



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

Deliverable D2.2b

RESILINK resource sharing digital platform – v2

Responsible Editor:	ORANGE
Contributors:	UPPA
Document Reference:	RESILINK D2.2b
Distribution:	Public
Version:	1.1
Date:	July 2025

CONTRIBUTORS TABLE

DOCUMENT SECTION	AUTHOR(S)
SECTION 1	C. Pham (UPPA)
SECTION 2	A. Cazaux (UPPA)
SECTION 3	A. Cazaux (UPPA)
SECTION 4	A. Cazaux (UPPA)
SECTION 5	C. Pham (UPPA) & M. Despland (Orange)

DOCUMENT REVISION HISTORY

Version	Date	Changes
V1.1	July 11 th , 2025	PUBLIC RELEASE
V1.0	June 16 th , 2025	FIRST DRAFT VERSION FOR INTERNAL APPROVAL
V0.1	April 25 th , 2025	FIRST RELEASE FOR REVIEW

EXECUTIVE SUMMARY

Deliverable D2.2b describes the RESILINK platform v2 that runs behind the RESILINK mobile app to implement all the functionalities for sharing resources. It also describes the open API approach to enable third-party applications to use the RESILINK platform to build their own exchange platforms. A use case will illustrate how a RESILINK server can be used for other applications.

TABLE OF CONTENTS

1. Introduction	5
2. Quick review of the RESILINK framework	5
3. Evolution of the RESILINK platform	7
3.1. RESILINK platform – v1	7
3.2. RESILINK platform – v2	8
4. RESILINK platform – v2	10
4.1. Architecture	10
4.2. Installing & deploying RESILINK platform v2	12
4.3. Use & Validation of RESILINK v2 for the Living-Labs	13
4.4. Test of v2 on Raspberry Pi	14
4.5. Future work	16
5. First test of open-API & Interoperability	17
5.1. Motivations	17
5.2. Orange use-case	18
5.2.1. General view	18
5.2.2. The story of LockItSolar	18
5.2.3. Technical view	19

1. INTRODUCTION

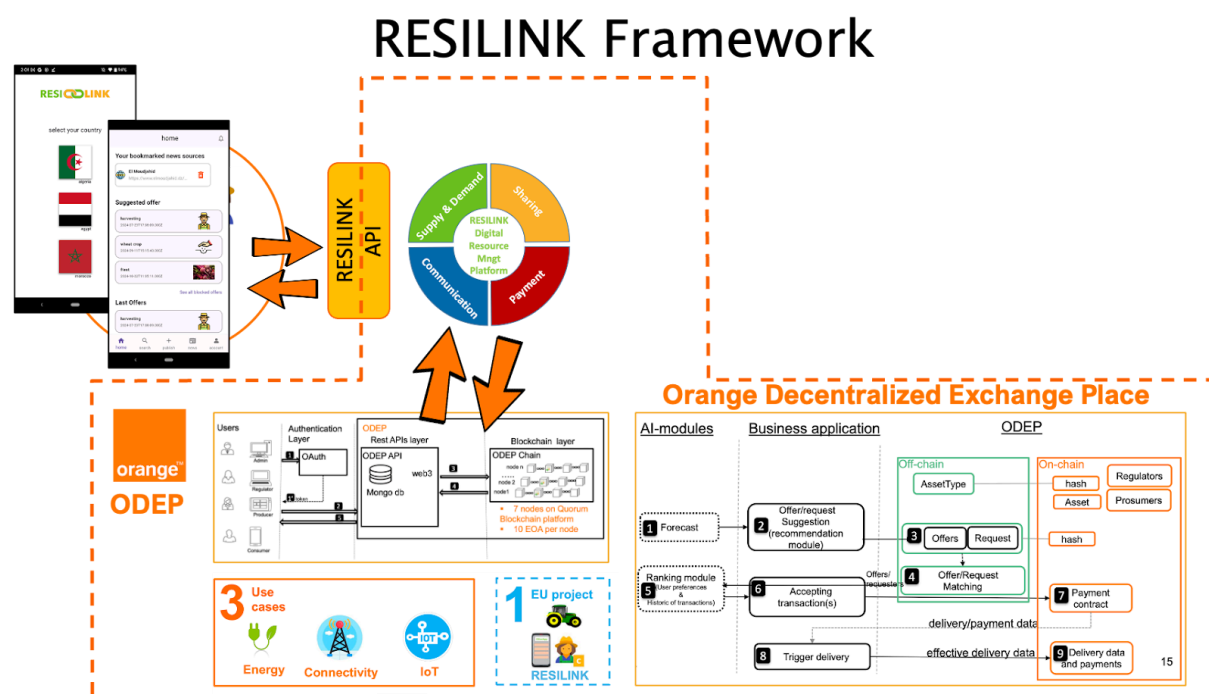
RESILINK develops a distributed digital resource management platform for real-time exchange of information on territorial resources and supplies & demands; connecting smallholders to new supply, sharing opportunities and distribution channels. In addition, RESILINK will incrementally use cutting-edge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

The lowest layer 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, referred to as **RESILINK mobile app**, that will be the main interface to simply, quickly and intuitively manage resources. An intermediate platform, referred to as the **RESILINK platform**, will stand between the RESILINK mobile app and the ODEP platform to implement additional functionalities on top of ODEP.

