Kene Biri

ICT4D

Group 7

Bogdan E(2553671), Petar G(2571287),

Shreyas S(2635171), Zainab S (2652346)

1. Summary of key idea

Livestock rearing in Mali contributes considerably to rural household production systems. To improve the economic positions of milk producers in small villages and the value chain of milk, the communication between milk producers and customers/resellers should be more efficient. Milk being a perishable commodity, there are several challenges to storage and logistics of it. Milk that is collected from the cattle should reach the cooperative within four hours in. A step towards achieving this is an application for the communication between the farmers and the cooperative. With a voice message application the cooperative can be notified faster to collect the milk. Currently, the cooperative uses a paper map on the wall to write the locations and other data about where to pick up the milk. The digitized solution offers the cooperative a better overview of all the locations from where to pick up the milk and to plan the logistics more efficiently. Milk producers from small villages collect milk from their cattle. In the proposed solution, the farmers/milk producers call the voice application and input the number of litres they have for pick up. In order for a better overview of orders, trackability and readability, the proposed application also incorporates a web interface in the form of a dashboard for an operator in the cooperative. The operator then informs the delivery person of the pick up address and the milk is delivered back to the cooperative from the farmer as soon as possible.

2. Actors and goals

Stakeholder	**Operational goal**	**Responsibility in the envisaged system**
Milk producer in small village	Send milk to cooperative as soon as possible and prevent it from getting spoiled	● Register themselves at the cooperative, ● Remember their id and ● Call when milk needs to be picked up.
Cooperative Operator	Gather more milk from the farmer and more efficiently(quickly)	● Manage orders.
Hosting organization (ICT business/NGO)	Provide the voice application and web interface	● Develop, test and maintain the system

Pick-up Driver	Pick up and deliver more milk, more efficiently	● Receives information from the operator in the cooperative as soon as possible

3. Context and scope

The stakeholders are the hosting organization, milk producers from small villages, pick-up drivers and the cooperative. The cooperative provides a supporting network and a stable economic position for the producers by introducing an efficient way of selling the milk without the milk going bad or losing its quality. The cooperative currently uses no digital technology and uses outdated hardcopies for keeping a track of all active orders which does not have much readability or classification in terms order statuses and overview. The web interface being developed aims to solve this issue by dividing the tasks into different categories based on their status. The hosting organizations concerns are that of designing, developing/implementing, testing and maintaining an application where demand and supply come together. The pick-up drivers are members of the cooperative. It is assumed that they can meet with the operator in the cooperative so that the latter can assign orders to the delivery persons. We also assume that the operator in the cooperative has sufficient knowledge and skill to manage orders by assigning drivers to orders(including informing the drivers of the pick up address) and mark orders as delivered. The scope of the scenario is focussed on improving the communication between farmers (milk producers) and the cooperative. Since the application has to be as simple as possible for the sake of better usability for the less technically adept farmers, our system takes into assumption that the farmers are already registered in cooperative, and the website operators job is to register these farmers in the system and inform them about their id (which is the primary key of their entry in the database). This also means, only valid/registered farmer ids are allowed to move on to step 2 to enter the number of litres. In case, invalid or unregistered farmer id or wrong farmer id is entered, the voice application requests to enter the id again. To minimize the number of steps in the voice application for the farmer, we have only used French language for all the menus. We also assume the farmers have practical literacy about the weight units of the milk that they want to be collected and the time when the milk is collected from the cattle. An obvious prerequisite for the whole system to work is phones for the farmers and a good internet connection for the cooperative

