Increasing Resilience of Smallholders with Multi-Platforms Linking Localized Resource Sharing

Deliverable D5.2b

Evaluation technical framework – v2

Responsible Editor:	ACICT
Contributors:	ORANGE, UPPA
Document Reference:	RESILINK D5.2b
Distribution:	Public
Version:	1.1
Date:	February 2024

Contributors table

DOCUMENT SECTION	AUTHOR(S)
SECTION 1	C. Pham (UPPA)
SECTION 2	S. Bradai (Orange)
SECTION 3	A. Cazaux (UPPA)
SECTION 4	A. Cazaux (UPPA)

Document revision history

Version	Date	Changes
V1.1	Feb 28th, 2024	Public release
V1.0	Jan 15th, 2024	First draft version for internal approval
V0.1	Nov 15th, 2023	First release for review

Executive Summary

Deliverable D5.2b describes the technical evaluation framework to target the final definition and component integration for the RESILINK digital platform. The document will elaborate on the ODEP platform and the associated tests to confirm ODEP’s functionalities and API for RESILINK’s needs, the proof-of-concept of the RESILINK mobile application, the preliminary study of the RESILINK digital platform and the preliminary study on deployment of all these components.

TABLE OF CONTENTS

1. Introduction

RESILINK's proposed approach is to increase smallholder's resilience by providing continuity of access to both resources and markets in crisis situations. It will empower the local agri-food value chain model by optimizing usage of local resources, promoting and generalizing local resource sharing approach and facilitating territorial markets. This local agri-food value chain model will also be integrated with the local e-commerce, supply & distribution channels. The concept of localized and short agri-food value chain will also impact on the agro ecological system by minimizing the food losses and contributing to the climate & environment changes with shorter food supply chains and logistics. As a result, new and local innovative services can be identified and created, enhancing further the smallholders' agri-food chain.

It develops a distributed digital resource management platform for real-time exchange of information on territorial resources and supplies & demands; connecting smallholders to new supply, sharing opportunities and distribution channels. In addition, RESILINK will incrementally use cutting-edge digital technologies to connect fields and farms resources, automatize and add intelligence in the agri-food value chain to provide simple application interfaces adapted to smallholders.

RESILINK has the clear ambition to make digital smart technologies attractive & accessible to smallholders. The proposed solutions will be simple to use on a daily basis so that its usage will become natural, even in non-crisis situations.

The core of the RESILINK digital resource management consists of the Orange Decentralized Exchange Place (ODEP) platform initially developed by Orange for energy and telecommunication ecosystems.

In addition to ODEP and to maximize RESILINK's usage by the end users, RESILINK will develop a mobile application that will be the main interface to simply, quickly and intuitively interact with the RESILINK digital resource management platform. An intermediate platform, referred to as the RESILINK platform, will also stand between the RESILINK mobile app and the ODEP platform to implement additional functionalities on top of ODEP.

This document describes these preliminary steps, in link with D1.1 "Smallholders' resource requirements and distribution channels in a local & territorial agri-food chain".

2. ODEP PLATFORM

2.1. Evolution of ODEP platform

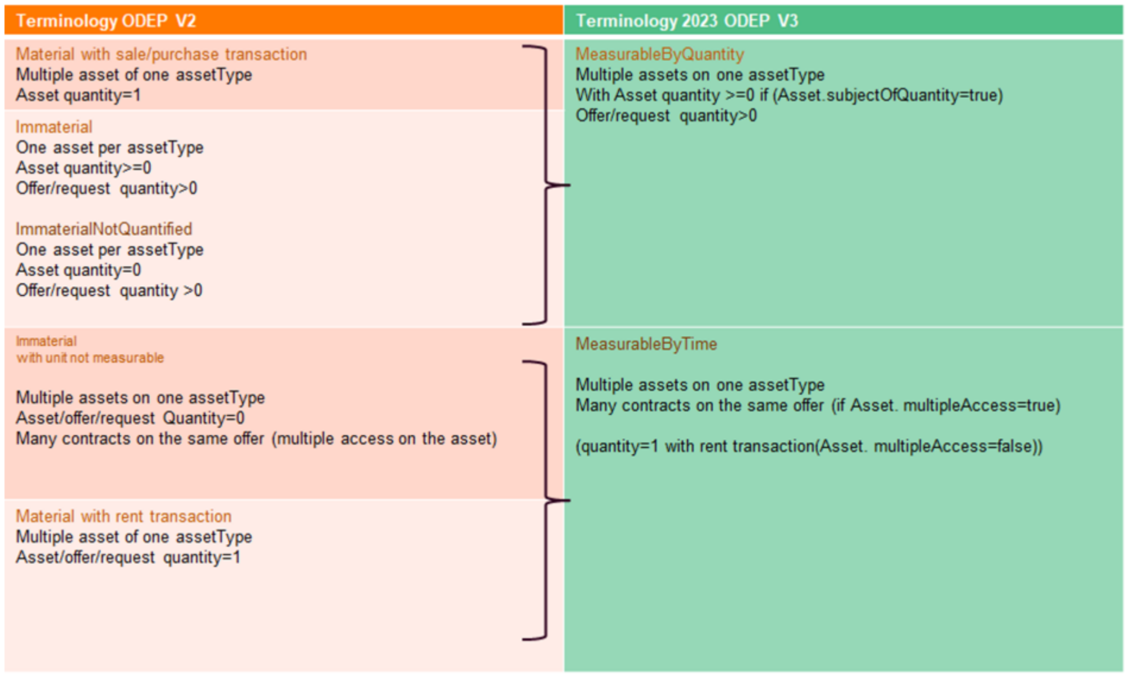
To meet functional and performance requirements of RESILINK project, we defined some evolutions on ODEP.

2.1.1. Data Volume limitation

Current version of ODEP stores all data within smart contracts in a totally on-chain manner. To enhance the performance of ODEP in terms of some metrics such as the number of transactions per second (TPS) and used gas, we adopt an on-chain/off-chain approach that separates what must be processed as digitally executed smart contracts as on-chain modules, from what must be processed as off-chain modules at the level of the REST API. We have coupled all services with a MongoDB so that we stored data that does not require a high level of security in this database. The configuration of the asset types, postulation of offers and requests will be processed as off-chain modules and their data will be stored on mongo database. However, we keep hashes registered with the Blockchain to verify the integrity of the data, mainly with payment transactions. This is done by comparing the hash stored in the Blockchain with the one calculated from data stored in MongoDB. The matching algorithm between offers and requests is fed by Mongo data and is performed in an off-chain module.

2.1.2. Asset type abstraction

The terminology used for the asset type creates some confusion: material; immaterial; immaterial not quantified; immaterial with unit not measurable. The goal is to update this terminology to make it more 'understandable' and appropriate. We abstract more common transactions and change the terminology according to the table below.



