POLITECNICO
MILANO 1863

DREAM project by Parcevaux - Pouliquen

Design Document

Deliverable:	DD
Title:	Design document
Authors:	de Parcevaux & Pouliquen
Version:	1.0
Date:	9-January-2022
Download page:	https://github.com/thekalipo/DeParcevauxPouliquen
Copyright:	Copyright © 2021, Parcevaux - Pouliquen – All rights reserved

Contents

List of Figures

1 Introduction

1.1 Purpose

As reported by the Food and Agriculture Organization of the United Nations [1], the Indian cultivated area went through a great expansion between 1970 and 2000, going from 140 million to 190 million hectares (ha). In the meantime, the number of farmholding skyrocketed, from 70.5 to 115.6 million. This is the reason why, the average size of farmholdings dropped from 2.30 to 1.41 ha.

Hence, the Indian government has to tackle many new farms with lower sizes, which may require some help or incentives.

In this contect, the United Nations Development Programme (UNDP) partnered with the state of Telangana to enhance good deviances in the state's agriculture through a data-driven approach.

- G.1: Allow farmers to get advices for optimizing their production

  G.1.1: Allow farmers to retrieve personalized suggestions if they perform poorly

  G.1.2: Allow farmers to discuss with other farmers about their issues

  G1.2.1: Allow farmers to create a discussion forum

  G1.2.2: Allow farmers to look for a specific topic among discussion forums

  G1.2.3: Allow farmers to send messages on a forum already created

  G1.2.4: Allow farmers to contact another farmer privately Allow farmers to send a specific help request

- G2: Allow farmers to get data that impact their production.

  G2.1: Allow farmers to access data about weather conditions and predictions

  G2.2: Allow farmers to access data about soil moisture

  G2.3 Allow farmers to access data about soil organic carbon

- G3: Allow policy makers to globally enhance the productivity of the farmers of their area

  G3.1: Allow policy makers to identify well and poorly performing farmers of their area, according to a chosen metric

  G3.2: Allow policy makers to incent well performing farmers

  G3.3: Allow policy makers to fetch best practices among farmers and provide them to others

  G3.4: Allow policy makers to send personalized suggestions to poorly performing farmers

1.2 Scope

1.2.1 World Phenomena

- Farmers seed their crops
- Farmers fertilize their crops
- Farmers measure the amount of fertilizer used for a specific cropping
- Farmers harvest their crops
- Farmers measure their production
- Water consumption figures are updated
- Soil moisture figures are updated

- Vegetation index figures are updated
- Rainfall conditions are updated
- Rainfall previsions are updated
- Global Positioning System (GPS) gets the farmer location

1.2.2 Shared Phenomena

- Farmer releases production data
- Farmer enters the fertilizers used
- Farmer enters the seed variety used
- Farmer enters the seed rate of the crop
- Farmer releases the amount of fertilizer used for cropping
- Farmer enters the start and end dates
- Farmer receives special incentive
- Farmer receives a request of best practices
- Farmer provides best practices
- Farmer visualizes the weather forecasts
- Farmer requests for help
- Farmer creates discussion forum
- Farmer searches for a discussion forum on a specific topic
- Farmer sends a message in a discussion forum
- Farmer sends a message to another farmer
- Farmer registers and provides personal data (mail, name)
- Farmer logs in
- Farmer provides exploitation data (location, type of production)
- Policy Maker registers and provides personal data (mail, name)
- Policy Maker logs in
- Policy Maker provides area he is responsible of
- Policy Maker asks for a ranking of the farmers he is in charge of, based on some metric
- Policy Maker searchs for a discussion forum on a specific topic
- Policy Maker sends a message to a farmer
- DREAM displays a notification to farmer for production release
- DREAM displays a notification to farmer for lack of soil or weather data

- DREAM displays a notification to farmer for new message from another farmer
- DREAM displays a notification to farmer for new message from a policy maker
- DREAM displays a notification to farmer for a suggestion from a policy maker
- DREAM displays a notification to farmer for help request proposal
- DREAM displays a notification to farmer for best practice
- DREAM displays a notification to farmer for e-voucher
- DREAM displays a notification to policy maker for help request from a farmer
- DREAM displays a notification to policy maker for new registration in his/her area
- DREAM displays a notification to policy maker for completion of production data

1.2.3 Machine Phenomena

- Identifies best performing farmers with regard to meteorological events
- Identifies worst performing farmers with regard to meteorological events
- Compute personalized suggestions concerning crops and fertilizers
- Fetchs weather forecasts
- Fetchs data from water irrigation system
- Fetchs user's location

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- release
- batch on a season, they are many release.
- release period
  - Rabi season
  - Kharif season
- analysis period
- help request
- incentive (e-voucher for us)
- vegetation index

1.3.2 Acronyms

- RASD: Requirement Analysis and Specification Document
- DREAM: Data-dRiven PrEdictive FArMing in Telengana
- GPS: Global Positioning System
- FAO: Food and Agriculture Organization
- UNDP: United Nations Development Programme
- UML: United Modeling Language
- PM: Policy Maker
- SQL: Simple Query Language
- NICES: National Inforamtion System for Climate and Environment Studies
- API: Application Programming Interface
- UI: User Interface
- UX diagram: User experience diagram

1.3.3 Abbreviation

- Gn: n^{th} Goal
- Rn: n^{th} Requirement
- PM: Policy Maker

1.4 Revision history

1.5 Reference Documents

References

[1] Food and Agriculture Organization of the United Nations. Fertilizer use by crop in india. https://www.fao.org/3/a0257e/A0257E02.htm#ch1, 2005.

