

The Food Safety Market: An SME-powered industrial data platform to boost the competitiveness of European food certification

D3.2 - Annual Releases of the TheFSM Data Platform

DELIVERABLE NUMBER	D3.2
DELIVERABLE TITLE	Annual Releases of the TheFSM Data Platform
RESPONSIBLE AUTHOR	Danai Vergeti (UBITECH)



Co-funded by the Horizon 2020
Framework Programme of the European Union

GRANT AGREEMENT N.	871703
PROJECT ACRONYM	TheFSM
PROJECT FULL NAME	The Food Safety Market: An SME-powered industrial data platform to boost the competitiveness of European food certification
STARTING DATE (DUR.)	01/01/2020 (36 months)
ENDING DATE	31/12/2023
PROJECT WEBSITE	www.foodsafetymarket.eu
COORDINATOR	Nikos Manouselis
ADDRESS	110 Pentelis Str., Marousi, GR15126, Greece
REPLY TO	nikosm@agroknow.com
PHONE	+30 210 6897 905
EU PROJECT OFFICER	Stefano Bertolo
WORKPACKAGE N. TITLE	WP3 Platform
WORKPACKAGE LEADER	UBITECH
DELIVERABLE N. TITLE	D3.2 Annual Releases of the TheFSM Data Platform
RESPONSIBLE AUTHOR	Danai Vergeti (UBITECH)
REPLY TO	vergetid@ubitech.eu
DOCUMENT URL	
DATE OF DELIVERY (CONTRACTUAL)	31 January 2021 (M12)
DATE OF DELIVERY (SUBMITTED)	31 January 2021 (M12)
VERSION STATUS	V1.0 Final
NATURE	Demonstrator (DEM)
DISSEMINATION LEVEL	Public (P)
AUTHORS (PARTNER)	Danai Vergeti (UBITECH), Dimitris Ntalaperas (UBITECH), Iosif Angelidis (UBITECH)
CONTRIBUTORS	Giannis Stoitsis (Agroknow), Charalampos Thanopoulos(Agroknow), Timotheos Lanitis (Agroknow), Panagiotis Rousis (Agroknow), Nikolaos Manouselis (Agroknow), Pavlin Gyurov (SAI), Branimir Rakic (PROSPEH)

REVIEWER		Branimir Rakic (PROSPEH)	
VERSION	MODIFICATION(S)	DATE	AUTHOR(S)
0.1	Table of contents	9/12/2020	Danai Vergeti (UBITECH)
0.3	Contribution to section 2	23/12/2020	Danai Vergeti (UBITECH)
0.5	Contribution to section 2	8/01/2021	Iosif Angelidis (UBITECH)
0.6	Contribution to sections 3	10/01/2021	Giannis Stoitsis (Agroknow), Charalampos Thanopoulos(Agroknow), Timotheos Lanitis (Agroknow), Panagiotis Rousis (Agroknow), Nikolaos Manouselis (Agroknow), Pavlin Gyurov (SAI)
0.8	Contribution to sections 2	13/01/2021	Branimir Rakic (PROSPEH)
0.9	Contribution to sections 1 and 4 and version for review	18/01/2021	Danai Vergeti (UBITECH), Iosif Angelidis (UBITECH)
0.95	Deliverable review	22/01/2021	Branimir Rakic (PROSPEH)
1.0	Final version	31/01/2021	Danai Vergeti (UBITECH)

PARTNERS		CONTACT
Agroknow IKE (Agroknow, Greece)		Nikos Manouselis (Agroknow) nikosm@agroknow.com
SIRMA AI EAD (SAI, Bulgaria)		Svetla Boytcheva (SAI) svetla.boytcheva@ontotext.com
GIOUMPITEK MELETI SCHEDIASMOS YLOPOIISI KAI POLISI ERGON PLIROFORIKIS ETAIREIA PERIORISMENIS EFTHYNIS (UBITECH, Greece)		Danai Vergeti (UBITECH) vergetid@ubitech.eu
AGRIVI DOO ZA PROIZVODNJU, TRGOVINU I USLUGE (Agrivi d.o.o., Croatia)		Tanja Matosevic (Agrivi d.o.o.) tanja.matosevic@agrivi.com
PROSPEH, POSLOVNE STORITVE IN DIGITALNE RESITVE DOO (PROSPEH DOO, Slovenia)		Ana Bevc (PROSPEH DOO) ana@origin-trail.com
UNIVERSITAT WIEN (UNIVIE, Austria)		Elizabeth Steindl (UNIVIE) elizabeth.steindl@univie.ac.at
STICHTING WAGENINGEN RESEARCH (WFSR, Netherlands)		Yamine Bouzembrak (WFSR) yamine.bouzembrak@wur.nl
TUV- AUSTRIA ELLAS MONOPROSOPI ETAIREIA PERIORISMENIS EUTHYNIS (TUV AU HELLAS, Greece)		Kostas Mavropoulos (TUV AU HELLAS) konstantinos.mavropoulos@tuv.at
TUV AUSTRIA ROMANIA SRL (TUV AU ROMANIA, Romania)		George Gheorghiu (TUV AU Romania) george.gheorghiu@tuv.at
VALORITALIA SOCIETA PER LA CERTIFICAZIONE DELLE QUALITA' E DELLE PRODUZIONI VITIVINICOLE ITALIANE SRL (VALORITALIA, Italy)		Francesca Romero (Valoritalia) francesca.romero@valoritalia.it
TUV AUSTRIA CYPRUS (TUV AU CYPRUS, Cyprus)		Sousanna Charalambidou (TUV AU CYPRUS) sousanna.charalambidou@tuv.at

ACRONYMS LIST

ABAC	Attribute Based Access Control
ABE	Attribute Based Encryption
ACL	Access Control List
API	Application Programming Interface
CP-ABE	Ciphertext-Policy Attribute Based Encryption
CRUD	Create, Read, Update, Delete
CSS	Cascading Style-Sheets
CSV	Column Separated Values
DAC-MACS	Data Access Control for Multi-Authority Storage Systems
DID	Decentralized Identified
DTO	Data Transfer Object
DoA	Description of Action
FAME	Fast Attribute-based Message Encryption
GDPR	General Data Protection Regulation
HTML	Hypertext Markup Language
JWT	JSON Web Token
KP-ABE	Key-Policy Attribute Based Encryption
PAP	Policy Administration Point
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PERM	Policy, Effect, Request, Matchers
PGP	Pretty Good Privacy
PIP	Policy Information Point
POJO	Plain Old Java Object
PRP	Policy Retrieval Point
RBAC	Role Based Access Control
RD-ABE	Revocable and Decentralized Attribute Based Encryption
REST	Representational state transfer
RSA	Rivest-Shamir-Adleman
SPA	Single Page Application
SSO	Single Sign-On
TheFSM	The Food Safety Market
UI	User Interface
URL	Uniform Resource Locator
UX	User Experience
VM	Virtual Machine
XACML	eXtensible Access Control Markup Language

EXECUTIVE SUMMARY

The current document entitled “Annual Release of TheFSM Data Platform” constitutes an accompanied report of TheFSM Platform demonstrator providing the preliminary efforts undertaken within the context of Tasks T3.2 “User, Identity & Application Management Layers”, T3.3 “Secure Storage & Information Exchange”, T3.4 “Blockchain-powered Smart Contracting Layer Secure Storage & Information Exchange” and T3.5 “Data Management, Indexing & Processing” of WP3. The purpose of this deliverable is to deliver the first demonstrator of the platform (M12). Towards this we **present TheFSM backbone infrastructure** that will support the implementation and deployment of TheFSM platform, the **technology stack** of TheFSM backbone infrastructure, the **access information for the services** of TheFSM backbone infrastructure, we present **TheFSM platform’s prototype** for M12 and additional **low-fidelity mock-ups** which were designed for all the main functionalities of the data market platform that will be delivered in future releases. Finally, we conclude the document.

TABLE OF CONTENTS

EXECUTIVE SUMMARY	6
1 INTRODUCTION	10
1.1 SCOPE.....	10
1.2 AUDIENCE.....	10
1.3 STRUCTURE	11
1.4 RELATION TO OTHER DELIVERABLES	11
2 THEFSM PLATFORM INFRASTRUCTURE	12
2.1 BACKBONE INFRASTRUCTURE	12
2.2 TECHNOLOGY STACK.....	13
2.2.1 Access Control and Authentication	13
2.2.2 Data Curation and Semantic Enrichment	15
2.2.3 Data Market Modelling	16
3 THEFSM PLATFORM DEMONSTRATOR	18
3.1 OVERVIEW	18
3.2 CRITICAL CONCEPTS	18
3.2.1 Request filtering.....	18
3.2.2 Data Encryption.....	20
3.2.3 Security Configuration and Conventions.....	21
3.3 THEFSM PLATFORM PROTOTYPE.....	22
3.3.1 Administrator	22
3.3.2 Security Layer	25
3.3.3 Data Management Layer	29
3.3.4 Design principles	31

4	THEFSM DATA MARKET MOCKUPS	36
4.1	THEFSM DATA MARKET	36
4.1.1	Welcome page.....	36
4.1.2	Dataset providers.....	37
4.1.3	Thematic categories of dataset packages.....	38
4.1.4	Creation of custom data package	39
4.1.5	Selection of datasets from categories and providers	40
4.1.6	Advanced filtered dataset search	41
4.1.7	Dataset sampling and purchase.....	42
4.1.8	Ordering a dataset	43
4.1.9	Dataset's metadata display.....	44
4.1.10	Versioning and delivery format during purchase process	45
4.1.11	Monetized dataset added to cart.....	45
4.1.12	Cart checkout	47
4.1.13	Login prompt when trying to checkout.....	48
4.1.14	On the fly registration of unregistered user.....	49
4.1.15	Billing information during checkout process.....	50
4.1.16	User comments and order finalization.....	51
4.1.17	Order confirmation prompt.....	52
4.1.18	Representative technological stack	52
5	CONCLUSIONS	54
6	REFERENCES	55

LIST OF FIGURES

Figure 1: The FSM platform component deployment.	12
Figure 2: The FSM platform infrastructure	13
Figure 3: Request filtering flow.	20
Figure 4: Welcome page.....	23
Figure 5: Login page	23
Figure 6: The dashboard.....	24
Figure 7: Roles management.	25
Figure 8: Users management.	26
Figure 9: ABAC policies management and authorization.....	27
Figure 10: Advanced policy query builder.	28
Figure 11: The policy enforcer testing tool.....	29
Figure 12: Dataset upload step 1, metadata input.....	30
Figure 13: Dataset upload step 2, dataset sampling and column filtering.	31
Figure 14: User edit modal.....	32
Figure 15: User gets feedback on successful edit.	33
Figure 16: Advanced date picker.	33
Figure 17: Dataset upload step 3, access policy setup and optional monetization.....	34
Figure 18: Dataset upload step 4, overview summary and final confirmation.	35
Figure 19: Welcome page.	36
Figure 20: Dataset providers.	37
Figure 21: Thematic categories of dataset packages.....	38
Figure 22: Creation of custom data package.....	39
Figure 23: Selection of datasets from categories and providers to form custom package.	40
Figure 24: Advanced filtered dataset search.....	41
Figure 25: Dataset sampling and purchase.	42
Figure 26: Ordering a dataset.....	43
Figure 27: Dataset's metadata display.....	44
Figure 28: Versioning and delivery format during purchase process.	45
Figure 29: Monetized dataset added to cart.....	46
Figure 30: Cart checkout.....	47
Figure 31: Login prompt when trying to checkout (user is not logged in).....	48
Figure 32: On the fly registration of unregistered user during cart checkout.	49
Figure 33: Billing information during checkout process.	50
Figure 34: User comments and order finalization.....	51
Figure 35: Order confirmation prompt.....	52

1 INTRODUCTION

1.1 Scope

The current document entitled “Annual Release of TheFSM Data Platform” constitutes an accompanied report of TheFSM Platform demonstrator providing the preliminary efforts undertaken within the context of Tasks T3.2 “User, Identity & Application Management Layers”, T3.3 “Secure Storage & Information Exchange”, T3.4 “Blockchain-powered Smart Contracting Layer Secure Storage & Information Exchange” and T3.5 “Data Management, Indexing & Processing” of WP3. The purpose of this deliverable is to deliver the first demonstrator of the platform (M12). Towards this end, the scope of the current report can be described in the following axes:

- **To present TheFSM backbone infrastructure** that will support the implementation and deployment of TheFSM platform. TheFSM backbone infrastructure is based on the concept of components dockerization. Each component that contributes an isolated functionality to the platform is a docker container, while the entire platform is deployed as a composed docker consisting of the components’ docker images. On the application layer, a set of well-established and powerful services is deployed in order to facilitate the implementation and operation of TheFSM platform’s services.
- **To document the technology stack of TheFSM backbone infrastructure** that enable the platform’s scalability, robustness and high performance. The technology stack is composed by a set of services that are built on top of mature, production-ready open source technologies and tools. For each service, a brief description of the provided functionalities and features is presented and the role of each service in TheFSM platform is highlighted.
- **To present TheFSM platform’s prototype** and additional low-fidelity mock-ups which were designed for all the main functionalities of the platform that will be delivered in future releases.
- **To document the access information for the services of TheFSM backbone infrastructure** that are utilised for the implementation purposes of TheFSM platform.

The current deliverable presents the first version of TheFSM platform that serves as a pillar for the implementation of the upcoming versions of the platform. The delivery of TheFSM platform’s backbone is a continuous process that will last till M36 as per TheFSM Description of Action. The platform’s backbone will incorporate the necessary refinements and enhancements according to the specifications and the feedback that will be continuously collected.

1.2 Audience

This document is structured in a way to properly present the documented functionality. It is oriented towards both technical experts familiar with the presented concepts, as also individuals who are not overly familiar with said concepts, but still need to inspect the prototype version of the platform presented here.

1.3 Structure

This demonstrator report is structured as follows.

- Section 2 describes the backbone infrastructure and technological stack the platform uses so far. We explain critical concepts, as they justify specific design decisions we opted for. The functional pages and all implemented navigation steps are thoroughly documented, providing details involving integration points between components etc. Of particular focus is the dataset upload steps/pipeline, since they involve multiple integration points and components of the platform, proving a well-designed synergy.
- Section 3 illustrates with mockups how we envision the final iteration of the platform to be, briefly explaining what each functional page will provide.
- Section 4 concludes the report by summarizing.

1.4 Relation to other deliverables

The current deliverable documents the preliminary efforts undertaken within the context of Tasks T3.2 “User, Identity & Application Management Layers”, T3.3 “Secure Storage & Information Exchange”, T3.4 “Blockchain-powered Smart Contracting Layer Secure Storage & Information Exchange” and T3.5 “Data Management, Indexing & Processing” of WP3. The main input of D3.2 is D3.1 - TheFSM Open Reference Architecture, D1.1 - Report on Requirements for TheFSM and D3.3 - Annual Report from TheFSM software components Integration & Testing. D3.2 presents the developed prototype of the first version of the platform, as well as low fidelity mockups of TheFSM Data Market, which are being developed under T3.5, started on M10. The provided functionality is derived from D3.1, where the functional, non-functional and technical requirements were presented. Furthermore, the components presented here are derived from the proposed architecture we presented in D3.1. Also, this document is closely related to D3.3, where we present our deployment, testing and development strategy, as well as documenting our decision on deploying the components as docker containers. The results of the first version of the platform will be carefully studied in order to improve the components already integrated up to this point, observing the strengths and potential limitations of the direction the mockup took, all with the aim to improve the integration and incorporate lessons learned towards integrating more services to the platform.

