POLITECNICO
MILANO 1863

Computer Science and Engineering

Requirements Analysis and Specifications Document

DREAM Data-driven Predictive Farming

Software Engineering 2 Project
Academic year 2021 - 2022

22 December 2021
Version 1.0

Authors:
Lorenzo IOVINE
Nicola LANDINI
Francesco LEONE

Professor:
Matteo Giovanni ROSSI

Deliverable:	RASD
Title:	Requirement Analysis and Verification Document
Authors:	Lorenzo Iovine, Nicola Landini, Francesco Leone
Version:	1.0
Date:	22-December-2021
Download page:	https://github.com/fraleone99/IovineLandiniLeone
Copyright:	Copyright © 2021, Lorenzo Iovine, Nicola Landini, Francesco Leone – All rights reserved

Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

Agriculture has a key role in India's economy and covid-19 pandemic has highlighted the need of building a resilient food system. This need is increased by the problems due to climate change that will impact everything from productivity to livelihoods across food and farm systems and is predicted to result in a 4%-26% loss in net farm income towards the end of the century. In addition, according to Harvard Business Review, food demand is expected to increase between 59% to 98% by 2050.
For this reason policy makers, citizens, agronomists, and farmers should share data and information to achieve better results.

Telangana region is an extended and populous state of India, whose economy is mainly driven by agriculture. To address the described above problem, Telangana's government want to design, develop and demonstrate anticipatory governance models for food system using digital public goods and community-centric approaches to strengthen data-driven policy-making in the state.

The application aims to enable the acquisition, communication and combination of data provided by Telangana policymakers, farmers and agronomists as:

- meteorological forecasts
- farmers' production
- amount of used water
- soil humidity
- agronomists' report

The product will allow policymakers to identify farmers who are performing well and those who are performing badly. As the first ones will receive special incentives and will be asked to provide useful best practices to others. Furthermore, the application will provide information regarding the results of the farmers who received help.

The product needs to provide farmers the ability to visualize data relevant to them based on their location and type of production. Farmers should be also able to insert in the system data about production and any problem they face. They should be allowed to request help suggestions by agronomists and create forums to discuss with agronomists or other farmers.

The application will allow agronomists to insert the area they are responsible for, receive information about requests for help, and answer them. Agronomists need to know data about weather forecasts and the best performing farmers in the area; they also need to visualize and update daily plan visits of farms. At the end of each day the agronomist need to confirm the daily schedule.

1.2 Scope

To represent the scope of the project we use the "The World and The Machine" model by M. Jackson. It contains the events which cannot be observed by the system ("The World"), those strictly related to the system ("The Machine"), and those in common between them.

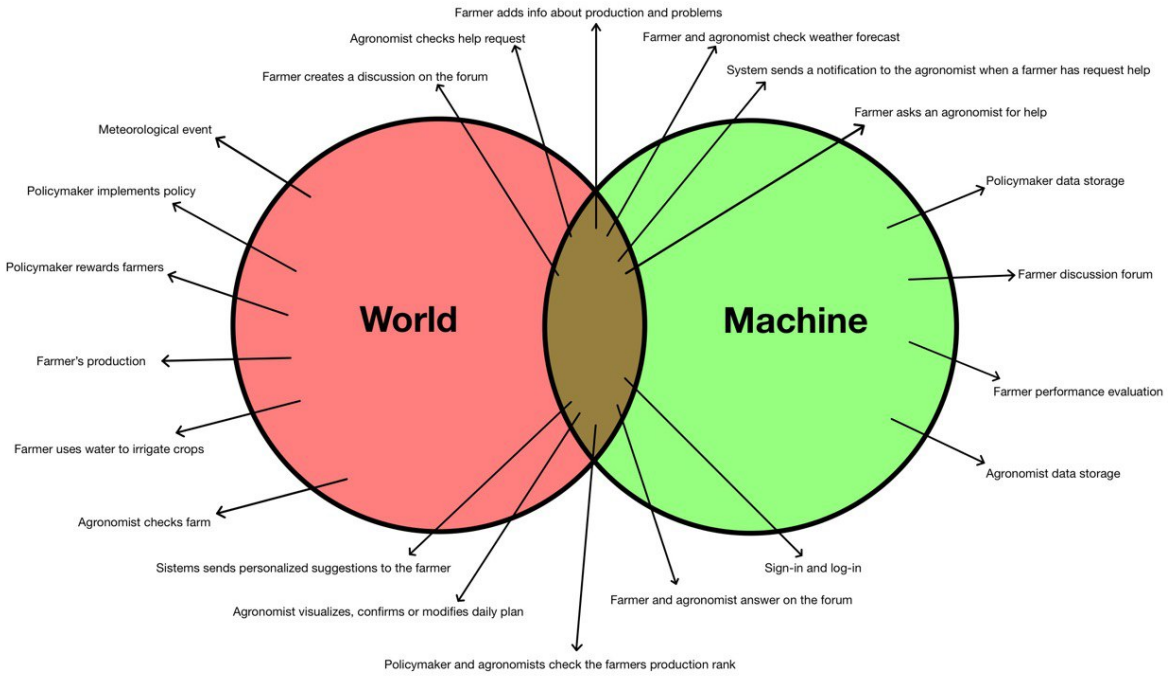

Figure 1: The World and the Machine diagram

1.2.1 World phenomena

- Meteorological event
- Policymaker implements policy
- Policymaker rewards farmers
- Farmer's production
- Farmer uses water to irrigate crops
- Agronomist checks farm

1.2.2 Machine phenomena

- Policymaker data storage
- Farmer discussion forum
- Farmer performance evaluation
- Agronomist data storage

1.2.3 Shared phenomena

Controlled by the World

- Policymaker and agronomist check the farmers production leaderboard
- Policymaker sign-in and log-in

- Farmer answers help request on the forum
- Farmer sign-in and log-in
- Farmer add information about his production
- Farmer add information about a problem he faces
- Farmer create a discussion on the forum about a problem
- Farmer checks weather forecasts
- Farmer asks an agronomist for help
- Agronomist sign-in and log-in
- Agronomist checks help requests
- Agronomist answers help request on the forum
- Agronomist answers privately help request
- Agronomist checks weather forecasts
- Agronomist visualizes daily plan
- Agronomist confirms or modifies daily plan

Controlled by the Machine

- System sends personalized suggestions to the farmer
- System sends a notification to the agronomist when a farmer has requested help

1.2.4 Goals

G1 The app should help and improve farmers' work-related activity

G2 The app should allow agronomists to oversee and improve farmers work

G3 The app should allow policymakers to control agriculture performance in Telangana

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

User Everyone who is interested in using *DREAM*

Farmer Person who owns a farm and needs to be helped using *DREAM*

Agronomist Specialized individual who wants to help and monitor farmers through *DREAM*

Policymaker Government official who wants to supervise the agricultural sector

Forum Section of *DREAM*, accessible for farmers and agronomists, used to discuss farmers' problems

1.3.2 Acronyms

DREAM Data-dRiven prEdictive fArMing

RASD Requirement Analysis and Specification Document

HTTPS Hypertext Transfer Protocol over Secure Socket Layer

UML Unified Modeling Language

MITM Man-In-The-Middle

1.3.3 Abbreviations

[Gn] n-th goal.

[Dn] n-th domain assumption.

[Rn] n-th functional requirement.

1.4 Revision history

v. 1.0 - 23/12/2021 Initial release

1.5 Reference documents

WeBeeP channel - Project Assignment

The World & The Machine - M. Jackson, P. Zave

1.6 Document structure

This document is structured in the following five main chapters:

Chapter 1 This chapter offers a brief description of the problem and required functionalities. It also contains the list of definitions, acronyms and abbreviations that could be found in this document.

Chapter 2 The second chapter offers a summary description about the overall organization of the system (through state machine diagrams), the hardware and the software costraints. It also contains a description of all the features, the actors who used them.

Chapter 3 This section contains mockups interfaces in order to explain those mentioned in Chapter 2. It also contains a description of functional requirement through some scenarios, use cases and sequence diagrams.

Chapter 4 The fourth chapter is a formal analysis of the model, made through the Alloy, including a graphic representation of it obtained from Alloy Tool.

Chapter 5 The fifth and last chapter contains the efforts spent by each contributor.

2 Overall Description

2.1 Product perspective

2.1.1 Class diagram

This section will present the high-level UML class diagrams of the application.

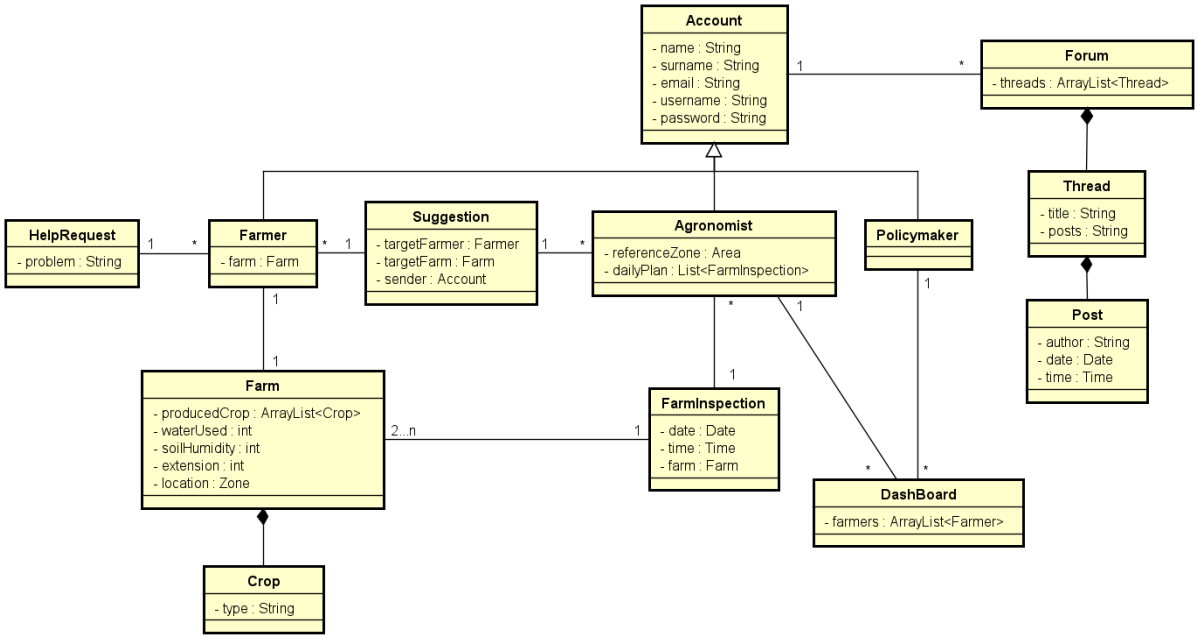

Figure 2: High level class diagram

Additional notes on the class diagram

- The agronomist attribute dailyPlan is a collection of FarmInspection ordered by date and time
- Agronomists can create a suggestion responding to a farmer help request

2.1.2 State machine diagrams

The following diagrams are meant to give a high-level description of the states' evolution during the system processes.