4. Use case scenario script

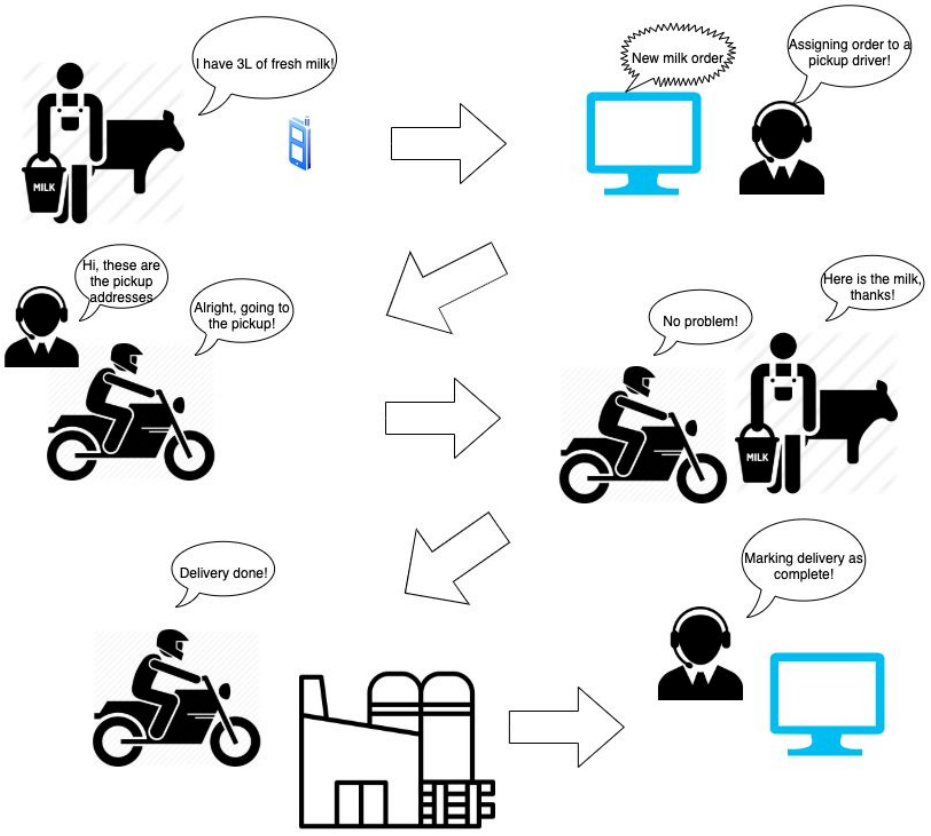

Fig.1. Use case scenario

5. Interaction and communication

We present 2 diagrams that model the interaction of the Farmer with the interface, and the second one which models the interaction between Operator in the Cooperative and the pick-up driver. Farmer interacts with the voice interface to give information about who he is and how much milk he has. Based on that the order is stored in the database and displayed at the web interface in Cooperative. Cooperative operator then schedules these deliveries to a pick-up driver, who receives an SMS alert once there are orders.