2 THEFSM PLATFORM INFRASTRUCTURE

2.1 Backbone Infrastructure

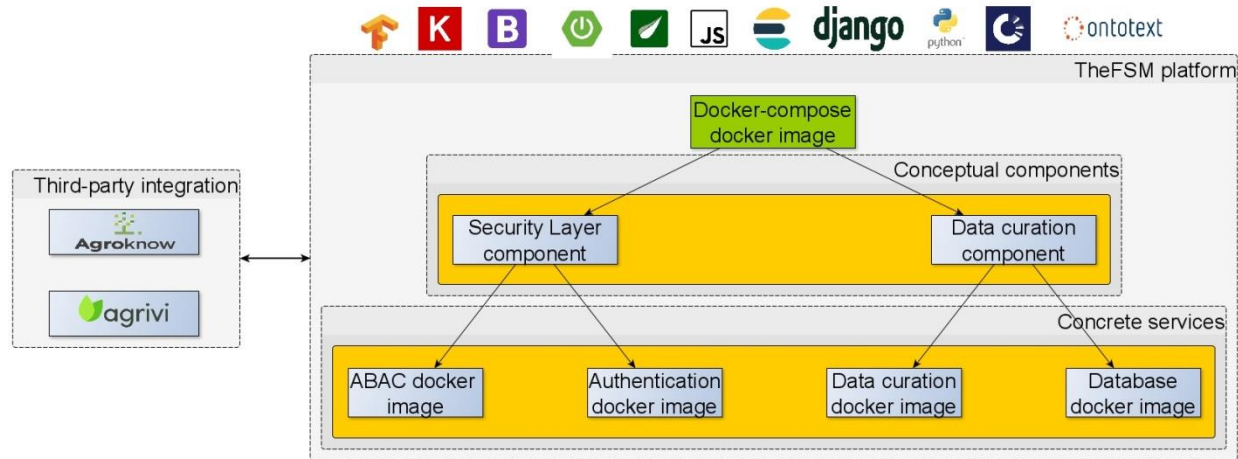


Figure 1: TheFSM platform component deployment.

The deployment of the technical solution of TheFSM Platform into production involves the deployment of a single **docker-compose image** (Figure 1) representing the entire platform (for a thorough documentation involving the dockerization process of the components and how we deploy into production, kindly consult with D3.3).

TheFSM deployment (Figure 1) is further divided into conceptual components, as described in the architecture provided in D3.1 TheFSM Open Reference Architecture. **Each conceptual component consists of one or more docker images** for encapsulation, scalability and turning them into microservices, greatly enhancing their utilization. Additionally, the platform communicates with the external third-party applications of Agroknow and Agrivi and other external datasources, also allowing for future integration of developers' APIs, who have expressed interest in processing the offered data of the platform.

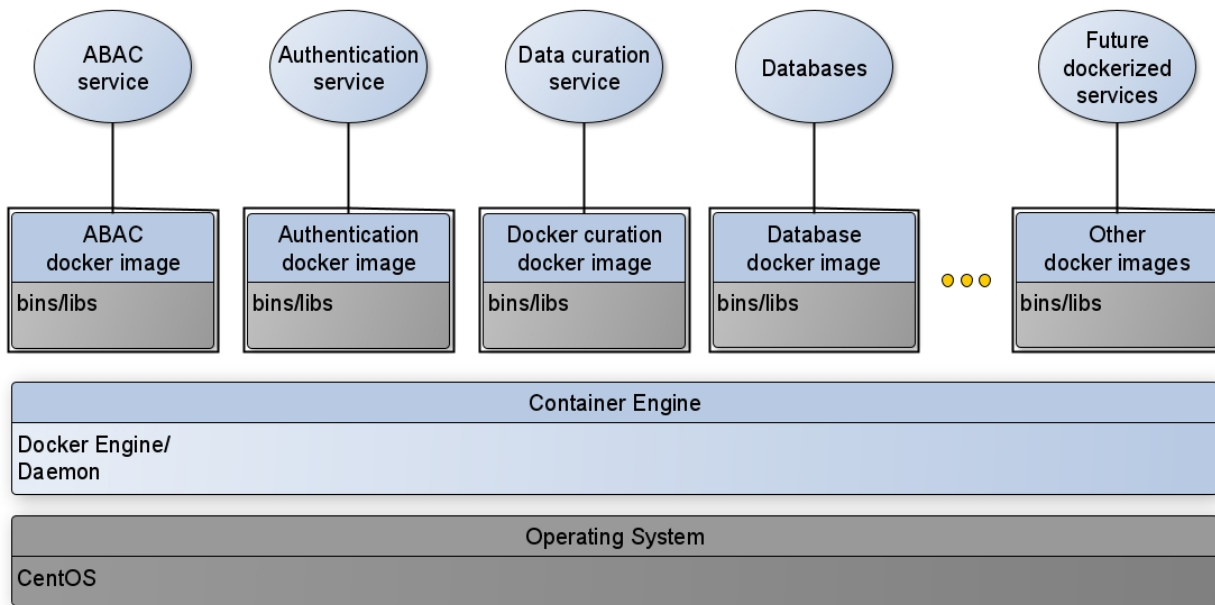


Figure 2: The FSM platform infrastructure

The above figure illustrates the same infrastructure, this time inside the Virtual Machine (VM) the platform will be hosted. More specifically:

- The bottom level represents the operating system (CentOS) of the VM.
- The middle level represents the container engine user (Docker's Engine and Daemon).
- The top level represents all dockerized services we currently have successfully deployed and integrated, as well as future dockerized services we will add in the next iterations

During the development of the platform more docker images and conceptual components will be added to the infrastructure diagram. During our internal testing (stress tests included) of the platform's functionality, we did not observe any scalability or performance issues. Nevertheless, we designed the provided services and the database layer to be scalable.

The detailed description of the integration strategy, the deployment process, as well as the technical integration and testing of TheFSM Platform is provided in "D3.3 - Annual Report from TheFSM software components Integration & Testing".

2.2 Technology Stack

TheFSM Platform applies the **containerization paradigm**, as mentioned above. This allows the development of **secure, scalable, isolated applications, enabling the selection among a variety of technologies and tools**, depending on the needs of each microservice. Based on this, the technology stack of each layer and/or each component is provided below:

2.2.1 Access Control and Authentication

Authorization and Access Control Engine:

- Web application backend level:
 - Spring Boot¹: A well-known and industry-standard framework for Java development.
 - JCasbin²: A well-known and industry-standard library involving ABAC (Attribute Based Access Control). JCasbin is the Java flavour of Casbin, an authorization library that supports access control models like ACL, RBAC, ABAC for Golang, Java, C/C++, Node.js, Javascript, PHP, Python, .NET (C#), Delphi, Rust, Dart/Flutter and Elixir. It is another well-known library used in the industry by companies such as Intel, VMWare, Docker, Orange, Cisco, Microsoft, Verizon, RedHat, IBM and T-Mobile. We opted for JCasbin due to its simple, yet powerful PERM (Policy, Effect, Request, Matchers) meta-model.
- Frontend/UI level:
 - Javascript with jQuery³: Enhanced capabilities due to jQuery. Additionally, extra plugins/libraries are used, such as:
 - SweetAlert⁴: Pretty-modal popup for user notification/feedback.
 - Datatables⁵: Very powerful and well-known library for data tables management.
 - QueryBuilder⁶: Extremely flexible query builder for policies creation and management.
 - Steps⁷: Library enabling easy creation of forms with multiple steps.
 - Validation⁸: Powerful form validator. Guarantees no errors in user's input.
 - Papa-parse⁹: CSV processing library. Required for column manipulation of datasets.
 - Bootstrap¹⁰: HTML/CSS library for beautiful and responsive UI design.
- Security:
 - Spring Security¹¹: A module of Spring's library for security. Enforces HTTPS requests and provides data encryption capabilities. Also required for hybrid data encryption (fully documented later on).
 - JSEncrypt, HybridCryptoJS¹²: Libraries for RSA and Symmetric encryption for Javascript. Required for hybrid data encryption (fully documented later on).
- Documentation/API:

¹ <https://spring.io/projects/spring-boot>

² <https://github.com/casbin/jcasbin>

³ <https://www.javascript.com/> and <https://jquery.com/>

⁴ <https://sweetalert2.github.io/>

⁵ <https://datatables.net/>

⁶ <https://querybuilder.js.org/>

⁷ <http://www.jquery-steps.com/>

⁸ <https://jqueryvalidation.org/>

⁹ <https://www.papaparse.com/>

¹⁰ <https://getbootstrap.com/>

¹¹ <https://spring.io/projects/spring-security>

¹² <https://travistidwell.com/jsencrypt/> and <https://github.com/juhoen/hybrid-crypto-js>

- The provided API and DTOs are thoroughly documented via SpringFox and Swagger UI¹³. It is possible to inspect the entire exposed API, the required parameters, the object structure of DTOs, as well as even test requests and simulate user authentication for testing the provided functionality.

User Authentication Mechanism:

- The authentication module has been implemented according to the approach recommended by the W3C Decentralized Identifiers standard¹⁴:
 - Node.js based Express framework
 - Javascript DID resolver libraries
 - OriginTrail Open source Command Executor framework
 - Ethereum Blockchain DID implementation interfaces (ethr and ERC725 methods)
 - Open source Web3.js library for the Ethereum blockchain DID interface

2.2.2 Data Curation and Semantic Enrichment

Data Curation and Semantic Enrichment services are based on the Ontotext Platform and are shipped as a Docker image and can be deployed with Docker compose along with all dependent components and services. For the development of the Data Curation component, SAI's technological stack was used:

- kong: the API gateway.
- semantic-objects¹⁵: schema management service.
- graphdb¹⁶: triple store. GraphDB is an enterprise ready Semantic Graph Database, compliant with W3C Standards. Semantic graph databases (also called RDF triplestores) provide the core infrastructure for solutions where modelling agility, data integration, relationship exploration and cross-enterprise data publishing and consumption are important.
- mongodb¹⁷: document store. MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database, MongoDB uses JSON-like documents with optional schemas.
- graphiql¹⁸: GraphQL playground. GraphiQL is the reference implementation of this monorepo, GraphQL IDE, an official project under the GraphQL Foundation.

¹³ <https://swagger.io/>

¹⁴ <https://www.w3.org/TR/did-core/>

¹⁵ <http://platform.ontotext.com/3.2/semantic-objects/semantic-objects.html>

¹⁶ <https://graphdb.ontotext.com/>

¹⁷ <https://www.mongodb.com/>

¹⁸ <https://github.com/graphql/graphiql>

- fusionauth¹⁹: out of the box security authentication service, integrating seamlessly.
- fusion-db²⁰: the fusion internal database. FusionDB is a simple and powerful Multi-model database. It allows storage, management and querying of millions of Documents and Key/Value pairs using XQuery.
- fusion-search: the fusion internal search store.
- grafana²¹: A very popular and well-used dashboard observability tool, metrics and logging operations.
- influxdb²²: open-source time series database used for logs. InfluxDB empowers developers to build IoT, analytics and monitoring software. It is purpose-built to handle the massive volumes and countless sources of time-stamped data produced by sensors, applications and infrastructure.
- telegraf²³: open source server agent to help collect metrics from stacks, sensors and systems.

At the core of Ontotext Platform is the Semantic Object service. A declaratively configurable service for querying and updating knowledge graphs. Users can write powerful GraphQL queries that uncover deep relationships within the data while not having to be overly concerned about the underlying database query language. Data can be modified and validated against configured data shapes with simplicity and ease. The Semantic Object service transpiles GraphQL queries and mutations into the Platform's different data storage layer query syntaxes. Queries and mutations are invoked, joined and validated, providing a simplified developer experience for those use-cases when GraphQL is sufficient.

All the data curation and semantic annotation services are described in more detail in D2.1 - Data Models & Representations, D2.2 - Data Services and D2.3 - Report on Data Population.

2.2.3 Data Market Modelling

For the development of the Supplier & Product Risk Assessment Models, the incident Prediction Models, the risk prediction models and the supplier risk prediction models, the following software stack is used:

- Django framework
- Python 3.9
- elasticsearch

¹⁹ <https://fusionauth.io/>

²⁰ <https://fusiondb.com/>

²¹ <https://grafana.com/>

²² <https://www.influxdata.com/>

²³ <https://www.influxdata.com/time-series-platform/telegraf/>

- Keras library
- Prophet Library
- Apache airflow
- Kibana
- Logstash

3 THEFSM PLATFORM DEMONSTRATOR

3.1 Overview

This section is dedicated to the documentation of the developed prototype of the first version of the platform. Based on the release plan of TheFSM Platform and since the ABAC interface are of top priority, the aforementioned mechanism had to be implemented first. ABAC and the security services consist the platform's core prototype. Additional functionalities regarding data curation and management and user authentication were developed and integrated in iterations, based on the component dependencies and the integration plan provided in D3.3. Temporary user interfaces have been developed, to enable communication and demonstrate the backend services which will be replaced by the final UI of TheFSM Platform in the next release.

3.2 Critical Concepts

The current section provides the main concepts related to the backend functionalities of TheFSM Platform v1.0. The following research and technological issues **will be analyzed in detail in the second version of D3.1 - TheFSM Open Reference Architecture** (M15). Nevertheless, we provide a short description of these concepts for a better the comprehension of the provided features, since some of them consist updates of the first version of the architecture and are introduced for the first time in the technical deliverables of the project.

3.2.1 Request filtering

When interacting with the platform, it is paramount to protect users' data with fine-grained enforcement of a set of policies. To that end, all resources and APIs should be protected; an action should be completed only if the requester is both authenticated and authorized. Those two concepts function at a different level and are both necessary to ensure requests are filtered with certainty. Next, we analyse both concepts.

3.2.1.1 Authentication

Authentication ensures ensuring requests are not random or exploited by a malicious entity. Simply put, it enforces a way for a requester to prove that they are who they claim to be. This can be achieved by cross-checking a set of properties a user has, such as their log in location, the devices they are using etc. In TheFSM is realized using **token based authentication**, as well as, **decentralized identity management**.

TheFSM authentication mechanism provides a token to the user upon login, forcing the user to use the token in all subsequent requests to prove they are who they claim to be. For the FSM platform, PROSPEH undertakes the task of providing the authentication service for issuing DID's which uniquely identify each entity (actual or conceptual) throughout the entire platform. Upon registration, each user is assigned a DID which is the claim to their identity, while they also get the Default User role initially (they can obtain roles with more privileges later). Excluding

registration, all other requests are rejected unless the requester **provides the JWT generated by the DID authentication service**. In addition, when providing one, it is validated against the DID authentication service in order to ensure their claim is valid. If valid, the request must also be filtered through authorization in order to be granted, as described next. This entire workflow is explained in brief steps as well as visually in Section 3.2.1.3.

3.2.1.2 Authorization

At this point, what is certain is that the requester is who they claim to be. However, we require a more fine-grained access, therefore the request should also be authorized. A basic user should not be able to execute actions of administrators. Unauthorized users should never be able to access a dataset they are not supposed to (this is particularly important towards achieving the platform's goals).

For the purposes of TheFSM, we opted to employ ABAC (Attribute Based Access Control) for authorization. To that end, we utilize an ABAC specification, via JCasbin.

In order to maximize flexibility and ensure robustness of policy evaluation, we use the allow-and-deny effect expression. Additionally, to guarantee the maximum amount of fine-grained control in FSM, we use ABAC. Currently, the attributes accompanying a user are:

- Their DID (unique identifier)
- The UNIX timestamp of the request
- The user's roles
- The year of the request
- The month of the request
- The day of the request
- The day name of the request (this is useful for policies denying access in specific days by name)
- The hour of the request
- The minute of the request

3.2.1.3 Request filtering revisited

Having explained all important concepts, we can now present the full workflow of how requests are filtered in the platform.

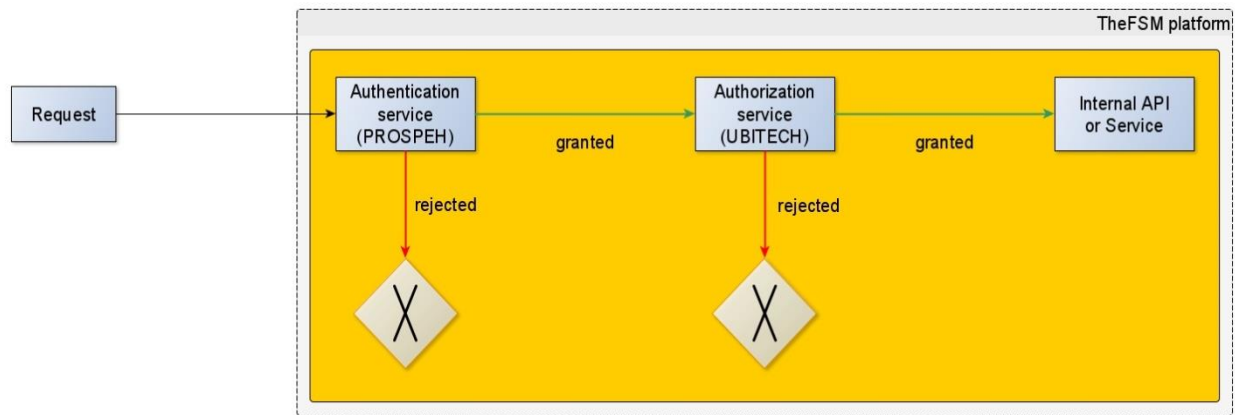


Figure 3: Request filtering flow.