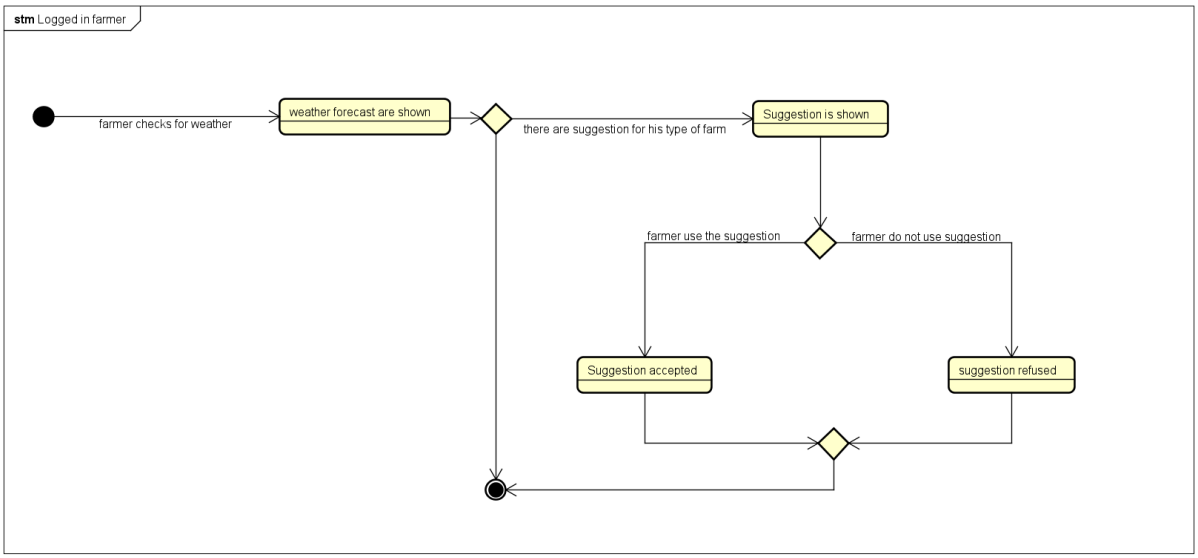

Figure 3: Statechart of a farmer checking weather forecast

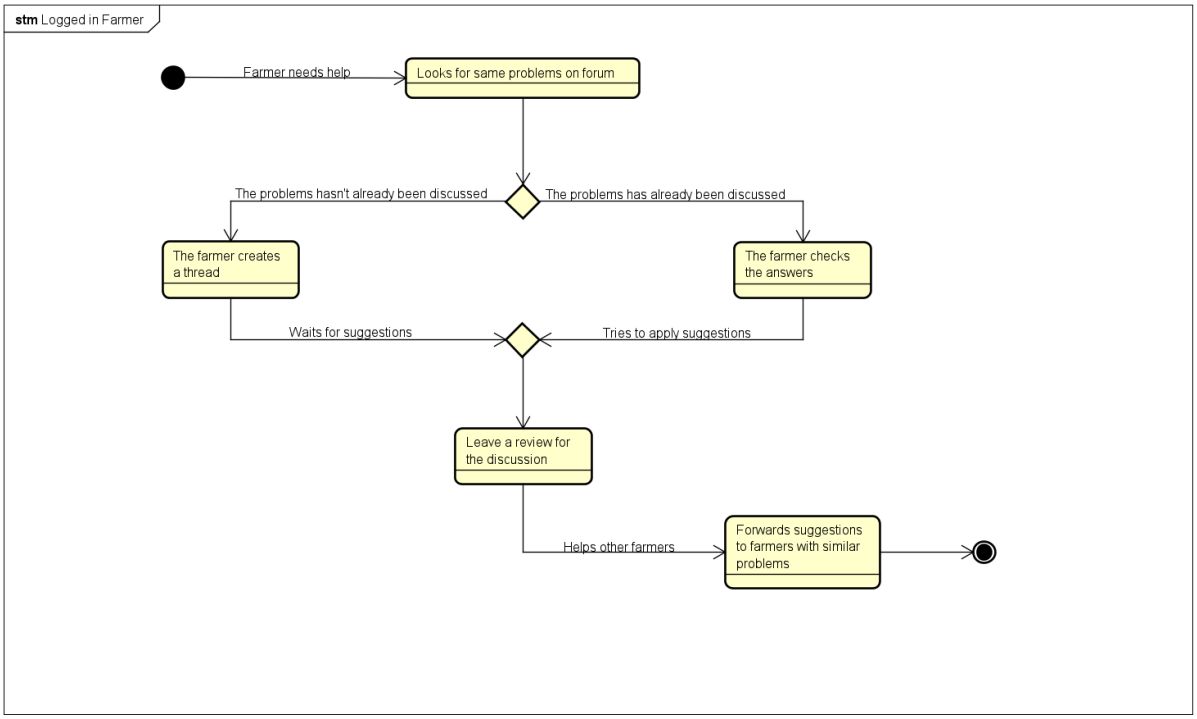

Figure 4: Statechart of the lifetime of a discussion thread on the forum

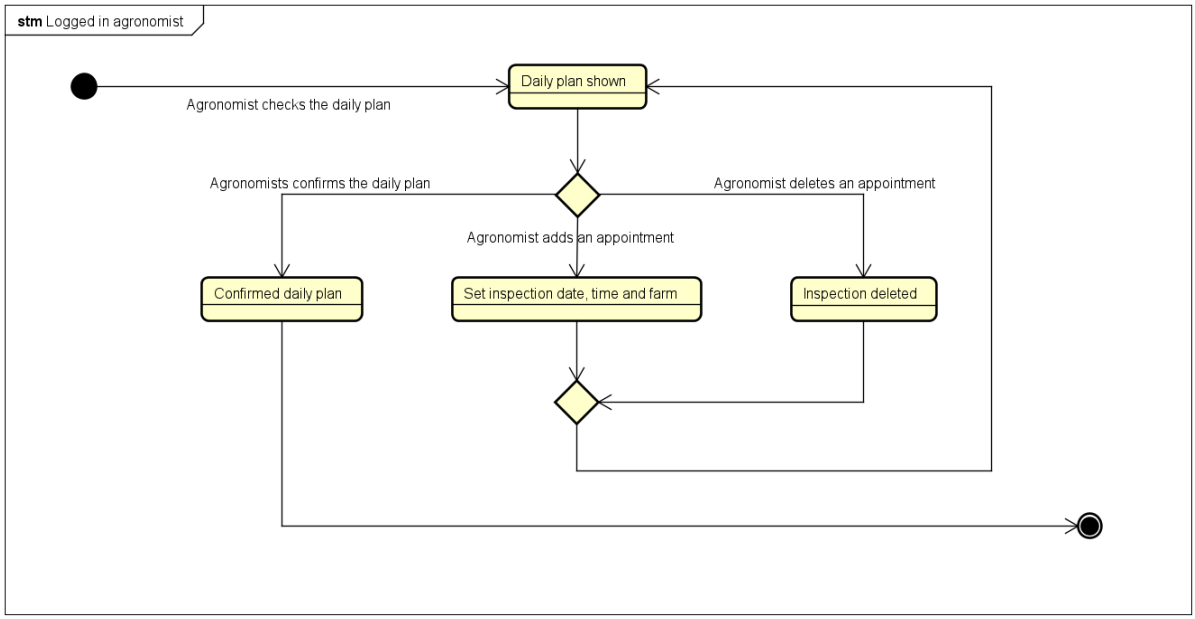

Figure 5: Statechart of an agronomist managing his daily plan

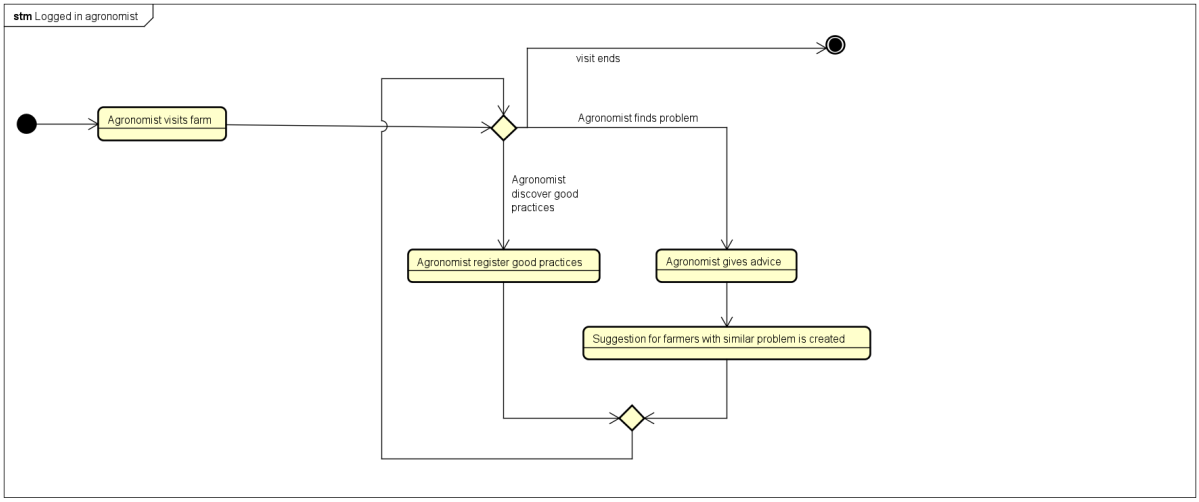

Figure 6: Statechart of the lifetime of a farm visit

2.2 Product functions

DREAM offers Farmers the ability to ask for help from other farmers or an agronomist, they can check the weather forecast and see if there are suggestions for them. *DREAM* create a leaderboard of the farmers that both agronomist and policymakers can visualize. The first ones can discover good practices and help farmers that are performing badly, policymakers instead can see the best farmer, reward them and see if the ones who accepted suggestions are performing better. In addition, *DREAM* offers agronomists the possibility of visualizing, modifying or confirming their daily plan.

Common user functions These functions are available to all Users:

- **Registration and Login**
  User should be able to create a personal account for the App using personal email and password. In this process he has to select his role.
  By logging in with their account, user can access the other functions.

Basic Farmer functions These functions are available to all logged-in Farmers:

- **Check weather forecast**
  DREAM retrieves meteorological short-term and long-term data from an external source and makes them available to farmers. They can check the forecasts on the app and if there are suggestions for their farm these are shown to them. These suggestions are created when an agronomist helps a farmer in a similar situation.

- **Request help**
  A farmer in case of a problem can ask for the direct help of an agronomist. He can use a section to send a help request, when this functionality is used a notification is sent to the agronomist responsible for the location of the farm with informations about the problem and the farm.

- **Create discussion on forum**
  DREAM has a forum section on which farmers can create a discussion specifying the type of problem and eventual specificity of their farm.

- **Reply to forum discussion** After access to the forum section of *DREAM*, the farmer can select a recent, not closed discussion and post a reply.

- **Close a discussion** A farmer after receiving an effective reply can mark the answer as effective and close the discussion.

- **Insert production data** A farmer periodically insert in the app the amount of crops produced, the type and the amount of water used.

Agronomist base functions These functions should be accessible to all logged-in Agronomists

- **Insert area he is responsible for**
  From the home page, the agronomist can open a geographical representation of Telangana and select the area he is responsible for. *DREAM* checks if the selected territory is already overseen by another agronomist. In that case, the app asks the user to repeat the process; otherwise, the selection is saved on *DREAM*'s database.

- **Reply to forum discussion**
  After access to the forum section of *DREAM*, the agronomist can select a recent, not closed discussion and post a reply.

- **Reply to help request**
  On receiving a private help request from a farmer, the agronomist receives a notification containing the sender and the problems they need to face.

- **Visualize information area**
  DREAM retrieves meteorological short-term and long-term data from an external source and makes them available to agronomists on the app.
  In addition, *DREAM* offers the possibility to the agronomist of visualizing the best performing farmers in his related area; these farmers are selected through thresholds (concerning the production and especially the resilience to meteorological adverse events) created by the agronomist of the corresponding area.

- **Daily plan management**
  From his personal area, the agronomist can visualize the daily plan on a calendar. The user has three possible choices: add or delete an appointment and confirm the plan; the first two can be repeated several times until the confirmation. In case of deletion of a visit with a given farmer, *DREAM* reinserts automatically the same appointment, according to the agronomist's availability, only if with that farmer there are less than two visits planned for this year.
  The app suggests to the agronomists the farmers that are under-performing thus need to be visited more often.
  At the end of each day, the agronomist has to confirm the execution of the daily plan or specify the deviations from that.

- **Insert good practices**
  The agronomist after the visit to a farm can add good practices that will be available to farmers with similar conditions and crops.

Basic Policy maker functions These functions are available to all logged-in Policy makers:

- **Visualize farmers' performance**
  DREAM provides policy makers with a dashboard with comprehensive data about farmers in Telangana. The dashboard shows for each area the farmers that are performing well, the ones who are performing badly, and the ones who are exceeding the thresholds by far (the last ones are those who will receive special incentives).
  In addition, there is a section that shows the result of farmers who recently received help.

2.3 User characteristics

DREAM is meant to be accessible for both specialized and unspecialized users.

Users

- **Farmer**: people that own a farm and may need help
- **Agronomist**: someone who is specialized in agriculture activity and is responsible for a given area
- **Policy maker**: people who oversight the agriculture process of Telangana

2.4 Assumptions, dependencies and constraints

D1 Each user creates only one account

D2 The information provided by the farmer is correct

D3 The thresholds provided by the agronomists are reasonable for the zone they are responsible for

D4 Most of the farmers that receive suggestions will enact them

D5 The answers to the forum's threads are consistent

D6 A privately help request to an agronomist is replied

D7 Users will be instructed to use the application if possible

D8 Agronomist is entitled to choose only his area of responsibility

D9 Each area has only one responsible agronomist

D10 Each farmer owns only one farm

3 Specific Requirements

3.1 External interface Requirements

The *DREAM* frontend is a web application that can be accessed from web browsers, both from mobile and desktop devices. The following section will give a comprehensive description in terms of hardware, software and communication interfaces.