1.6 Document Structure

2 Architectural Design

2.1 Overview: high-level components and their interaction

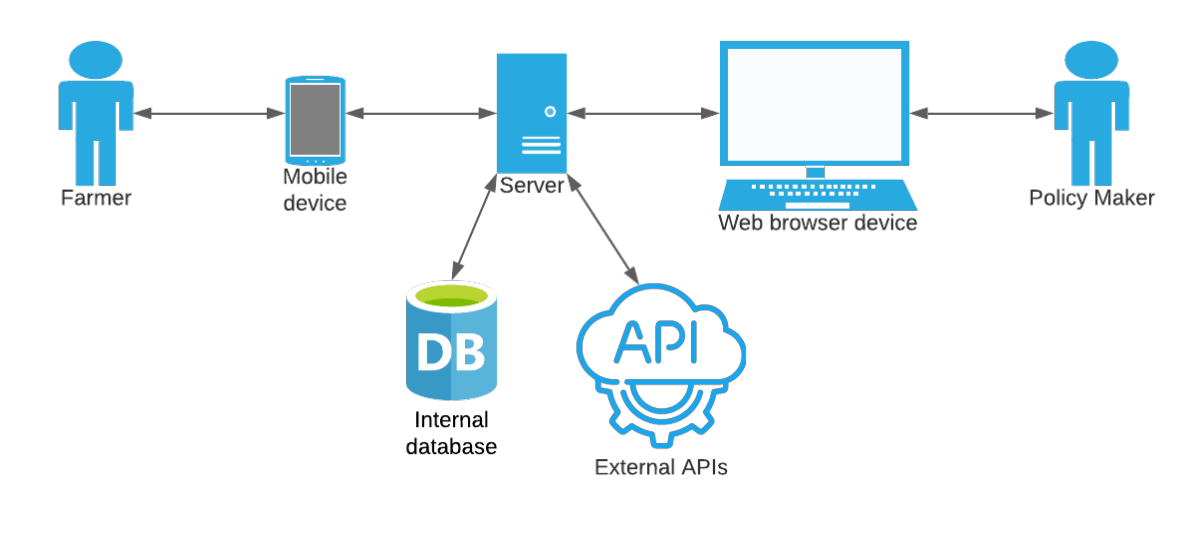


Figure 1: Physical architecture diagram

2.2 Component view

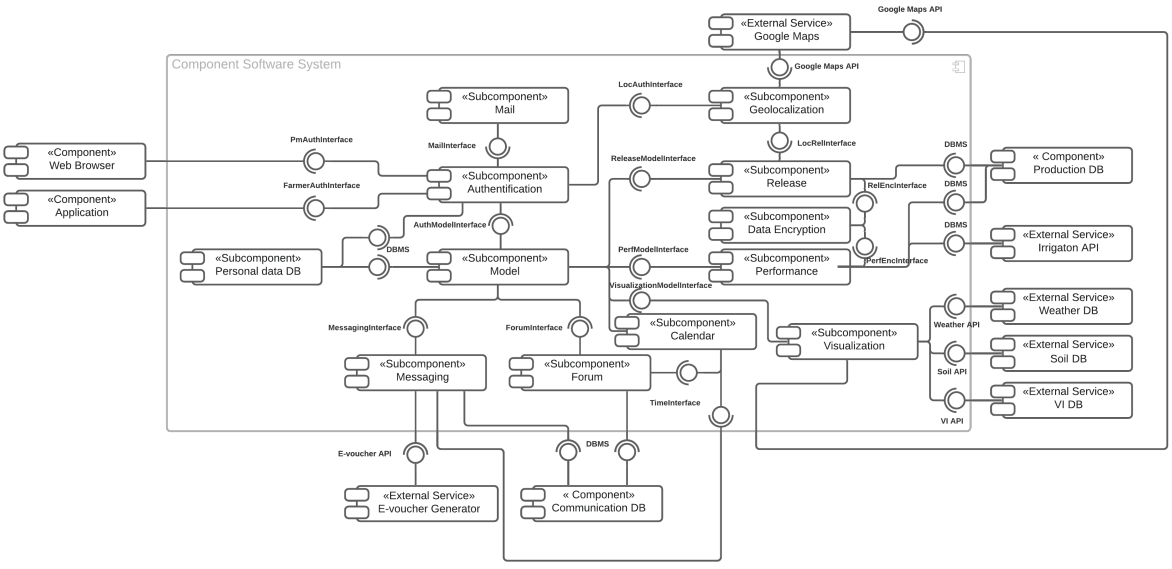


Figure 2: Component diagram

2.3 Deployment view

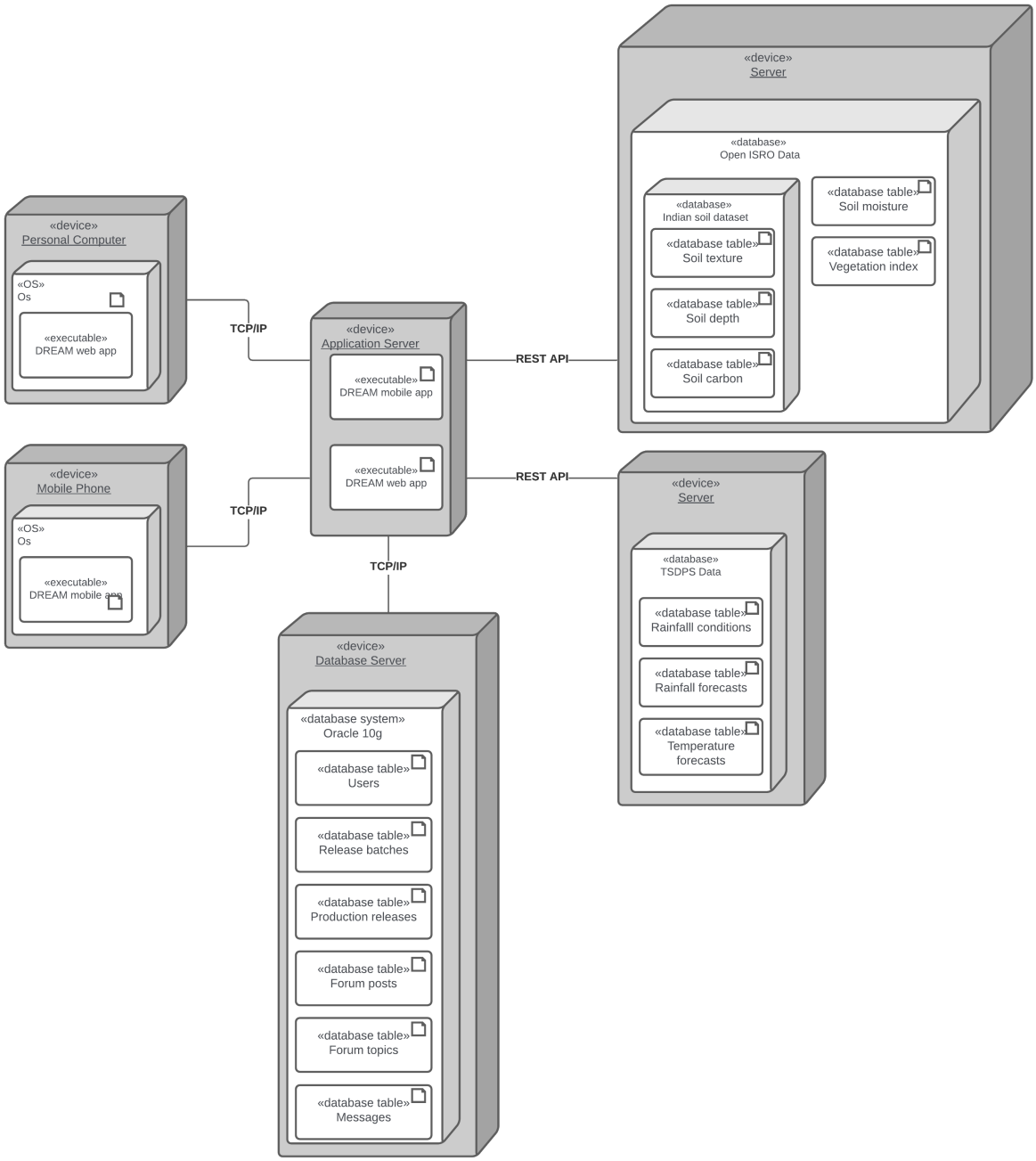


Figure 3: Deployment diagram

2.4 Runtime view

The following sequence diagrams aims at presenting the links between the component during the principal uses of the System by the different users.