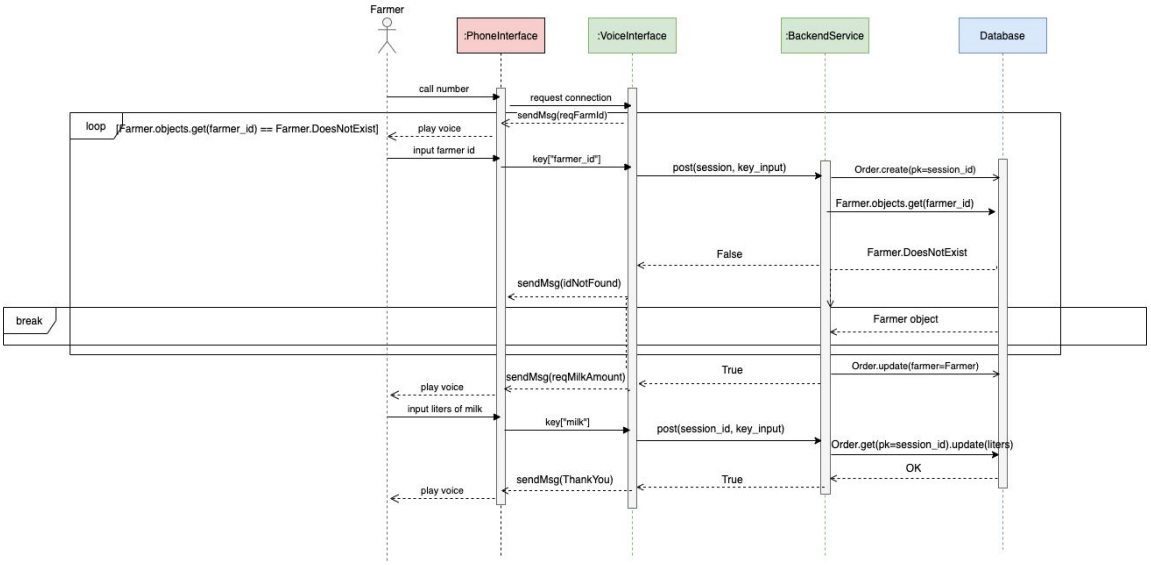

Fig.2. Sequence Diagram with farmer as actor

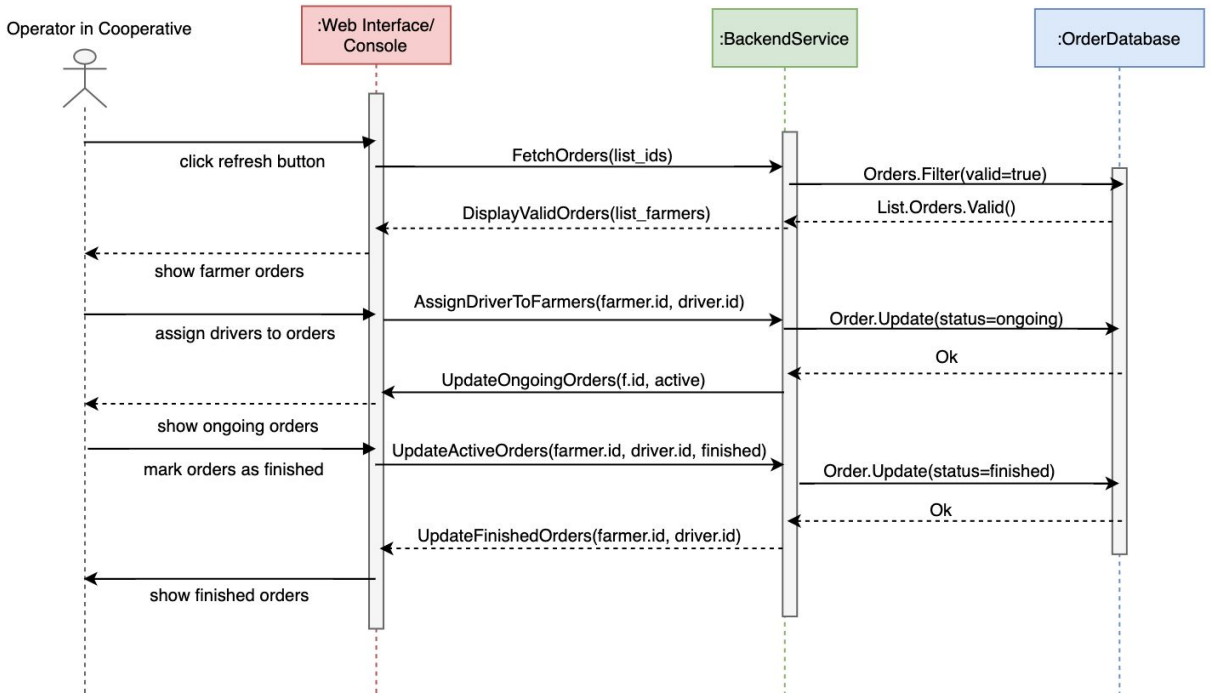

Fig.3. Sequence Diagram for Operator in the Cooperative as actor

6. Information concepts

This class diagram represents relationship between farmer and driver

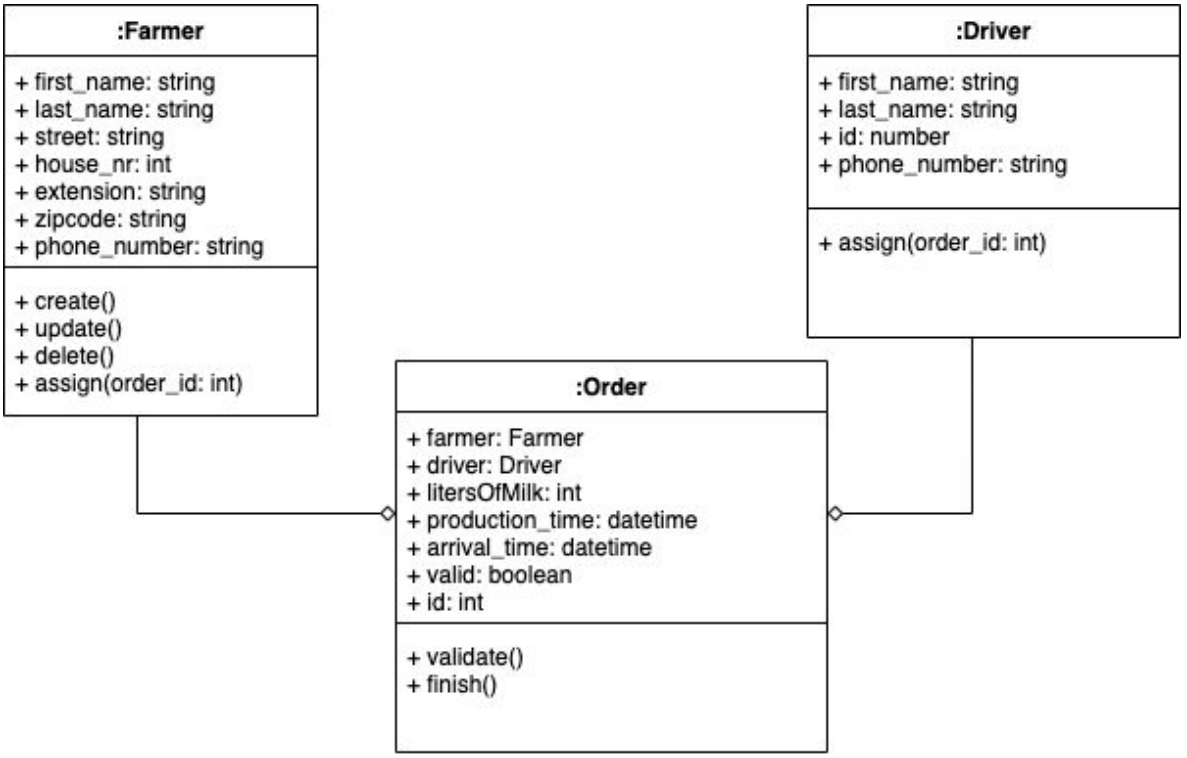

Fig. 4 . Class diagram

7. Technology infrastructure

The following deployment diagram depicts where the individual components of the application can be/are deployed:

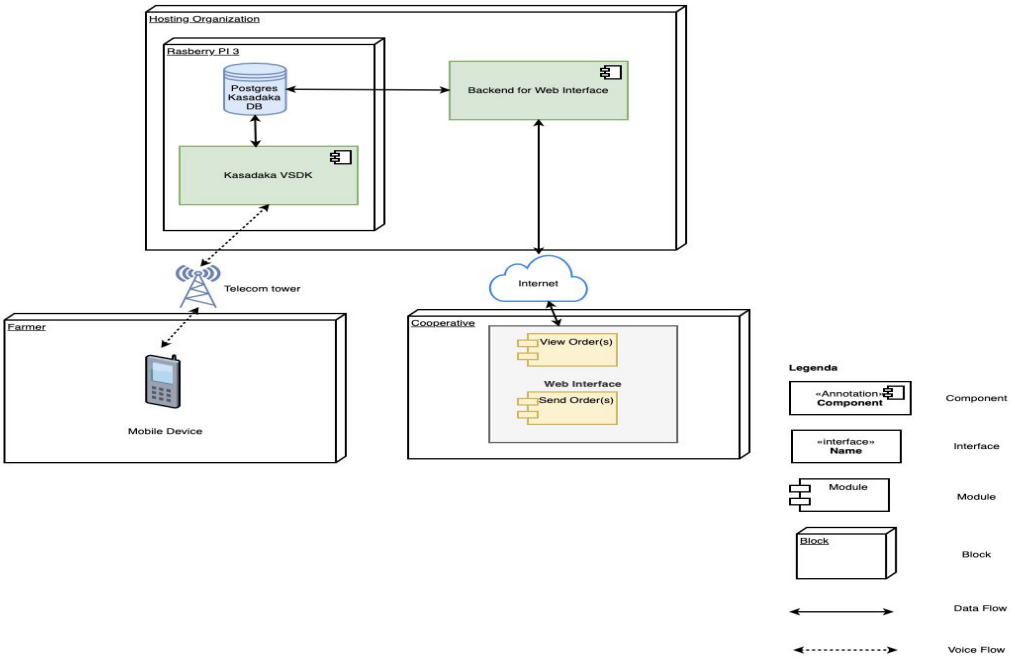

Fig. 5 . Deployment diagram

8. Cost considerations

The stakeholders of the application are the cooperative, the farmers and the drivers. In this sense, they represent the parties that must support all the costs (investment, development, operational, outsource). However, the application does not require high costs, as the main things that are needed is a Raspberry-Pi for hosting both the web and voice applications, with a price around 40 euros, mobile phones with basic calling functionality for the farmers and drivers that do not already own one, and basic phone subscriptions to a telecommunication provider for the farmers and drivers that do not yet own one in order to be able to make calls to the voice application. Moreover, a computer is needed in order to run the web interface at the cooperative. All these costs must be gathered for an initial investment and in order to keep the solution operational. As for maintenance and development there are no costs required.