As shown in the figure above, the following steps take place:

- A request is submitted to and API.
- The request is forwarded to PROSPEH's authentication service in order to authenticate the user. If the authentication fails, the request is rejected.
- The request is filtered through ABAC's policies in order to check if the request is authorized to execute. If authorization fails, the request is rejected.
- The request is executed. If data transfer between the end user and the platform is involved, the data are always travel encrypted.

3.2.2 Data Encryption

An important facility offered by the platform is data encryption. Due to GDPR regulations and due to dealing with sensitive information, legal contracts and protected datasets, it is paramount to ensure the data remains protected. To that end, we guarantee that while data is being transferred between the end user and the platform, they are always encrypted. Next, we will describe our proposed protocol for the platform's needs.

For obvious reasons, it is very risky to use Symmetric Encryption, because once a malicious user obtains the symmetric key, they immediately have access to the data as well. However, a great strength of Symmetric Encryption is that the key does not need to be large, as is the case in RSA. Furthermore, it is possible to combine the two into what is known as Hybrid Encryption. Many Hybrid Encryption schemes and protocols exist, some among them being well known (e.g., PGP in mail transmission).

The key idea behind it is that instead of encrypting the data with the public key, we can encrypt the symmetric key which will be used to actually encrypt the data and send both to the receiving user. Since they can decrypt the message with their private key, they can then decrypt and obtain the symmetric key with which they can decrypt the data they received. For the purposes of the

platform, we utilize **Hybrid Encryption**. Let's assume that user U wishes to send (upload) a dataset to the platform (P). With steps, the protocol is as follows:

- **U** generates a symmetric key **K_U**. Since the key is generated randomly on every data transfer, it is a one-time key only, preventing attackers from leveraging its use.
- **U** asks **P** for its public key **PK_P**.
- **U** encrypts the dataset **D_U** with **K_U**. The result is **ED_U**.
- **U** encrypts **K_U** with **PK_P**. The result is **EK_P**.
- **U** sends **ED_U** and **EK_P** to **P**.
- **P** receives both.
- **P** uses the private key **SK_P** to decrypt **EK_P**. The result is **K_U**.
- **P** uses **K_U** to decrypt **ED_U**. The result is **D_U**.
- **P** can now use the data as needed.

Respectively, when the platform sends a dataset to a user (e.g., dataset download), the protocol is as follows:

- **P** generates a symmetric key **K_P**. Since the key is generated randomly on every data transfer, it is a one-time key only, preventing attackers from leveraging its use.
- **P** asks **U** for its public key **PK_U**.
- **P** encrypts the dataset **D_P** with **K_P**. The result is **ED_P**.
- **P** encrypts **K_P** with **PK_U**. The result is **EK_U**.
- **P** sends **ED_P** and **EK_U** to **U**.
- **U** receives both.
- **U** uses the private key **SK_U** to decrypt **EK_U**. The result is **K_P**.
- **U** uses **K_P** to decrypt **ED_P**. The result is **D_P**.
- **U** can now use the data as needed.

Finally, we decided against using other schemes and instead utilize the Hybrid Encryption scheme we described simply because ABAC's policy engine is already strong enough to protect resources, especially combined with the authentication service provided by PROSPEH. If a user is not authenticated and authorized to interact with a resource, it is simply impossible for them to do so.

3.2.3 Security Configuration and Conventions

When developing the prototype presented here, we opted for the following security configuration setup and made the following conventions:

- By default, we have a set of pre-built policies in the platform for fine-grained access. These policies can never be edited by administrators since they provide the barebone fine-grained protection of data and APIs.

- A super administrator account is created by default and it can never be edited or deleted. The reason for this is we always wish to have a permanent administrator account that can create moderators and take important actions.
- We have setup the platform with a pair of RSA keys (public and private key) for secure data transfer.
- SSL communication and HTTPS are enforced by default. HTTP requests are redirected as HTTPS in order to avoid exposing sensitive information of any kind.
- When data travels from/to end users, they are always encrypted.
- The interface of the platform is designed following the Single Page Application (SPA) paradigm. This ensures fast performance and responsiveness, providing a near-native experience to the end user.
- The backend is stateless to avoid security compromises and also to force users to be authenticated with every request.

3.3 TheFSM Platform Prototype

The current section presents **TheFSM Platform v1.0 demonstrator**. The demonstrator covers the **core backend services** regarding: **access control, user authentication and authorization** and **data security at transit and at rest** and **decentralized identity management** (T3.2, T3.3, T3.4), as well as, **the integration with the backend data curation services** (D2.2, M12) provided the main steps of the data asset upload, mapping, policies definition, storage and encryption-decryption. The UI developed for the demonstrator will be updated and replaced with the final UI adopted by TheFSM Platform, developed within the tasks of TheFSM Data Market (T3.5).

The relevant **release plan** for the different versions of the platform, the **coverage** of the exact **components and features** of the delivered v1.0, as well as, the implemented **integration points** are provided in detail in “D3.3 Annual Report from TheFSM software components Integration & Testing”.

3.3.1 Administrator

3.3.1.1 User Login

Upon visiting the web platform, the user is welcomed with the following screenshot:

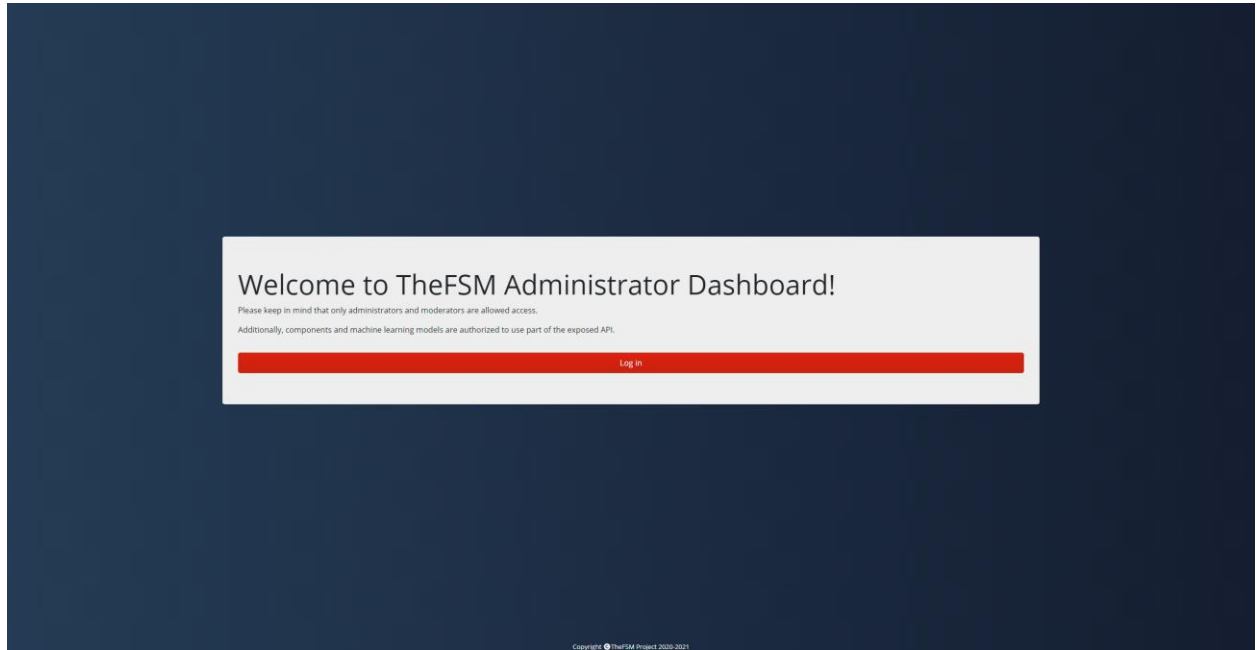


Figure 4: Welcome page.

The user is then prompted to put their username and password in order to successfully login, as shown in the next screenshot.

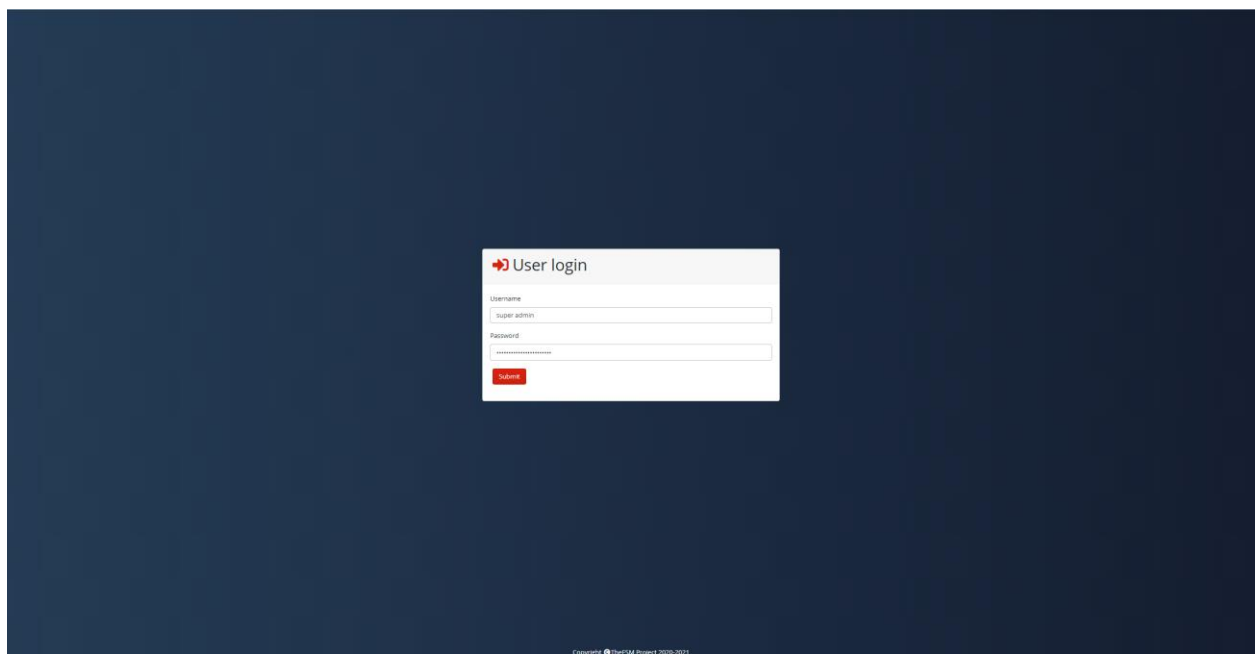


Figure 5: Login page

From this point on, all subsequent requests need **to attach the JWT the user receives upon the success of the operation**. Obtaining the JWT indicates that the user has been **successfully authenticated** by the authentication service and their **DID claims validated**. Every requests forwards this JWT to the authentication service in order to verify the request's user is authenticated. Additionally, the request is filtered through the **specified ABAC policies** and it passes through if the user's attributes grant them access.

3.3.1.2 Main Dashboard

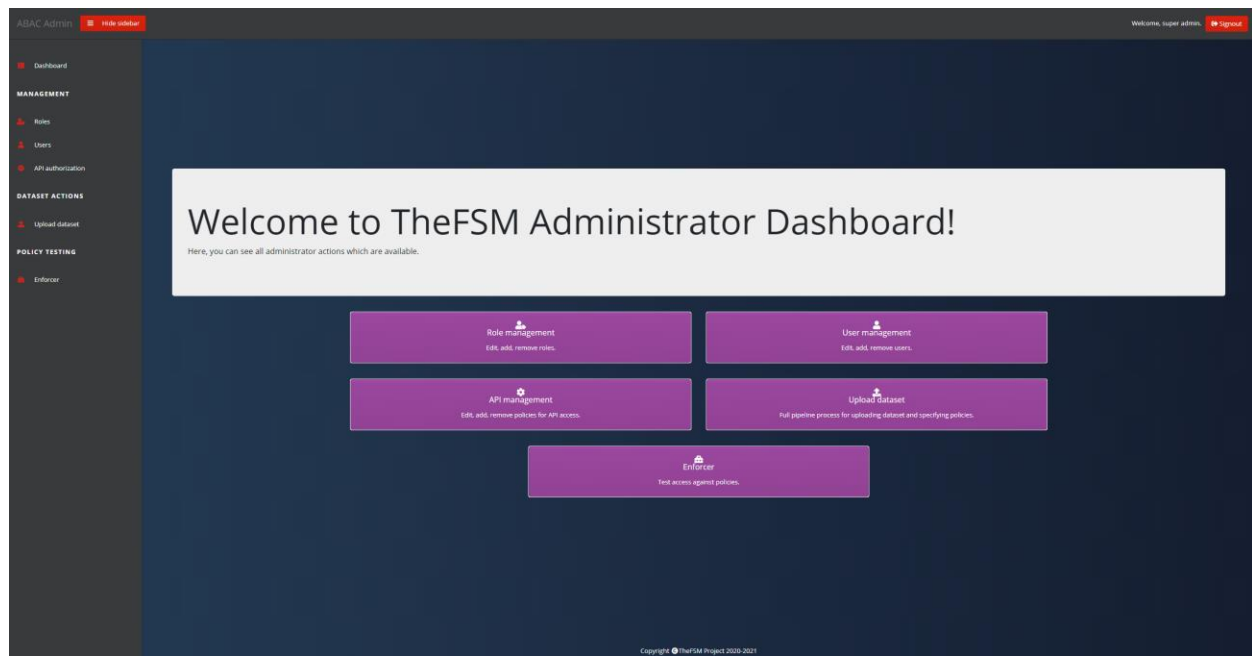


Figure 6: The dashboard.

Upon successful authentication and authorization (only the super administrator and moderators can visit this page), the user is redirected to the dashboard as shown in the above screenshot.

3.3.2 Security Layer

3.3.2.1 Roles management

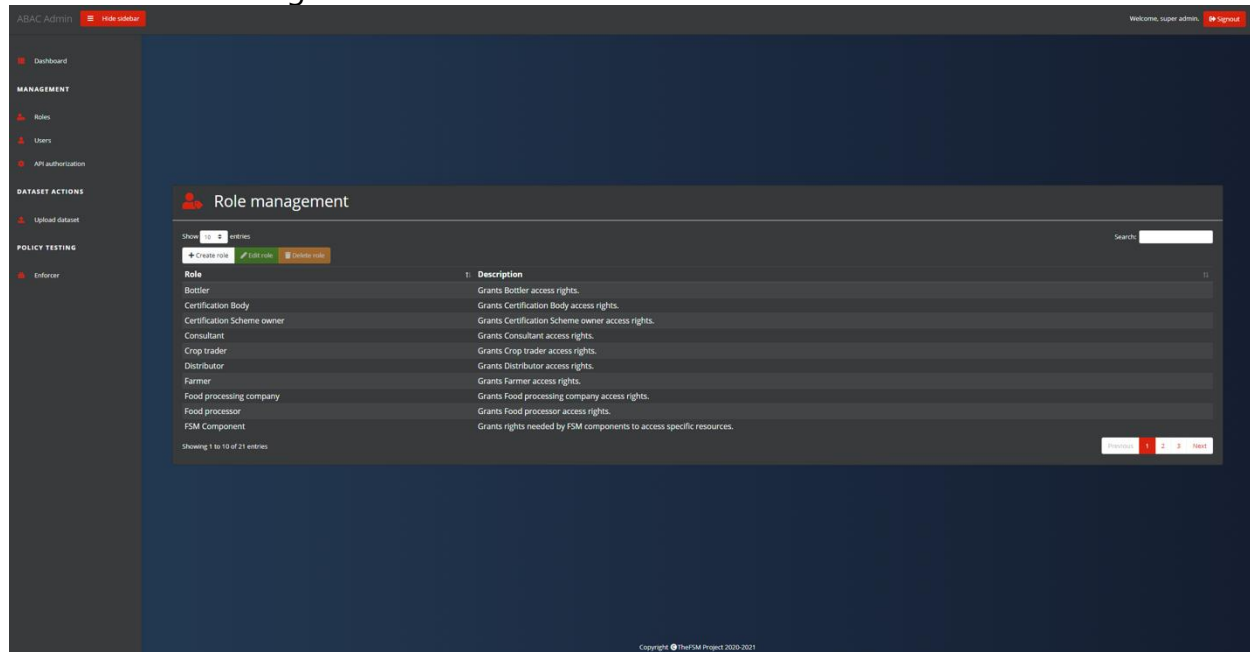


Figure 7: Roles management.