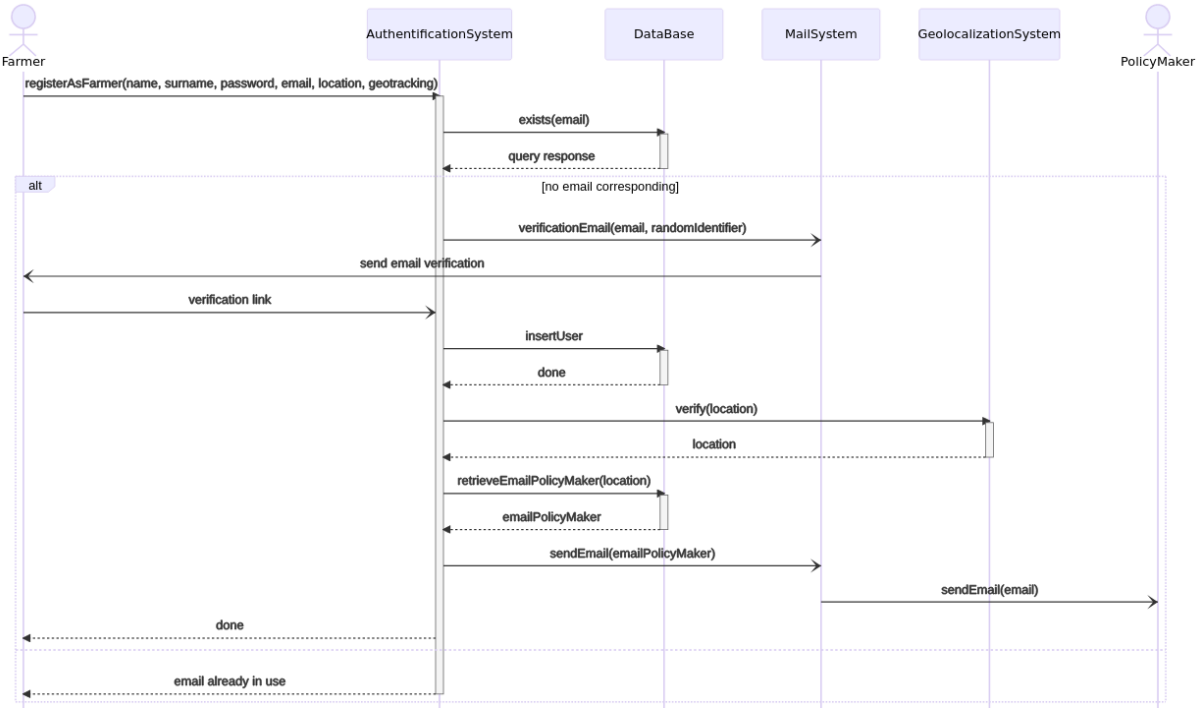


Figure 4: Registration of a Farmer

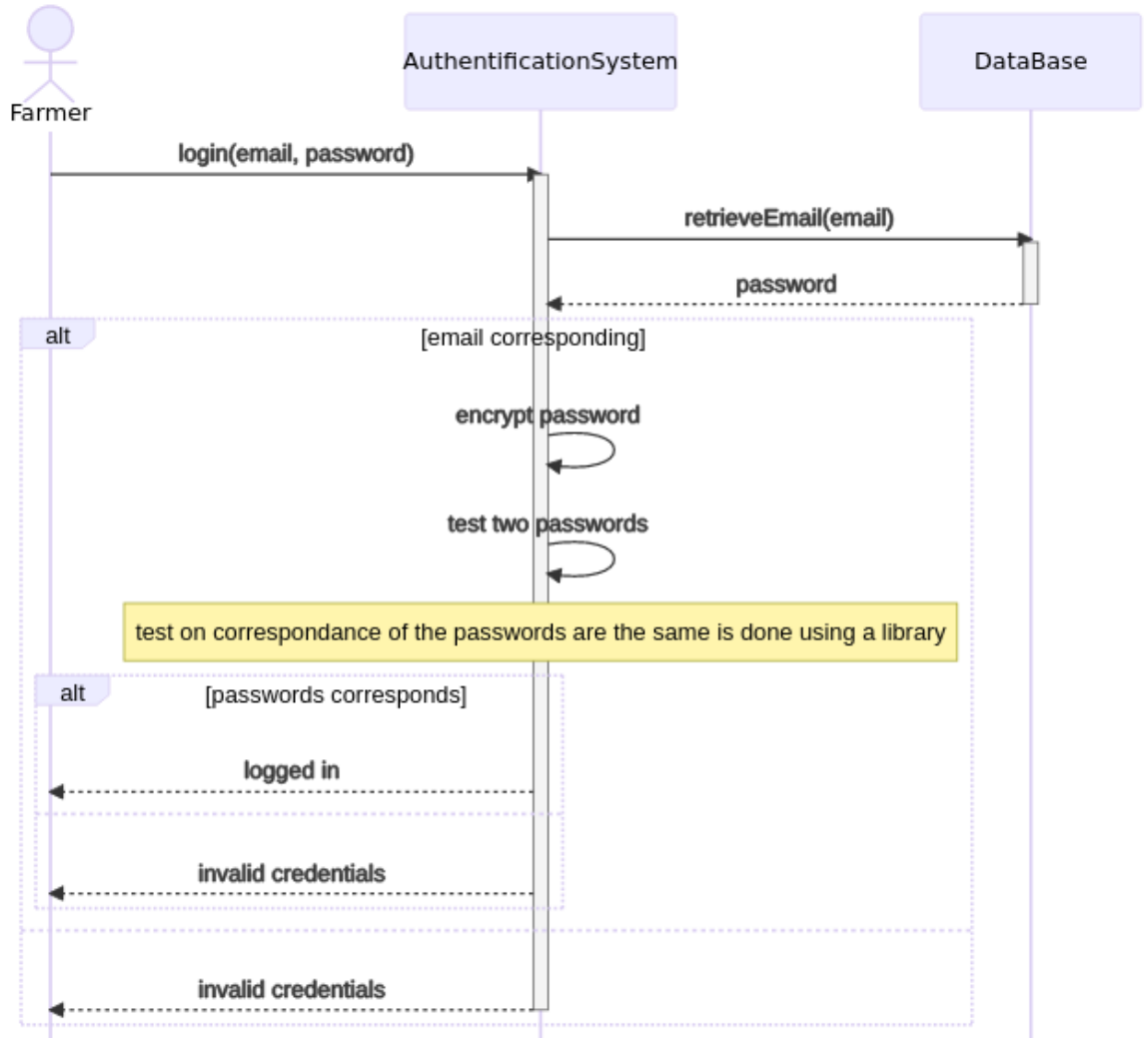


Figure 5: Login of a Farmer