9. Feasibility and sustainability

Technical feasibility.

The project does not require advanced technologies in order to be achieved. As described at the previous point, the project only requires basic mobile phones, basic subscription to a telecommunication provider, a computer and a Raspberry-pie to act as a server.

Business and (socio-)economic feasibility and sustainability of the scenario.

Selling milk is one of the most important sources of income for a farmer in Mali. For the distribution of milk most of these farmers use different ways of reaching customers. They can choose from selling/bringing the milk to a collection centre, selling the milk via a middleman or directly to consumers. The benefit of bringing it to a collection centre is that the process and sales are protected and regulated, whereas selling via a middleman or directly to consumers is not. The downside of such centres is that there are no pick-up services. It is also not profitable for farmers to bring the milk, when it's a small amount, to a collection centre. This makes it more attractive for farmers to sell the milk via other forms of distribution. Not only farmers suffer from these inefficient forms of distribution, but also consumers do not know in most cases where the milk comes from when it is bought via a retailer. This scenario creates opportunities to add (socio)economic value for the stakeholders in our use case as well as for the consumers. Even though consumers are not part of our use case, eventually they also contribute to the economic status of the milk producers..

With the voice application and the web interface farmers do not have to bring milk to the collection centre, but is being picked-up by drivers that are members of the cooperative, this adds convenient value and economic value for farmers. Farmers both save time and money. Adding pick-up service and when farmers use this form of distribution creates jobs for the pick-up drivers. By providing their services more efficiently the NGO creates social value.

Adding propositional values for the stakeholders increases the sustainability of the digitized solution. More and more people use mobile phones in Mali. There are at least four major mobile operators of which Malitel and Orange Mali are two of them. This means that people will get more familiar with mobile phones and thus increase the chances of using and maintaining the digitized solution for the distribution of (fresh) milk.

Possible goal conflicts and dependencies in the collaboration between the actors in the scenario.

As far as our scope of the scenario goes there are no goal conflicts in the collaboration between the actors. An important dependency is that between the farmers and the cooperative operator. The cooperative operator is the one that manages the orders. If there is something wrong with the orders or placing the order the operator should provide a solution as soon as possible. When an operator does not work adequately it can cause waste of milk for farmers and for the pick-up drivers to not be able to do their job. The operator is the one that should communicate problems and solutions to the other actors. It should be easier to find pick-up drivers , however because of the knowledge and skills that are needed for the voice application and web interface it might be difficult to find an operator when a current operator does not work adequately.

Important general preconditions for the scenario to work.

The most important preconditions are the literacy level of the farmers and the protection/regulation of the collection centres. The literacy rate in Mali for the population aged 15 years and older is 35,47%. Because the illiteracy rates are high in Mali the voice application is developed in a way that it is easy in use. As milk is the main source of income for farmers it is important that the distribution process is regulated and protected by authorities. Knowing that the process is being protected farmers will build trust in this distribution form and more formers will make use of the digitized solution.

10. MoSCoW

MoSCoW attributes	Requirements
Must have	- The voice application must be user friendly (easy to understand/navigate). - The voice application must be easily accessible. - The voice application must operate in the local language (French). - The web application must display the new orders, ongoing orders and finished orders. - The web application must display the registered farmers and drivers. - The web application must offer a way to assign drivers to orders. - The web application must offer a way to transition an order from new to ongoing and from ongoing to done. -
Should have	- The web application should have a way to register farmers. - The web application should have a way to register drivers. - The web application should have a way to remove drivers. - The web application should have a way to remove farmers. - The farmer should
Could have	- The application could have a feature to send SMS with the pickup locations, to the drivers.
Won't have	- The application prototype won't contain other languages than French -

11. Prototype Description

Description of the design decisions of the application and how these decisions are made.

The design decisions of our application are based on the feedback from the instructors and our assumptions. Based on feedback from assignment-1, we decided to use only one language i.e French for the application due to ease-of-use and understanding. Since we wanted the proposed solution to run without internet and much investment costs, the web interface is built as a part of the Kasadaka-VSDK. This ensures that it can be deployed locally without needing to pay for cloud providers and hosting services. We also realized that milk production contributes significantly in terms of economy for the rural population in Mali and so there would be a lot of farmers. This meant that we also extended the Kasadaka-VSDK to support multi-digit key input. This means we input and store more numbers than nine which in turn lets us store a big number of farmers.