3.1.1 Common users interfaces

Login and Registration
When first opening the application, all the users are presented with the login page. If not already registered, it is provided a registration button. In this section they are asked to provide: first name, second name, a *username*, an *e-mail address* and a *password*. The interface also shows a small button to switch to the login page, in case the user has already signed in on another device or the session has expired.

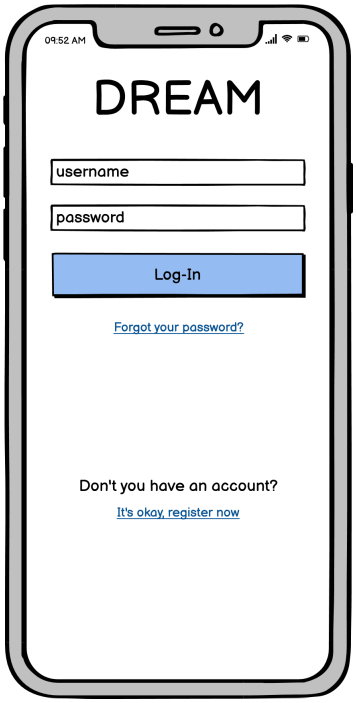

Figure 7: Log-in interface

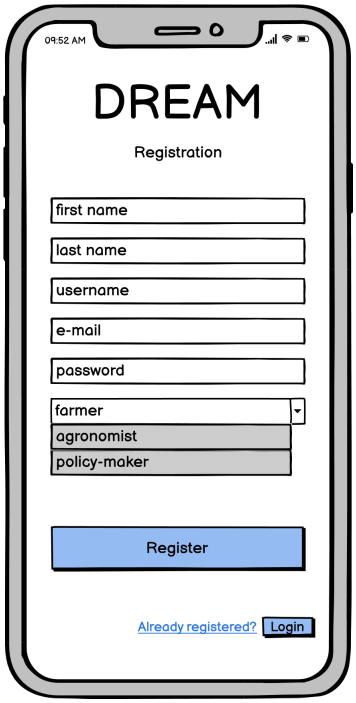

Figure 8: Registration interface

Farmer's home page

Provides different options. The user can insert data concerning production, checks weather forecasts, privately request help, and reach the forum.

Figure 9: Farmer's home page

Agronomist's home page

Provides the possibility of selecting the area of responsibility, answering private help requests, checking the weather, accessing the forum, managing the daily plan, and checking the dashboard

Figure 10: Agronomist's home page

Policymaker's home page

Provides a table containing values for all farmers in the system. Every field of this table can be ordered. There are also graphs that sum up data about production.

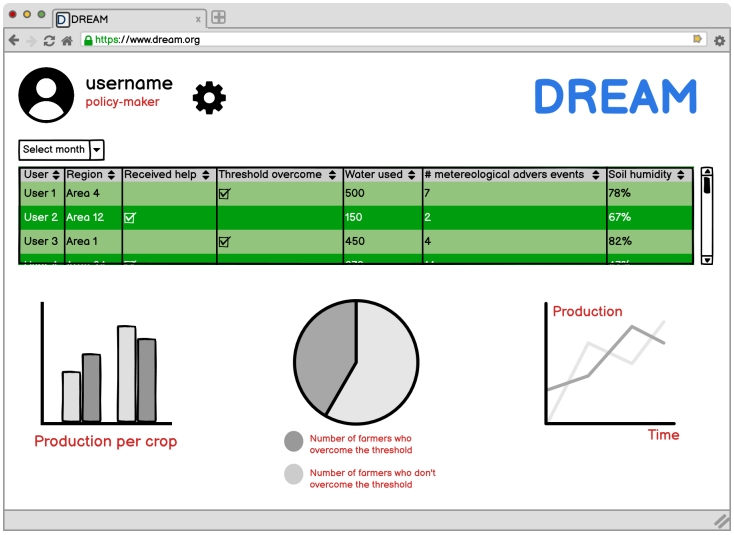

Figure 11: Policymaker's home page

Forum - home page

The forum section is accessible to both the farmers and the agronomists. Agronomists have the possibility of taking a look at the requests and answer to them. Farmers, as agronomists, can answer to the forum threads but can also create a new discussion.

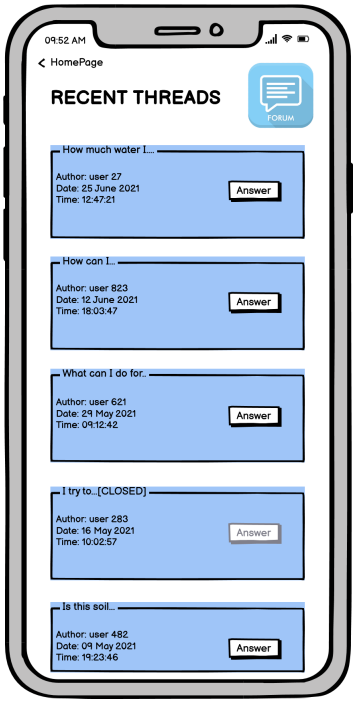

Figure 12: Agronomist's forum home page

Figure 13: Farmer's forum home page

Daily plan - home page

The daily plan home page presents a calendar in which appointments day are marked in red. By clicking on a date is possible to manage it, by adding or removing an appointment.

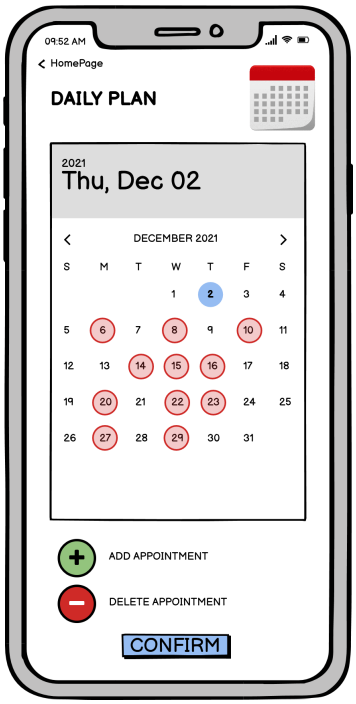

Figure 14: Daily plan management home page

3.2 Functional Requirements

3.2.1 Goal-Requirement mapping

G1 The app should help and improve farmers' work-related activity

- **R1** The system shall keep track of the farmer who requested help and of the suggestion that they received
- **R2** The system shall keep track of historical data for each zone
- **R3** The system shall keep track of meteorological events
- **R4** Farmers shall be able to create a help request for the agronomist specifying the type of problem
- **R5** The system shall be able to record data about production inserted by the farmer
- **R7** Farmers shall be able to create discussion threads on the forum
- **R8** Farmers shall be able to reply to discussion on the forum
- **R9** The system shall send a notification to the Agronomists when a farmer has requested help
- **R10** After an agronomist has registered a best practice or a suggestion to a farmer, the system should infer the farmers with similar characteristics and send them the same suggestion
- **R11** The systems shall be able to connect to an external service to retrieve meteorological forecasts
- **R12** The system shall give Farmers the possibility to see weather forecasts
- **R13** The system shall be able to manage a forum

R14 The system shall be able to record data about visits of agronomists

R17 The system shall give Farmers the possibility to register

R18 The system shall give Farmers the possibility to login

R19 The system shall give the Farmers the possibilty to visualize personalized suggestion

R20 Agronomists shall be able to reply to discussion on the forum

R27 Agronomists shall be able to insert best practices discovered during inspections

R31 The system shall be able to localize and keep track of each farm

G2 The app should allow agronomists to oversee and improve farmers work

R1 The system shall keep track of the farmer who requested help and of the suggestion that they received

R2 The system shall keep track of historical data for each zone

R3 The system shall keep track of meteorological events

R4 Farmers shall be able to create a help request for the agronomist specifying the type of problem

R5 The system shall be able to record data about production inserted by the farmer

R6 Agronomist shall be able to insert significant prodution thresholds for each zone

R7 Farmers shall be able to create discussion threads on the forum

R9 The system shall send a notification to the Agronomists when a farmer has requested help

R10 After an agronomist has registered a best practice or a suggestion to a farmer, the system should infer the farmers with similar characteristics and send them the same suggestion

R11 The systems shall be able to connect to an external service to retrieve meteorological forecasts

R13 The system shall be able to manage a forum

R14 The system shall be able to record data about visits of agronomists

R15 The system shall be able to manage a calendar

R16 The system shall organize Agronomists' calendar in such a way that they visit each farm at least twice a year

R19 The system shall give the Farmers the possibilty to visualize personalized suggestion

R20 Agronomists shall be able to reply to discussion on the forum

R21 The system shall give Agronomists the possibility to see weather forecasts

R22 Agronomists shall be able to add or remove an appointment from daily plan

R23 At the end of each day Agronomists shall be able to confirm or communicate deviations from the daily plan

R24 The system shall give Agronomists the possibility to register

R25 The system shall give Agronomists the possibility to login

R26 Agronomists shall be able to visualize farmers' performance through the dashboard

R27 Agronomists shall be able to insert best practices discovered during inspections

R28 Agronomists shall be able to visualize disussion on the forum

R29 The system shall be able to keep track of all the areas and the agronomist responsible for them

R30 The system shall give Agronomists the possibility to know every farmer for which he is responsible

G3 The app should allow policymakers to control agriculture performance in Telangana

- **R1** The system shall keep track of the farmer who requested help and of the suggestion that they received
- **R2** The system shall keep track of historical data for each zone
- **R3** The system shall keep track of meteorological events
- **R5** The system shall be able to record data about production inserted by the farmer
- **R6** Agronomist shall be able to insert significant prodution thresholds for each zone
- **R14** The system shall be able to record data about visits of agronomists
- **R31** The system shall be able to localize and keep track of each farm
- **R32** The system shall give Policymakers the possibility to register
- **R33** The system shall give Policymakers the possibility to login
- **R34** Policymaker shall be able to visualize farmers' performance through the dashboard
- **R35** Policymaker shall be able to visulize data inserted by farmers and agronomists
- **R36** Policymaker shall be able to visualize comparative performance of the farmers who received help
- **R37** The system shall give policymakers the ability to identify well-performing and underperforming farmers

3.2.2 Scenarios

Scenario 1 Daily plan confirmation
Tom is an agronomist that has just finished his work shift, so he logs in to the app and clicks on the daily plan icon. The daily appointment with farmers just completed, follows exactly the plan previously determined, so he presses the confirmation button.

Scenario 2 Daily plan modification
Martha is an agronomist that is planning his appointments with farmers. She logs in to DREAM and, from the home page, opens the daily plan section; here she decided to remove an appointment planned for the 10th of December selecting the corresponding date and then the deletion icon that updates the daily plan. After that, Martha, from the calendar, selects the 28th of December, and then she presses on the addition icon below: doing that another section is opened in which she can insert information about the appointment to plan (date, time, and the farm that she wants to visit). Confirming the operation, she will come back to the daily plan section and the daily plan is updated.

Scenario 3 Agronomist's area selection
Marvin is an agronomist that has just completed the registration. After the registration and the log in, he decides to select his area of responsibility from the map icon on the home page. In this area, a political map of Telangana is shown and Marvin selects his responsibility zone. No errors occur so Marvin can select the confirmation button and his area is stored.

Scenario 4 Forum discussion creation
Anthony is a farmer and has problems with soil humidity in his fields, so he decides to post a question on the forum. In order to do that, he logs in and clicks on the forum icon. In the forum section, he selects the add discussion icon: another section is opened in which he writes his question and post it.

Scenario 5 Farmer insert data
Jack is a farmer who recently finished collecting his harvest. He opens the app, logs in to the system. On his home page, he selects the Insert Data icon, he now can insert the amount and the type of crop he has harvested and then confirm.