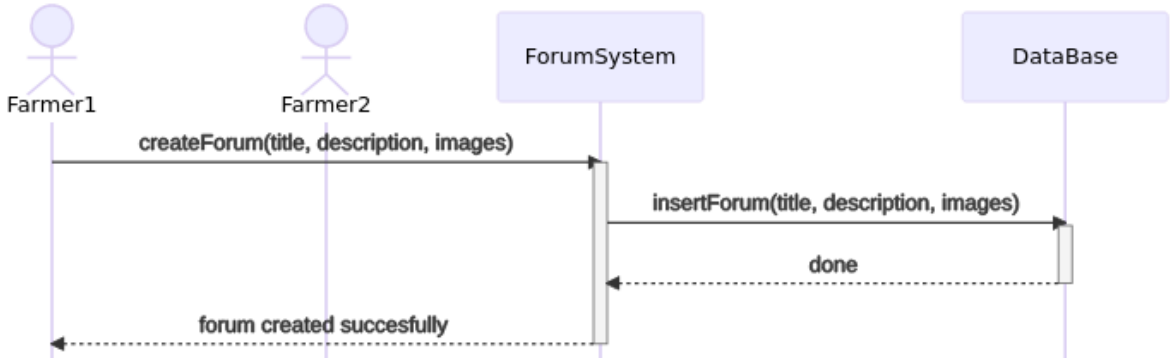


Figure 6: Creation of a forum

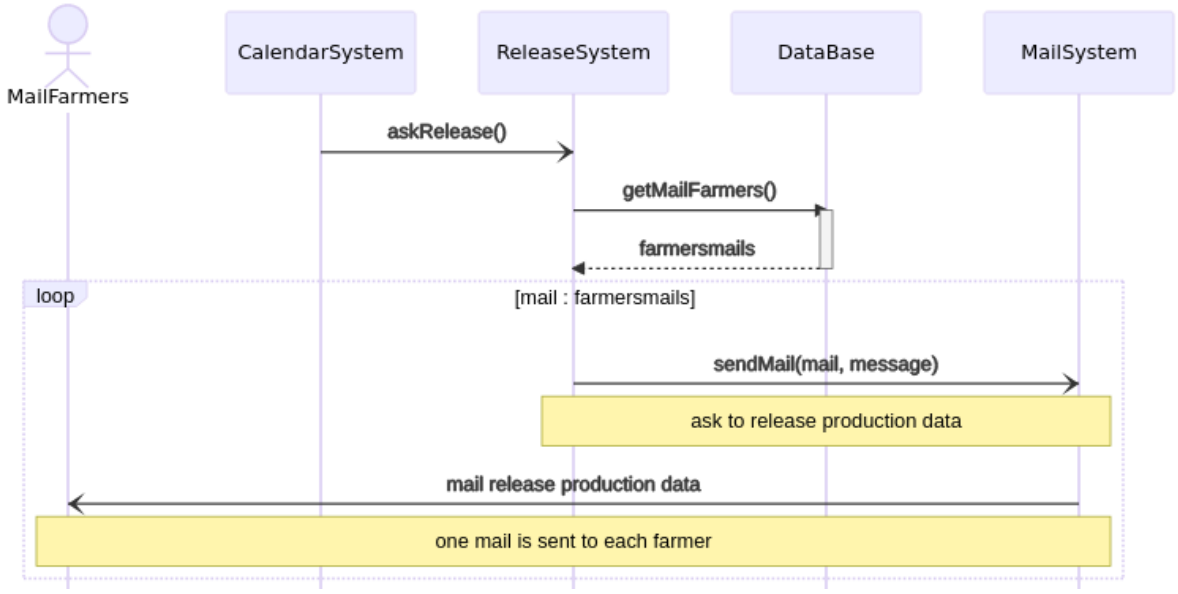
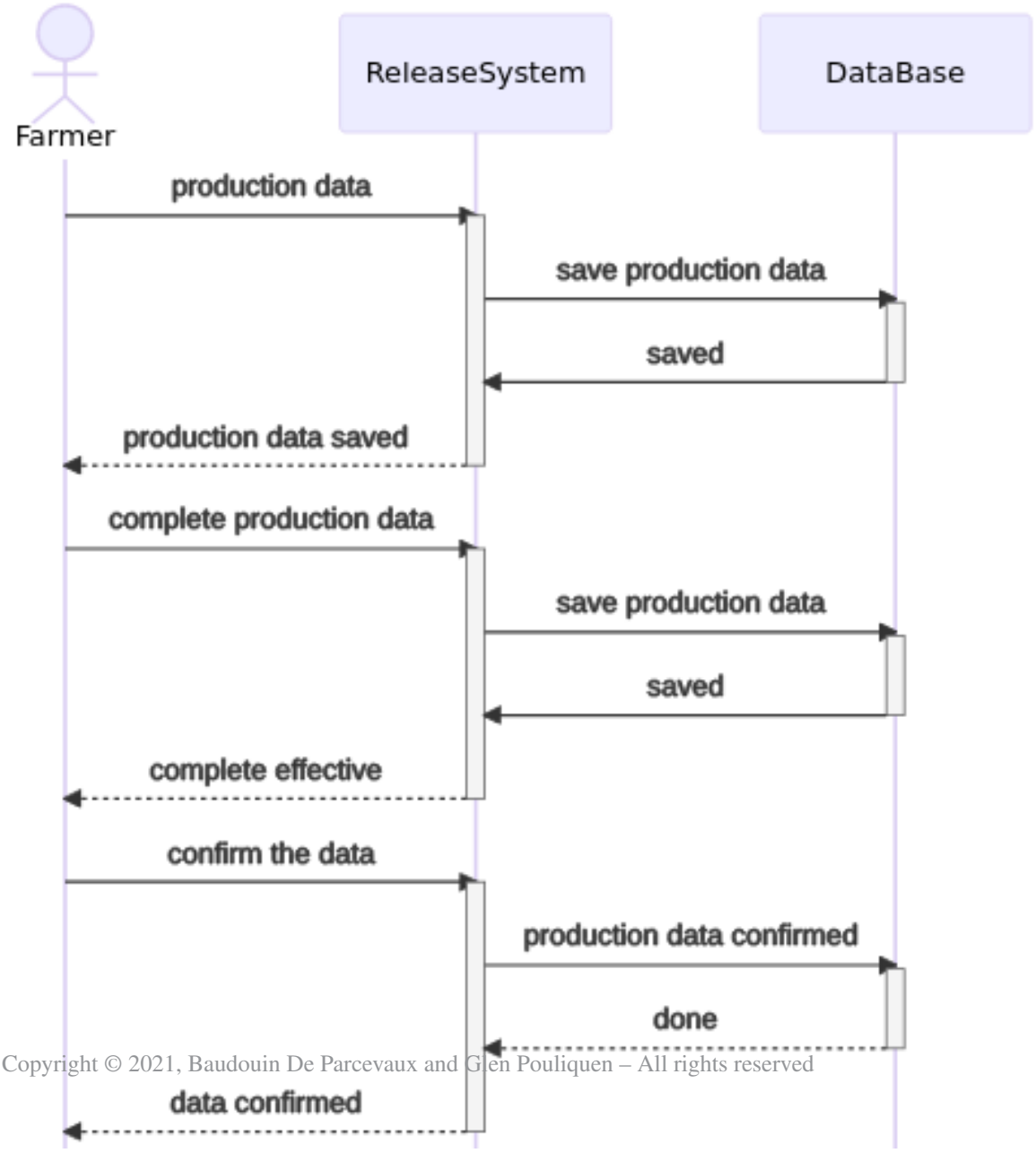