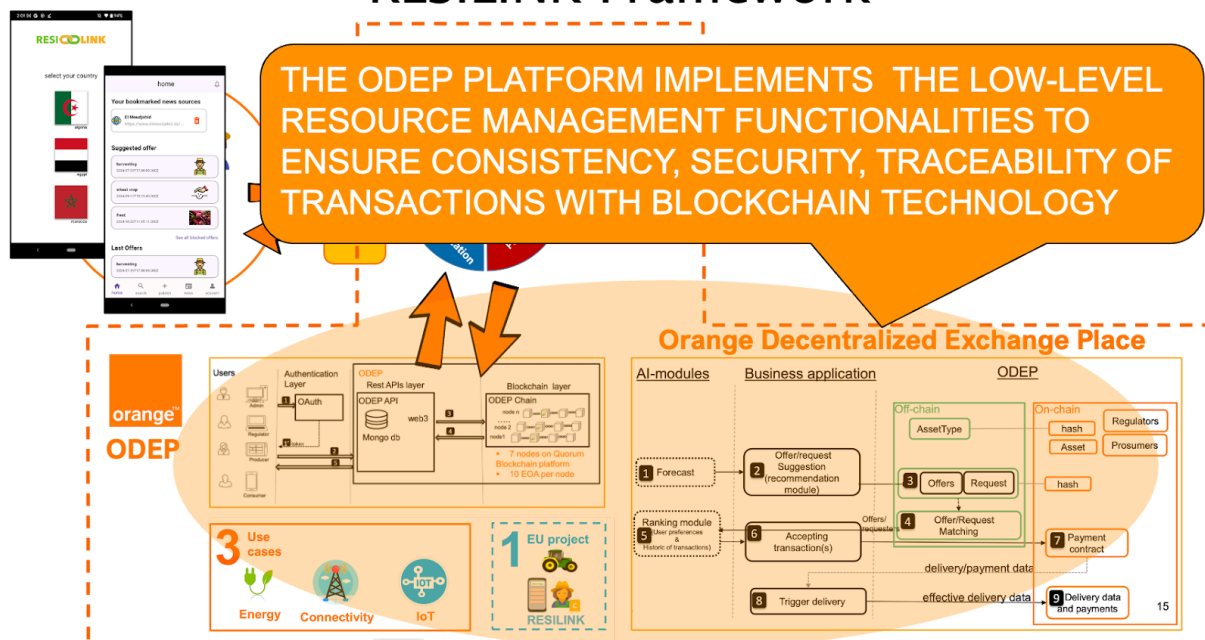
2. QUICK REVIEW OF THE RESILINK FRAMEWORK

Big picture



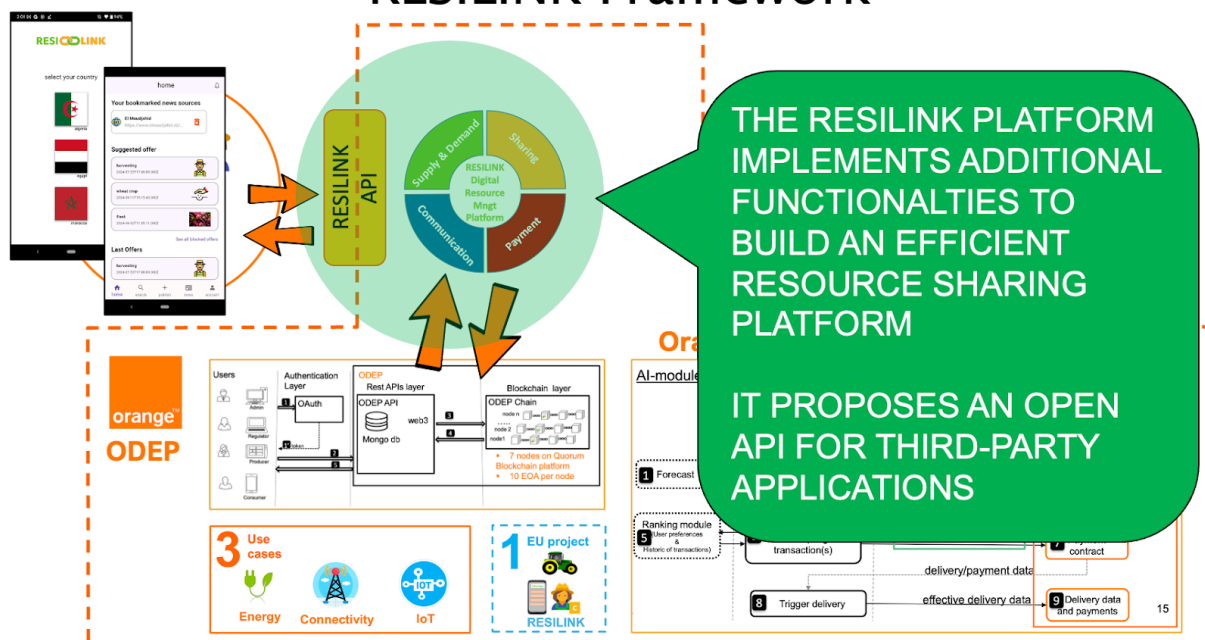
ODEP layer

RESILINK Framework

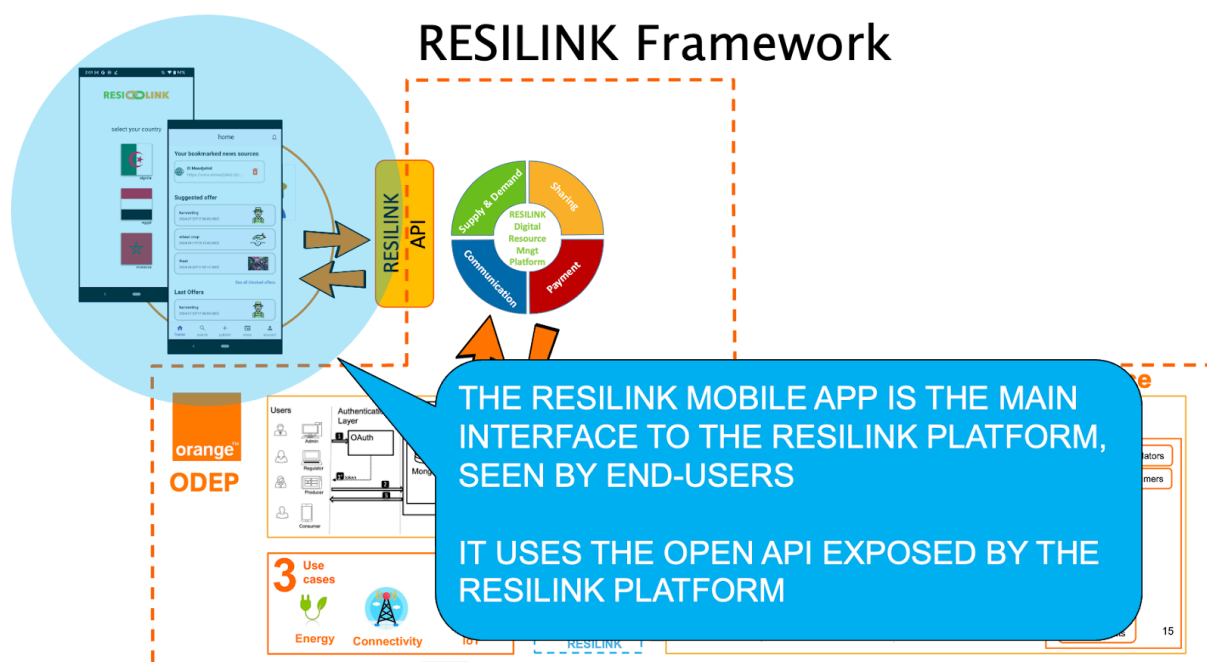


RESILINK platform

RESILINK Framework



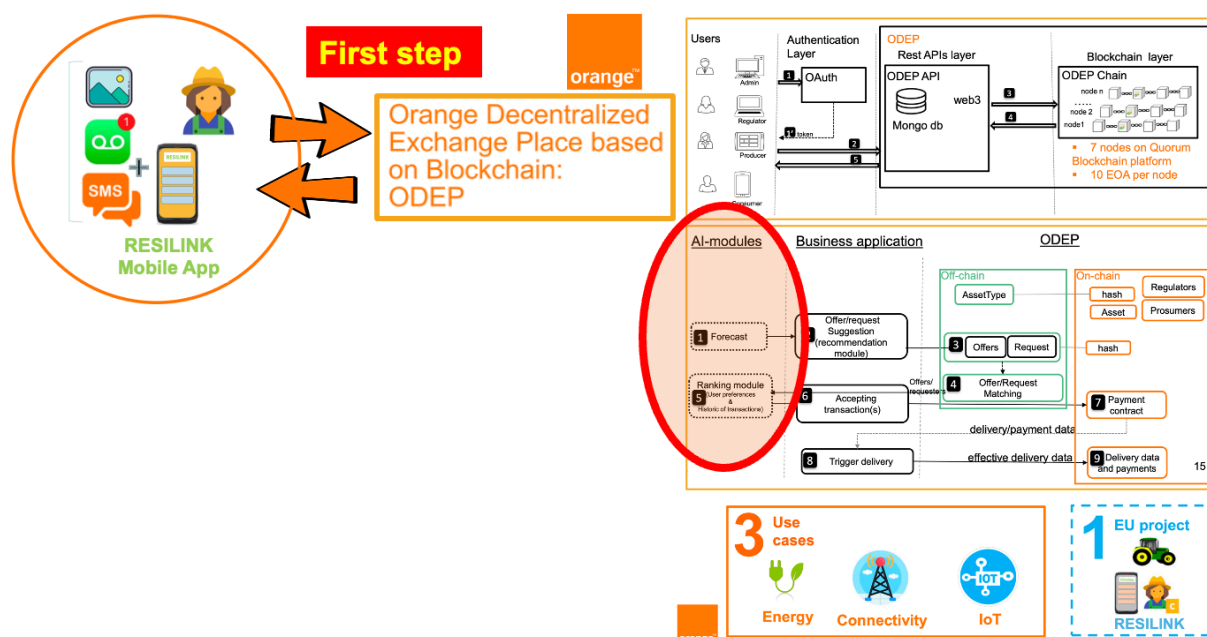
RESILINK mobile application



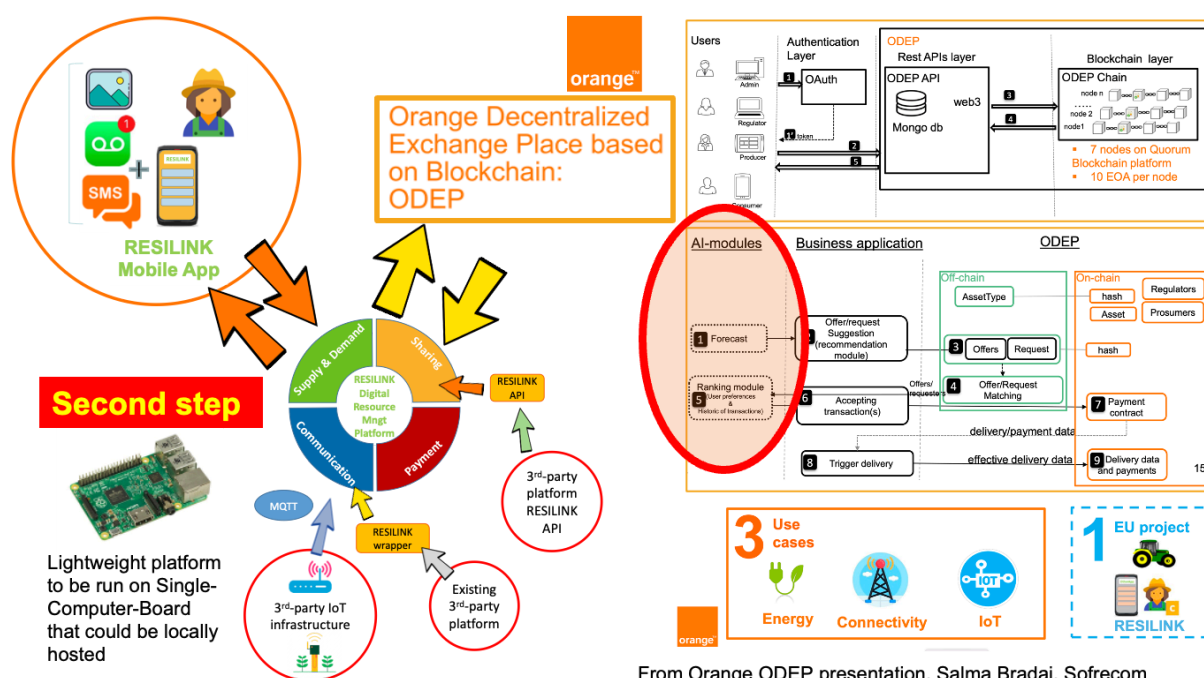
3. EVOLUTION OF THE RESILINK PLATFORM

3.1. RESILINK platform – v1

RESILINK implements a dedicated intermediate RESILINK platform to serve as a front-end to the ODEP platform. The illustration below depicts this concept: the first step was the proof-of-concept demonstrator at the very beginning of the project.



From Orange ODEP presentation. Salma Bradai, Sofrecom