Scenario 6 Farmer Help Request
Marty is a farmer who is having some problems in the cultivation of a crop and because he can't wait for a response on the forum decides to request help from an agronomist. He logs in on the app, selects the request help icon, inserts a brief description of the problem and then confirms that he wants to ask for the help of an agronomist. Victor is one of the agronomists responsible for the zone of Marty. After Marty's request, he receives a message containing a description of the problem. Victor conveys that the problem of Marty is a serious one so he decides to schedule a visit to his farm through the app functionality.

Scenario 7 Policymaker checks performance of farmer who received help
Tom is an official of the Telangana Department of Agriculture, after the recent flood he wants to know how many farmers have requested help and if this help has been effective. He logs in on the website of DREAM and can see the dashboard with some aggregate data. To see which farmers requested help he can select the previous month from the choice box and check which user had requested help, then he can see if in the current month the users who received help have overcome the thresholds set by the agronomist. He then wants to know who are the best performing farmers so he orders the table to show the users who have the greatest number of meteorological events but overcome the threshold nevertheless.

3.2.3 Use cases

Name	User creates an account
Actor	User
Entry condition	• The User has not created an account yet • The User has to be a farmer, an agronomist or a policy maker
Event flow	1. The User opens the app 2. The app asks for login or registration 3. The User clicks on *"Register now"* 4. The User inserts his first name, last name, username, e-mail, password and role 5. The User clicks on *"Register"* 6. The app notifies to User that the registration is confirmed
Exit condition	• User account is added to the database • The User can login
Exceptions	• The User already exists

Name	User logs in
Actor	Registered User
Entry condition	• The User has an account
Event flow	1. The User opens the app 2. The app asks for login or registration 3. The User inserts their username and password 4. The User clicks on *"Log-In"* 5. The User receives confirmation from the server 6. The Customer is brought to the home page
Exit condition	• The User is logged in • The User can use other features
Exceptions	• The User insert wrong credentials • The insert username doesn't exist in the app database

Name	User checks weather forecasts and suggestions
Actor	Agronomist and farmer logged-in
Entry condition	• The User has logged in
Event flow	1. The User presses on *"Weather forecasts"* icon 2. The User is brought to the *"Weather forecasts"* page 3. The User inserts the interested area and day 4. The User visualizes the weather forecasts 5. The User visualizes personalized suggestions 6. The User return to the homepage
Exit condition	• The User can use other features
Exceptions	• The inserted area doesn't exist

Name	User answers forum discussions
Actor	Agronomist and farmer logged-in
Entry condition	• The User has logged in
Event flow	1. The User presses on *"Forum"* icon 2. The User is brought to the *"Forum"* page 3. The User scrolls the page and presses on an open discussion 4. The User answers using his knowledge 5. The app creates an answer with the creator username, the date and the time 6. The user can choose to answer another discussion or return to the homepage
Exit condition	• The User can use other features
Exceptions	• The answer is empty

Name	Farmer asks for help
Actor	Farmer logged-in
Entry condition	• The Farmer has logged in
Event flow	1. The Farmer presses on *"Help request"* icon 2. The Farmer is brought to the *"Help request"* page 3. The Farmer inserts a brief description of the problem 4. The Farmer specify the area in which his farm is located 5. The Farmer clicks on *"Send request"* 6. The app forwards the request to the agronomist responsible for that area 7. The app notify to the Farmer that the request has been sent 8. The Farmer is brought to the homepage
Exit condition	• The Farmer can use other features
Exceptions	• The request is empty • There is no agronomist responsible for that area

Name	Agronomist selects the area he is responsible for
Actor	Agronomist logged-in
Entry condition	• The Agronomist has logged in
Event flow	1. The Agronomist press on *"Select your area"* icon 2. The Agronomist is brought to the *"Select your area"* page 3. The app opens a map of Telangana's State divided by areas 4. The Agronomist press on the area he is responsible for 5. The app notify to the Agronomist that the operation was successful 6. The Agronomist is brought to the homepage
Exit condition	• The Agronomist can use other features • The selected area is added to the database
Exceptions	• The selected area has already been taken

Name	Users logs out
Actor	Users logged-in
Entry condition	• The User has logged in
Event flow	1. The User press on *"Settings"* icon 2. The User is brought to the *"Settings"* page 3. The User press on *"Log-out"* button 4. The User is brought to the *"Log-in"* page
Exit condition	• The User is logged out

Name	Agronomist adds appointment
Actor	Agronomist
Entry condition	• The Agronomist is logged in
Event flow	1. The Agronomist opens the *"Daily Plan"* section 2. The Agronomist is brought to a page with a calendar 3. The Agronomist selects the day that we wants to modify 4. The Agronomist presses the *Add appointment* button 5. The Agronomist inserts additional information about the visit 6. The Agronomist presses the *Confirm* button 7. The Agronomist receives confirmation from the server
Exit condition	• The new event is added to the calendar of the agronomist • Agronomist can see the new event on the calendar
Exceptions	• The day is already full

Name	Farmer creates forum discussion
Actor	Farmer
Entry condition	• The Farmer is logged in
Event flow	1. The Farmer opens the *"Forum"* section 2. The Farmer is brought to the *"Forum"* page 3. The Farmer presses the *Create Discussion* button 4. The Farmer inserts the question and additional information 5. The Farmer presses the *Confirm* button 6. The Farmer receives confirmation from the server
Exit condition	• The new discussion is added to the forum • The Farmer can see his question on the forum
Exceptions	• The question is empty

Name	Farmer inserts data
Actor	Farmer
Entry condition	• The Farmer is logged in
Event flow	1. The Farmer opens the *Insert Data* section 2. The Farmer is brought to the *Insert Data Form* 3. The Farmer compiles the form inserting the type of crop the amount of crop and the amount of water used 4. The Farmer presses the *Confirm* button 5. The Farmer receives confirmation from the server
Exit condition	• The new production data is added to the database
Exceptions	• The Farmer inserted invalid data

Name	Agronomist inserts threshold
Actor	Agronomist
Entry condition	• The Agronomist is logged in
Event flow	1. The Agronomist opens the *Dashboard* section 2. The Agronomist is brought to the *Dashboard* page 3. The Agronomist presses on the *insert threshold* button 4. The Agronomist is brought to the *Insert Threshold form* 5. The Agonomist insert the threshold 6. The Agronomist receives confirmation from the server
Exit condition	• The new threshold is added to the database
Exceptions	• The Agronomist inserted invalid data

Name	Policymaker sees dashboard
Actor	Policymaker
Entry condition	• The Policymaker is logged in
Event flow	1. The Policymaker opens the *Dashboard* section 2. The Policymaker is brought to the *Dashboard* page 3. The Policymaker presses on the *water used column* of the table to order the farmer by water used 4. The Policymaker see the table and decides the farmer to reward
Exceptions	• The Table is empty

Name	Agronomist replies to help request
Actor	Agronomist
Entry condition	• The Agronomist is logged in
Event flow	1. The Agronomist receives an help request 2. The Agronomist open the *Help Request* section 3. The Agronomist sees a list of help requests 4. The Agronomist select the help request he want to answer 5. The Agronomist insert an answer to the problem 6. The Agronomist presses *Send* button 7. The Agronomist receives confirmation from the server
Exit condition	• The answer is sent to the farmer • The Agronomist can visualize his answer
Exceptions	• The Agronomist answer is empty