Terminology ODEP V2	Terminology 2023 ODEP V3
Material with sale/purchase transaction Multiple asset of one assetType Asset quantity=1 Immaterial One asset per assetType Asset quantity>=0 Offer/request quantity>0 ImmaterialNotQuantified One asset per assetType Asset quantity=0 Offer/request quantity >0	MeasurableByQuantity Multiple assets on one assetType With Asset quantity >=0 if (Asset.subjectOfQuantity=true) Offer/request quantity>0
Immaterial with unit not measurable Multiple assets on one assetType Asset/offer/request Quantity=0 Many contracts on the same offer (multiple access on the asset) Material with rent transaction Multiple asset of one assetType Asset/offer/request quantity=1	MeasurableByTime Multiple assets on one assetType Many contracts on the same offer (if Asset. multipleAccess=true) (quantity=1 with rent transaction(Asset. multipleAccess=false))

2.1.3. Payment By instalments

This involves settling the payment based on a defined periodicity. Two types of payments will therefore co-exist:

- Total payment at the end of the period.
- Periodic payment: Periodic payment is triggered with the beginning of delivery, and then it is the responsibility of the business application to send interim measurements for payment.

2.2. Dedicated RESILINK ODEP instance

A dedicated instance of ODEP is instantiated for RESILINK exclusive usage.

- Authentication server: In order to call our APIs, a user will need an access token for his client application. Access token is used to identify the client application and allow it to make calls to the APIs. We are based on Orange Authorization server's implementation (OAuth), which is based on the Client Credentials Grant flow of the OAuth 2.0 specification. All APIs are secured via a valid access token retrieved from the Authentication layer. The Authentication server is accessible on this link:

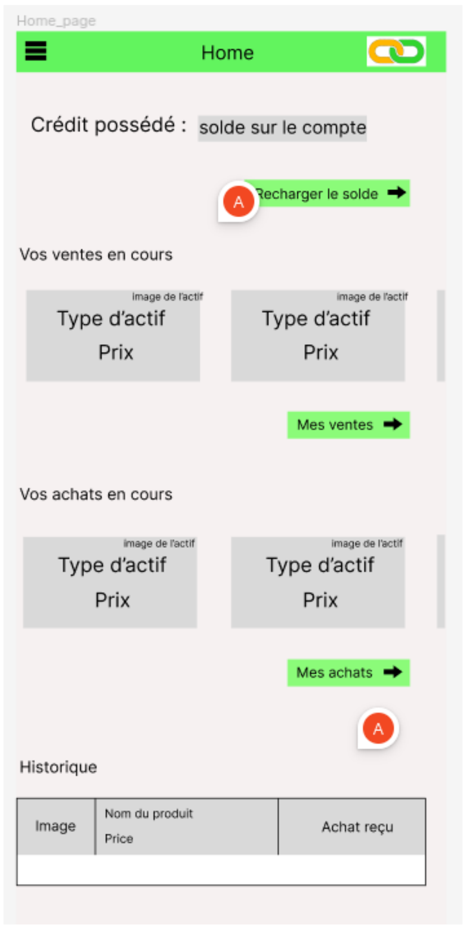
  http://90.84.194.104:4000/oauth/api/v1.0.0/api-docs/#/signin/signinUser

- ODEP-REST server: use smart contracts addresses and ABI (Application Binary Interface) values, which are the interfaces' values of the generated smart contracts, to trigger different transactions. Some off-chain treatments are developed on this side, such as offers/requests matching and ranking per price, sending notifications and also treating Key Performance Indicators data. It is accessible on this link: http://90.84.194.104:10010/docs/#/

3. RESILINK MOBILE APPLICATION

3.1. Architecture & link with ODEP

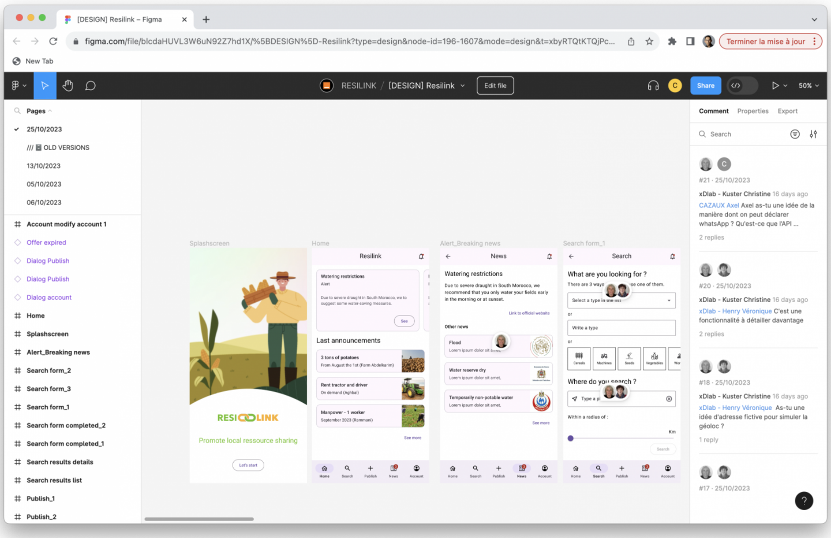
At first, we went for a really simple design with clear blocks. The main idea was to help users, especially those less familiar with smartphones, easily find different paths on an application page (like the example shown on the home page).