Figure 7: System asking to farmers to put their production data



2.5 Component interfaces

In the following diagram, the DBMS interfaces are not covered since they are supposed to execute basic queries that can be handled by any system.

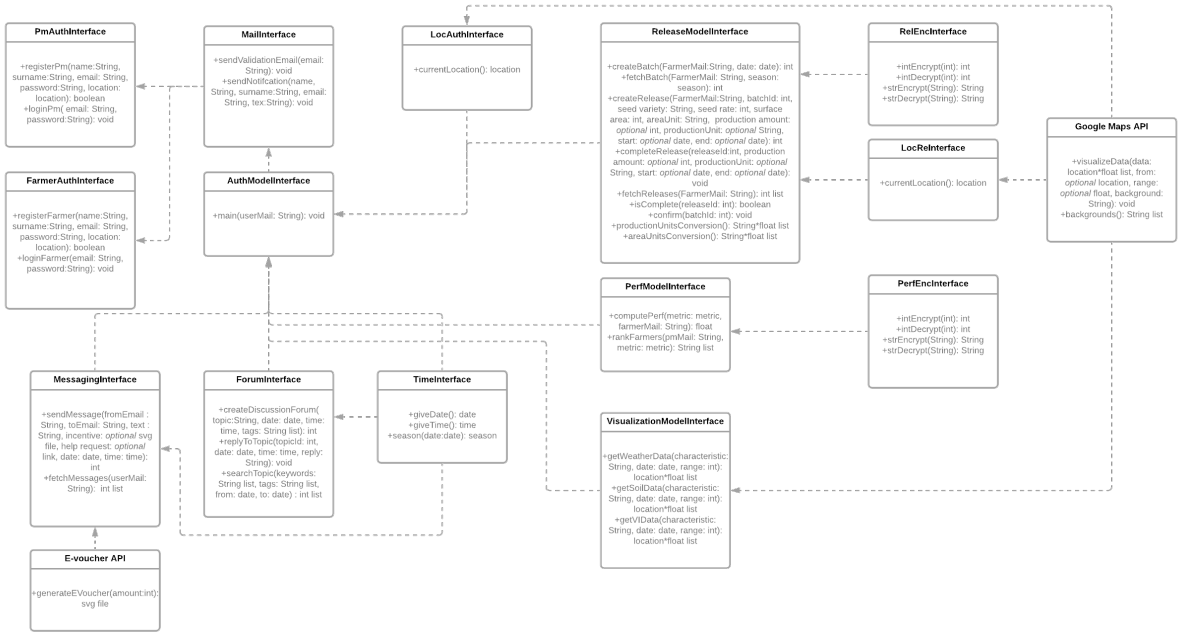


Figure 9: Component interfaces diagram

2.6 Selected architectural styles and patterns

Since the application is distributed and explicitly designed to support interface between users and the system, the optimal architecture is a client/server one.

As the system integrates multiple data sources, a three-tier architecture is more adequate. The system is thus divided into a Client Layer, which is the entry point for users, the Business Layer which is embedded in servers, and the Data Layer.

For the sake of simplicity, praticality and security, the architecture is designed as a thin client one.

3 User Interface Design

The user interface will be responsive and should be multiple platform. Even if it should be mainly done for phones.
Where as the application for policy makers should be also responsive but mainly optimized for computer.

First we will present the interfaces for farmers. They are not exhaustive but the goal here is to give a good overview of what the app will look like. The figure 10 represents the general login page. Depending on the mail the user will then be a farmer or a policy maker.