3.3 Sequence diagrams

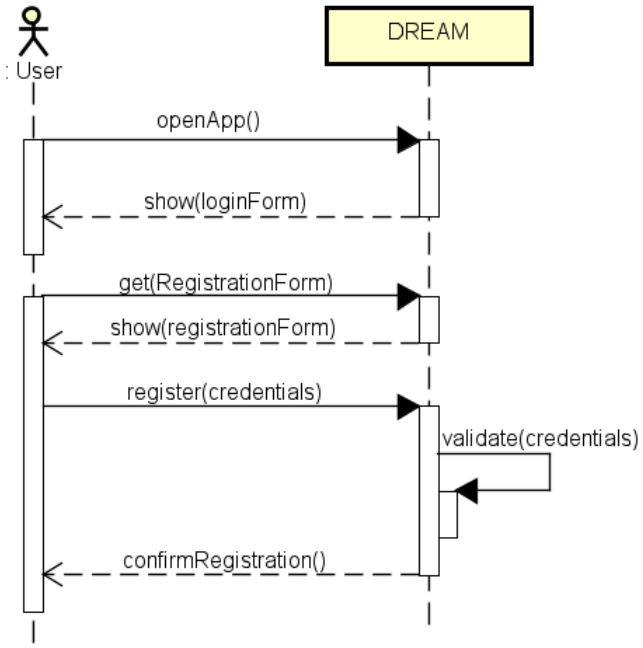

Figure 15: Registration sequence diagram

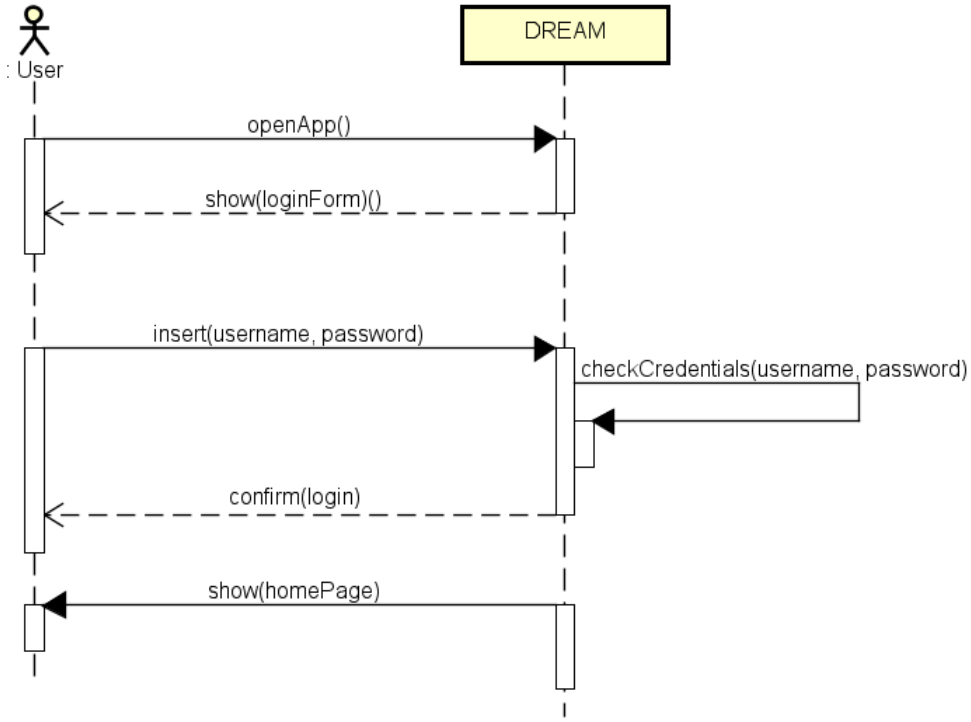

Figure 16: Log-in sequence diagram

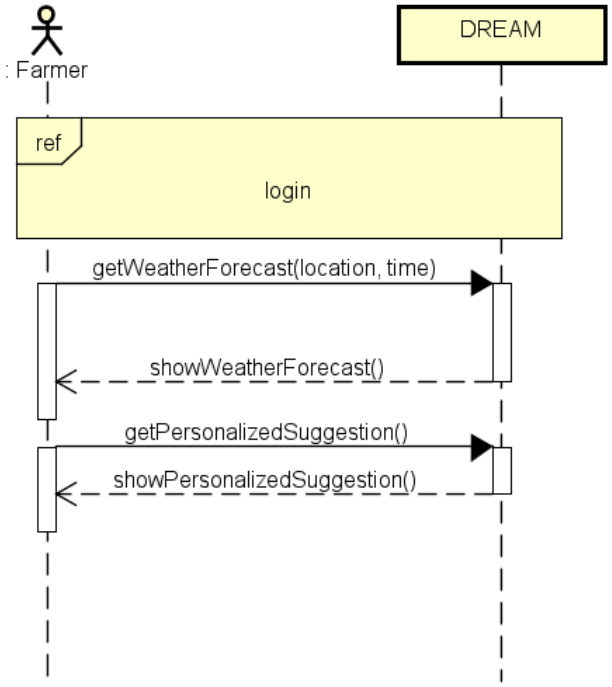

Figure 17: Sequence diagram of weather forecast check

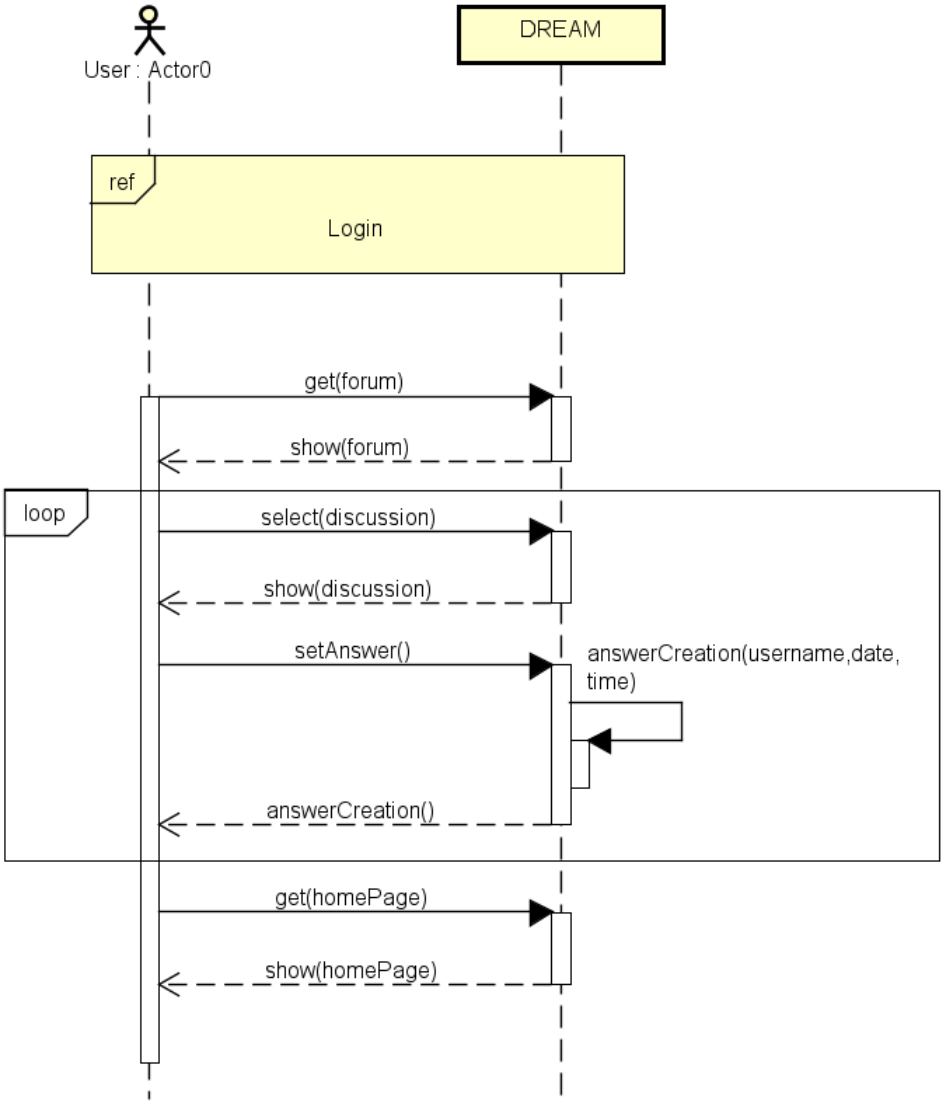

Figure 18: Sequence diagram of users' answer on the forum

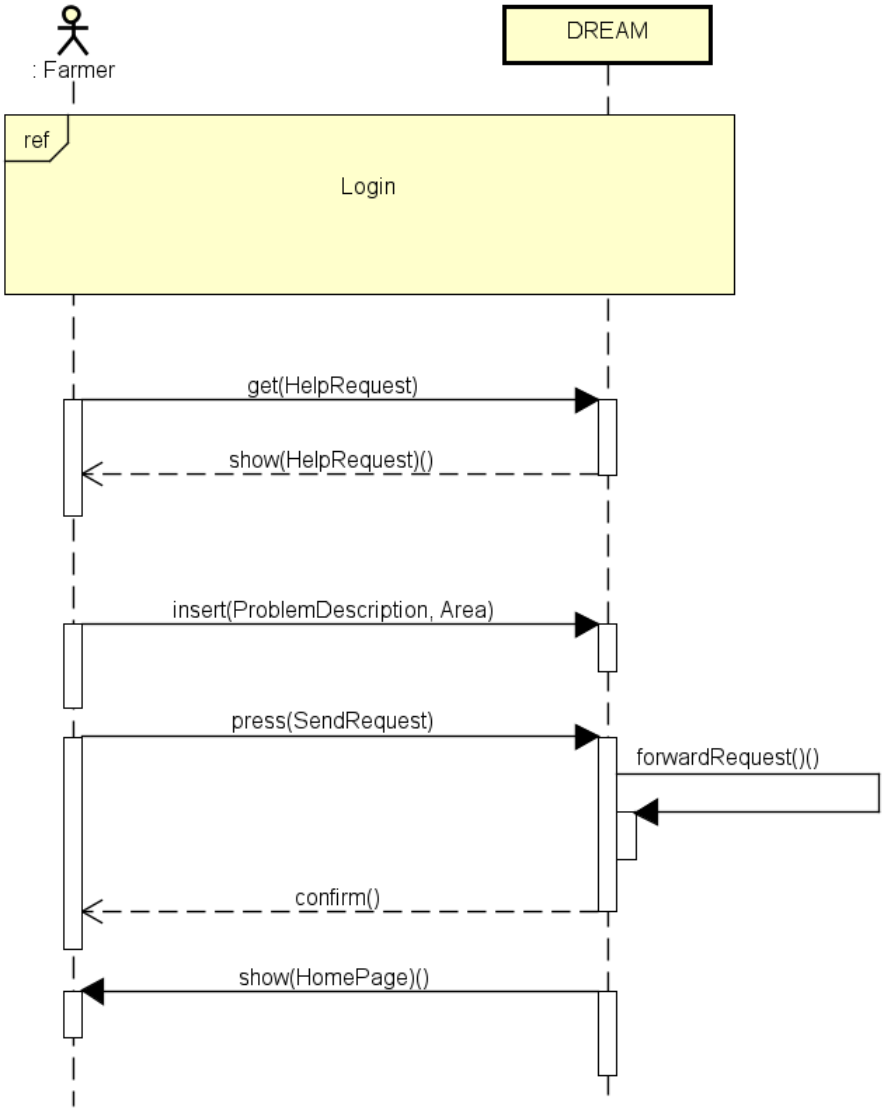

Figure 19: Farmer requesting help sequence diagram

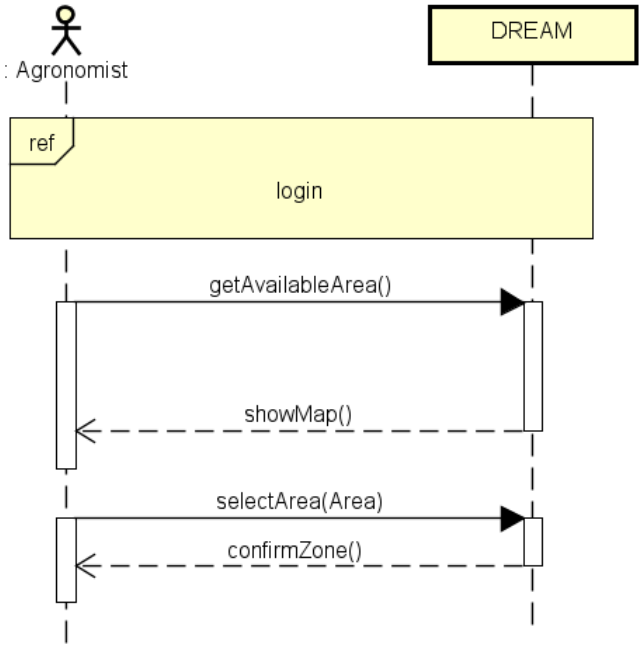

Figure 20: Sequence diagram of the area selection

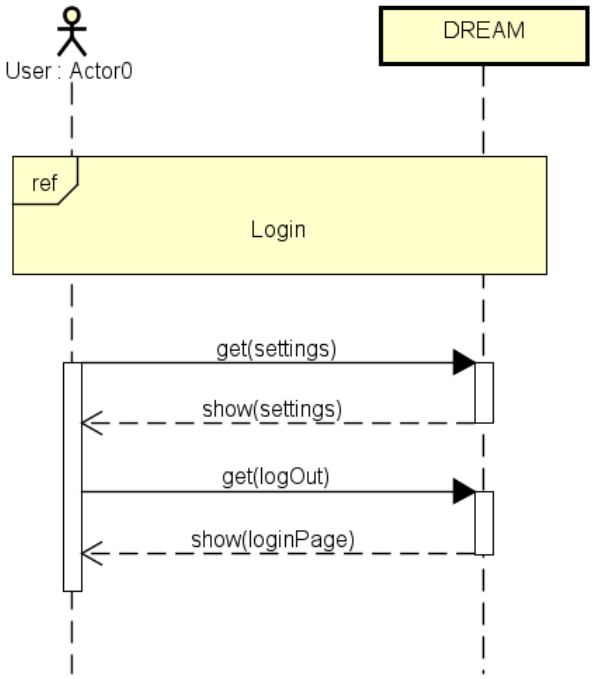

Figure 21: Sequence diagram of log out

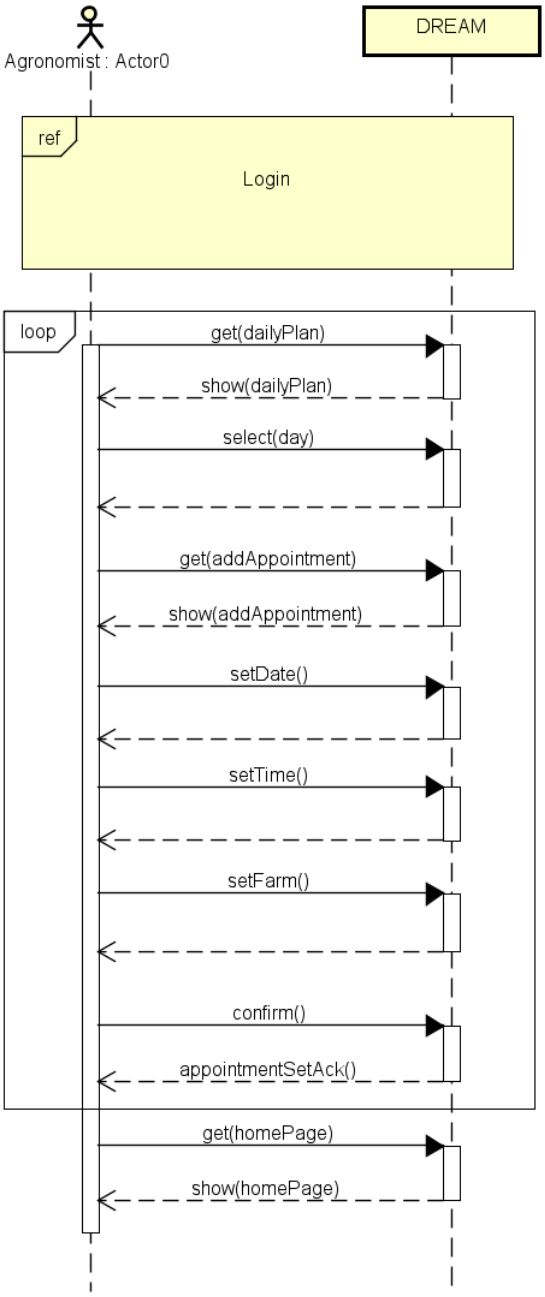

Figure 22: Sequence diagram of the addition of an appointment

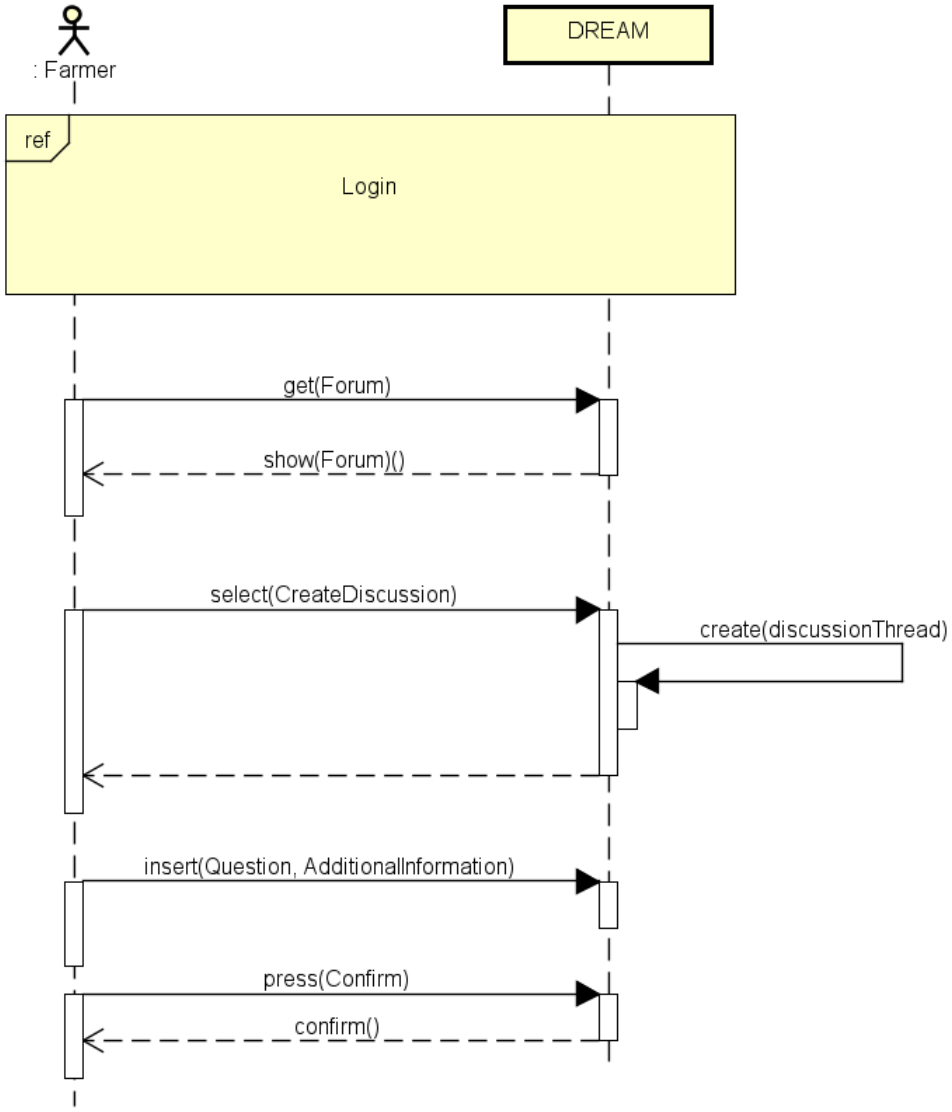

Figure 23: Sequence diagram of the creation of a discussion on the forum

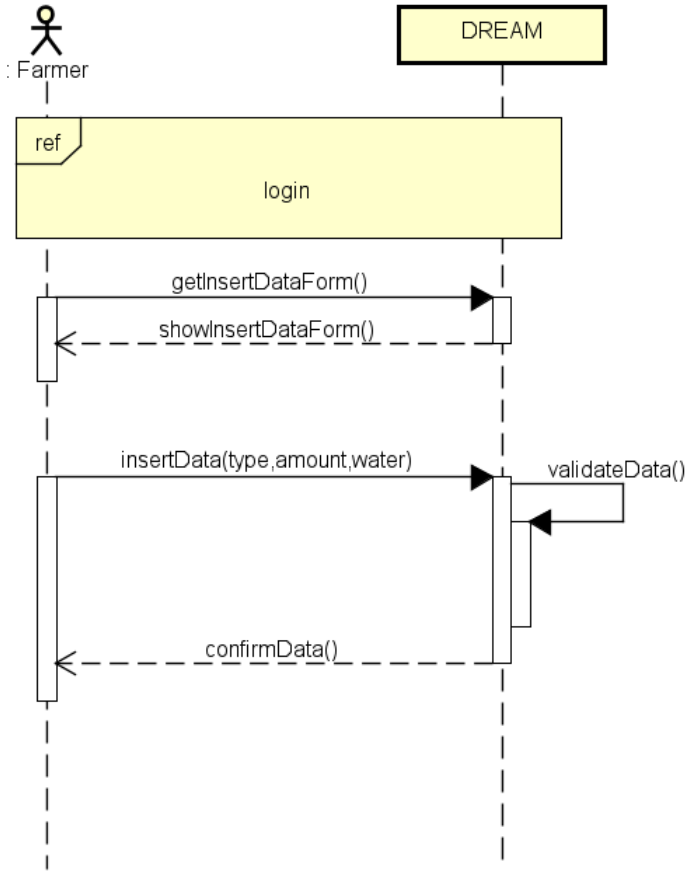