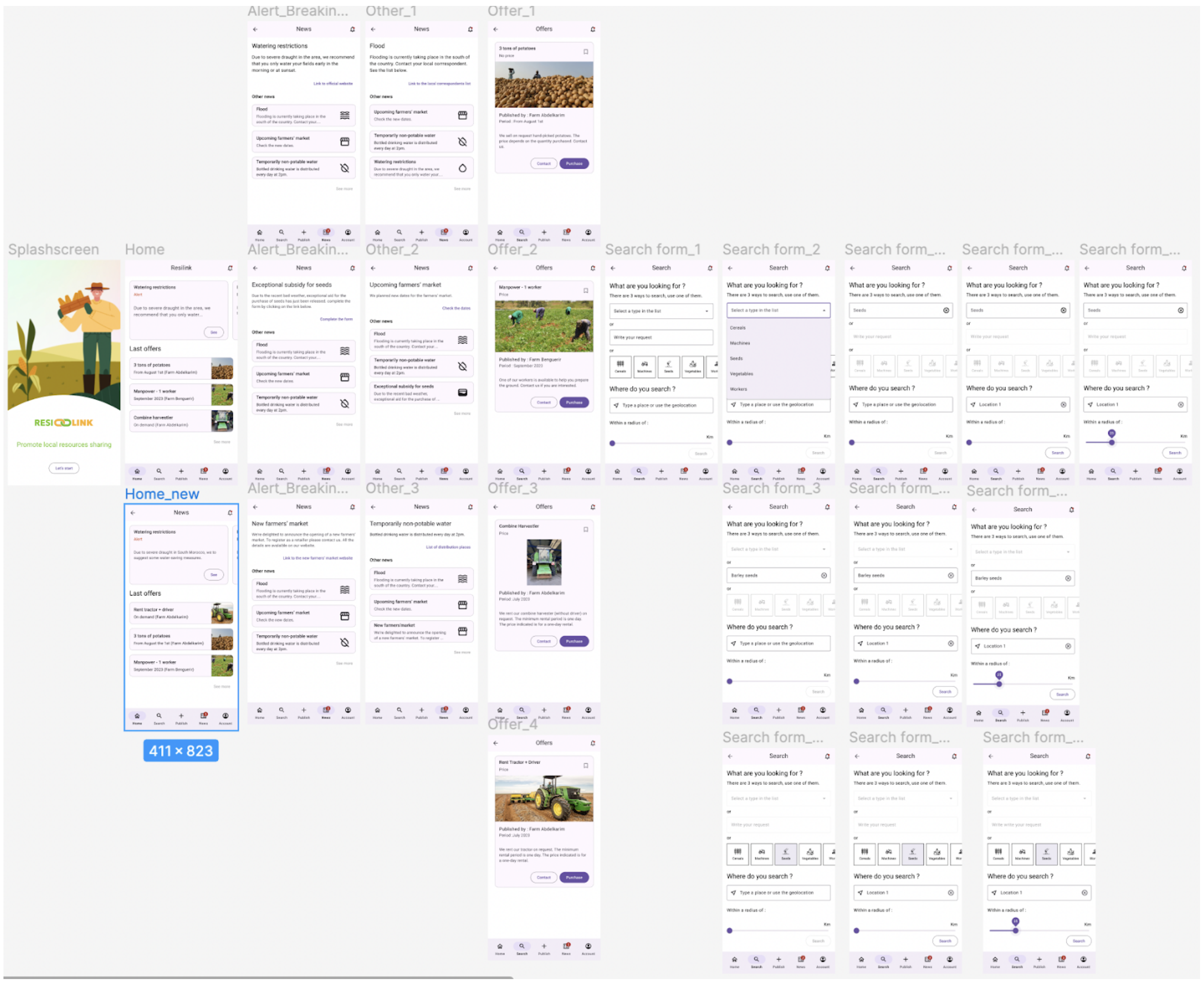


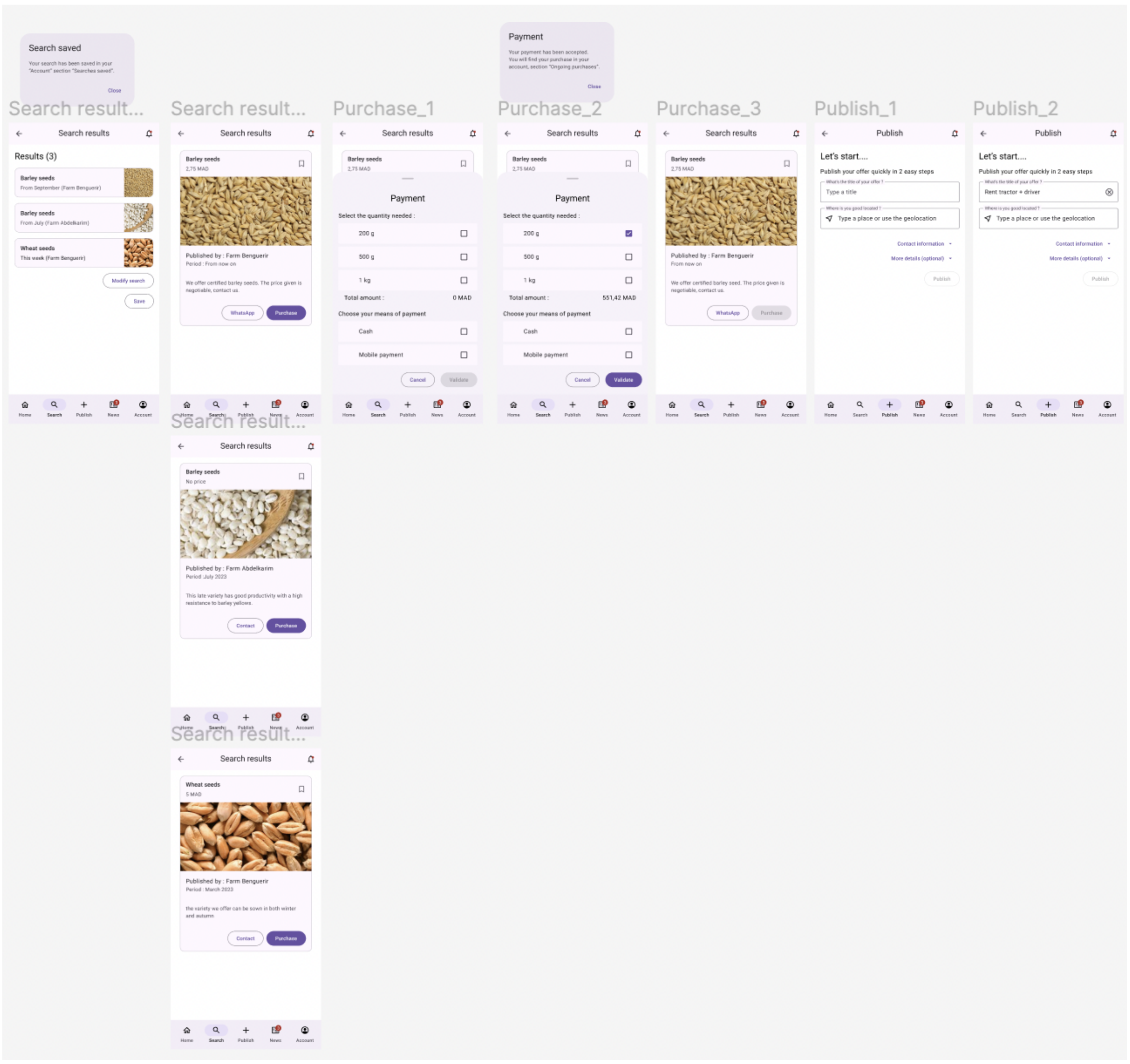
Furthermore, the initial app versions mandated users to possess an account for ODEP API functionality access, and consequently, access to diverse services and products. This version guided users straight to the login screen before reaching the app's services. After the General Meeting in Rabat in June 2023, several decisions have been made, particularly regarding the application's user experience. As ODEP functions as a REST API, each service utilization mandates requests, alongside a user key. In the app's early stages, a user login facilitated seamless interaction with ODEP, evolving into a more contemporary approach incorporating access to the "store." This approach necessitates only one requirement: having an account configured as a public account shared among all users.

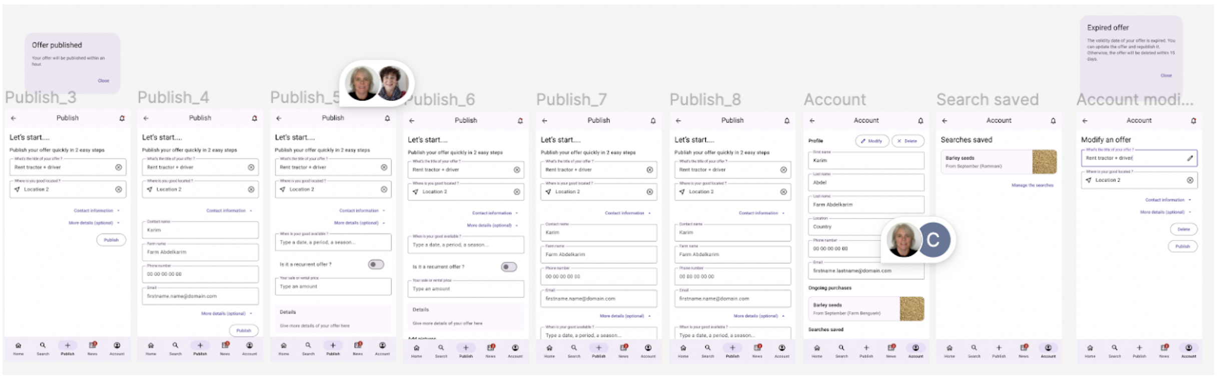
3.2. New FIGMA-based mockup

A new application mock-up was therefore created using Figma to meet this new vision of the application. The following figure shows each available screen.