As with all conceptual entities, roles are displayed in a dynamic table. The user can sort the rows in ascending or descending order, while fuzzy search is also supported to filter rows. Additionally, the user can select how many rows they wish to see before results are paginated. All CRUD operations are supported. The table displays the name of each role, accompanied by a short description of what the role indicates with respect to privileges

3.3.2.2 Users Management

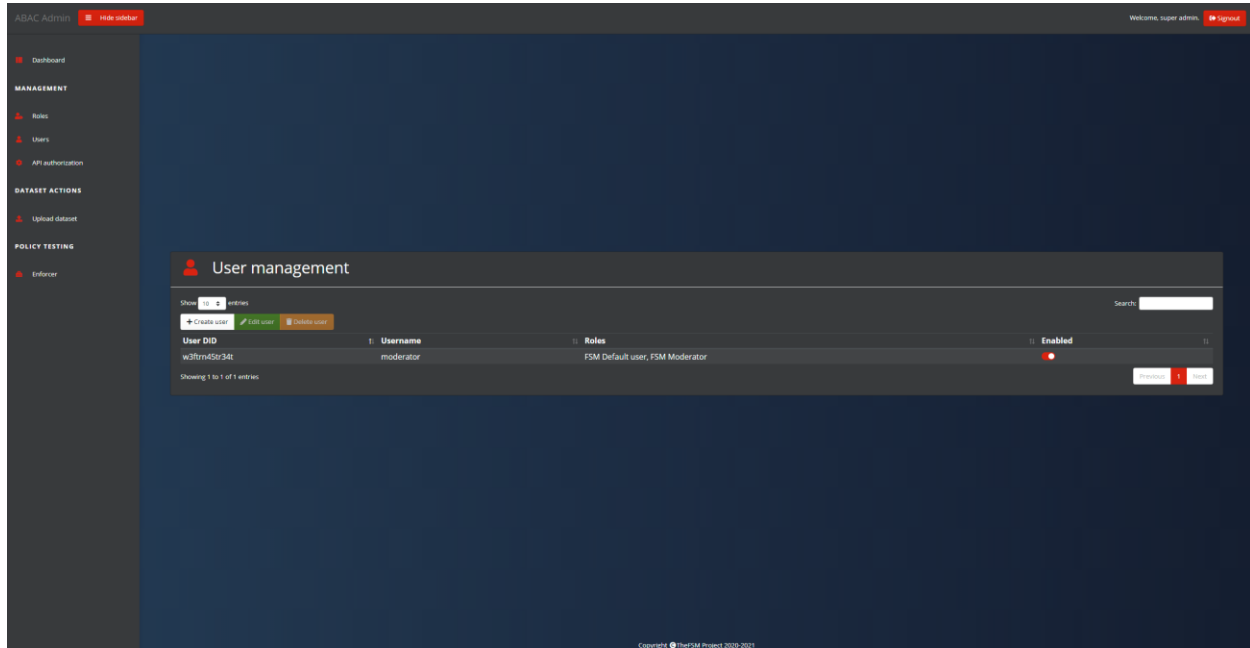


Figure 8: Users management.

User management is similar to roles management in many ways. However, creating new users is a pipeline involving multiple subcomponents of the platform. More specifically, when creating a new user, the following steps take place under the hood:

- The user is assigned a new DID by the authentication service.
- The list of available roles to assign to the user is provided by ABAC's roles management service.
- The user creation request is filtered through the authentication service and then through ABAC's policies. Since only administrators and moderators can access this API (and the UI), the request is authorized.
- The new user is stored by ABAC's users management service.

The relevant integration points and sequence diagrams are presented in D3.3 - Annual Report from TheFSM software components Integration & Testing.

The table displays the following fields:

- The user's DID.
- Their username.
- A list of roles they have, separated by commas.
- Enabled status. If a user is disabled, they won't be able to take any action even if they otherwise would. This provides even more fine-grained control over resource access.

3.3.2.3 Policy Management

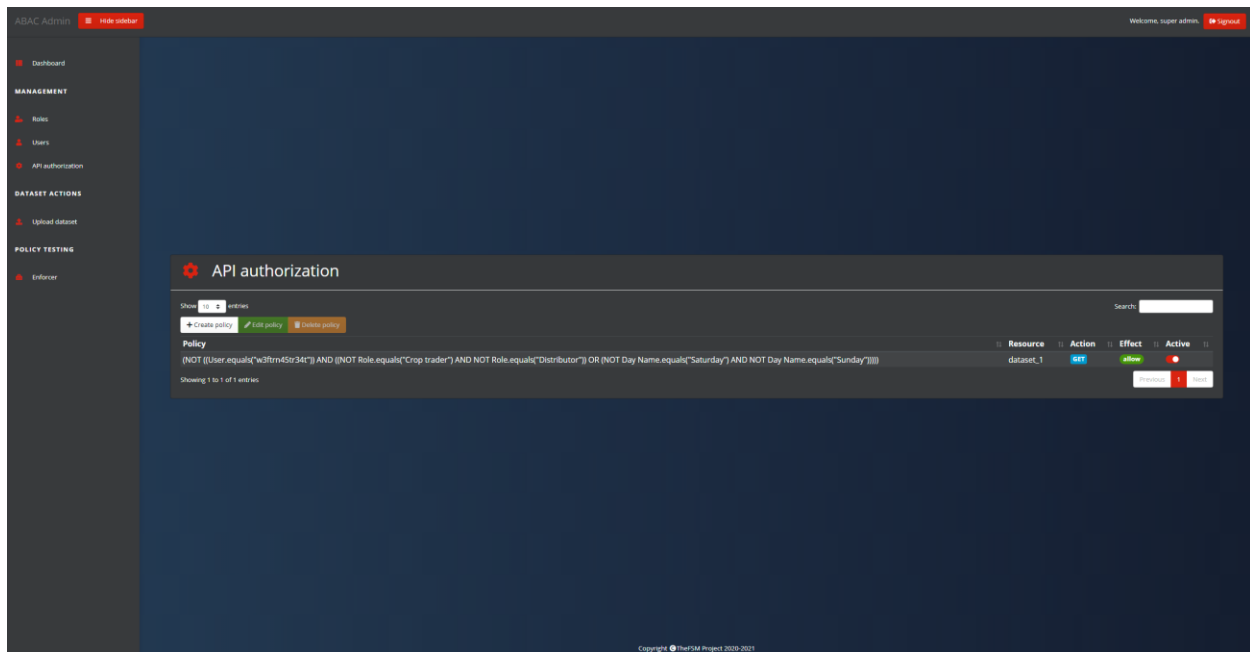


Figure 9: ABAC policies management and authorization.

The provided functionality of the policy management is one of the most fundamental parts of the platform in order to ensure fine-grained access to all resources. The table shows all policies used by the super admin and the moderators. More specifically, it shows:

- The actual condition the policy requires to have an effect.
- The resource this policy is applied on.
- The action this policy and resource is applied.
- The effect this policy will have if it matches with a user's attributes. Can be "allow" or "deny", effectively granting or denying them access.
- The active status of the policy. Policies might need to be temporarily disabled (e.g., for maintenance), which is what this field facilitates.

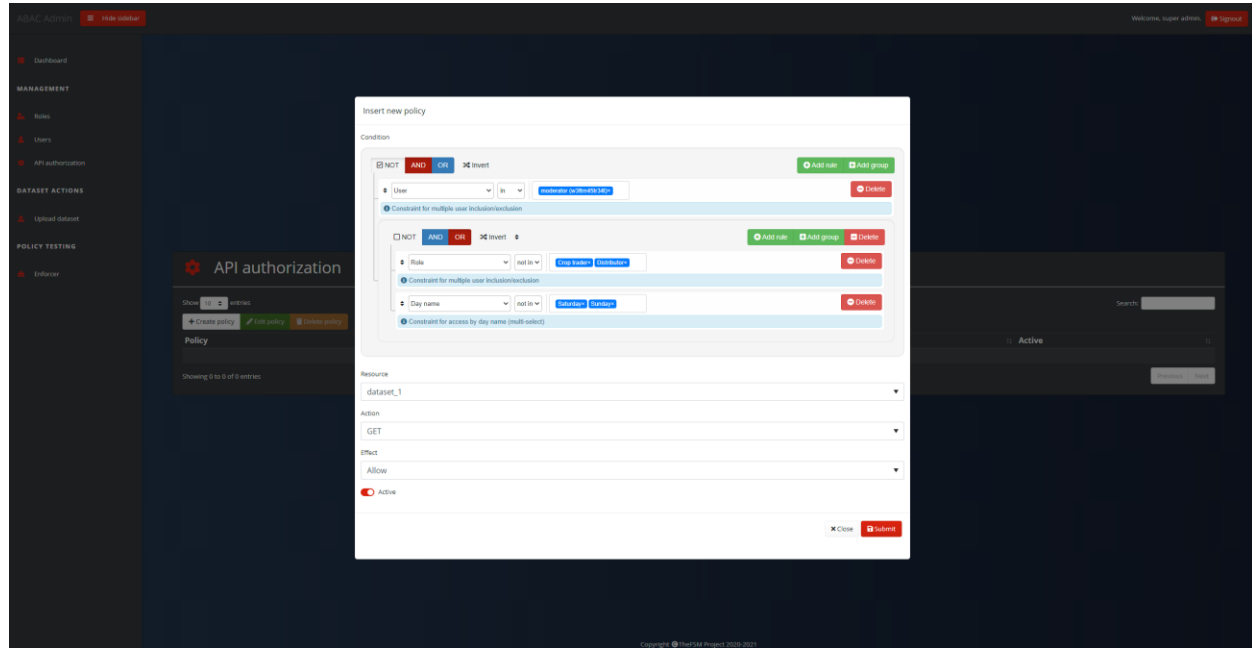


Figure 10: Advanced policy query builder.

The above screenshot illustrates the advanced query builder when adding new policies to the platform. Since the condition(s) which must be met when enforcing policies is critical, building those conditions must avoid error input at all costs. Consequently, all policies are built with the multi-condition builder shown above. The builder supports AND/OR conditions between individual rules, as well as groups, while it also supports negation (NOT). These can be applied at any level of the required condition. Rules or entire groups can be deleted, while it is also possible to drag and drop rules and/or groups, allowing quick, on the fly corrections and minimizing the number of actions a user must do. Each rule also has a small description, explaining to the administrator what the selected filter is trying to accomplish. The depth and complexity of conditions is infinite, allowing any desired effect to be enforced. The builder supports the following fields in condition building:

- Arbitrary access (anyone can access).
- User filtering. Users must belong (respectively, NOT belong) to a set of selected users.
- Role filtering. Users must have (respectively, NOT have) a role from a set of selected roles.
- Year filtering. The year of the request must be (respectively, NOT be) greater than, less than, equal, less or equal than, greater or equal than or between two years.
- Month filtering. The month of the request must belong (respectively, NOT belong) to a set of selected months.
- Day filtering. The day of the request must be (respectively, NOT be) greater than, less than, equal, less or equal than, greater or equal than or between two days.
- Day name filtering. The day of the request must belong (respectively, NOT belong) to a set of selected days. Instead of numerical, this filters day names. It can be useful for example when restricting access to a resource during weekends.

It is worth noting that all filters are validating before submission, preventing moderators and the super admin from making mistakes, while multi-select fields such as users, roles, months and day names are filled by autocomplete/search/tag selection.

3.3.2.4 Policy Enforcement Testing

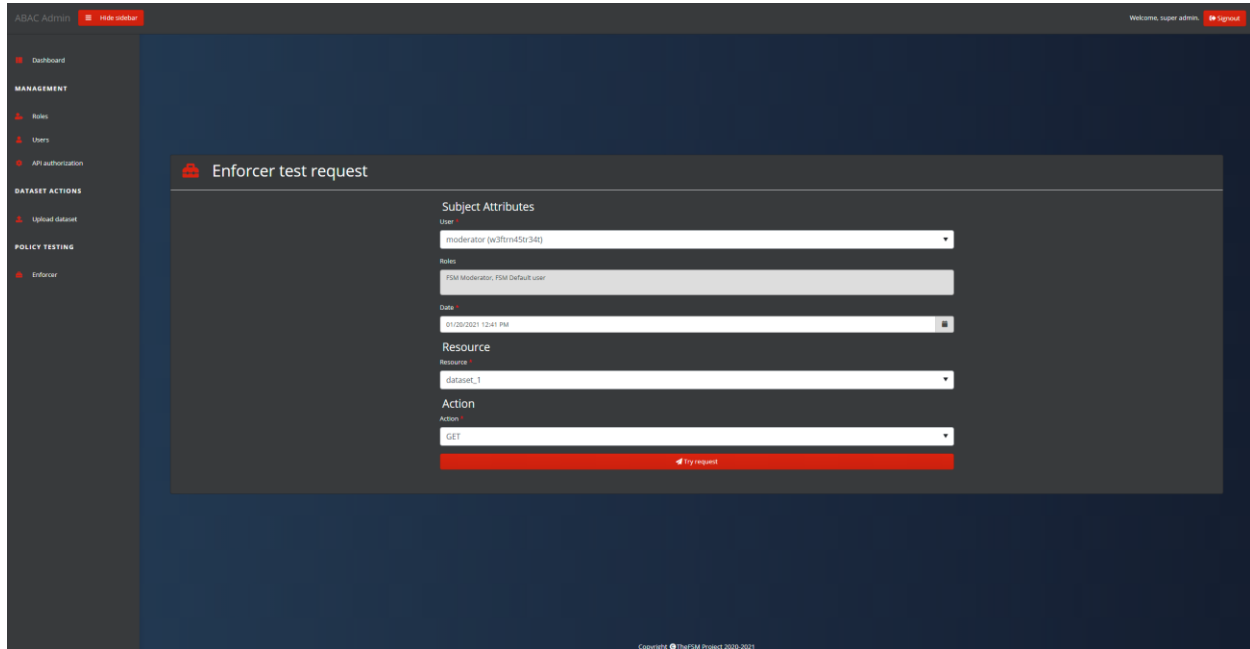


Figure 11: The policy enforcer testing tool.

This screenshot shows the policy enforcer testing tool. After setting up new policies or editing existing ones, it is useful for administrators and moderators to have the ability to test their setup with simulated requests, which is what this tool does. The form consists of simulated attributes a user would have (it assumes a user is already authenticated since it is testing the policy enforcement, not the authentication). Additionally, a resource, desired action and date are selected. The request is being tested against the ABAC enforcement engine and a visual status notifies the administrator or moderator about whether the request would go through.

3.3.3 Data Management Layer

3.3.3.1 Data Asset Management and Policies Definition

The steps presented in the current section will be further developed and refined until the next iteration. It is one of the most complex integration points in terms of interaction flows, as it involves multiple components and services.

The dataset management process consists of four main steps in total, each explained separately. These steps are identified as **prerequisites and are prioritized** to be implemented first regarding

the data management process, supported by TheFSM Platform. The relevant sequence diagrams are provided in “D3.3 Annual Report from TheFSM software components Integration & Testing”.

The form is dynamic and supports navigation to previous steps until submission, allowing the user to quickly change/correct their input without losing their progress.