In the second step (RESILINK platform v1) The RESILINK platform will function as an API intermediary between the ODEP API and the RESILINK mobile app.

3.2. RESILINK platform – v2

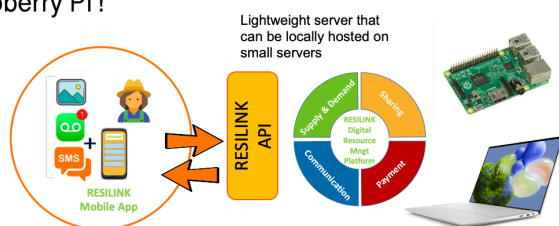
The main motivations behind the RESILINK platform v2 are:

1. Provide a very lightweight server that could be deployed out-of-the-box on laptops or even single board computers such as a Raspberry Pi
2. This lightweight server should be able to manage most of RESILINK platform core functionalities: news feeds, publication of offers, search for offers and contact users
3. Enable the deployment of several lightweight RESILINK servers, where each server could have a geographical scope to efficiently manage crisis in a very local manner
4. Incremental deployment of servers could then be also enabled to provide a very level of flexibility
5. Enable a full platform-of-platforms approach where the RESILINK server can be deployed, possibly adapted and used, through the proposed open API, for other applications, thus stimulating local innovation

These motivations were defined from the very beginning of the project as illustrated by the following 3 slides.

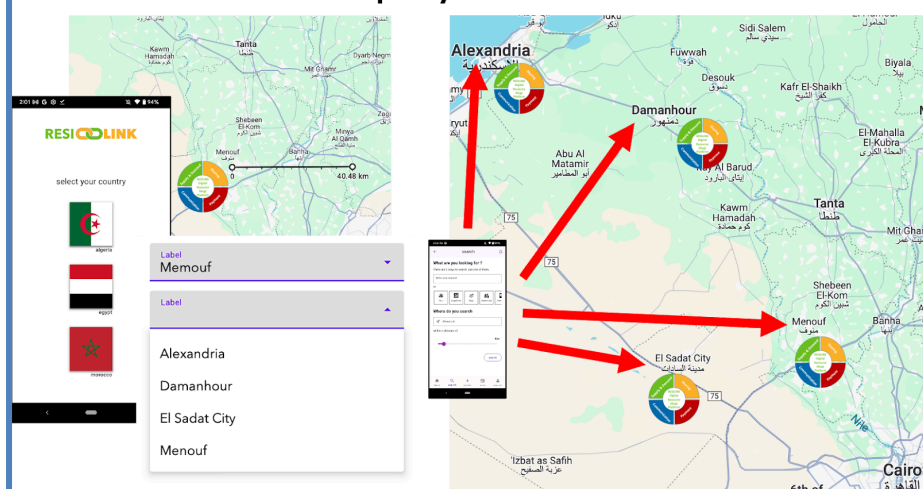
Light-weight servers & fast deployment

- The RESILINK digital platform server can run on a small server, even on a Raspberry Pi!



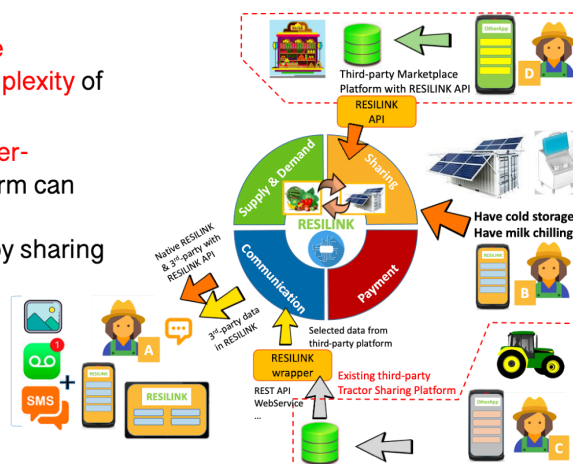
- A RESILINK server can be **locally deployed** by local city government agencies, agriculture cooperatives, agriculture services, ... in 1 hour!