Search saved
Your search has been saved in your "Account" section "Searches saved".
Close
Payment
Your payment has been accepted. You will find your purchase in your account, section "Ongoing purchases".
Close
Search result...
Search result...
Purchase_1
Purchase_2
Purchase_3
Publish_1
Publish_2
Search results
Results (3)
Barley seeds
From September (Farm Benguerir)
Barley seeds
From July (Farm Abdelkarim)
Wheat seeds
This week (Farm Benguerir)
Modify search
Save
Barley seeds
2,75 MAD
Published by : Farm Benguerir
Period : From now on
We offer certified barley seeds. The price given is negotiable, contact us.
WhatsApp
Purchase
Payment
Select the quantity needed :
200 g
500 g
1 kg
Total amount :
0 MAD
551,42 MAD
Choose your means of payment
Cash
Mobile payment
Cancel
Validate
Publish
Let's start....
Publish your offer quickly in 2 easy steps
What's the title of your offer ?
Type a title
Rent tractor + driver
Where is you good located ?
Type a place or use the geolocation
Contact information
More details (optional)
Publish
Home
Search
Publish
News
Account
Barley seeds
No price
Published by : Farm Abdelkarim
Period : July 2023
This late variety has good productivity with a high resistance to barley yellows.
Contact
Purchase
Wheat seeds
5 MAD
Published by : Farm Benguerir
Period : March 2023
the variety we offer can be sown in both winter and autumn.
Contact
Purchase

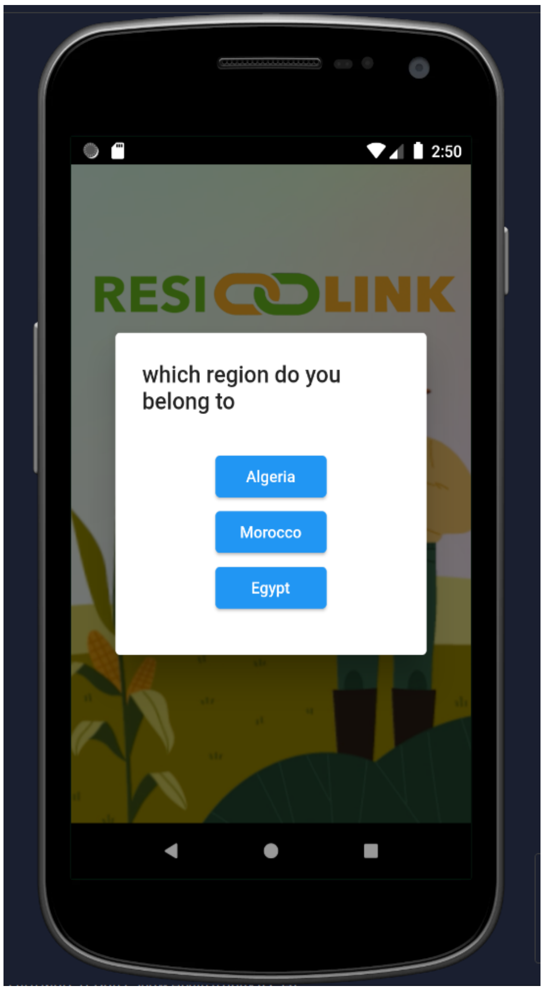

Offer published
Your offer will be published within an hour.
Close
Expired offer
Close
Publish_3
Publish_4
Publish_5
Publish_6
Publish_7
Publish_8
Account
Search saved
Account modi...
Publish
Let's start....
Publish your offer quickly in 2 easy steps
Rent tractor + driver
Location 2
Contact information
More details (optional)
Publish
Karim
Farm Abdelkarim
00 00 00 00 00
firstname.name@domain.com
Type a date, a period, a season...
Is it a recurrent offer ?
Type an amount
Details
Account
Profile
Modify
Delete
Karim
Abdel
Farm Abdelkarim
Country
00 00 00 00 00
firstname.lastname@domain.com
Ongoing purchases
Barley seeds
From September (Farm Benguerir)
Searches saved
Barley seeds
From September (Farmsel)
Manage the searches
Modify an offer
Delete
Publish
Home
Search
Publish
News
Account

3.3. Proof of concept

In this proof of concept (PoC), as mentioned earlier, we will use the Flutter framework and the ODEP API to craft our application. As it stands, the application starts with a loading screen, requiring the user to wait while a JSON configuration file is fetched from GitHub and processed by the software. Later, this configuration file will be moved and hosted by an intermediary platform.

On this screen, the program checks the application's local configuration registry to determine if a language has been previously selected. If so, it stores this value and applies it to the application; if not, it defaults to the language defined in the smartphone's system and sets this as the default language in local configuration registry.

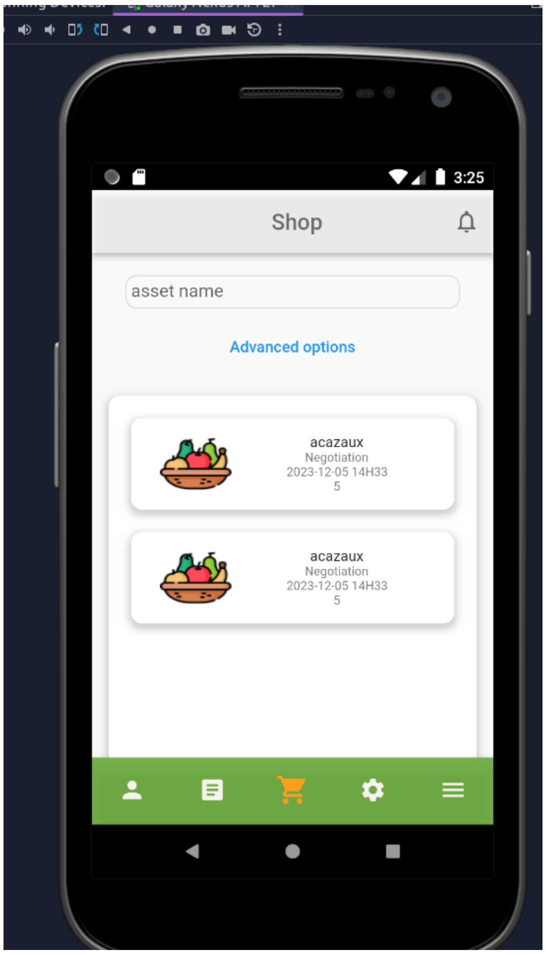
The same logic applies to verifying a utility connection. The process involves checking if a user structure is registered in local memory. If it exists, the application sends a connection request to the API to acquire the user's key; if not, a connection is established with the pre-created public account, allowing the user access to specific functions. Additionally, an asynchronous function runs concurrently to fetch the key corresponding to the currently connected account every two hours.



Following that, the user is directed to the "store" page. Here, utilizing the navigation bar, they can sequentially access the "profile," "news," "store," "settings," and "menu" pages. From this page, the user can initiate a search either by pressing the advanced search button or simply typing a name into the search bar.