Front- and back-end implementation results.

Back-end

Apart from the original functionality of the application/framework, the following features have been added(Fig. 6)

- Originally, Kasadaka-VSDK did not record the key input values which meant we could not register the number of litres of milk to be picked up or the ID of the farmer. So we added support for recording the user-input.
- Additionally, as discussed in previous sub-question, the functionality of the back-end i.e has been extended to allow for multi-digit input.

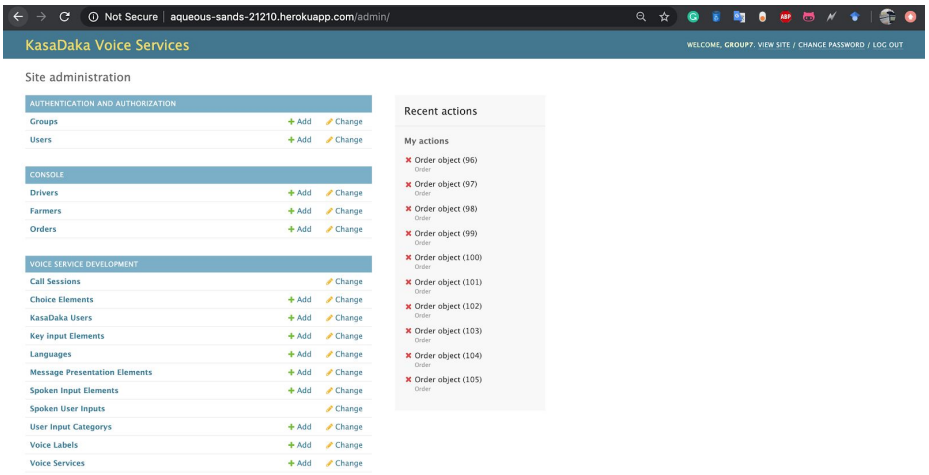

Fig. 6 . Backend/ Django Admin Interface diagram

Front-end

The frontend is a console (a screenshot in Fig. 7) built inside Kasadaka-VSDK (URLs/endpoints have been added in appendix) which has the following functionalities:

- Refresh orders to retrieve the list of valid orders
- Assign drivers to orders under the “New orders” tab (under Orders) and save it
- Mark orders as finished under the “Ongoing orders” tab (under Orders) and save it
- View list of finished orders under the “Finished orders”(under Orders) tab
- Create and delete farmers
- Create and delete drivers