Incremental deployment



Stimulating local innovation?

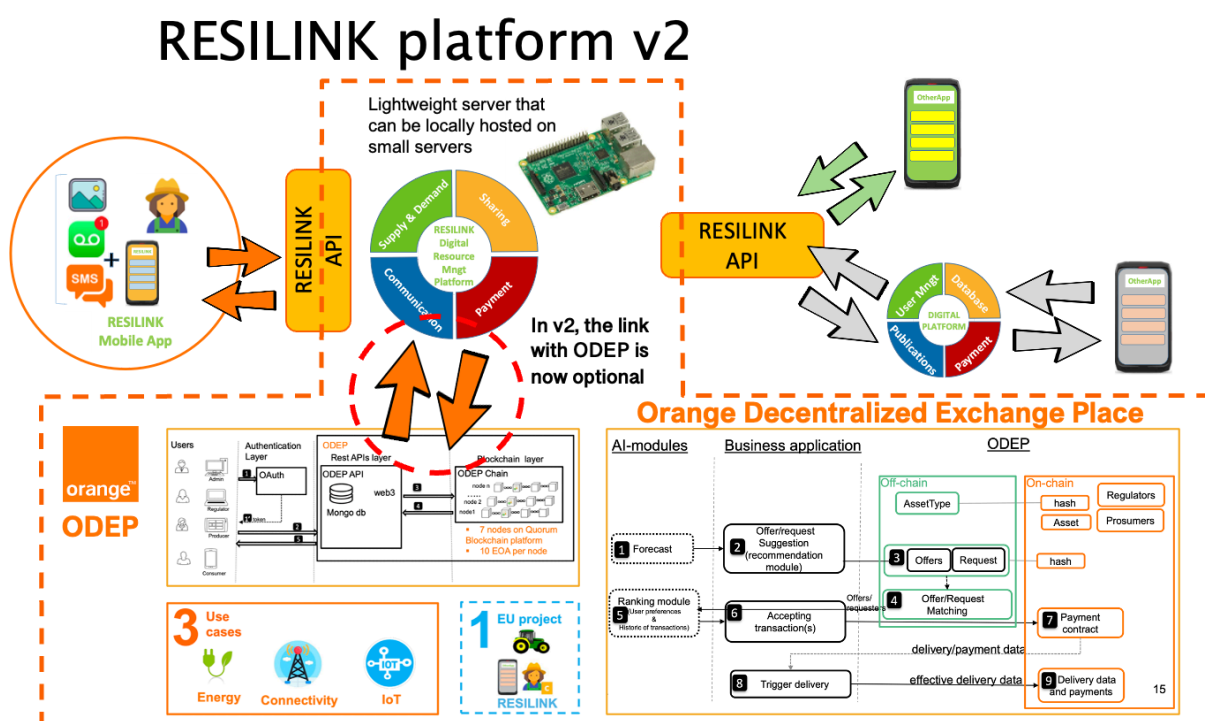
- The open API will **reduce development time & complexity** of new platforms
- All platforms **can fully inter-operate** and a new platform can benefit from all the other platforms' communities by sharing users & contents
- RESILINK develops a consistent API to **enable fast development and deployment** platforms



4. RESILINK PLATFORM – v2

4.1. Architecture

The architecture of the RESILINK platform v1 has been modified to make the link with ODEP optional in order to be able to provide a lightweight server that could be deployed out-of-the-box. **This would mean that some of ODEP functionalities are now embedded into the RESILINK server.**



In the initial setup, the RESILINK mobile application relied on a single backend server (RESILINK platform v1) acting as an intermediary between the app and the Orange Decentralized Exchange Place (ODEP) API. This setup enabled peer-to-peer exchanges based on smart contracts and virtual currency management, fully supported by ODEP's blockchain-based infrastructure.

With the deployment of RESILINK platform v2, the architecture has evolved to be able to remove the dependency on the ODEP API when needed. The platform can now directly embed most backend functionalities required by the mobile application, acting no longer only as a relay, but as a fully autonomous backend service. It handles core logic, data persistence, and user interaction workflows independently from ODEP.

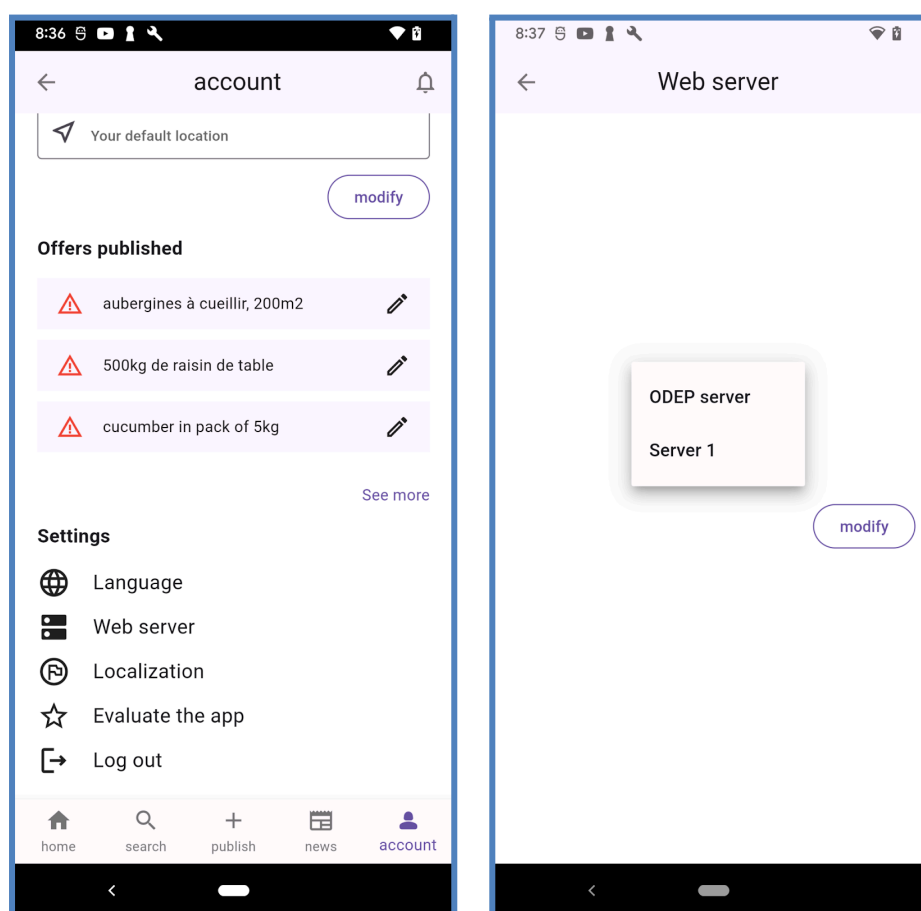
With RESILINK platform v2 running without ODEP, the blockchain-based smart contract functionality with virtual currency transactions or smart contract execution is therefore not supported. In exchange, the new platform benefits from significantly improved **performance and reduced latency**, as all interactions between the mobile application and the backend are now direct. There is no longer a need to forward requests through an intermediary service.