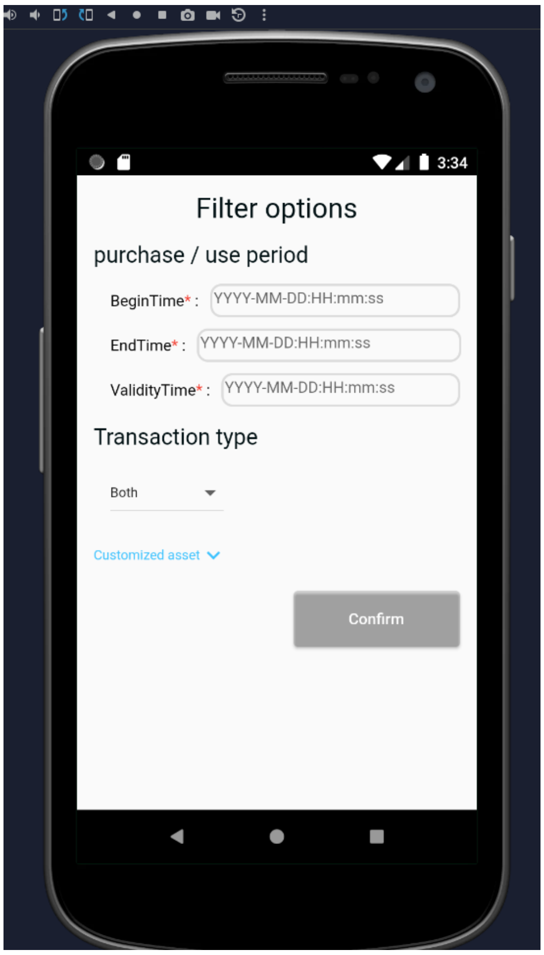
The shop page displays available and valid offers for sale, encompassing services, products, manpower, etc. It provides crucial details such as the offer's image, the offerer's nickname, price, the deal's end date, and the remaining quantity if applicable. Furthermore, if the user wishes to filter offers, they can click on "Advanced options" to explore restrictions that can be added to the filter.

For more detailed information about an offer, tapping on it will open a pop-up displaying all the relevant details and suggesting the user initiate a contract.

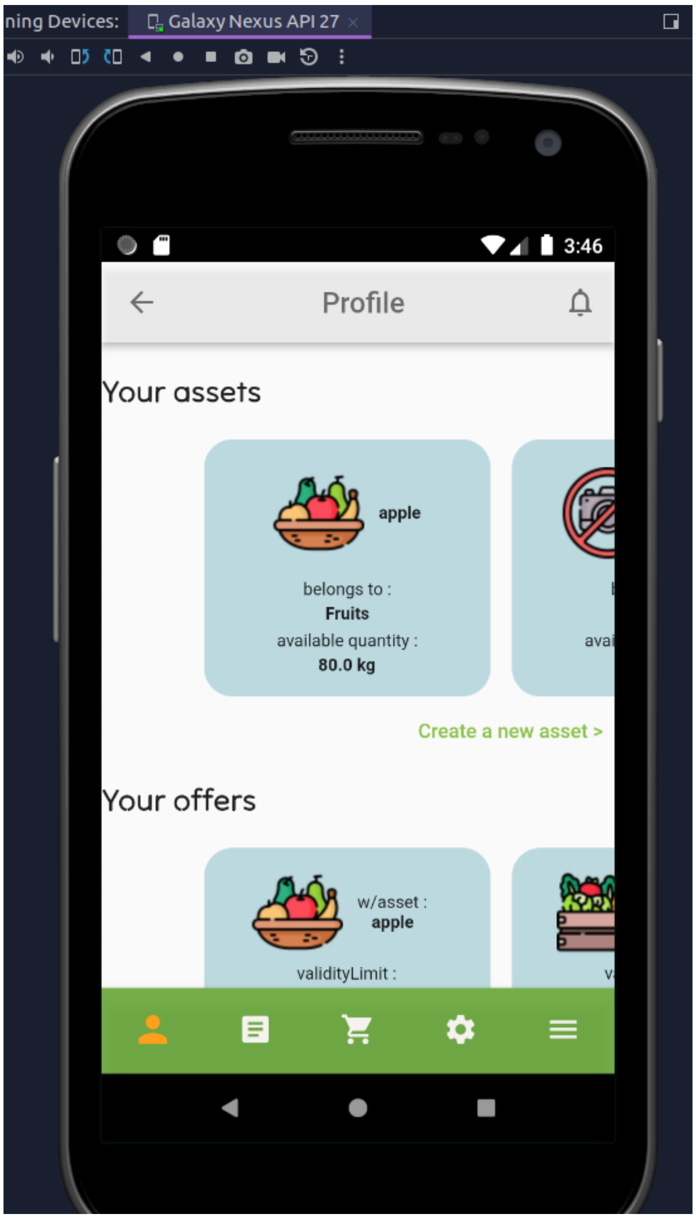


acazaux
Negotiation
2023-12-05 14H33
5

acazaux
Negotiation
2023-12-05 14H33
5

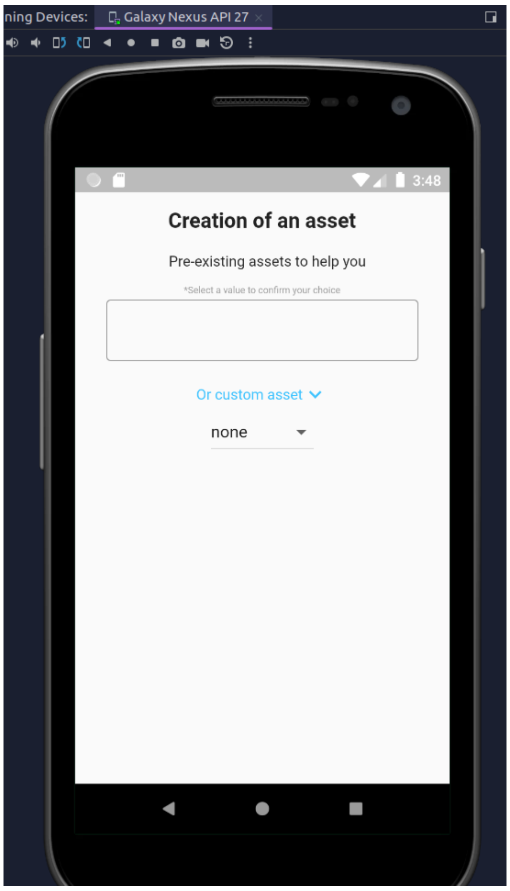

3:34
Filter options
purchase / use period
BeginTime* : YYYY-MM-DD:HH:mm:ss
EndTime* : YYYY-MM-DD:HH:mm:ss
ValidityTime* : YYYY-MM-DD:HH:mm:ss
Transaction type
Both
Customized asset
Confirm

On the profile page, once logged in, the user will be able to see a list of his assets, offers and contracts. They can create an offer and an asset via this page, as well as modify or delete their assets/offers/contracts.



When generating an asset, the user needs to choose the desired asset type and then input the attributes for the asset. There's an option to select an asset from a predefined list, which will automatically fill in some of the fields.