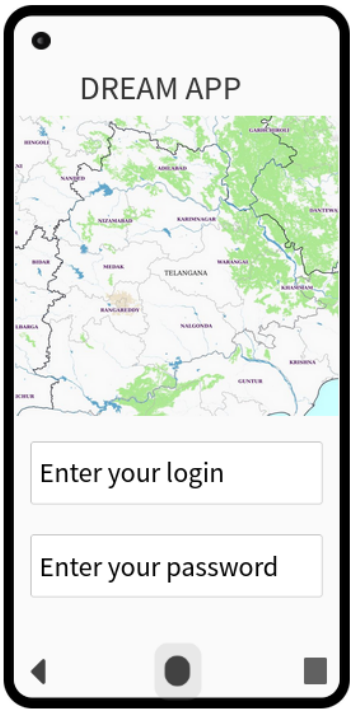


Figure 10: login from a phone

The figure 11 shows the view of the forum on a phone, this view as the others will also be visible from a computer as all the app will be responsive. Next to it the figure 12 gives access to the main data simply and in a rapid way.

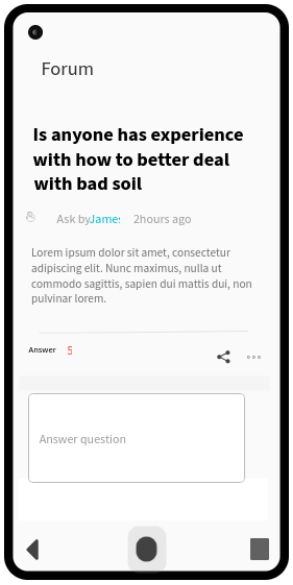


Figure 11: forum from a phone

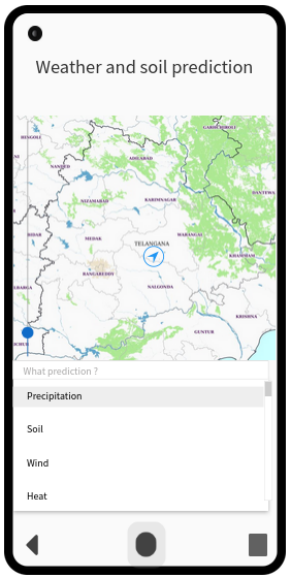


Figure 12: have access to data

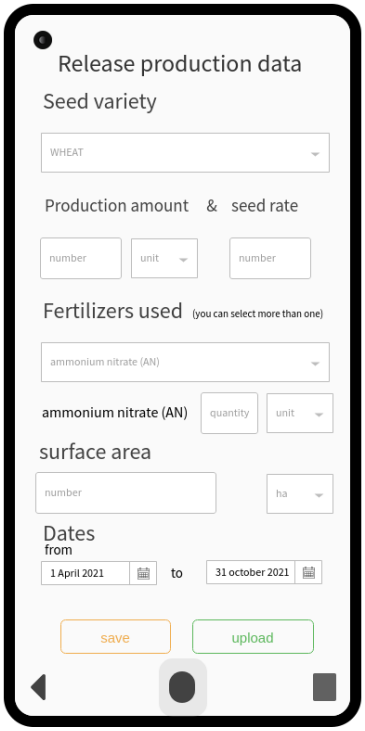


Figure 13: release data

Figure 14: Policy Maker comparing farmer on different factors

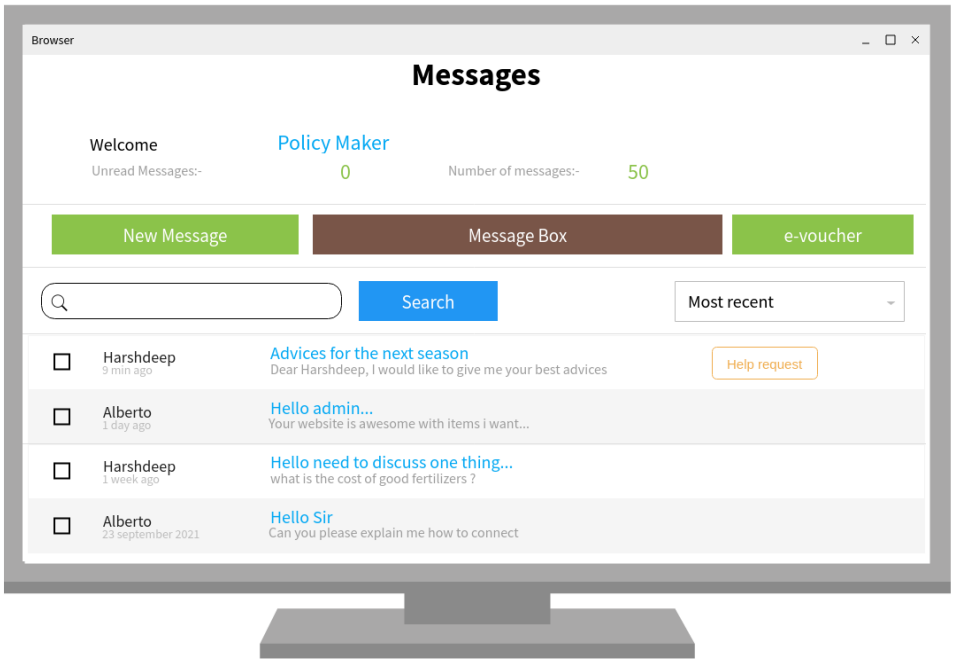


Figure 15: Policy Maker messages

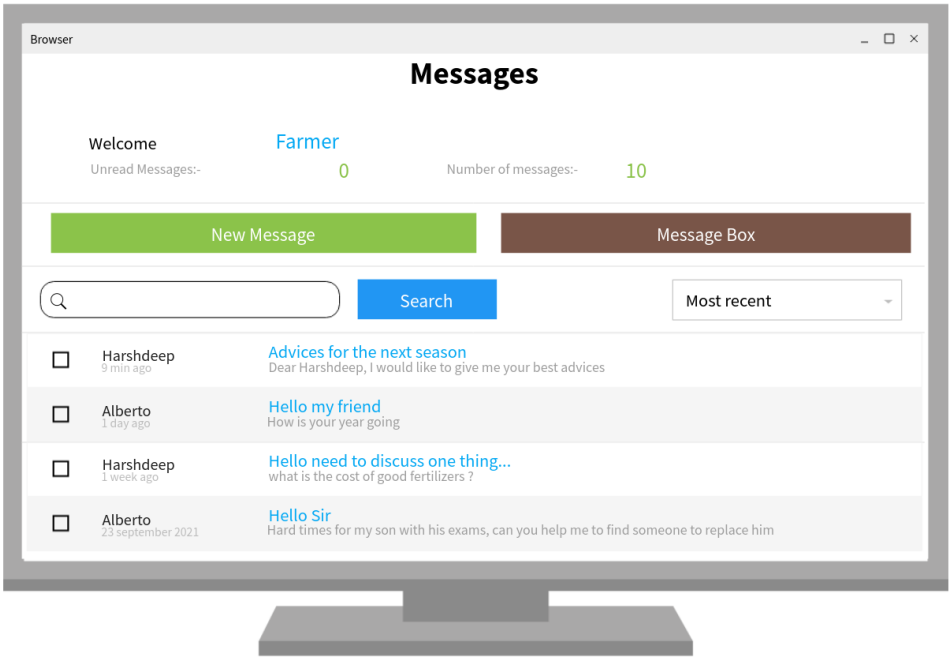


Figure 16: Farmers messages

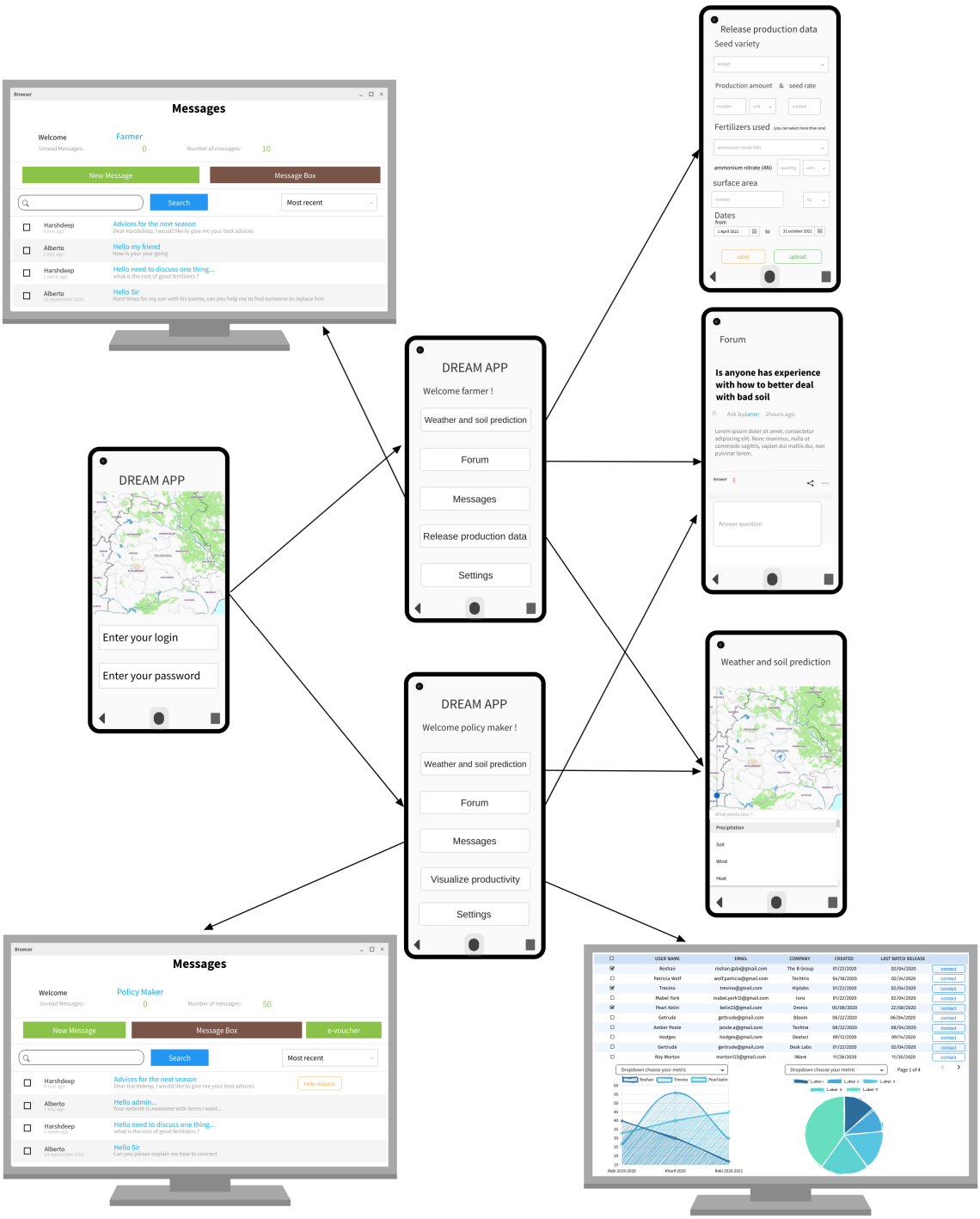


Figure 17: UX diagram showing the different path when using the application

4 Requirements Traceability

The main goal of this document is to better explain what we have defined in the RASD document. In the following list we will explain how the requirements map to the design elements that are defined in this document.

- R1: The system should integrate a calendar that rythms the alternance of cropping season/release period/analysis period.

  Components : Calendar

- R2: Two weeks before the end of the release period, DREAM should recall the farmer to confirm the batch of production (if the farmer didn't confirm it yet).

  Components : Calendar, Message, Mail and Release

- R3: Two weeks before the end of the analysis period, DREAM should recall the policy maker to identify well and poorly performing farmers (if not done).

  Components : Calendar, Performance, Messaging and E-voucher

- R4: Two weeks before the end of the analysis period, DREAM should recall the policy maker to send messages/incentives/help proposal to farmers, if some of the well or poorly performing ones weren't contacted yet.

  Components : Calendar, Performance, Messaging and E-voucher

- R5: DREAM should enable messaging between farmers and a farmer and a policy maker from the same area.

  Components : Messaging, Mail and Geolocalization

  Mockup : figure 15

- R6: When an area is given, DREAM should fetch the policy maker in charge of it.

  Components : Geolocalization

- R7: DREAM should enable any farmer to create a discussion forum, with a title, tags and filters.