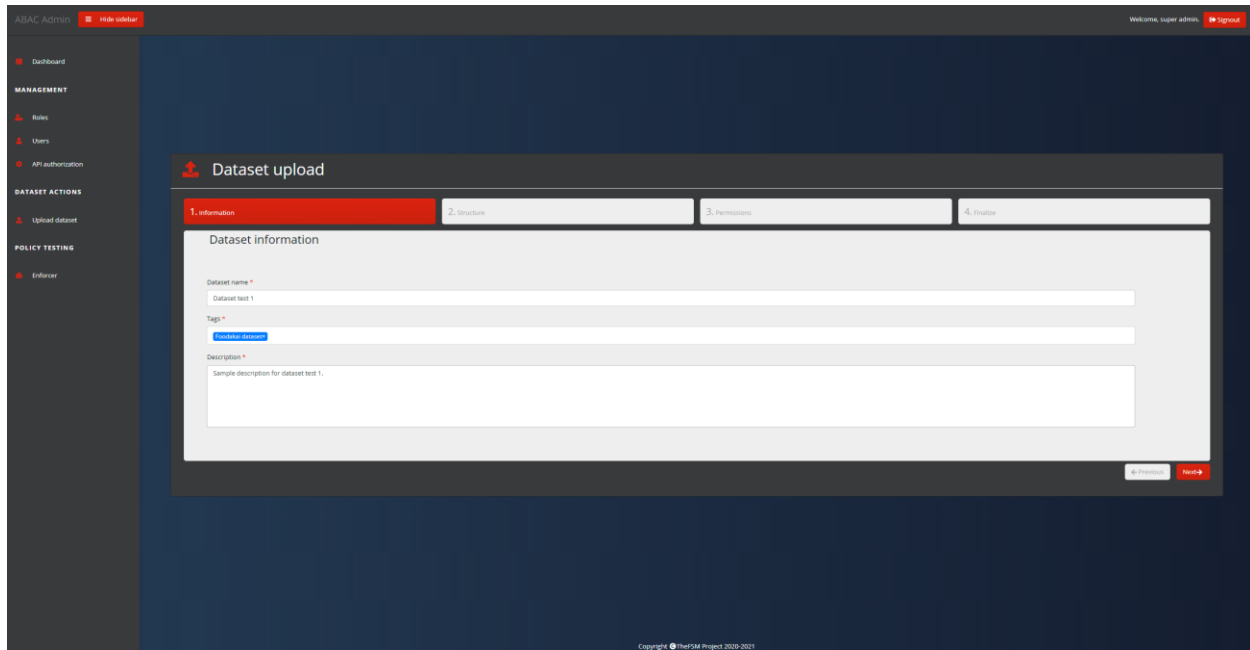
The screenshot shows a web application interface for 'Dataset upload'. The top navigation bar includes 'ABAC Admin', a 'Hide sidebar' button, and a user profile 'Welcome, super admin' with a 'Logout' button. A dark sidebar on the left contains a menu with 'Dashboard', 'MANAGEMENT' (with sub-items 'Roles', 'Users', 'API authorization'), 'DATASET ACTIONS' (with sub-item 'Upload dataset'), and 'POLICY TESTING' (with sub-item 'Enforcer'). The main content area is titled 'Dataset upload' and features a progress bar with four steps: '1. Information' (active), '2. Structure', '3. Permissions', and '4. Review'. The 'Dataset information' form includes a 'Dataset name' field with the value 'Dataset test 1', a 'Tags' field with a 'Fetch tags from backend' button, and a 'Description' field with the value 'Sample description for dataset test 1'. At the bottom right of the form are 'Back' and 'Next' buttons. A copyright notice 'Copyright © TheFSM Project 2020-2021' is visible at the very bottom.

Figure 12: Dataset upload step 1, metadata input.

The first step of the process, “Dataset Information”, gathers metadata about the dataset about to be uploaded. Such metadata include a dataset name, a small summary and thematic tags (obtained by a backend service). As this pipeline becomes more refined, more metadata will be included.

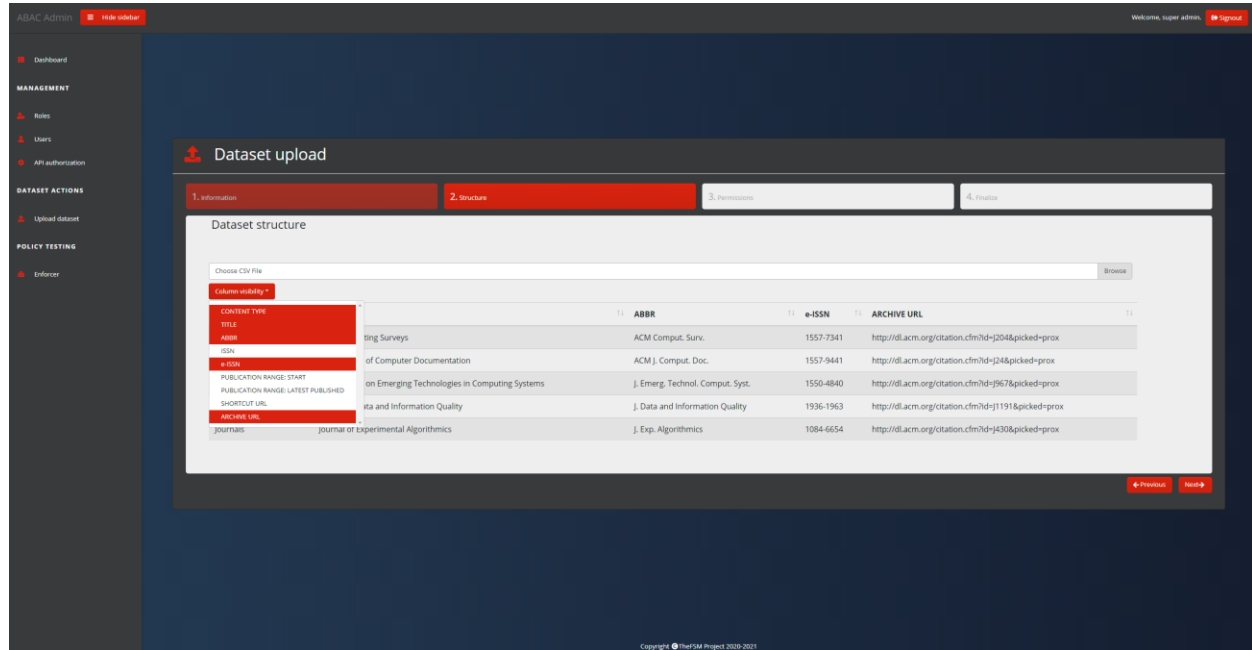


Figure 13: Dataset upload step 2, dataset sampling and column filtering.

The next step is “Dataset structure”. Here, the user selects a file (ex. .csv) which is their local dataset to be uploaded. The dataset is encrypted with the hybrid encryption approach explained above (Section 3.2.2) and it reaches the platform. This is to ensure their data remain protected. Upon receiving, the platform decrypts the dataset (important note: nothing is stored in the platform’s database so far) and displays it visually as shown above. To avoid clutter and for performance reasons, the first five rows are displayed. At this point, the user can select the columns they wish to keep from the dataset when finalizing the upload process.

3.3.4 Design principles

In this subsection, we provide a few examples illustrating the simplification of the interactions of the user with the application, as well as visual feedback, both aspects being crucial for a good user experience. When developing applications, it is paramount to make a user’s experience as good as possible, as this greatly increases the probability of the user taking an interest in the application. To that end, and taking into consideration **Nielsen’s principles**, we have taken the following into consideration:

- We build a clean interface, avoiding color polluting.
- We build an interface where colors can be correlated to specific actions.
- The user interface is fluid and responsive. It has been tested in smaller devices such as tablets and smartphones, doing our best to display the same content in limited space without subtracting from the overall aesthetics.
- All buttons are accompanied by an indicative icon, offering a quick visual clue to the user, rather than forcing them to read through actions.

- Actions are offered in as few steps as possible.
- We avoid overwhelming the user with options.
- Actions having opposite effects are as distant as possible.
- Similar actions are grouped.
- All forms requiring user input are always validated before submission, preventing the user from making mistakes.
- Wherever possible, autocomplete and tag selection from preselected values is provided to the user.
- Date input is always correct, due to an advanced date picker tool.
- The user is visually notified about the success or failure of their actions. Whenever possible, they also get a description of a potential error status. This is especially important for actions taking time to complete, as the user cannot be sure if the application has stopped working without at least a visual clue.
- Interactions should be as fast as possible.

In TheFSM Platform prototype development the **backend optimizations** and the **Single Page Application design** we opted for (as explained above) ensures a near-native user experience, like the user interacting locally instead of a web app.

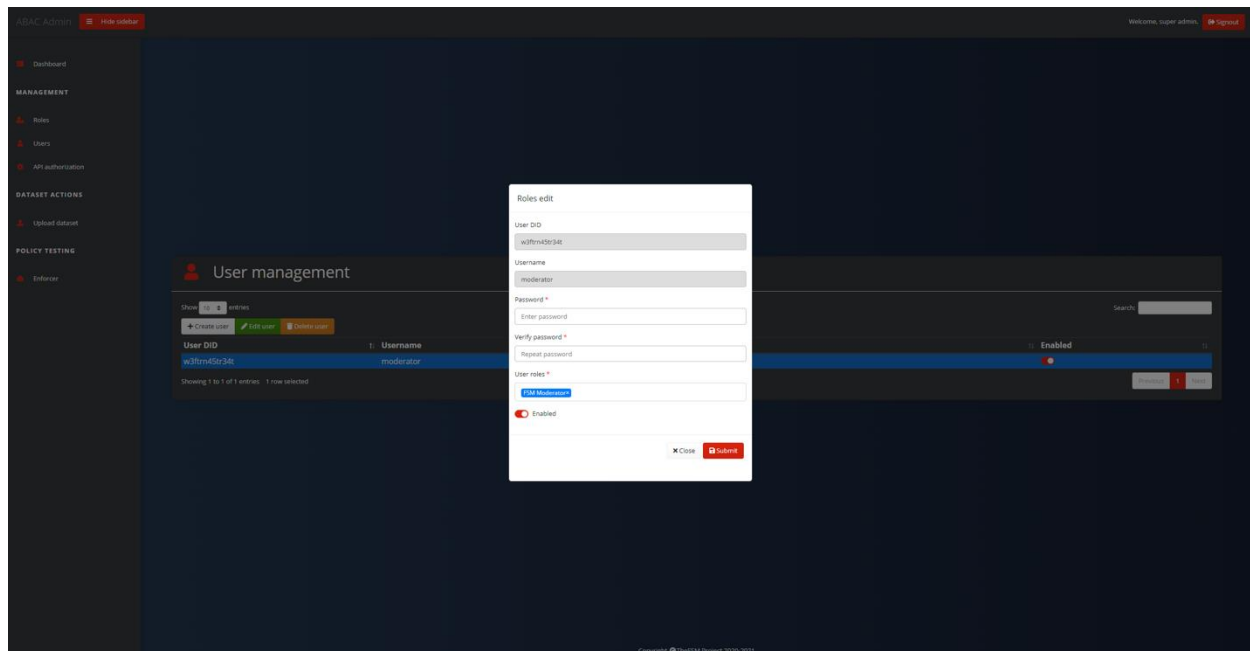


Figure 14: User edit modal.

The above screenshot shows an edit modal. It only appears when the user requests this action, keeping the interface clean. All fields are validated before submission and the user is notified about any fields requiring corrections. Additionally, the roles input supports autocomplete, search and tag selection, preventing error input.

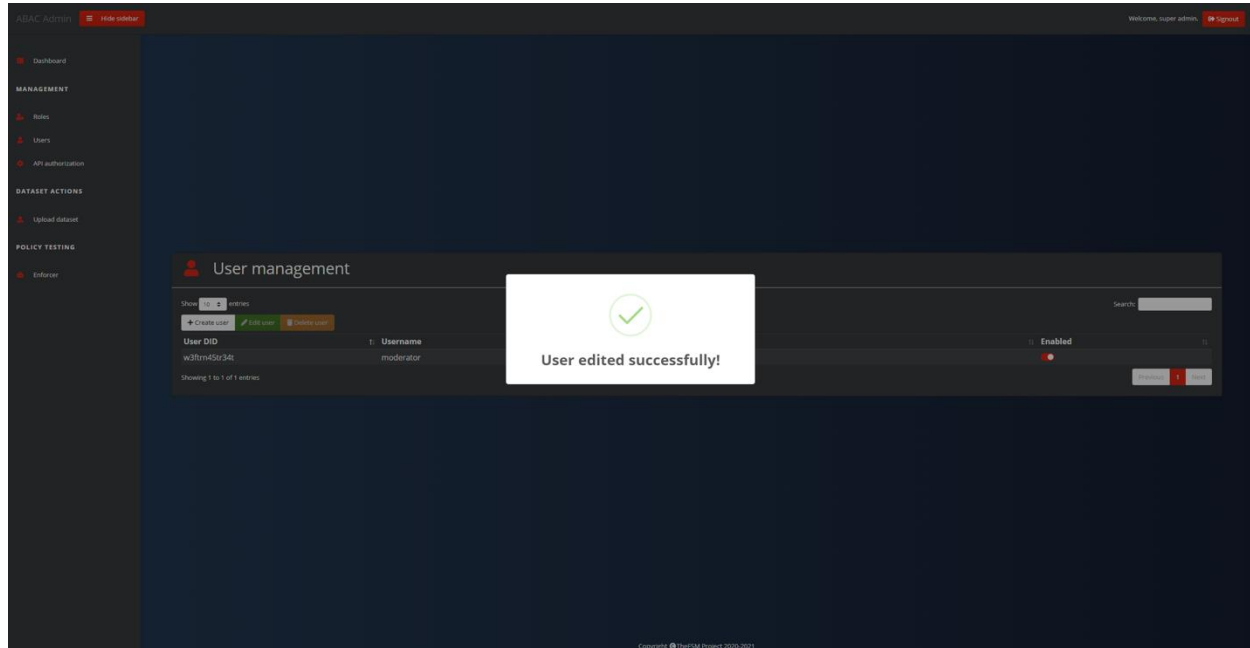


Figure 15: User gets feedback on successful edit.

Upon form submission and, provided the operation was successful, the user is notified visually about the operation status, as shown in the screenshot above.

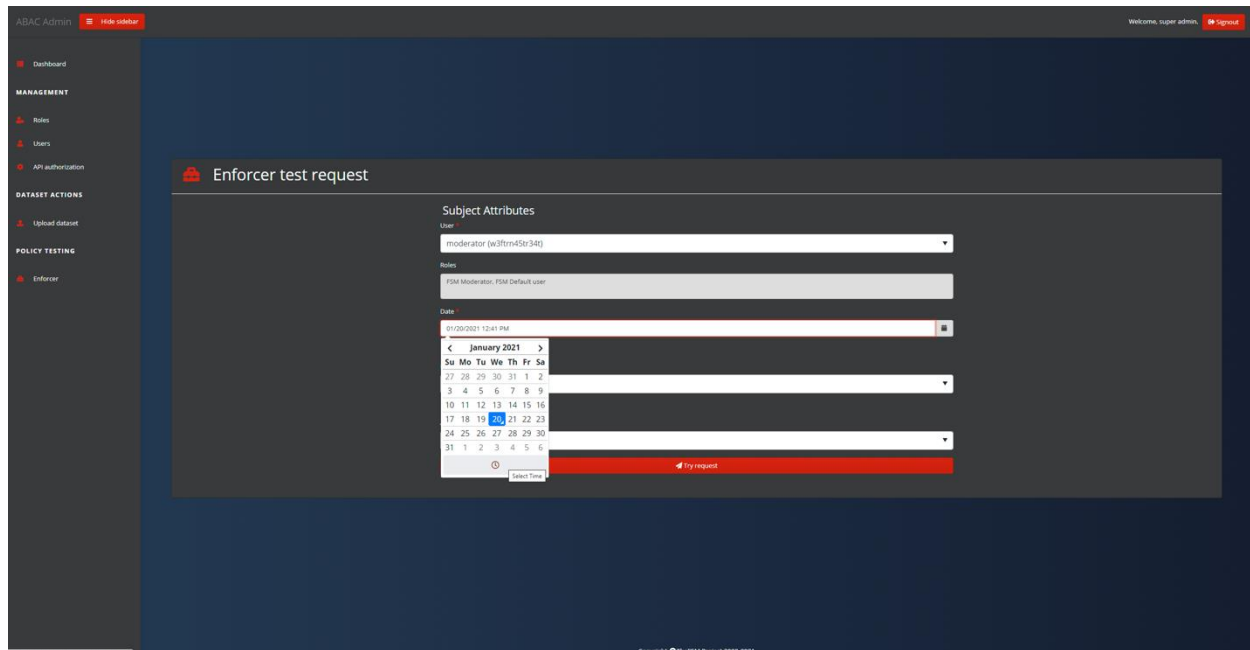


Figure 16: Advanced date picker.