Furthermore, for specific types of asset specifications, the user will be presented with multiple choices, allowing them to finalize the asset based on their preferences.



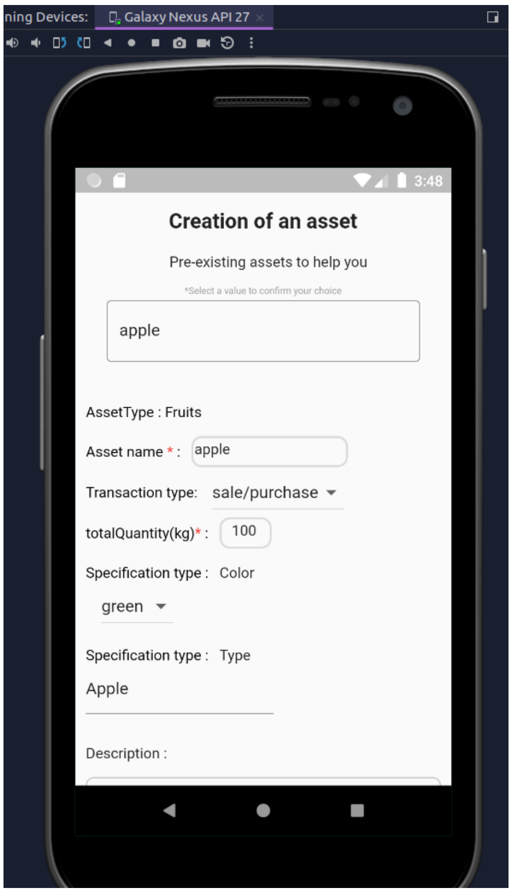
Creation of an asset

Pre-existing assets to help you

*Select a value to confirm your choice

Or custom asset

none



Creation of an asset

Pre-existing assets to help you

*Select a value to confirm your choice

apple

AssetType : Fruits

Asset name * : apple

Transaction type: sale/purchase

totalQuantity(kg)* : 100

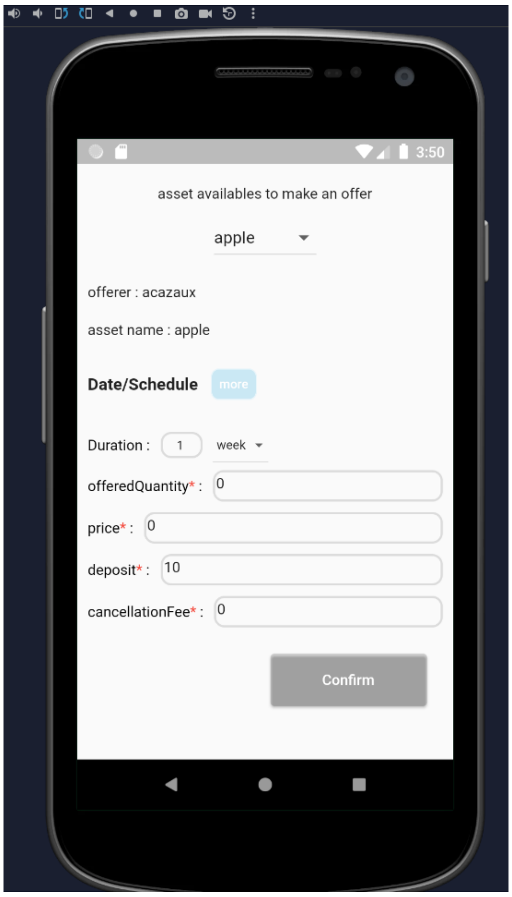
Specification type : Color

green

Specification type : Type

Apple

Description :



asset availables to make an offer

apple

offerer : acazaux

asset name : apple

Date/Schedule more

Duration : 1 week

offeredQuantity* : 0

price* : 0

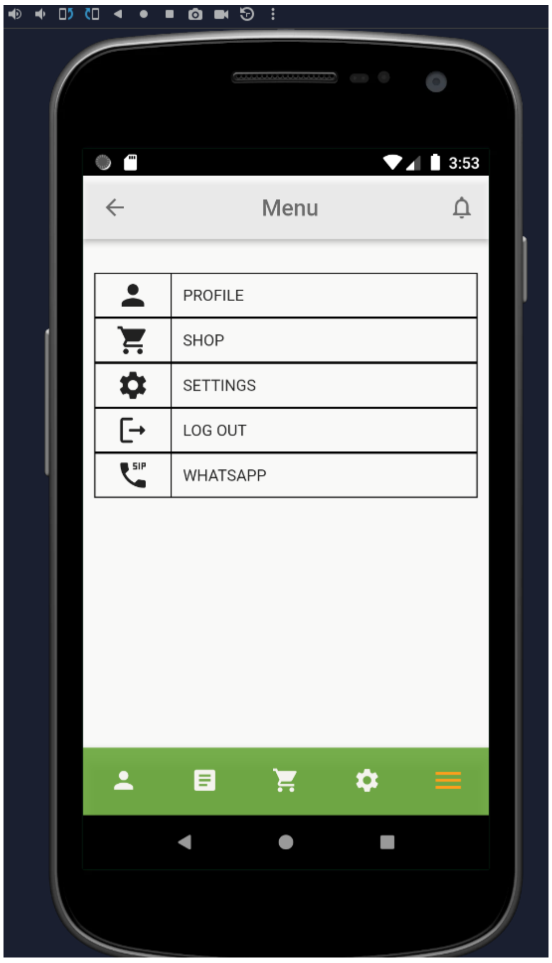
deposit* : 10

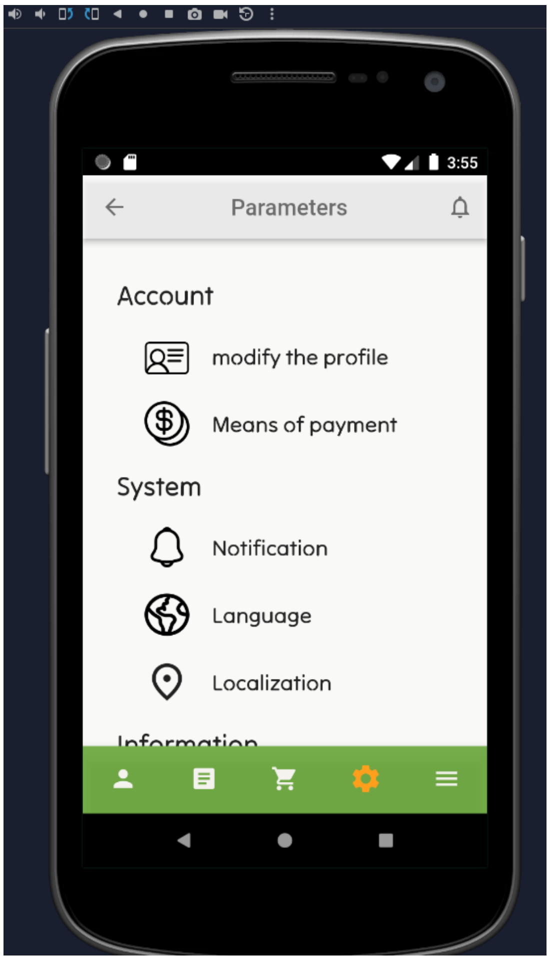
cancellationFee* : 0

Confirm

To generate an offer, the user needs to pick the asset associated with the offer. Once an asset has been selected, the user can proceed to input all the necessary values required by the asset for the offer to be successfully created in the database.