Figure 24: Farmer insert data concerning production, sequence diagram

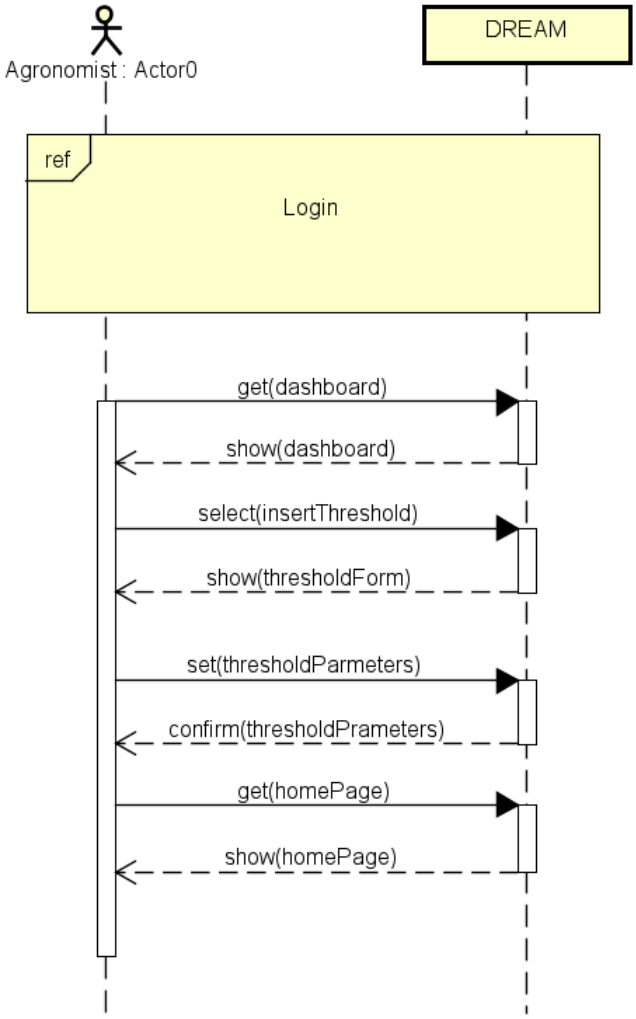

Figure 25: Sequence diagram of threshold creation by an agronomist

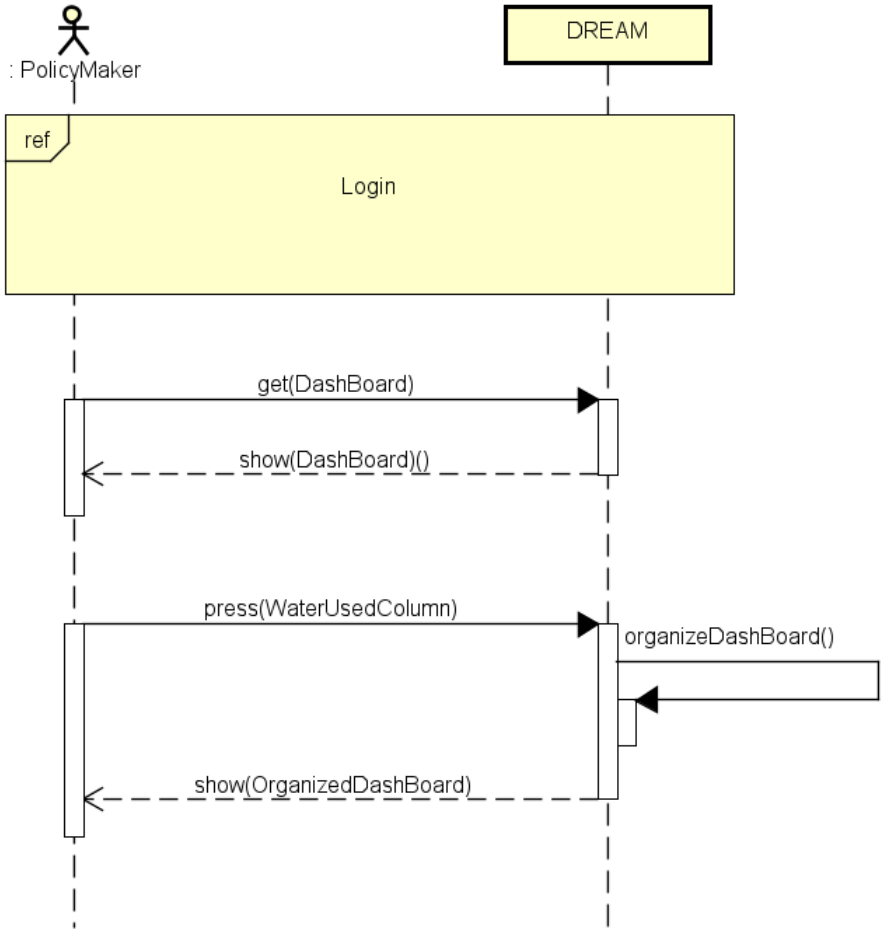

Figure 26: Sequence diagram of dashboard check by policy maker

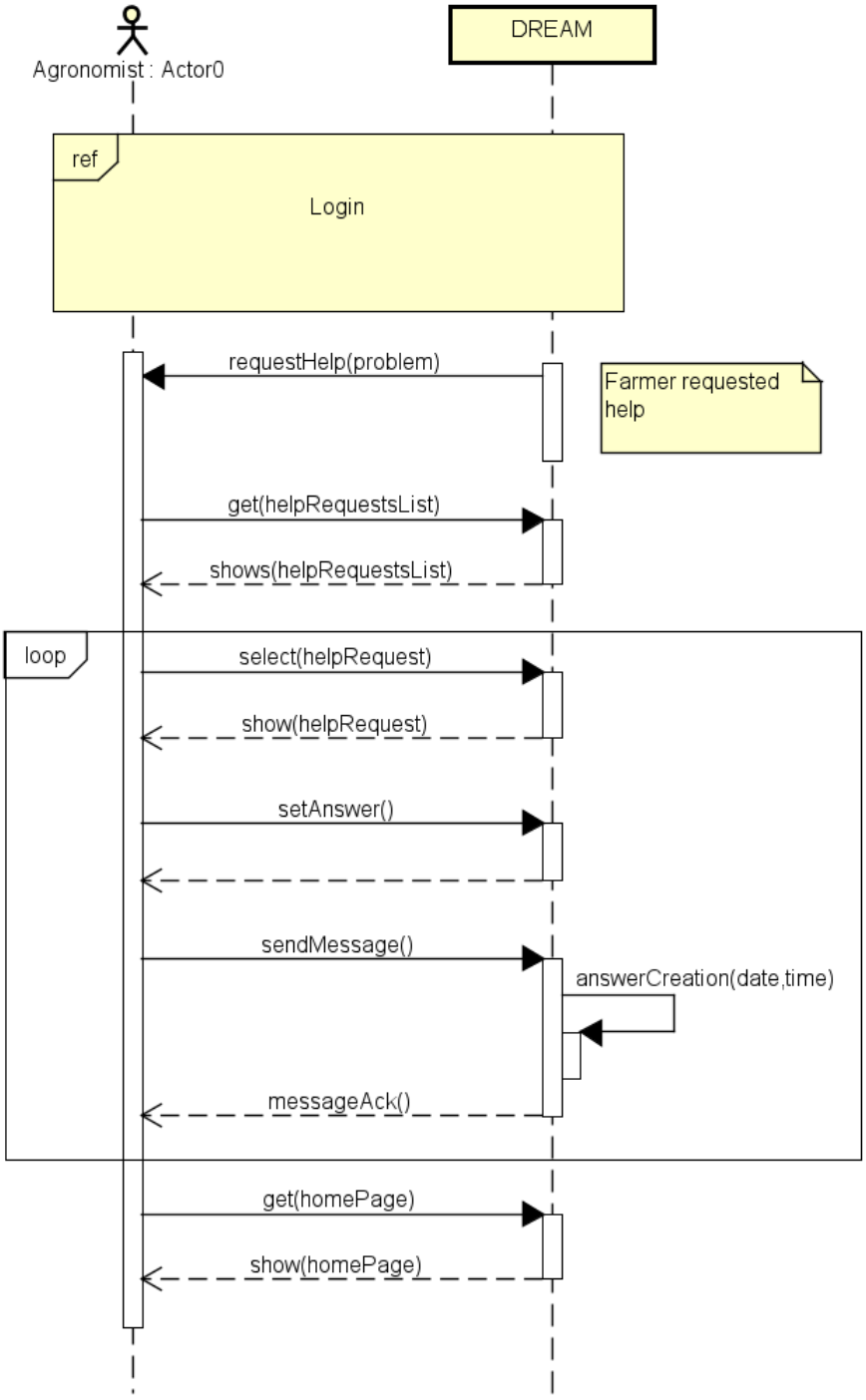

Figure 27: Sequence diagram of an agronomist replying to help requests

3.4 Performance Requirements

Though in these cases time is not critical, the requested performances are:

- Weather forecasts update must be done within 5 seconds.
- When farmer adds production data on the app, the server should be updated within 15 seconds.
- When someone answers or requests help on the forum, its post need to be add within 10 seconds.
- Daily plan modifications have to be scheduled within 0.5 seconds.
- Private help requests and answers, need to be sent to the receiver within 3 seconds.

