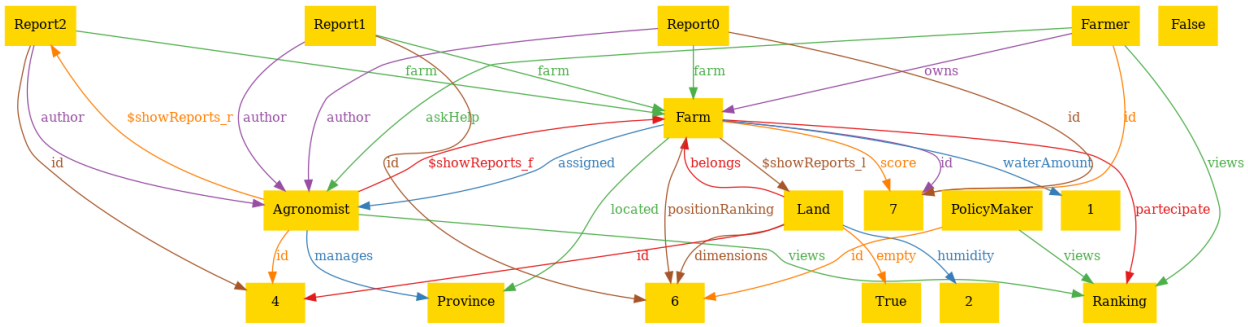
Farmer0, Farmer1 and the agronomist assigned to their farms (Agronomist) simultaneously view the ranking.

2.Verify that farmer requests for help are being handled correctly



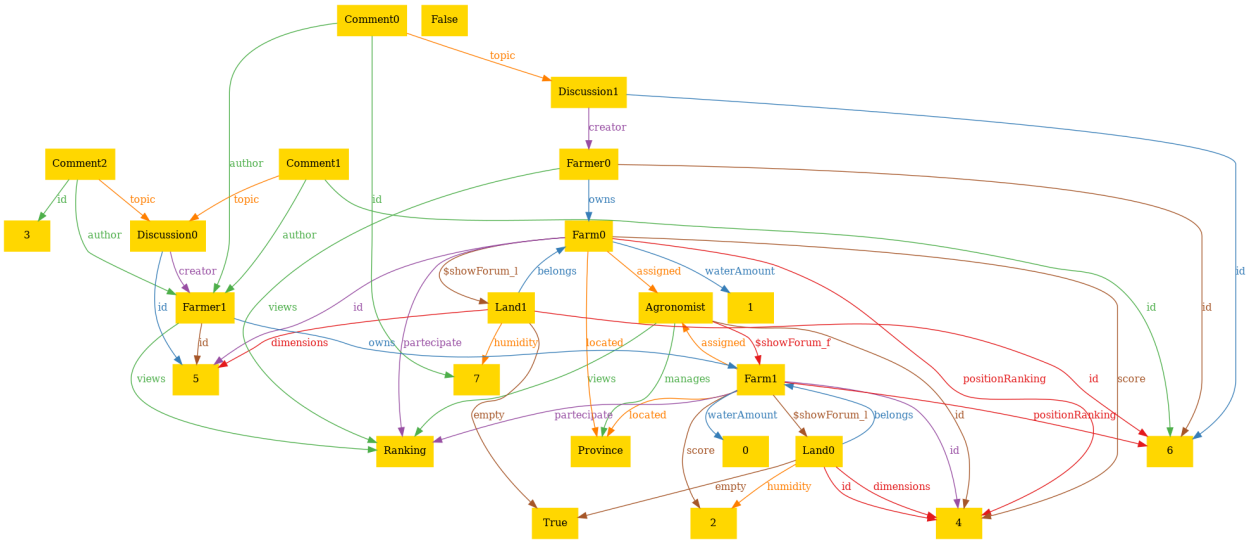
Farmer0 asks for help through the relative askHelp function and the agronomist associated with his farm (Agronomist, Farm0), will first visit the farm and then write Report2 in which there will also be useful advice. It is important to note that the writing of a report is not necessarily linked to the request for help, as for example Report0 and Report2 show, these are due to the periodic visits that agronomists make to the farms.

3.Reports are properly linked to other entities



As you can see from the example, the three reports were written by the same agronomist and refer to the same farm.

4.The forum works correctly and allows farmers to share ideas



There are two discussion (Discussion0, Discussion1) created by two different users (Farmer0, Farmer1) and the various comments can belong to either one discussion or the other. All constraints are respected and the forum works correctly.

4.4 Result of Assertions

Executing "Check oneDiscussionCreator"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11447 vars. 828 primary vars. 31901 clauses. 157ms.
No counterexample found. Assertion may be valid. 15ms.

Executing "Check oneAgronomistFarm"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11447 vars. 828 primary vars. 31901 clauses. 32ms.
No counterexample found. Assertion may be valid. 0ms.

Executing "Check oneCommentAuthor"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11447 vars. 828 primary vars. 31901 clauses. 47ms.
No counterexample found. Assertion may be valid. 0ms.