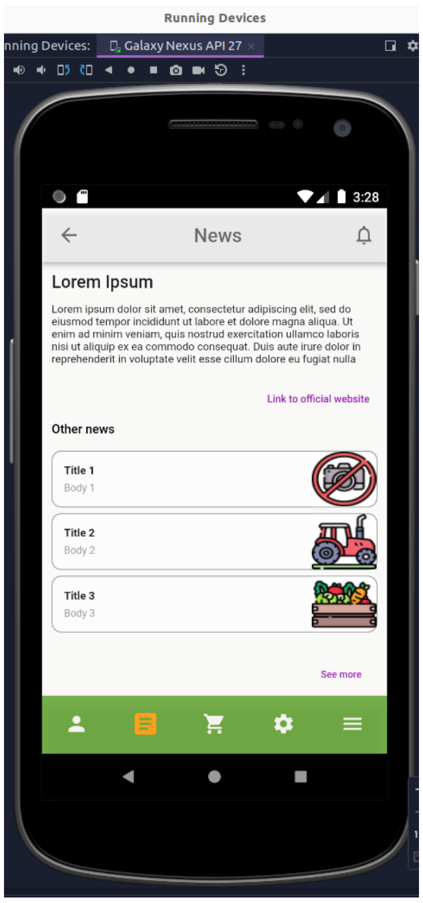
In the main navigation (right figure), there are the settings and menu screens. The purpose of the menu is to provide an additional link between all the pages and the new features that will be added over time. This is where they will be available.





The settings page (right figure) contains various parameters, some of which can be accessed even without the user being logged in. The parameters currently implemented are user profile modification and application language change.

The news page displays the latest article in the list, with a direct link to the article's web page and a list of articles below it.



Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla

Link to official website

Other news

Title 1
Body 1

Title 2
Body 2

Title 3
Body 3

See more

3.4. Main difficulties for the proof of concept

One important issue is that the ODEP API is currently unavailable for iOS. Since the ODEP API operates as a REST API, utilizing a web link, it poses a security concern. Unfortunately, iOS does not accept any interaction with a non-secure website (https yes / http no).

Some issues have been addressed (solutions may change):

- Capturing photos from the gallery or camera.
- Storing photos using the IMGUR API, enabling the storage and retrieval of images through an HTML link and fetching.
- Pre-completion of assets and a word list to aid in filling offers based on assetType, saved in an internal JSON.
- Implementation of a system to retrieve a configuration file containing data related to the user's country upon the application's initial launch or request.

Some problem still need to be solved:

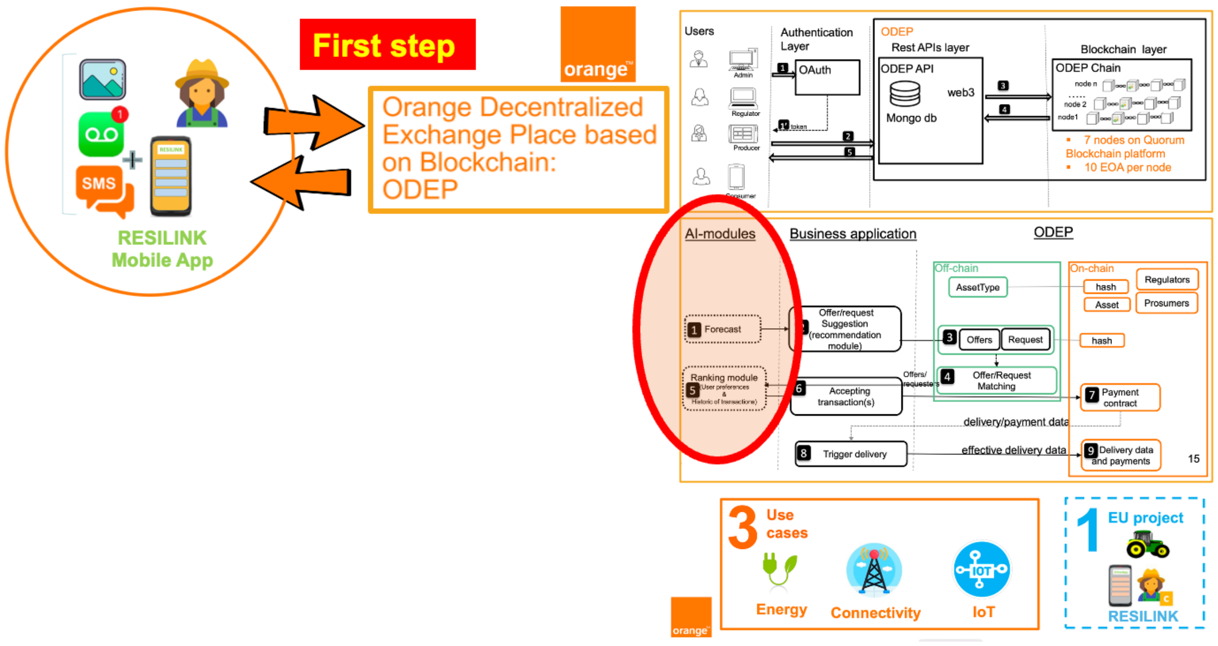
- Consistent, modern and easy-to-understand UX.
- IOS permissions not fully tested.
- Have a real server for photos. IMGUR has a maximum size for storing photos (in progress).

3.5. Deployment of the mobile application

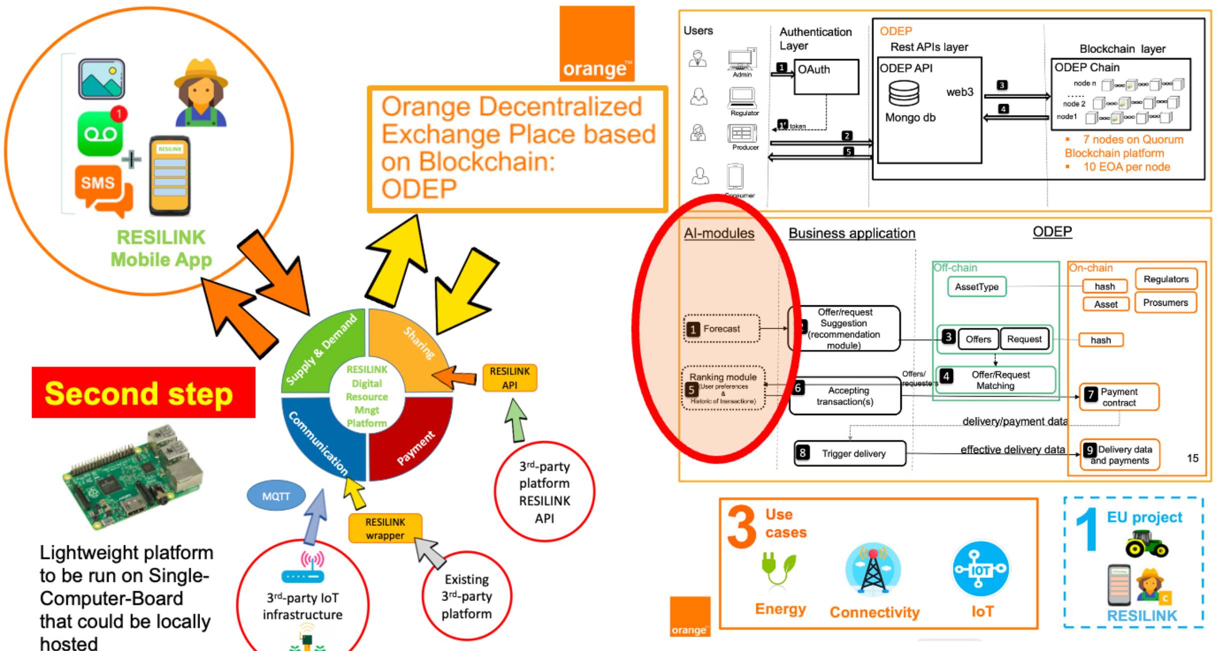
A revised deployment method has been explored and chosen, building upon the deployment method previously tested in D5.2a. The updated approach involves placing the application on Dropbox cloud, with a QR code linked to the URL provided by Dropbox. By simply scanning the QR code, the application file will be installed. The QR code is accessible on the shared drive RESILINK-PROJECT/RESILINK-MOBILE-APP or below.