The RESILINK platform — both in v1 and v2 — stores its data in a MongoDB cluster. This cluster includes two main databases:

- One dedicated to **application data** (e.g., assets, users, offers, ratings).
- One dedicated to **logging events and system activity** (e.g., API calls, connection attempts).

While this architecture was already in place with v1, **the v2 platform manages a broader scope of data**, reflecting the shift in responsibilities from ODEP to the RESILINK backend.

To support testing, development, and regional deployments, the RESILINK mobile application now allows users to **select which server instance to connect to**—either RESILINK platform v1 (still relying on ODEP) or platform v2. This server-selection option appears in the user options section since **data is not shared between the two platforms**. RESILINK v1 continues to interact with the ODEP environment and its data, while v2 operates with its own independent backend and database.



By default, the application connects to RESILINK platform v2, which is now the preferred deployment version to run the various evaluation program and Living Lab piloting. This

flexibility in backend selection enables the platform to support multiple localized deployments, such as running independent server instances on edge devices (e.g., Raspberry Pi 5), each potentially tied to a specific subdomain and user community. **This is also motivated by the fact that smallholder farmers are still more willing to negotiate and pay services using traditional methods.**

4.2. Installing & deploying RESILINK platform v2

To support interaction between the RESILINK mobile application and the ODEP infrastructure, the RESILINK Platform v2 acts as an intermediate software layer. It enhances data processing, manages user interactions, and integrates additional business logic. The platform source code is publicly available on GitHub at the following URL:

 https://github.com/ZiQuwi/RESILINK_Render_Server/tree/MainWithoutODEP

The project can be downloaded directly from the website or cloned using Git. Be sure to select the MainWithoutODEP branch when cloning. A complete README is available in the repository to guide through dependencies, setup, and server usage, but the essential steps are summarized below.

Once the project has been retrieved, the first requirement is a working MongoDB cluster. Two databases are expected:

- Logs (for example) for storing logs related to API usage and user actions.
- Resilink (for example) for storing application-related data.

These databases must be initialized with the appropriate collections. The following script illustrates the required setup:

```
use Logs # or any database name
db.createCollection("ConnectionLogs")
db.createCollection("DeleteLogs")
db.createCollection("GetLogs")
db.createCollection("PatchLogs")
db.createCollection("PutLogs")

use Resilink # or any database name
db.createCollection("Asset")
db.createCollection("AssetType")
db.createCollection("AssetTypeCounter")
db.createCollection("News")
db.createCollection("Offer")
db.createCollection("Rating")
db.createCollection("prosumer")
db.createCollection("user")
```

In addition to the databases, a configuration file must be created in the root directory of the project. This file must be named RESILINK_Server.env and should include all necessary environment variables used by the server.

Here is a sample configuration:

```
IP_ADDRESS=resilink-dp.org
PORT=9990
SWAGGER_URL=https://resilink-dp.org
ENCRYPTION_KEY=b32c32aac9c6afd06ab3554415de5edbafc14ef97cc6d0e4ffa678220a57b39f
TOKEN_KEY=f0d8cd085ada735ac45c30e3368b5b4c87a8e7fb9828a2289af5065bad05b015
DB_URL=mongodb+srv://<username>:<password>@<cluster-url>/Resilink
DB_LOGS_URL=mongodb+srv://<username>:<password>@<cluster-url>/Logs
```

This file should be customized with your own MongoDB credentials and network information.

Before running the server, make sure Node.js is installed on the system. Then, from the root of the project, install the project dependencies using: `npm install`. Once installation is complete, the server can be started with the following command: `node src/index.js`.

The platform will be available at the defined IP address and port, with Swagger documentation accessible via the `SWAGGER_URL` specified in the environment configuration.

This setup enables RESILINK Platform v2 to serve as a reliable middleware between mobile clients and the decentralized ODEP infrastructure, ensuring smooth management of resources, users, and transactional data.

4.3. Use & Validation of RESILINK v2 for the Living-Labs

The RESILINK platform v2 is currently deployed on a professional web hosting provider to ensure continuous server availability (24/7), which marks a significant improvement in reliability and performance compared to the previous v1 deployment model. This deployment architecture replaces the initial intermediate role of the platform and provides a full backend service dedicated to the RESILINK mobile application.