Executing "Check provinceFarmAgronomistConnection"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11567 vars. 834 primary vars. 32154 clauses. 32ms.
No counterexample found. Assertion may be valid. 16ms.

Executing "Check oneCommentDiscussion"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11471 vars. 828 primary vars. 31943 clauses. 31ms.
No counterexample found. Assertion may be valid. 15ms.

Executing "Check oneFarmLand"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11471 vars. 828 primary vars. 31943 clauses. 16ms.
No counterexample found. Assertion may be valid. 16ms.

Executing "Check oneFarmFarmer"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11459 vars. 828 primary vars. 31940 clauses. 32ms.
No counterexample found. Assertion may be valid. 0ms.

Executing "Check agronomistReportFarmConnection"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11543 vars. 834 primary vars. 32112 clauses. 16ms.
No counterexample found. Assertion may be valid. 10ms.

Executing "Check oneFarmerFarm"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11471 vars. 828 primary vars. 31943 clauses. 16ms.
No counterexample found. Assertion may be valid. 15ms.

Executing "Check askHelpIgnored"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11437 vars. 825 primary vars. 31871 clauses. 16ms.
No counterexample found. Assertion may be valid. 16ms.

Executing "Check oneProvinceFarm"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11471 vars. 828 primary vars. 31943 clauses. 32ms.
No counterexample found. Assertion may be valid. 0ms.

Executing "Check oneRankingPosition"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11747 vars. 825 primary vars. 32809 clauses. 16ms.
No counterexample found. Assertion may be valid. 0ms.

4.5 Result of Predicates

Executing "Run showRanking"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11404 vars. 819 primary vars. 31868 clauses. 16ms.
Instance found. Predicate is consistent. 34ms.

Executing "Run showAskHelp"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11400 vars. 819 primary vars. 31860 clauses. 15ms.
Instance found. Predicate is consistent. 47ms.

Executing "Run showReports"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11386 vars. 819 primary vars. 31814 clauses. 16ms.
Instance found. Predicate is consistent. 47ms.

Executing "Run showForum"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11400 vars. 819 primary vars. 31860 clauses. 16ms.
Instance found. Predicate is consistent. 31ms.

Executing "Run showGeneral1"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11393 vars. 819 primary vars. 31837 clauses. 16ms.
Instance found. Predicate is consistent. 32ms.

Executing "Run showGeneral2"
Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
11393 vars. 819 primary vars. 31837 clauses. 16ms.
Instance found. Predicate is consistent. 32ms.

5 Effort Spent

Topic	Pietro Valente Hours	Andrea Seghetto Hours	Total Hours
Discussion of the problem	together	together	10h
Introduction: Graphic layout	0h	0.5h	0.5h
Introduction: Purpose	3h	0h	3h
Introduction: Scope	0h	3.5h	3.5h
Introduction: World Phenomena, Shared Phenomena, Goals	together	together	5h
Overall Description: Product perspective	3h	3h	6h
Overall Description: Product functions	0h	1.5h	1.5h
Overall Description: User characteristics	0.5h	0h	0.5h
Overall Description: Assumptions, dependencies and constraints	together	together	3h
Overall Description: Graphic layout	1h	0h	1h
Revisiting Introduction to maintain consistency	together	together	5h
Specific requirements: External Interface Requirements	5h	4h	9h
Specific requirements: Functional Requirements discussion	together	together	2h
Specific requirements: Functional Requirements	3h	2.5h	5.5h
Specific requirements: Performance Requirements	0h	0.5h	0.5h

Topic	**Pietro Valente Hours**	**Andrea Seghetto Hours**	**Total Hours**
Specific requirements: Graphical layout	2h	0h	2h
Specific requirements: Design Constraints	1.5h	0h	1.5h
Specific requirements: Software System Attribute	0h	2h	2h
Formal analysis using Alloy	10h	9.5h	19.5h
Final revision of the document with changes	4h	4h	8h
Effort tracking	together	together	2h
	59.5h	**59.5h**	**95.5h**

References

[1] Specification Document: "01. Assignment RDD AY 2021-2022.pdf"

[2] Slides of the lectures

[3] All diagrams have been made with https://app.diagrams.net/

[4] Weather forecast site: https://www.tsdps.telangana.gov.in/aws.jsp

[5] Google maps API site: https://developers.google.com/maps

[6] Data on the Telegana population: https://en.wikipedia.org/wiki/Rythu_Bandhu_scheme#:~:text=The%20scheme,-The%20scheme%20offers&text=There%20is%20no%20cap%20on,make%20a%20living%20from%20agriculture.

[7] All the screen and possible interface has been made using Android Studio: https://developer.android.com/studio?gclid=CjwKCAiA-9uNBhBTEiwAN3IlNM0jGYMa-ogbqUe1ZYPbhrkRsx64WxpMt115G2om6klG5qiQ9f0IyhoCE4MQAvD_BwE&gclsrc=aw.ds

[8] Alloy tool used throughout section 4: https://alloytools.org/