4. RESILINK PLATFORM

Building on the studies outlined in section 2, RESILINK suggests the implementation of a dedicated intermediate RESILINK platform to serve as a front-end to the ODEP platform. The illustration below depicts this concept.



From Orange ODEP presentation. Salma Bradai, Sofrecom



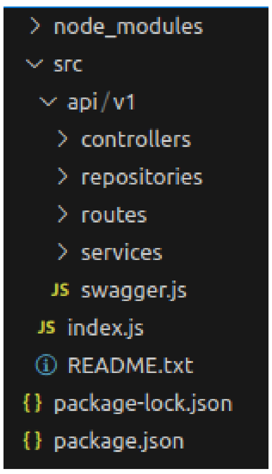
From Orange ODEP presentation. Salma Bradai, Sofrecom

The server will function as an API intermediary between the ODEP API and our mobile application. Currently, the server is not deployed for open access but is exclusively available for developers.

Similar to the ODEP API, this server will also leverage Swagger technology. This approach allows for the automatic generation of documentation, streamlining testing, and facilitating seamless interchange between different versions of our server in the future.

4.1. Architecture

Since we are still in the developmental phase, we chose a simple server architecture and employed a backup file as our temporary database. It is important to note that a genuine database will be required for effective scaling in the future. The current server is structured as follows – it is the result of a first PoC carried out by Manuel Munier from UPPA with the help of Marc Despland from Orange.



```
> node_modules
v src
  v api / v1
    > controllers
    > repositories
    > routes
    > services
    JS swagger.js
  JS index.js
  ⓘ README.txt
{} package-lock.json
{} package.json
```

In the root directory, the server configuration files with packages and versions, along with various modules essential for server operation, will be situated.

The "src" child folder will house the server code, encompassing the "index.js" file and the "api" folder. The "index.js" file manages incoming calls (get, post, etc.) and defines the paths to the files/folders to be utilized, specifying the server version.

Within the version-specific folder (e.g., v1), there are the "swagger.js" file, along with the "controllers," "repositories," "routes," and "services" folders.

The swagger file facilitates the definition of how URLs for accessing documentation are formed and allows the addition of extra functionalities.

Files within the "controllers" folder serve as controllers responsible for managing HTTP requests, validating data, and interacting with the appropriate services.

The "services" folder contains files with the application's business logic, handling data manipulation, communication with models, and execution of domain-specific tasks.

In the "routes" folder, files define API entry points and determine which controller methods should be called in response to HTTP requests.

Finally, in the "repositories" folder, files are responsible for accessing data from databases, file systems, or other sources. They encapsulate data access logic and enable the rest of the application to work with data in an abstract manner.

4.2. Proof of concept

Orange and UPPA investigated the technology building blocks: NodeJS, Javascript, and conducted tests on HTTP REST API.

The PoC developed by Manuel Munier consists of two servers. The initial API intended for RESILINK, designed to act as an intermediary platform for data filtering, is coded in Node.js (Javascript) using Express and Swagger technology to simplify server construction and documentation.

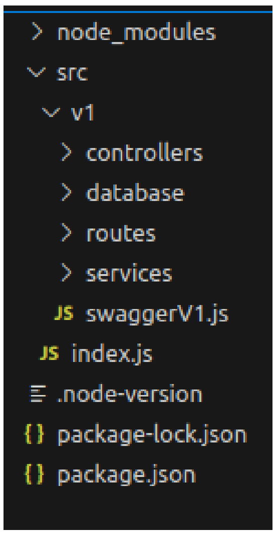
In addition to this server, a second Java server, also utilizing Swagger, serves as the ODEP API. Its role is focused on data retrieval and communication.

As of now, the PoC is operational, with both servers communicating effectively with each other.

4.3. Deployed platform

The platform is currently hosted on a free server provided by the hosting service Render, and one can access the Swagger documentation for the platform here (https://resilink-api.onrender.com/v1/api-docs/).

The platform is a server platform constructed with Node.js, hence utilizing Javascript. The use of Javascript allows us to leverage Swagger, a documentation library that is also applicable to a Java server, and Express, a framework enhancing the robustness of server-side applications. Additionally, we have integrated a MongoDB (NoSQL) database with our server. This is particularly beneficial for storing data in memory that ODEP doesn't consider (e.g., a user's telephone number).



```
> node_modules
v src
  v v1
    > controllers
    > database
    > routes
    > services
    JS swaggerV1.js
  JS index.js
≡ .node-version
{} package-lock.json
{} package.json
```

The server architecture mirrors that of the PoC, with the "repositories" folder now labeled as "database."

The server's purpose is to redirect and transform data transmitted to and received from ODEP. All requests initially directed to ODEP have been replicated with the same header (path after the domain URL). The distinction lies in the domain name, and requests are now directed to an HTTPS address rather than HTTP, making it compatible with iPhones. All data transformation and calculation functions are now transferred from the mobile application to the server to enhance the telephone's performance.

Furthermore, it is possible to store additional data and associate it with other data in the database. If users choose to provide them, we can retrieve and save them in the database, associating it with their ODEP ID for future retrieval. To avoid complications with data setup, we simply need to exclude them from the data intended for transmission to ODEP.

ACRONYMS LIST

Acronym	Explanation
ABI	Application Binary Interface
API	Application Programming Interface
APK	Android Package Kit
HMI	Human Machine Interface
HTTP	Hyper Text Transfer Protocol
JSON	JavaScript Object Notation
ODEP	Orange Decentralized Exchange Place
PoC	Proof-of-Concept
QR code	Quick-Response code

PROJECT CO-ORDINATOR CONTACT

Pr. Congduc Pham

University of Pau

Avenue de l'Université

64000 PAU

FRANCE

Email: Congduc.Pham@univ-pau.fr

ACKNOWLEDGEMENT

This document has been produced in the context of the PRIMA RESILINK project. The RESILINK project consortium would like to acknowledge that the research leading to these results has received funding from the European Union through the PRIMA program.