The above screenshot shows the advanced date picker (a reusable UI element in other parts of the platform). Instead of forcing the user to input a date in a specific format and potentially make all sorts of mistakes (order of day, month, year, separator delimiter, 24 hour or 12 hour format

etc.), the date picker enabled the user to interact with a calendar and select the exact date they wish. The calendar supports direct year, month and day selection, while also allowing the user to select the exact hour and minute as needed, showing explicitly the 12/24 hour status. While selecting the date, the input date shown in the field is automatically formed (the user is not allowed to manually input anything there). TheFSM Platform will further apply the best practices mentioned previously, to ensure a successful user experience.

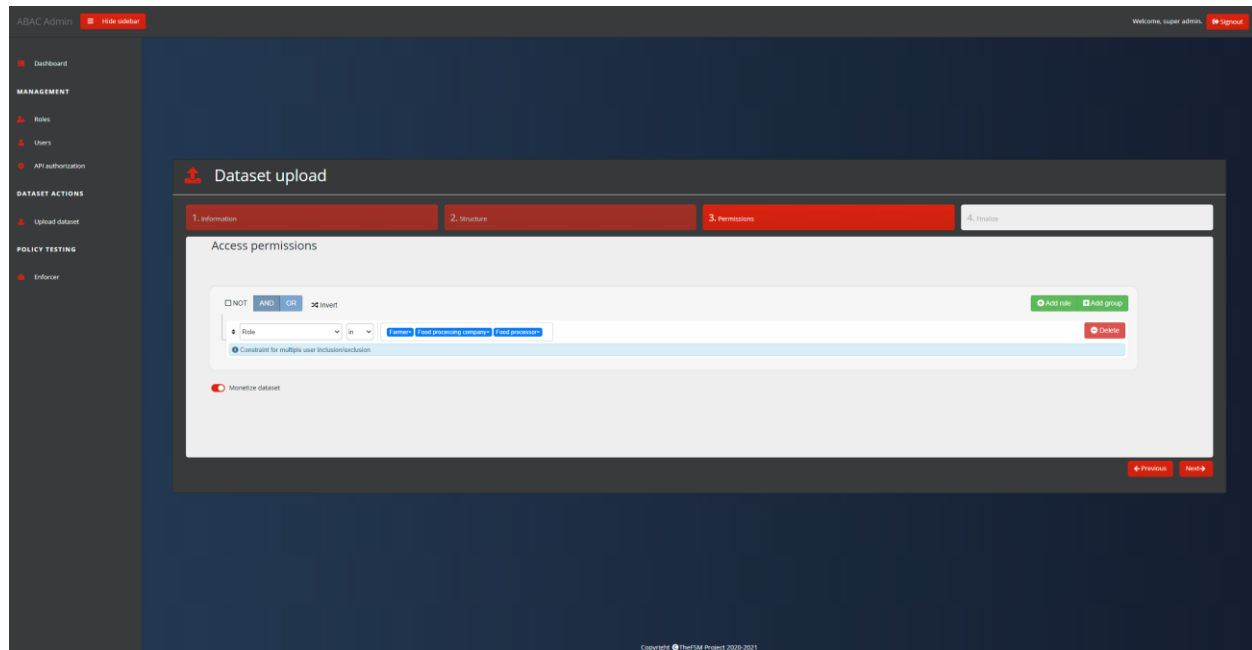
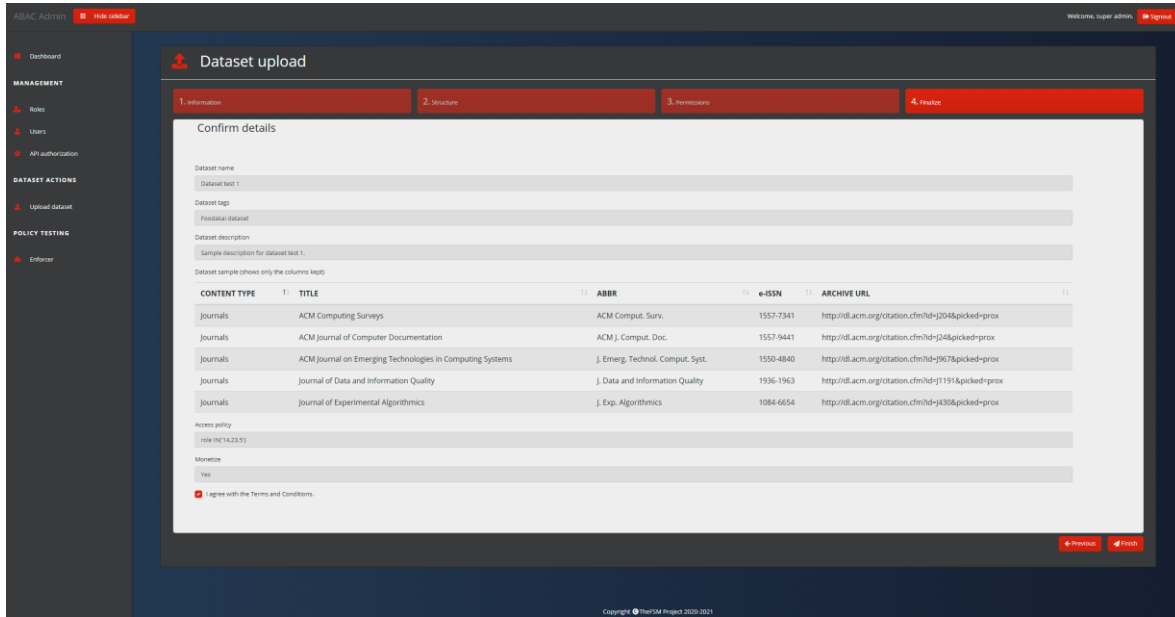


Figure 17: Dataset upload step 3, access policy setup and optional monetization.

Next, “Access permissions” follows. This step contains a query builder almost equivalent to the one used in administrators’ policy enforcement (it needs to be adapted to avoid setting up certain conditions that a regular user should never be able to enforce on their own). Roles and users for potential access conditions are obtained from ABAC’s roles and users management services, respectively. After setting up the access permissions condition(s), the user can optionally select to monetize their dataset.



Dataset upload

1. Information 2. Structure 3. Monetization 4. **Finalize**

Confirm details

Dataset name:

Dataset tags:

Dataset description:

Sample description for dataset test 1:

Dataset sample (show only the columns kept)

CONTENT TYPE	TITLE	ABBR	e-ISSN	ARCHIVE URL
Journals	ACM Computing Surveys	ACM Comput. Surv.	1557-7341	http://dl.acm.org/citation.cfm?id=12048&picked=prox
Journals	ACM Journal of Computer Documentation	ACM J. Comput. Doc.	1557-9441	http://dl.acm.org/citation.cfm?id=12048&picked=prox
Journals	ACM Journal on Emerging Technologies in Computing Systems	J. Emerg. Technol. Comput. Syst.	1550-4840	http://dl.acm.org/citation.cfm?id=12048&picked=prox
Journals	Journal of Data and Information Quality	J. Data and Information Quality	1936-1963	http://dl.acm.org/citation.cfm?id=12048&picked=prox
Journals	Journal of Experimental Algorithms	J. Exp. Algorithms	1084-6654	http://dl.acm.org/citation.cfm?id=12048&picked=prox

Access policy:

Monetize:

☒ I agree with the Terms and Conditions.

Figure 18: Dataset upload step 4, overview summary and final confirmation.

Finally, step 4 is “Confirm details”. This step provides an overview summary of everything the user has required so far, allowing them to inspect their information with a quick glance and opt to finalize the process. It is important to note that the process can only be finalized **after** the user has checked that they agree with the terms and conditions of the platform (the last checkbox in the above screenshot).

Upon finalization and before being stored, the dataset is forwarded to Data Handler and Data Staging components, which will process the columns of the dataset, mapping them to proper semantic entities from the Semantic Mapper and utilizing the dataset’s metadata to enable analytics on the dataset when the dataset is finally stored. If this operation succeeds, the authentication service will generate a new DID for the dataset.

Three more phases occur under the hood, but are optional. Depending on the dataset tag, the dataset might need to go through the OTN DLT services infrastructure (e.g., when dealing with smart contracts), while the user is prompted to a few more steps, if monetization is involved. Finally, if the dataset is tagged to be an Agroknow or Agrivi dataset, additional steps utilizing Agroknow’s or Agrivi’s services/components might be required (important note: this step is a work in progress as of the moment of writing this report). After all steps are successfully completed, only then is the dataset actually stored in the platform in the Secure Storage.

4 THEFSM DATA MARKET MOCKUPS

4.1 TheFSM Data Market

This section includes the low fidelity mockups of TheFSM Data Market, developed under T3.5, started on M10. TheFSM Platform prototype, presented in the previous section and TheFSM Data Market will be integrated both aesthetically and technically in the next release of the platform. T3.5 started on M10, thus, the provided mockups try to cover as many functionalities as possible. Their scope is to drive the conceptualization of TheFSM platform from the end-user's perspective. The current mockups will be followed by high fidelity mockups (delivered until M14) which will guide the development of the platform UIs.

4.1.1 Welcome page

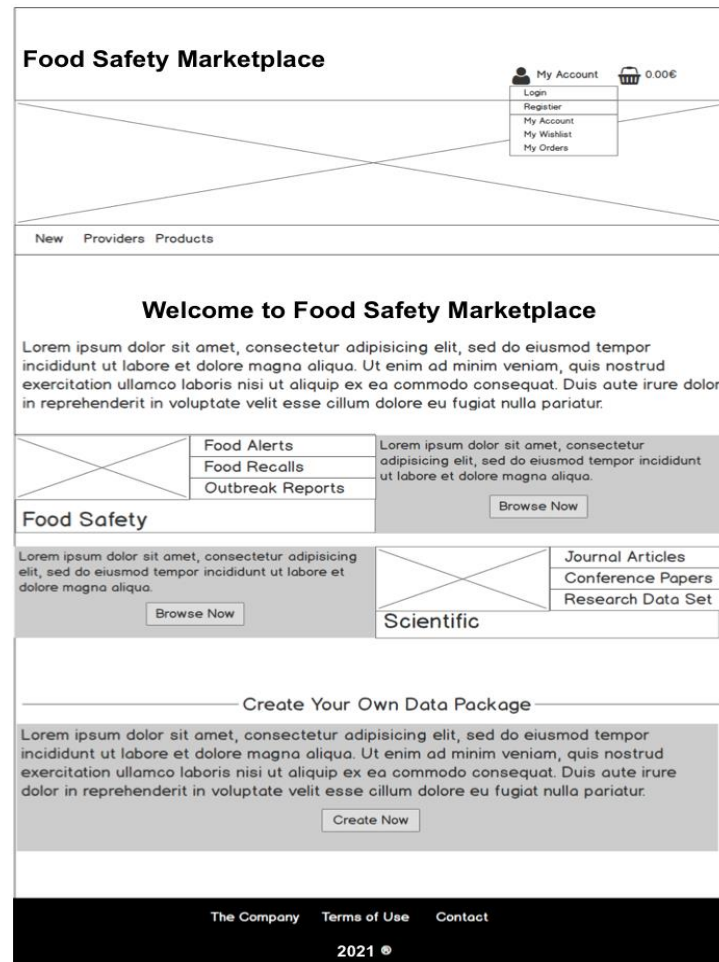


Figure 19: Welcome page.

The above page is the welcome page when the end user visits the marketplace. They are very limited at this point since they are not logged in, but they can still browse dataset categories and create their own data package.

4.1.2 Dataset providers

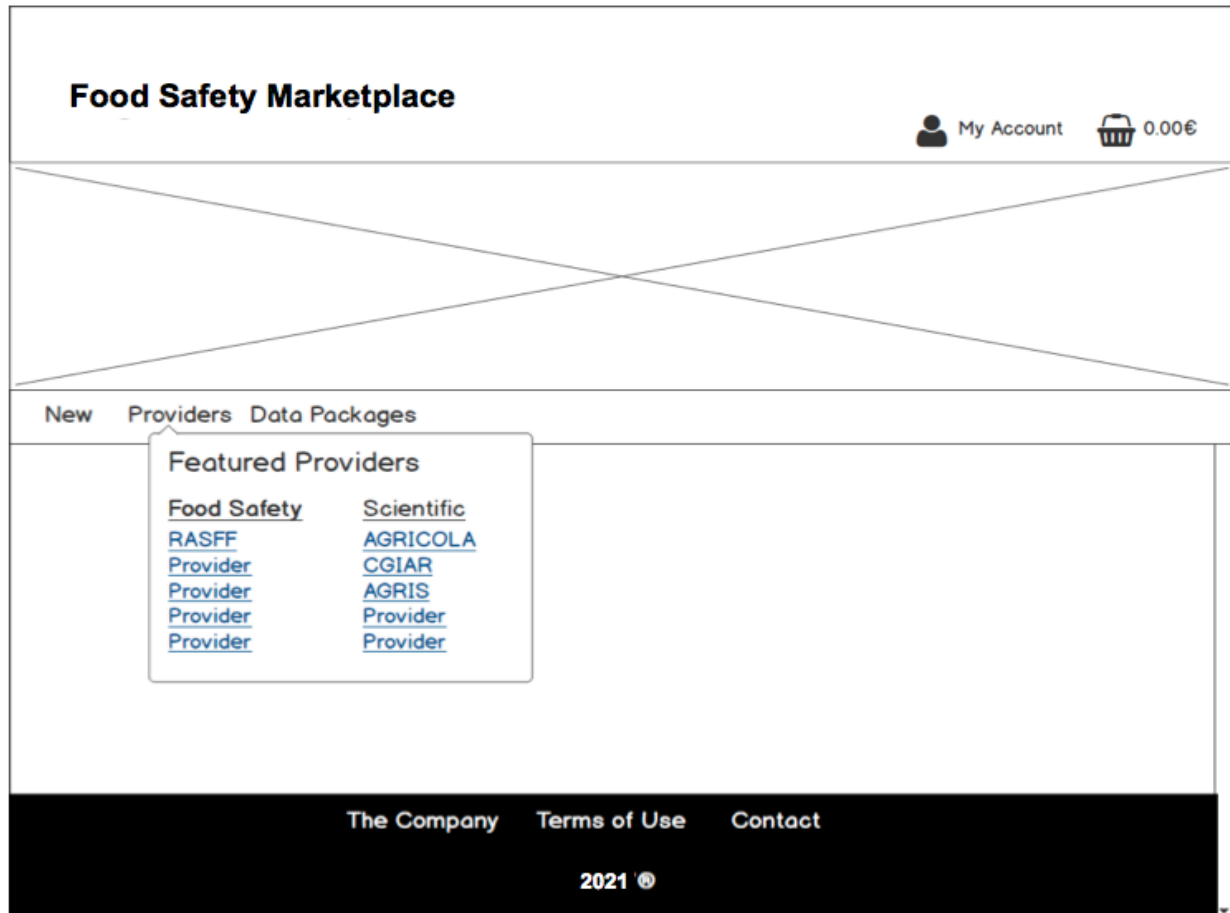


Figure 20: Dataset providers.

This page shows the data providers, as a quick means to inspect datasets of said provider.