As with version 1, access to the platform's database is managed through secured connections, using SSH tunnels to interact with dedicated MongoDB instances. This mirrors the v1 setup, where the platform connects to MongoDB Atlas clusters via specific connection URIs. However, in the near future, with small-scale deployed servers (such as raspberry), deployment will be carried out with a local base.

In terms of functionality, the RESILINK mobile application maintains a consistent user experience across both platform versions. A guest account — labelled “public” — allows users to browse recent offers without the need for registration. This ensures a low-barrier entry point to explore the ecosystem. However, creating and publishing new offers, or initiating contact with offer owners (e.g., via WhatsApp), still requires user authentication.

One notable addition in the application's user interface is the ability to switch between RESILINK platform v1 and v2 from the user profile settings. This feature enables the user to

explicitly choose which backend server they wish to interact with. It is important to note that data—particularly offers—are not shared or synchronized between the different platforms. Offers created on platform v2 will only be visible when the application is connected to platform v2, and similarly for v1.

This current configuration allows for flexibility in deployment, especially in regions with different infrastructural capabilities, and supports future scalability of the RESILINK ecosystem across multiple localized servers.

4.4. Test of v2 on Raspberry Pi

In order to evaluate the feasibility of deploying the RESILINK platform v2 on low-cost edge devices, a first deployment test was conducted on a single-computer board Raspberry Pi 5 with 4GB of main memory. For this purpose, a dedicated installation script was developed to automate the deployment process and minimize manual operations.



The script proceeds through the following main steps:

- **0. Check Wi-Fi connectivity and system time**
The script verifies internet access and checks the accuracy of the system clock, which is essential for secure communications and package management.
- **1. Install Git if not present**
Ensures Git is installed, as it is required to clone the RESILINK project repository.
- **2. Clone the RESILINK project**
The platform code is retrieved from the public GitHub repository and placed into the working directory on the device.
- **3. Create .env configuration file**
A `.env` file is generated to store environment variables such as IP address, port, encryption and token keys, and MongoDB connection URLs.
- **4. Install Docker**
Docker is installed to simplify the setup of the local MongoDB database.

- **5. Set up MongoDB via Docker**

Instead of connecting to an external MongoDB cluster as in the standard RESILINK v2 deployment, the script pulls and runs a Docker container for MongoDB. This allows the Raspberry Pi to host its own local database instance and improves portability.

- **6. Install Node.js**

Node.js and npm are installed if not already available, to support execution of the RESILINK backend.

- **7. Install project dependencies**

The necessary Node.js dependencies for the project are installed via `npm install`.

- **8. Create a systemd service**

A systemd service is defined to automatically launch the Node.js server at boot time and ensure high availability.

- **9. Enable and start the service**

The service is registered, enabled, and started, ensuring that the RESILINK server runs in the background and persists after reboots.

Compared to the standard deployment of the RESILINK v2 platform, which uses an external MongoDB Atlas cluster, this Raspberry Pi deployment leverages a local Dockerized MongoDB instance. This design choice avoids the complexity of provisioning and securing new database clusters for each edge deployment, while maintaining compatibility with the application's data structure.

The test was completed successfully. The server was accessible via the Raspberry Pi's IP address, with open ports ensuring proper network communication. The architecture proves suitable for deploying lightweight, decentralized RESILINK nodes in distributed environments.

A test on a local network can be easily realized as illustrated below.



Smartphone connected to the WiFi and using the local RESILINK server

Smartphone running as WiFi access point through its 3G/4G/5G connection

Raspberry Pi 5 connected to the WiFi and running the RESILINK server on the local network

4.5. Future work

Several key developments are planned for future iterations of the platform.

A primary objective is the design and implementation of a **master server**, acting as a central registry where individual RESILINK platform instances can register themselves. This registry will enable the RESILINK mobile application to dynamically discover and list available platforms, allowing users to select from a growing network of servers based on location, services, or performance criteria. This discovery mechanism will strengthen the decentralised architecture of RESILINK by facilitating autonomous deployments while maintaining a unified user experience.

In addition to platform discovery, future work includes the **automated creation and management of subdomains** for each deployed server instance. This feature would simplify and standardise the process of deploying new platforms by providing a dedicated and accessible endpoint for each server, without requiring complex DNS or hosting configurations by the deployer.


Another important enhancement under consideration is the **communication between multiple RESILINK platforms**. This would allow a user connected to a primary platform to query and retrieve data from other platforms designated as secondary. A typical use case involves a user wishing to access offers or resources distributed across different geographic areas or thematic servers. In such scenarios, the mobile application would connect to the user's selected primary server, which in turn would forward specific requests to the secondary platforms, aggregate the responses, and return a consolidated dataset to the user.

These developments aim to extend the reach, flexibility, and efficiency of the RESILINK platform, enabling broader deployment across multiple territories and fostering a more integrated ecosystem that serves both local needs and cross-regional opportunities.

5. FIRST TEST OF OPEN-API & INTEROPERABILITY







5.1. Motivations

As indicated in D4.2a "First report software templates demonstrating RESILINK's open API", RESILINK wants to promote a platform-of-platforms approach where the ecosystem is not anymore based on isolation of platforms and competition between them.