  Component : Forum

  Mockup : figure 11

- R8: When requested, DREAM should provide discussion forums ordered by semantic proximity of a given topic.

  Components : Forum, Geolocalization

- R9: DREAM should enable unlimited answers in the thread of a forum.

  Component : Forum

- R10: When a location is asked and the user has agreed so, DREAM should fetch the position thanks to the GPS functionality of a device.

  Component : Geolocalization

- R11: Given a date and a location, DREAM should fetch from the external databases the closest and latest weather conditions.

  Component : Visualization with the externals API (Wheather, Soil, VI)

  Mockup : figure 12

- R12: Given a date, a location and a number of days, DREAM should fetch from the external databases the closest and most adequate (with respect to the number of days of prediction) weather predictions.

  Components : Visualization with the external Wheather API and Geolocalization

- R13: Given a date and a location, DREAM should fetch from the external databases the closest and latest soil moisture data (that is, figures and locations).

  Components : Visualization with the external Soil API and Geolocalization

- R14: Given a date and a location, DREAM should fetch from the external databases the closest and latest soil organic carbon data (that is, figures and locations).

  Components : Visualization with the external Soil API and Geolocalization

- R15: Given a date and a location, DREAM should fetch from the external databases the closest and latest vegetation index data (that is, figures and locations).

  Components : Visualization with the external VI API and Geolocalization

- R16: Given a list of figures and locations, DREAM should display them on an interactive map, with a nice legend.

  Component : Visualization

  Mockup : figure 12

- R17: On a move, the map displayed should be reactive. That is to say: it should query new values to fit in the new spatial frame.

  Component : Visualization

  Mockup : figure 12

- R18: DREAM should have a clock synchronous with the Indian time zone and Gregorian calendar.

  Component : Calendar

- R19: Given a metric and a batch from a location, DREAM should fetch the required data at the date that correspond to the batch. The system should then correctly compute the performance.

  Component : Performance, Geolocalization, Production DB, Release, Calendar

- R20: The system should be able to rank a list of farmers during a given season with respect to a metric.

  Component : Performance

- R21: Given a two dates and a farmer identity, DREAM should fetch from the water irrigation system the total amount of water consumed during the time-lapse by the farmer.

  Component : Performance with the Irrigation API

- R22: If any necessary piece of information is missing at some point of a process, DREAMS should display an error message.

  a check will be done by all the components and an error will be prompted in the front end.

- R23: Given a user identity, DREAMS should be able to access all the personal data of this user.

  Component : Personal data DB, the application will only collect the necessary data

- R24: When a policy maker decides to make an incentive, DREAMS should create via an external API an e-voucher of the amount wanted.

  Components : Messaging with the E-voucher Generator

  Mockup : figure 15

- R25: Given a farmer identity, DREAM should be capable of fetching the corresponding batch.

  Component : Release and Production DB

5 IMPLEMENTATION, INTEGRATION AND TEST PLAN

5.1 Implementation and integration

In this part will be presented the steps in order to implement and integrate the DREAM project. Here is our route plan :

We will first start by creating component by component trying to obtain as soon a possible a first viable solution that some users will be able to use. Asking them feedback on the application. It would be useless to develop all the system and see at the end that it will not be used by farmers because it is not ergonomic, too complicated or that is doesn't correspond to their needs.

For this two servers will be used : a testing server and deployment server. First people will build the application on the test server and when some viable functionality ready, pushed into the production server. This will be used for testing by some selected users (farmers and policy makers).

To do that first of all a continuous integration will be effective with a continuous delivery. When code will be pushed on the development server all the tests will be run before that the new code is accepted. To store the code we will use GitHub or GitLab and use workers.

We will bring development and operation teams together using the devops practice. Devops require more work at the beginning but will permit a better work.

So here will be the beginning for putting into place the DevOps process flow.

1. Use an agile development process
2. Adapt the processes for continuous integration and continuous delivery
3. Automate the software development
4. Automate the software testing
5. All this using continuous deployment

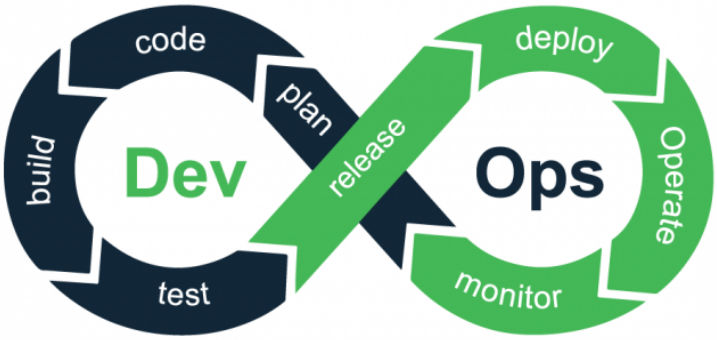


Figure 18: DevOps schema

After the setup of the DevOps process flow the application will be built using the following order. Of course as we use DevOps, changes can and will be made, so we can see the following as a guideline (see figure for the components 3).

1. Make available some components to make a first version like Authentication, Release, Visualization, mails and the different databases related to them, data Encryption (important part but as it is not visible from the user point of view can be done after)
2. forum

3. messages
4. calendar and localization

All this using continuous deployment and taking into account the feedback from users, operators ... Once a good system is present and that all the security and privacy is taken into account the development server will be shared to a bigger number of person. Until it is consider ready to be used by all, at that point it will used in all the state.

5.2 Test Plan

Testing should be done during all steps of implementations and integration.

Unit testing should be done for all the individual parts that will be developed. And if possible should be done by another person than whom did the code. Each part have to be tested individually to be sure that it can be used in other parts without errors and bugs. When a new component or change is done in a part all the previous Unit test should be run for all the parts.

At the same time global tests should be performed testing the good working of parts together. The tests will be run automatically before each deployment on the test and production server. To ensure proper implementation and integration of the DREAM system.

6 Effort Spent

6.1 Baudouin De Parcevaux

Task	Time spent
Overview	1h
Component diagram	4h
Deployment view	3h
Component interfaces	3h
DD	2h
Total	13h

6.2 Glen Pouliquen

Task	Time spent
User Interface design	5h
sequence diagrams	2h
requirements traceability	1h
Implementation, integration and test plan	3h
DD	4h
Total	15h

References

[1] Food and Agriculture Organization of the United Nations. Fertilizer use by crop in india. https://www.fao.org/3/a0257e/A0257E02.htm#ch1, 2005.