4.1.3 Thematic categories of dataset packages

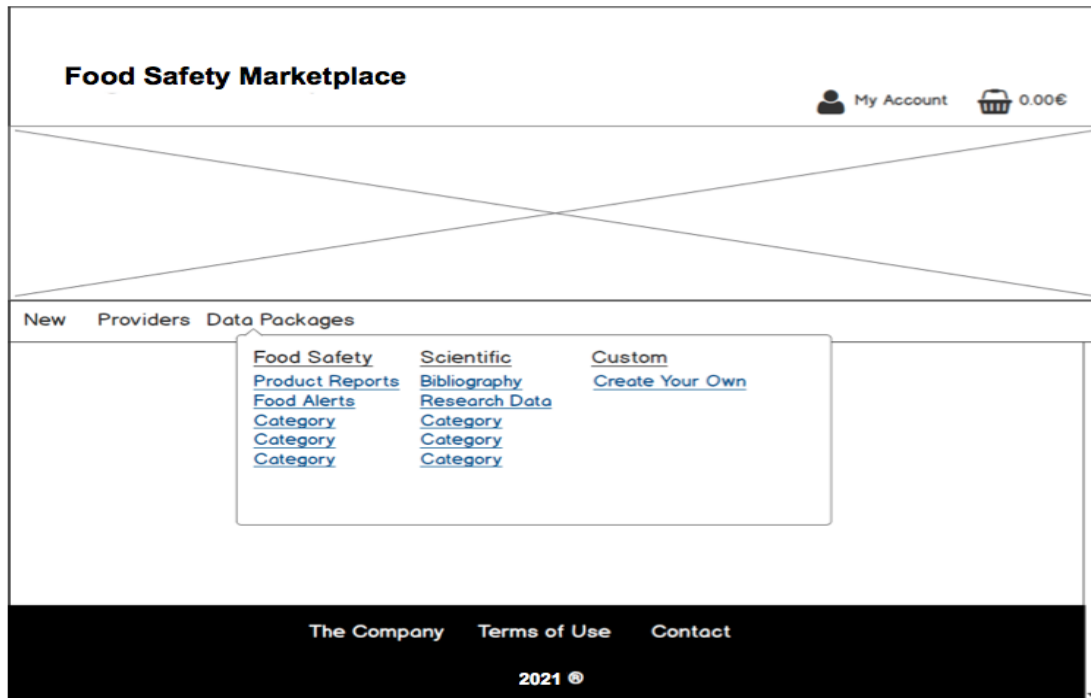
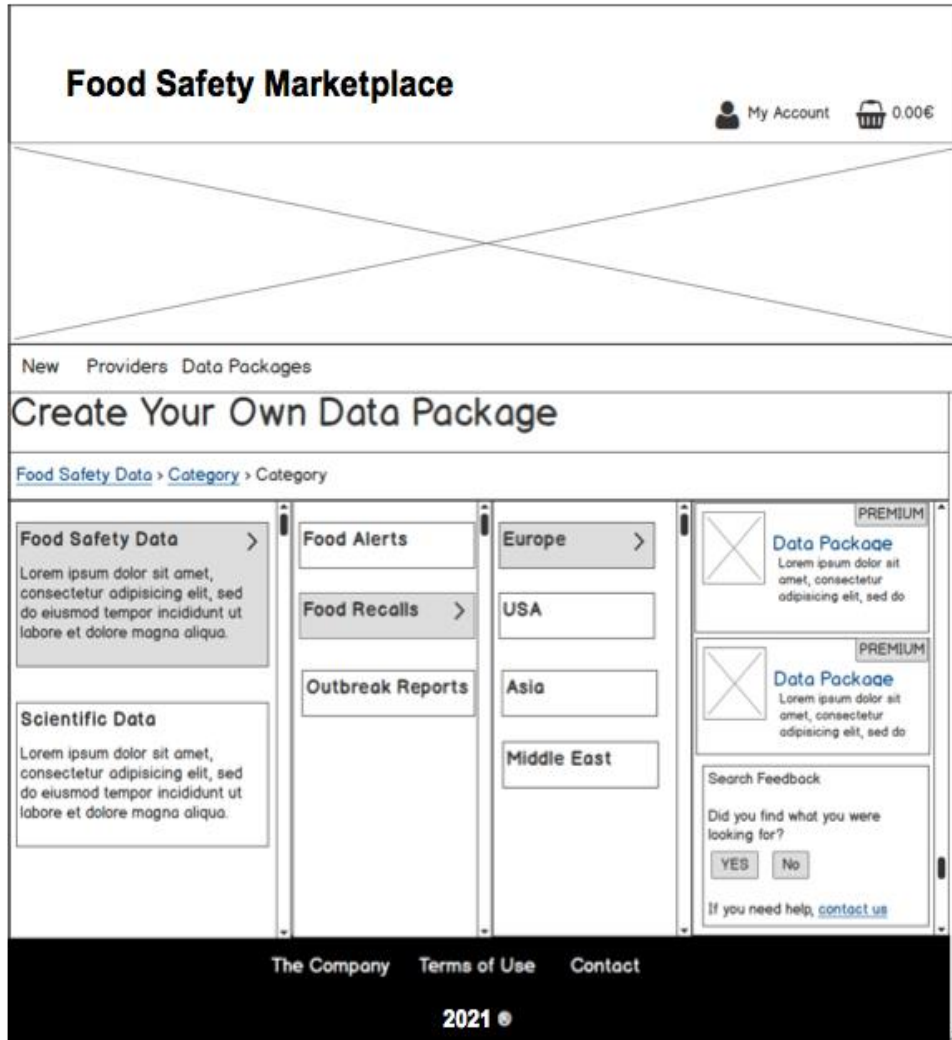


Figure 21: Thematic categories of dataset packages.

This page displays a shortcut to existing data packages according to predefined categories, additionally offering the option to create a custom data package.

4.1.4 Creation of custom data package



The screenshot shows a web interface for 'Food Safety Marketplace'. At the top, there's a header with the site name and user account information ('My Account' and a shopping cart icon showing '0.00€'). Below the header is a large placeholder area with a large 'X' across it. The main content area is titled 'Create Your Own Data Package' and includes a breadcrumb trail: 'Food Safety Data > Category > Category'. The interface is divided into several sections:

- Left Sidebar:** Contains two main categories: 'Food Safety Data' and 'Scientific Data', each with a description and a right-pointing arrow.
- Center Column:** Features three stacked boxes: 'Food Alerts', 'Food Recalls', and 'Outbreak Reports', each with a right-pointing arrow.
- Right Column:** Includes a 'Europe' box with a right-pointing arrow, followed by three input fields for 'USA', 'Asia', and 'Middle East'. Below these is a 'Search Feedback' section with a question 'Did you find what you were looking for?' and 'YES'/'No' buttons, and a link 'If you need help, contact us'.
- Rightmost Column:** Displays two 'Data Package' items, each marked as 'PREMIUM' and featuring a placeholder image and a description.

At the bottom, there's a footer with links for 'The Company', 'Terms of Use', and 'Contact', and a copyright notice '2021'.

Figure 22: Creation of custom data package.

This page displays user input while setting up a custom data package. Options include data type (food safety or scientific data), subtype (alerts, recalls, reports) and source area (Europe, USA, Asia, Middle East) etc. All this information serves as metadata also.

4.1.5 Selection of datasets from categories and providers

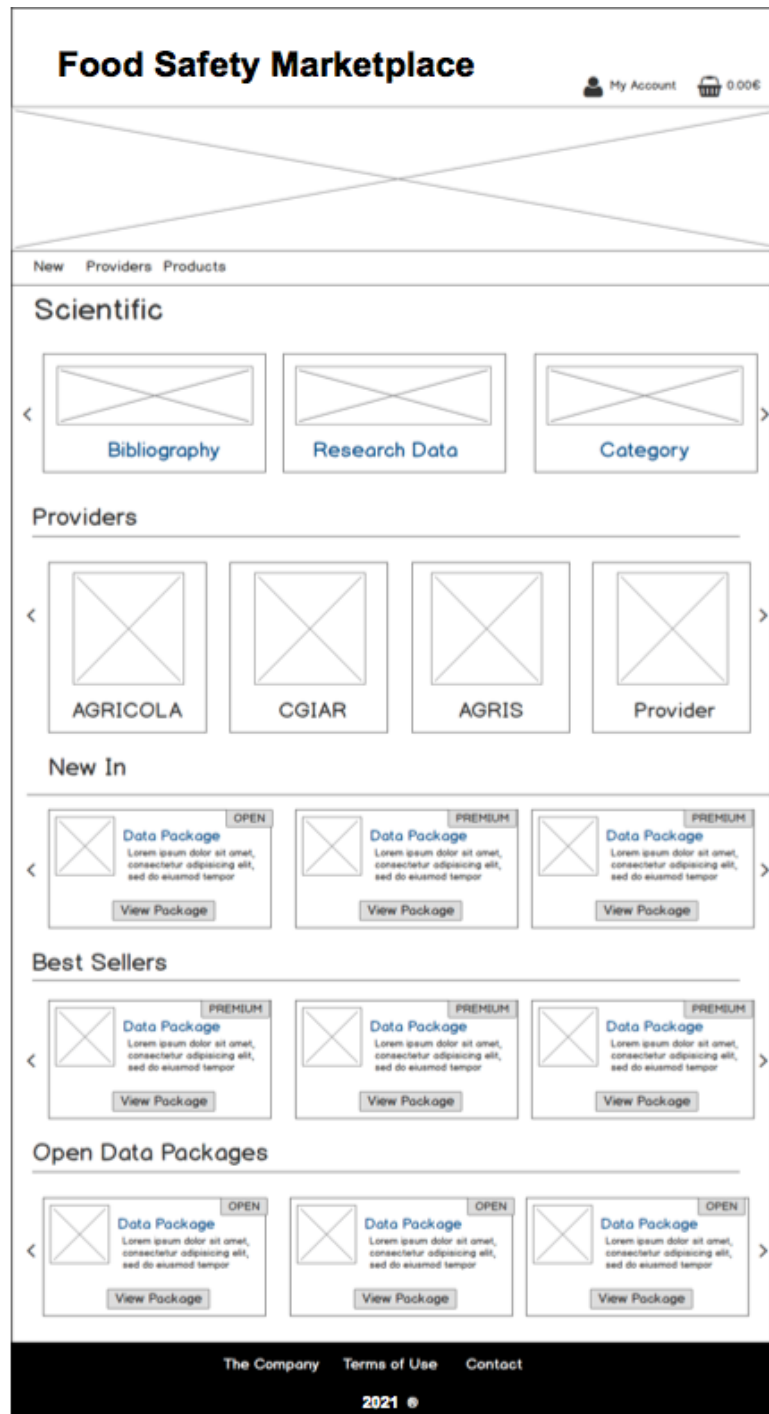


Figure 23: Selection of datasets from categories and providers to form custom package.

This page displays selection of datasets from various categories and data providers in order to form a custom package.

4.1.6 Advanced filtered dataset search

Food Safety Marketplace

My Account
 0.00€

[New](#)
[Providers](#)
[Products](#)

Refine By

Year

X Clear

1965

Now

Size

X Clear

☐ 0 - 100 records
 ☐ 100 - 1.000 records
 ☐ 1.000 - 10.000 records
 ☐ 10.000 - 100.000 records
 ☐ > 1M records

Providers

X Clear

☐ AGRIS
 ☐ AGRICOLA
 ☐ Provider
 ☐ Provider

View More

Type

X Clear

☐ Premium
 ☐ Open

Delivery

X Clear

☐ API
 ☐ File Download

Category

X Clear

☐ Option 1
 ☐ Option 2
 ☐ Option N

Sort By

Price: High to Low

Data Package

PREMIUM

Lorem ipsum dolor sit amet,
 consectetur adipiscing elit,
 sed do eiusmod tempor

View Package

Data Package

PREMIUM

Lorem ipsum dolor sit amet,
 consectetur adipiscing elit,
 sed do eiusmod tempor

View Package

Data Package

PREMIUM

Lorem ipsum dolor sit amet,
 consectetur adipiscing elit,
 sed do eiusmod tempor

View Package

Data Package

PREMIUM

Lorem ipsum dolor sit amet,
 consectetur adipiscing elit,
 sed do eiusmod tempor

View Package

Data Package

OPEN

Lorem ipsum dolor sit amet,
 consectetur adipiscing elit,
 sed do eiusmod tempor

View Package

Data Package

PREMIUM

Lorem ipsum dolor sit amet,
 consectetur adipiscing elit,
 sed do eiusmod tempor

View Package

[Previous](#)

1

2

3

4

5

[Next](#)

Search Feedback

Did you find what you were looking for?

YES

No

If you need help, [contact us](#)

[The Company](#)
[Terms of Use](#)
[Contact](#)

2021 ©

Figure 24: Advanced filtered dataset search.

D3.2 | Annual Releases of TheFSM Platform

41

This page displays potential filters used in advanced dataset search. The filters include parameters such as categories, delivery type, dataset type, provider, size and upload year.

4.1.7 Dataset sampling and purchase

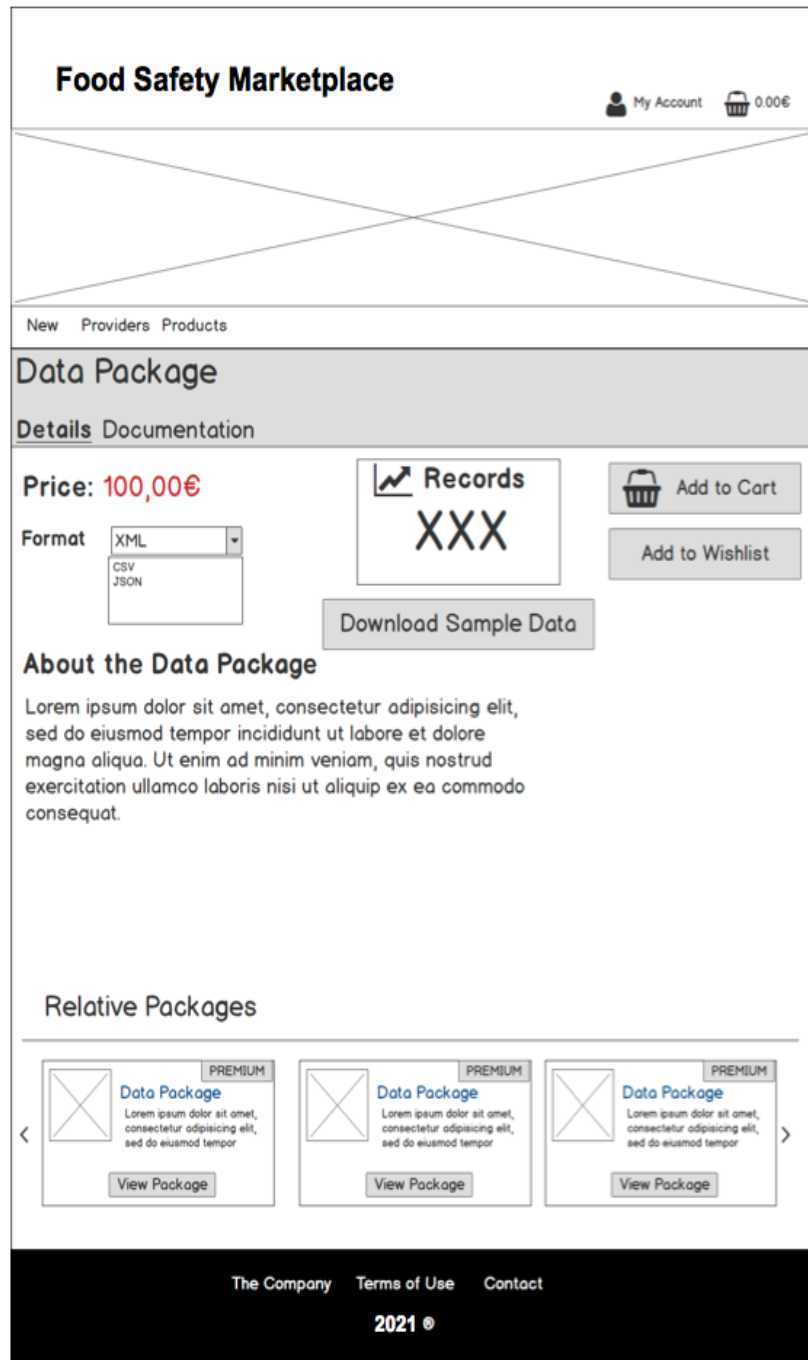


Figure 25: Dataset sampling and purchase.