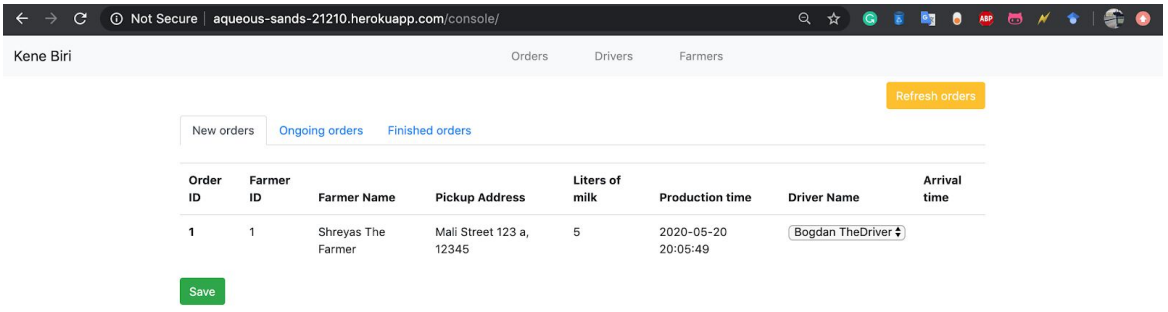

Fig. 7 . Console/Web Interface

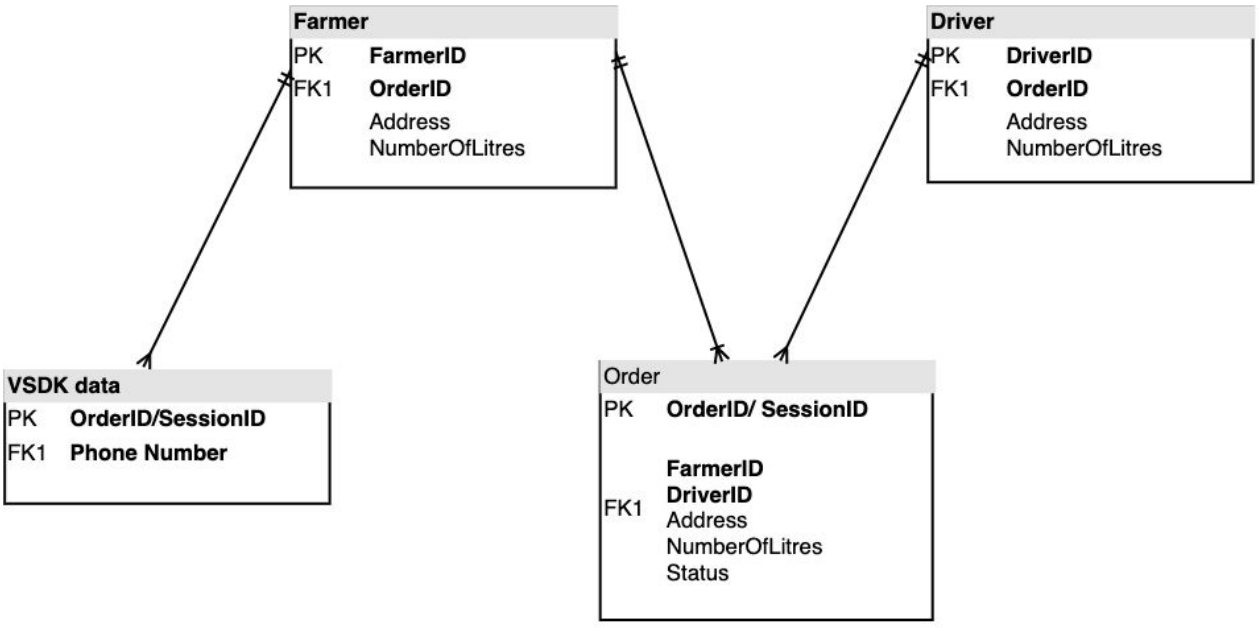

Fig. 8. Data model diagram

12. Pointer to the Application Code

URL for the voxeo VXML code is in Appendix A2.

13. Pointer to how to access the application

1. The application's Heroku dashboard link: https://dashboard.heroku.com/apps/aqueous-sands-21210
2. Heroku app link/ Django admin interface : http://aqueous-sands-21210.herokuapp.com/admin/
   login with credentials
   - Username: group7
   - Password: ict4dgroup7
3. Web Interface/Dashboard for operator: http://aqueous-sands-21210.herokuapp.com/console/

The starting url: http://aqueous-sands-21210.herokuapp.com/vxml/start/3
To run the application:
- Goto voicexml switcher
- Enter the starting url and other fields
- Call the voice service on the number: 020-3697664

After entering the id and litres of milk to picked up during the call, go to link in point 2 (Django admin interface).
Go to the link in point 3 (Web interface) to refresh the list of orders in point 3 mentioned above. In the web interface, under the main tab/header of Orders, assign orders and save in the "New Orders" tab, following which mark an ongoing order as finished under the "Ongoing orders" tab and finally click on "Finished orders" tab to look at the overview of the finished orders.

14. Short Usage scenario

Open a console interface on http://aqueous-sands-21210.herokuapp.com/console/. In a separate tab, create a farmer inside the admin interface: http://aqueous-sands-21210.herokuapp.com/admin/. In the overview of the farmers you can see their primary keys which are the id's assigned to them by the system. Next, using the voiceXML switcher lock the app (http://aqueous-sands-21210.herokuapp.com/vxml/start/3) and call the app using the number provided.
The app asks for the identification number of the farmer and waits for a number to be inserted. Please insert the id (primary key) of the farmer that you created. Here you can try to input a wrong id, the voice service will inform you that the requested id is not found, and it will ask you the id again.
Next, it asks how many liters of milk the farmer wants to deliver and waits for a number to be inserted. Go to the Kene Biri console and click "Refresh Orders" button. You can now see the new order pending. From here you can assign a driver (which you can also create in admin interface or use the already provided ones) and save the order. You can switch to the ongoing orders tab and mark the order as finished. The order will be displayed in the finished tab.

15. Feedback Questions

Questions from meeting with Anna Bon on April 7th, 2020:

1. What is the role of cooperative?
2. Who are the sellers and who are the customers in this use case?
3. Does the milk processing requirement have any influence on the actual use case?
4. Are there more sellers?
5. Should we define selling regions?
6. How do the customers get to know the sellers?
7. Can customers choose the seller?
8. Can we assume there is a one service per one cooperative?
9. Where is the bottleneck in the milk value chain?