3.5 Design constraints

3.5.1 Hardware limitations

Running *DREAM* requires a smartphone or computer and a web connection.

3.6 Software System Attributes

3.6.1 Reliability

DREAM should be available 24/7, except for maintenance that has to be performed only at night and has to last no longer than 2 hours.
The highest number of simultaneous accesses is expected in the harvest time.

3.6.2 Availability

The system must be available as much as possible; hence, during daytime is required a minimum value of 97%. During the night the availability could be lowered to 95%.

3.6.3 Security

The most critical problems concerning security are privacy, data integrity and authentication. For instance a MITM attack between policymaker and the server could be dangerous for private data. In order to overcome all these security problems, the communication between parties should be encrypted through HTTPS protocol.

3.6.4 Maintainability

The system must be designed in such a way that permits future addition of functionalities with minimum effort.

3.6.5 Portability

DREAM should be easily deployable on a dedicated machine, on a virtual private server or on a cloud. It is in any case usable on every web browser, from every device. The mobile application must be supported by iOS and Android.

4 Formal Analysis Using Alloy

5 Formal Analysis

5.1 Alloy Code

```
//SIGNATURES

abstract sig Account {
        name: one Name,
        surname: one Surname,
        username: one Username,
        email: one Email,
        password: one Password
}

sig Name {
}{
        no n : Name | n. ~name  = none
}

sig Surname {
}{
        no s : Surname  | s. ~surname = none
}

sig Email {}
{
        no disj a1, a2 : Account | a1.email = a2.email
        no e: Email | e. ~email = none
}

sig Username {}
{
        no disj a1, a2 : Account | a1.username = a2.username
        no u: Username | u.~username = none
}

sig Password {
} {
        no p: Password | p. ~password = none
}

sig Farm {
        area: one Area
} {
        #this.~inspection ≥ 2
        no f:Farm | f.~farm = none
        no f:Farm | f.~farms_dashboard = none
}

sig Post {
} {
        no p: Post | p.~posts = none
}

sig DailyPlan {
  visits: set Visit
}{
        no d:DailyPlan | d.~dailyplan = none
}

sig Visit {
        inspection: one Farm
}  {
        no v:Visit | v.~visits = none
}

one sig DashBoard {
        farms_dashboard: set Farm
```



```
}

sig Thread {
        posts: some Post,
} {
        no  t:Thread| t.~threads = none
}

one sig Forum {
        threads: set Thread,
}

sig Area {
}

sig PolicyMaker extends Account {
        dashboard: one DashBoard,
}

sig Farmer extends Account {
        farm: one Farm,
        forumF: one Forum,
        requests : set HelpRequest
}

sig Agronomist extends Account {
        forumA: one Forum,
        areas: one Area,
        dashboard: one DashBoard,
        dailyplan: one DailyPlan
}

sig HelpRequest{
        agronomist : one Agronomist
}{
        no h: HelpRequest | h. ~requests = none
}


-----------------------------------------------------------------------
//FACTS

//This fact ensures that a post is created by only one thread
fact aboutThread {
        all disj t1,t2: Thread| t1.posts & t2.posts = none
}

//This fact ensures that every farmer owns a farm
fact aboutFarms {
        no disj f1,f2: Farmer | f1.farm = f2.farm
}

//This fact contains constraints regarding agronomists:
        // - every agronomist owns a different area
        // - every agronomist has a different daily plan
        // - no visit belongs to different daily plans
fact aboutAgron {
        no disj a1, a2: Agronomist | a1.areas = a2.areas
        no disj a1, a2: Agronomist | a1.dailyplan = a2.dailyplan
        all disj d1,d2: DailyPlan | (d1.visits=none and d2.visits=none) or (d1.visits &
            ↪ d2.visits = none)
}

//This fact ensures that any farm belongs to different areas
fact aboutArea {
        all disj a1,a2: Area |  (a1.~area=none and a2.~area=none) or (a1.~area & a2.~area
            ↪  = none)
}

//This fact ensures that every daily plan owns visits to farmer belonging to the
    ↪ corresponding agronomist
fact aboutDailyPlan{
```



```
        all d:DailyPlan | (d.visits.inspection.area ≠ none) implies (d.visits.inspection.
            ↪ area = d.~dailyplan.areas)
}

//This fact contains constraints regarding help requests:
        // - every help requests can be sent only to the corresponding agronomist
        // - every farmer sends different requests
fact aboutHelpRequest{
        all h: HelpRequest | h.agronomist.areas = h. ~requests.farm.area
        all disj f1, f2: Farmer | f1.requests & f2.requests = none
}


-----------------------------------------------------------------------
//ASSERTS

assert agronomistArea {
        no disj a1, a2: Agronomist | {
                a1.areas = a2.areas
        }
}

assert helpRequest {
        all a1, a2: Agronomist, f: Farmer, disj h1, h2: HelpRequest | {
                (h1.~requests = f and h2.~requests = f and a1.~agronomist = h1 and a2.~
                    ↪ agronomist = h2) implies (a1=a2)
        }
}

assert dailyPlan {
        all a1, a2: Area, disj v1, v2: Visit | {
                (v1.inspection.area = a1 and v2.inspection.area = a2 and v1.~visits.~
                    ↪ dailyplan = v2.~visits.~dailyplan) implies (a1 = a2)
        }
}


-----------------------------------------------------------------------
//PREDICATES

//A simulation that shows the daily plan of Agronomists w/ visits to Farmer of
//their area of responsibility
pred world1 {
        #Farmer = 3
        #Agronomist = 2
        #PolicyMaker = 0
        #Thread = 0
        #Post = 0
}
//run world1 for 6

//A simulation that shows the functioning of the dashboard
pred world2{
        #PolicyMaker = 2
        #Agronomist = 2
        #Farmer = 1
        #Visit = 2
}

//A simulation that shows the different types of help given to the farmers
pred world3{
        #Farmer = 3
        #Agronomist = 3
        #PolicyMaker = 0
        #Thread = 3
}

//A simulation that shows the registration
pred world4{
        #Farmer = 1
        #PolicyMaker = 1
        #Agronomist = 1
```



```
}

run world4 for 3


check agronomistArea for 10
check helpRequest for 10
check dailyPlan for 10


run world1 for 6
run world2 for 5
run world3 for 6
run world4 for 3
```