No "one app fits all"


- Competition → isolation → fragmented ecosystems









- → Platform-of-Platforms!
- Enabling a platform-of-platforms approach will promote a much wider and appealing ecosystem
- → Specialized platforms to better manage specific agricultural sectors
- → Discover resources/services from other platforms → no isolation



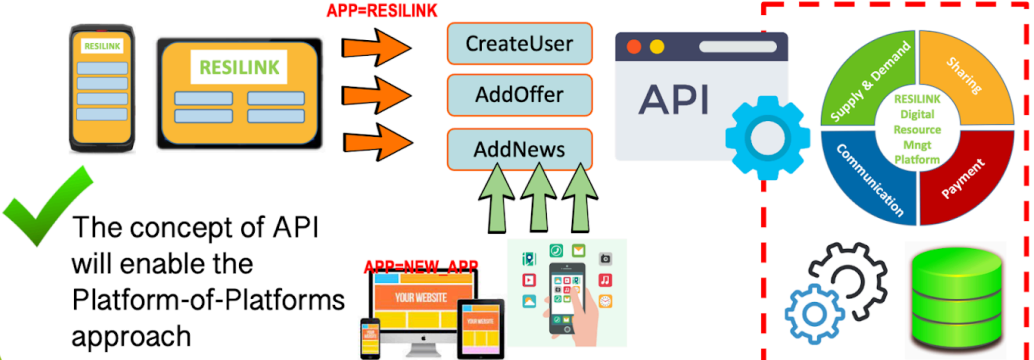
Prof. Congduc Pham
http://www.univ-pau.fr/~cpham






API: key to large adoption!

- API: Application Programming Interface
- Digital platforms > Applications thanks to API-oriented design
- → All functionalities/actions are accessible through an API call



Prof. Congduc Pham
http://www.univ-pau.fr/~cpham



5.2. Orange use-case

5.2.1. General view

The objective of Orange's demonstrator is to create a small application to show how to use RESILINK Open API.

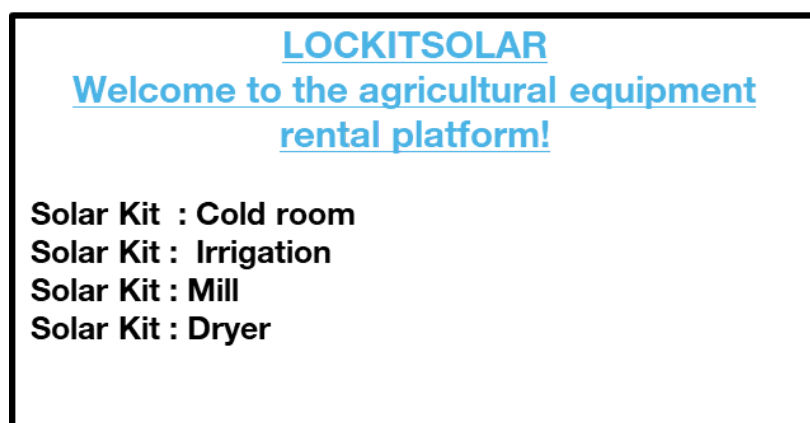
Linked to Orange Research projects in the MEA Area (Middle East and Africa), the suggested use case is a rental service of different solar powered tools for agriculture activities.

Today the offer of Orange Energies is a solar kit complete with different kinds of equipment, and in the context of professional activities, you can find some tools like solar Dryer, Solar Mill or a Chicken farm Kit.



5.2.2. The story of LockItSolar

Let's imagine a startup named LKS for LockItSolar. LKS offers to rent solar powered machines for agriculture.



To increase its visibility, LKS decided to become RESILINK partner. The first idea is to expose some promotion offers to Resilink users. But there seems to have other opportunities: publication of news on Resilink portal and display of targeted ads on LKS portal side.

As the **administrator** of LKS platform,

- 1) I want to publish a promotion offer on RESILINK platform

« One-month promotional offer period on irrigation kits, rental at 1 euro/month »

I can fill the form on LKS website, RESILINK API allows me to publish this offer on RESILINK platform.

- 2) I search for people to install irrigation kits for a particular region. I want to publish this information in RESILINK News.

« 6 months engagement, payment after each installation using Orange Money »

I describe the content of the news, RESILINK API allows to add this news on RESILINK platform.

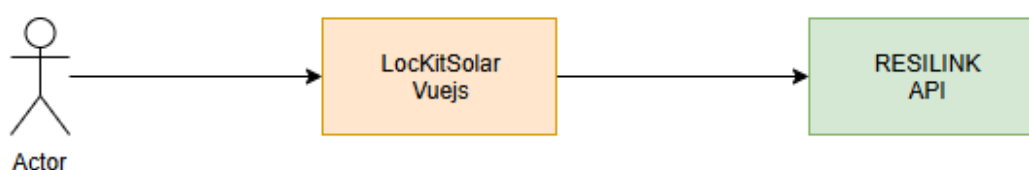
- 3) I configure the current Resilink offers to be displayed on the LOCKITSOLAR platform

« Selection of active offers in a geographical area (Morocco for example) linked to solar energy »

I mention the parameters for the search, RESILINK API allows searching in the current ads on RESILINK platform.

5.2.3. Technical view

For a normal service, we should have created a specific backend for the application that will communicate with the RESILINK API. However, since the idea is to create a sample application as simple as possible to demonstrate the use of the RESILINK API for creating a sharing resources service, we decided to create a backendless application that directly interacts with the RESILINK API.



The LockKitSolar is developed as a simple web app using the JavaScript framework Vue.js. It will be hosted on Google Cloud Platform with an Nginx component using Cloud Run to reduce hosting costs. Since we decided not to implement a backend, users must log in before using the service to avoid putting the password inside the JavaScript code exposed to the client.

ACRONYMS LIST

Acronym	Explanation
AES	Advanced Encryption Standard
API	Application Programming Interface
APK	Android Package Kit
DNS	Domain Name Server
HTTP	Hyper Text Transfer Protocol
IPv4/IPv6	Internet Protocol v4 / v6
JSON	JavaScript Object Notation
ODEP	Orange Decentralized Exchange Place
PoC	Proof-of-Concept
SSL/TLS	Secure Socket Layer/Transport Layer Security
URL	Universal Resource Locator

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.