Questions from the tutorial session with Francis Dittoh on April 29th, 2020:

1. Can we ask farmers to input the quantity of milk?
2. Can we use caller id as farmer id?
3. Can we have multi-number input in KaSaDaka?
4. How will the web interface interact with the KaSaDaka service?

16. Discussion of Scope and Fidelity

The prototype has a list of key features like the voice service and its interaction to the custom database models. We have also extended the KasaDaka with custom multi-digit input. Also we provided the cooperative operator with functionalities to assign orders to drivers and mark orders as finished.
However, there is a list of features we haven't had time to implement, but we would consider as necessary in order for the application to be deployed in the field.

Non-implemented Features:

1. **Implement i18n in the console.** This is critical in order for operators to use the application. The implementation is not challenging, we only need to add french translations.
2. **Add authorization to voice service.** Current implementation allows anyone to call-in and schedule an order if they correctly specify a farmer-id. This can be fixed by comparing the farmer's phone number and number of the caller.
3. **Add voice service for the driver.** Currently we assume that the drivers get the delivery addresses from the operator, but to automate and ease this process we could implement a new voice service for the drivers. They could call this voice-service and retrieve addresses for delivery that are scheduled to their name. This solution should implement text-to-speech in order for the addresses to be spoken to the drivers.
4. **Add local languages to voice service.** Currently our app supports only French. In order for it to be more relevant to local farmers, it should support local languages like Bambara
5. **Add other language support to Web interface and Django admin interface.** All Django admin interface and the console are both in English but we envisage that support be added for more languages in order for the operator to be more proficient at operating it.

17. Conclusions

In this document, a proof of concept has been implemented to demonstrate the usability and feasibility of our application to enhance and bolster the current methodology of collecting and pasteurizing milk. We have successfully expanded the existing framework to support recording of the user key input and also to support multi-digit input. The proposed solution uses a web interface built into the Kasadaka-VSDK application to allow the operator in the cooperative to assign drivers and manage orders. The features missing or scope for future extension to make it more robust and feature-rich have also been outlined in the previous section. We envision that our current implementation can be deployed as is or be extended and deployed in order for it to help the farmers, their financial status and the community as a whole.

Appendix 1

A1. List of URLs of our application- Kene-Biri

1. Django admin interface: http://aqueous-sands-21210.herokuapp.com/admin/
2. Django admin interface to create Drivers Objects: http://aqueous-sands-21210.herokuapp.com/admin/console/driver/
3. Web Interface/ Console for new, ongoing and finished orders: http://aqueous-sands-21210.herokuapp.com/console/
4. Web Interface/Console for creating and deleting new drivers: http://aqueous-sands-21210.herokuapp.com/console/drivers/
5. Web Interface/Console for creating and deleting new farmer: http://aqueous-sands-21210.herokuapp.com/console/farmers/

A2. Code of the Voxeo voice application

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version = "2.1" >
```

```
<form>
<field name="farmerid" type="number">
        <audio src="http://webhosting.voxeo.net/206674/www/hello.wav" />
</field>

<field name="milkliters" type="number">
  <prompt>
    <audio src="http://webhosting.voxeo.net/206674/www/milkliters.wav" /> <value expr="farmerid"/> <audio src="http://webhosting.voxeo.net/206674/www/milkliters2.wav" />
  </prompt>
</field>

<field name="milkhour" type="number">
  <prompt> <audio src="http://webhosting.voxeo.net/206674/www/you_have.wav" /> <value expr="milkliters" /> <audio src="http://webhosting.voxeo.net/206674/www/liters_of_milk.wav" /> <audio src="http://webhosting.voxeo.net/206674/www/what_hour.wav" /></prompt>
</field>

<field name="milkminutes" type="number">
  <audio src="http://webhosting.voxeo.net/206674/www/what_minute.wav" />
</field>

<block>
 <prompt>
 <audio src="http://webhosting.voxeo.net/206674/www/thank_you_final.wav" /><value expr="farmerid" /> <audio src="http://webhosting.voxeo.net/206674/www/and_you_have.wav" /> <value expr="milkliters" /> <audio src="http://webhosting.voxeo.net/206674/www/liters_of_milk_final.wav" /> <value expr="milkhour" /> <value expr="milkminutes" />
```