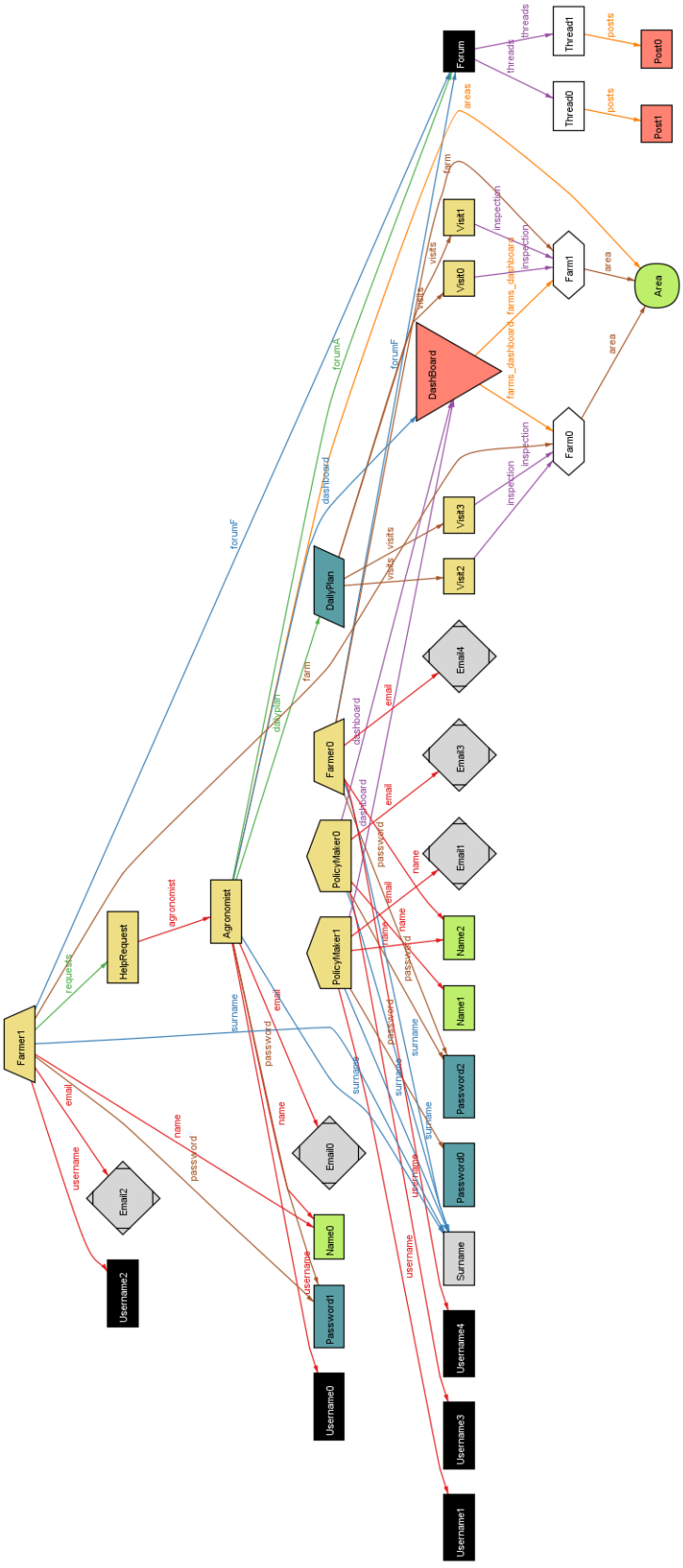

Figure 28: All the elements managed in the alloy analysis

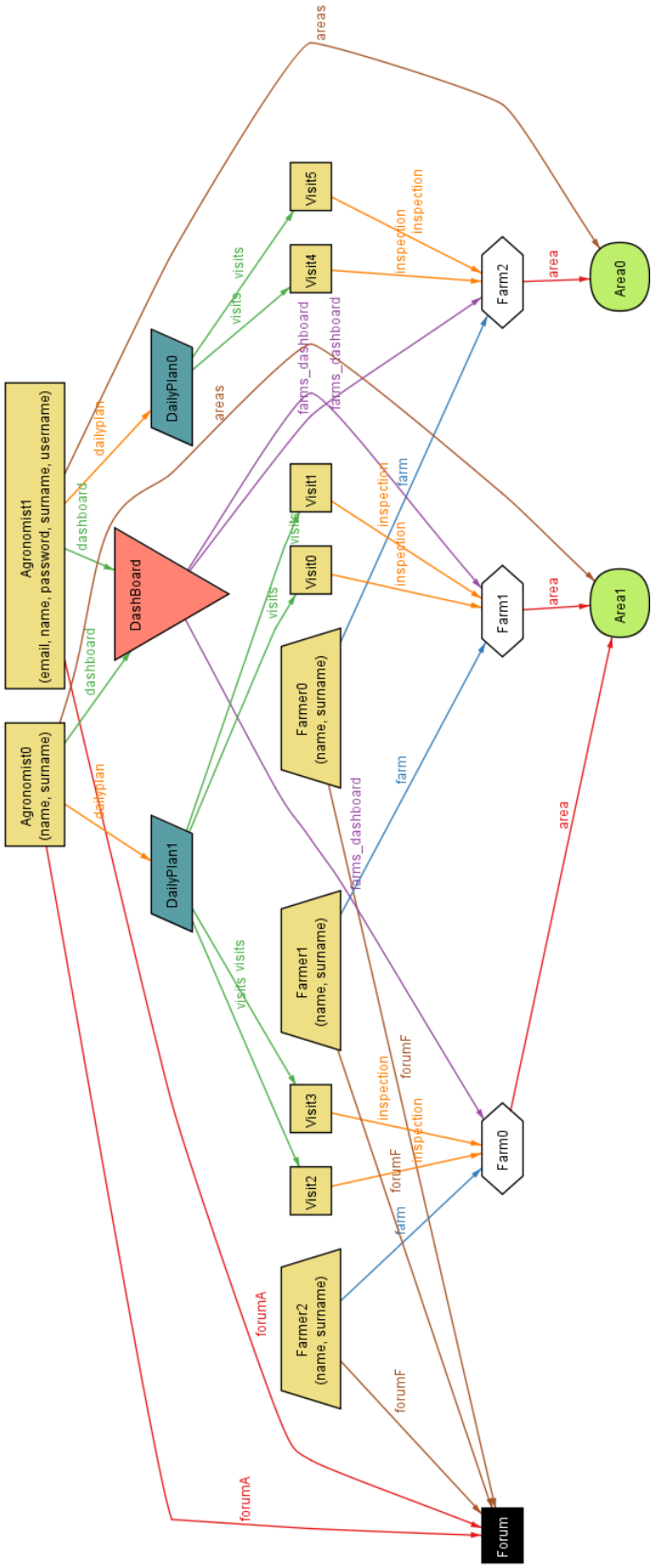

Figure 29: Representation of the execution of predicate world1

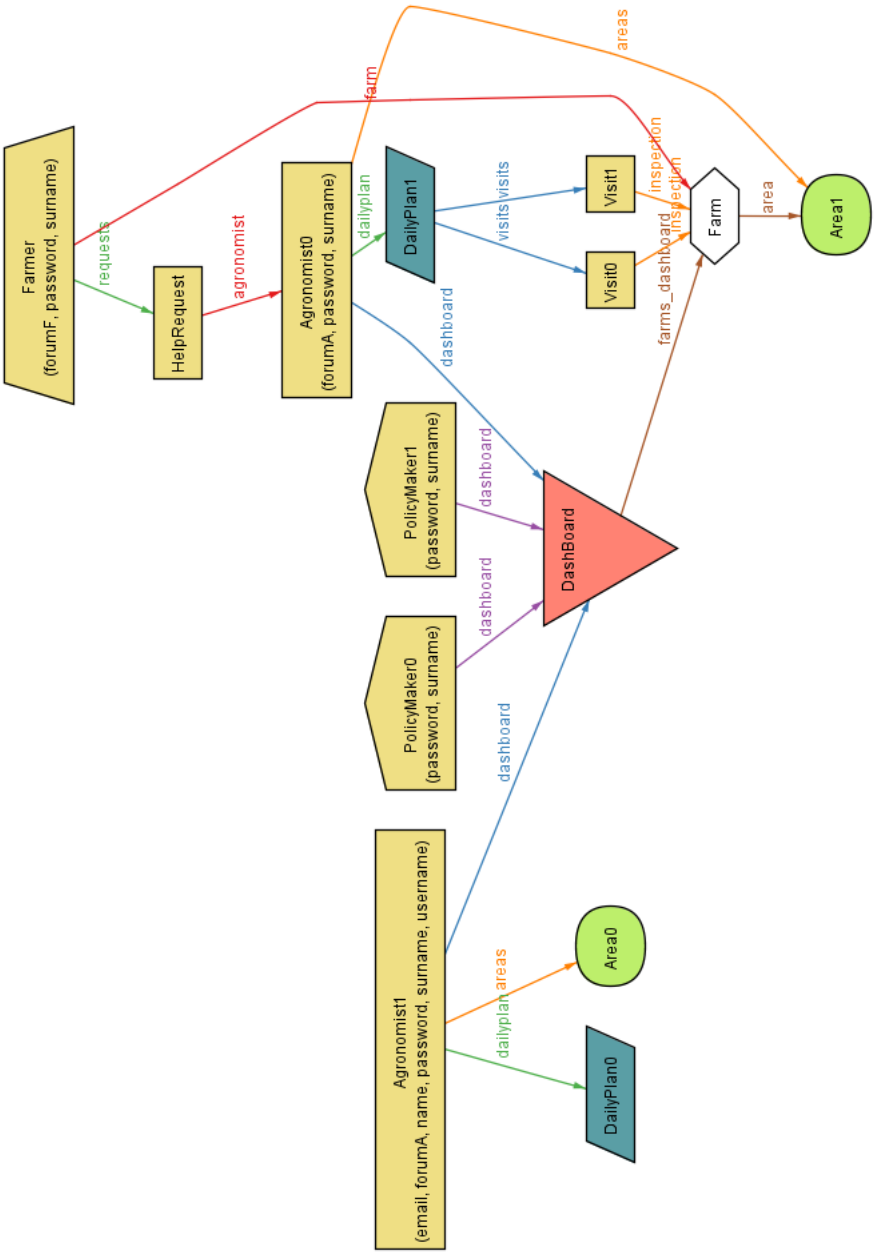

Figure 30: Representation of the execution of predicate world2

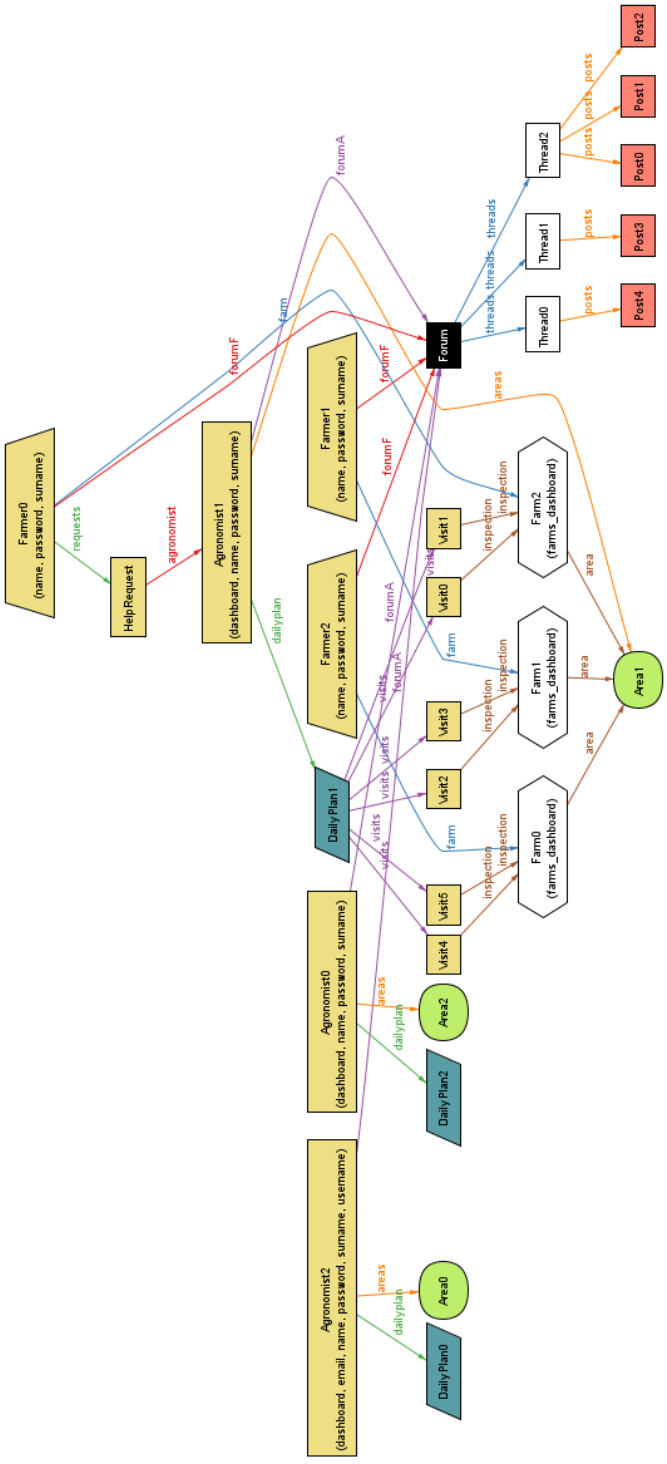

Figure 31: Representation of the execution of predicate world3

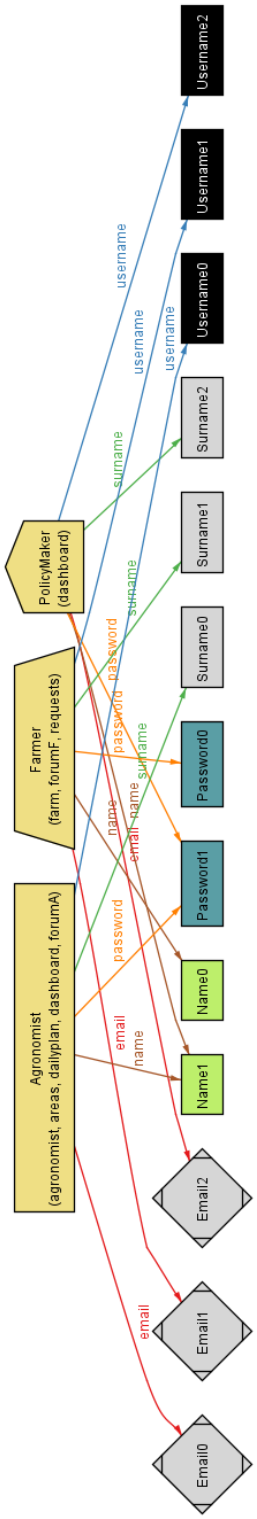

Figure 32: Representation of the execution of predicate world4

6 Effort Spent

Lorenzo Iovine

Date	Hours	Description
2021-11-29	7h	Introduction, Purpose, Scope & Goals
2021-11-30	5h	UML diagram, statecharts & World and the Machine
2021-12-01	7h	Product functions
2021-12-02	3h	Product functions, User characteristics & Assumptions
2021-12-02	4h	Mockups
2021-12-03	4h	Use cases
2021-12-04	2h	Use cases
2021-12-04	4h	Sequence diagram
2021-12-05	3h	Review
2021-12-06	5h	Review & Alloy
2021-12-07	3h	Alloy
2021-12-08	4h	Alloy
2021-12-10	5h	Alloy
	56h	

Nicola Landini

Date	Hours	Description
2021-11-29	7h	Introduction, Purpose, Scope & Goals
2021-11-30	5h	UML diagram, statecharts & World and the Machine
2021-12-01	7h	Product functions
2021-12-02	3h	Product functions, User characteristics & Assumptions
2021-12-02	4h	Mockups
2021-12-03	4h	Customer Interfaces
2021-12-04	2h	Customer Interfaces
2021-12-04	4h	Sequence diagram
2021-12-05	3h	Review
2021-12-06	5h	Review & Alloy
2021-12-07	3h	Alloy
2021-12-08	4h	Alloy
2021-12-10	5h	Alloy
	56h	

Francesco Leone

Date	Hours	Description
2021-11-29	7h	Introduction, Purpose, Scope & Goals
2021-11-30	5h	UML diagram, statecharts & World and the Machine
2021-12-01	7h	Product functions
2021-12-02	3h	Product functions, User characteristics & Assumptions
2021-12-02	4h	Requirements
2021-12-03	4h	Use cases
2021-12-04	2h	Use cases
2021-12-04	4h	Sequence diagram
2021-12-05	3h	Review
2021-12-06	5h	Review & Alloy
2021-12-07	3h	Alloy
2021-12-08	4h	Alloy
2021-12-10	5h	Alloy
	56h	