This page displays the dataset sampling capabilities before purchasing a premium dataset. Sampling is important, since it allows the end user to inspect a sample of the dataset before purchase, compared to making blind (and potentially risky) purchases.

4.1.8 Ordering a dataset

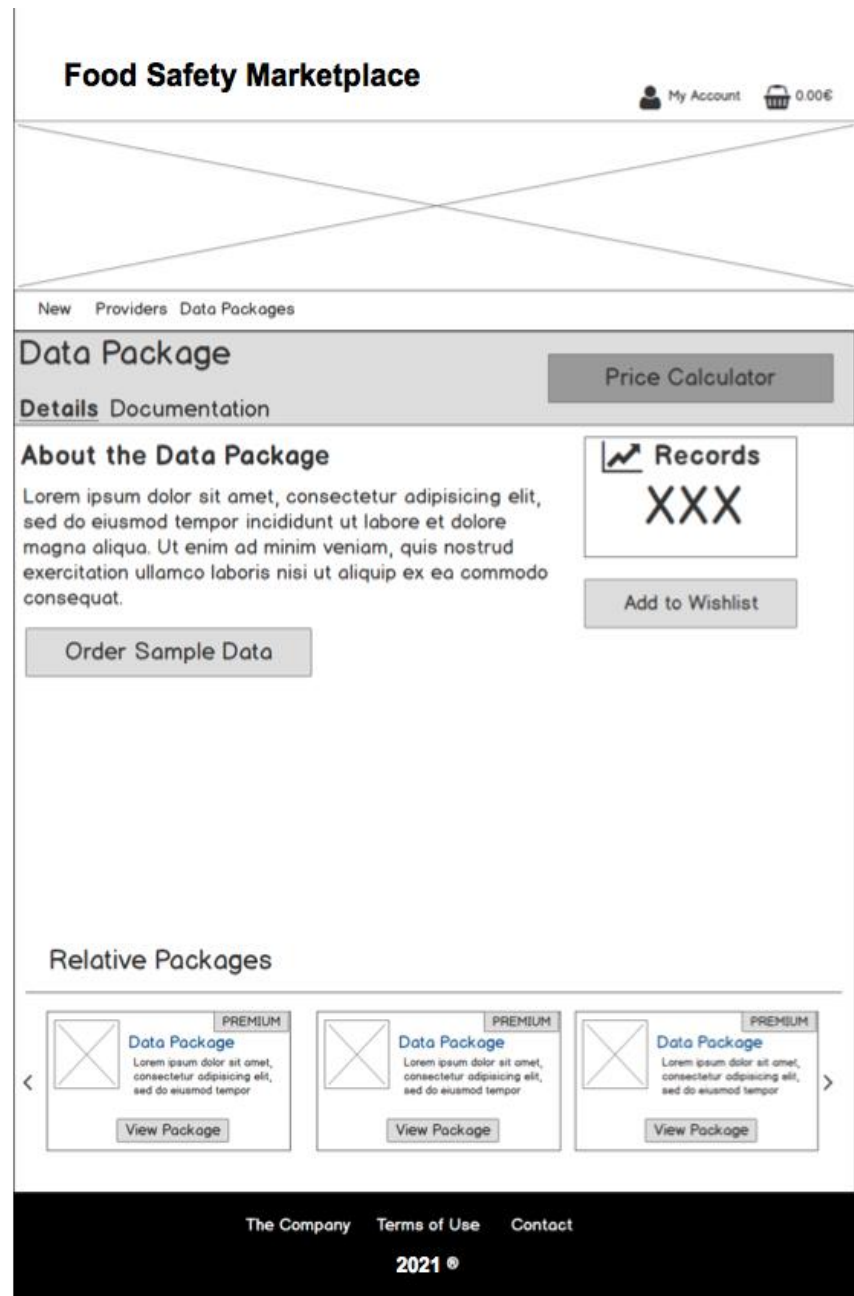


Figure 26: Ordering a dataset.

This page illustrates the ordering of sample data, if the dataset is not yet available.

4.1.9 Dataset's metadata display

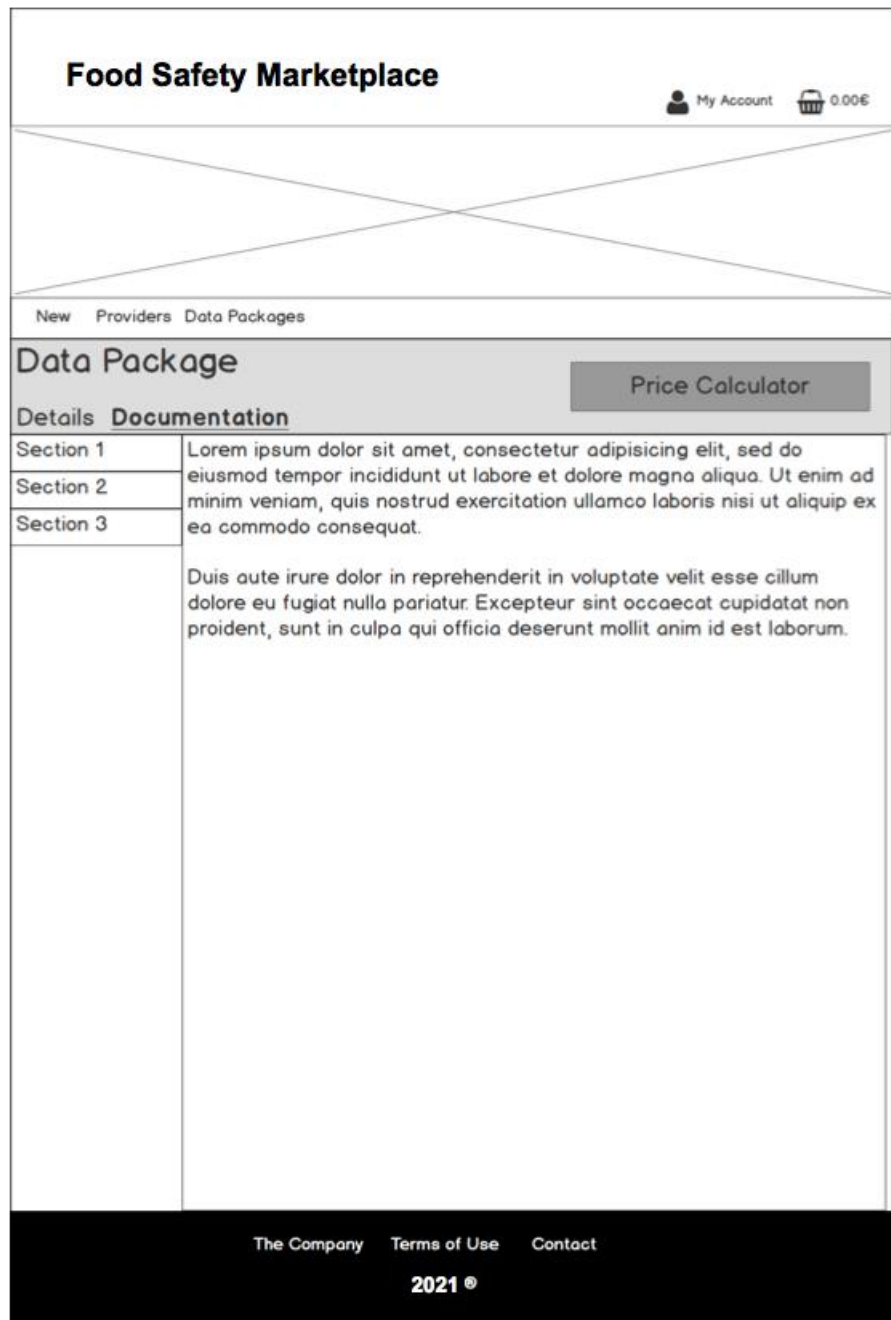
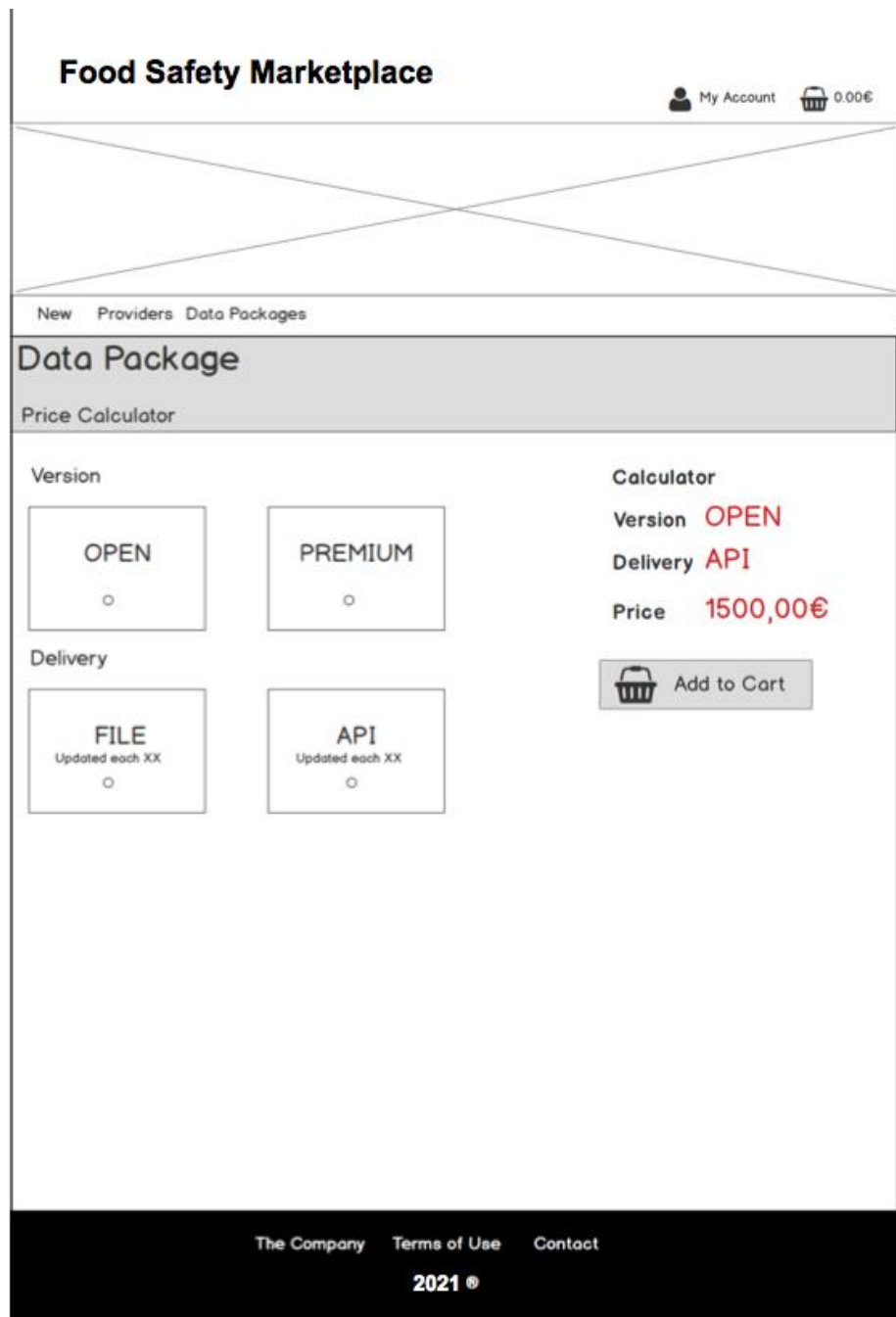


Figure 27: Dataset's metadata display.

This page displays the dataset's metadata.

4.1.10 Versioning and delivery format during purchase process



Food Safety Marketplace

My Account 0.00€

New Providers Data Packages

Data Package

Price Calculator

Version

OPEN

PREMIUM

Delivery

FILE
Updated each XX


API
Updated each XX

Calculator

Version **OPEN**

Delivery **API**

Price **1500,00€**

 Add to Cart

The Company Terms of Use Contact

2021 ®

Figure 28: Versioning and delivery format during purchase process.

This page displays versioning options and delivery format during the purchase process.

4.1.11 Monetized dataset added to cart

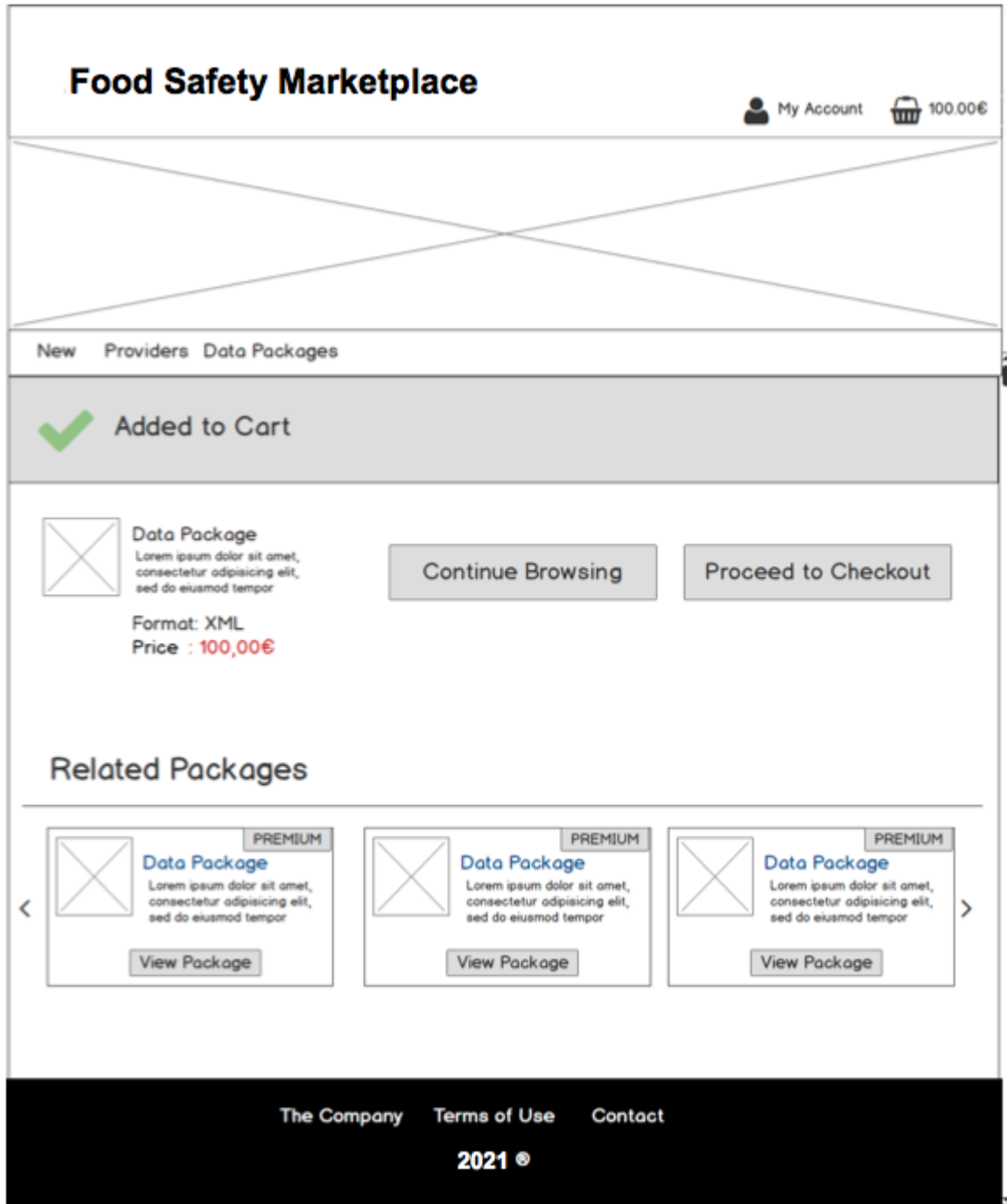


Figure 29: Monetized dataset added to cart.

This page displays visual feedback when adding a dataset to the cart.

4.1.12 Cart checkout

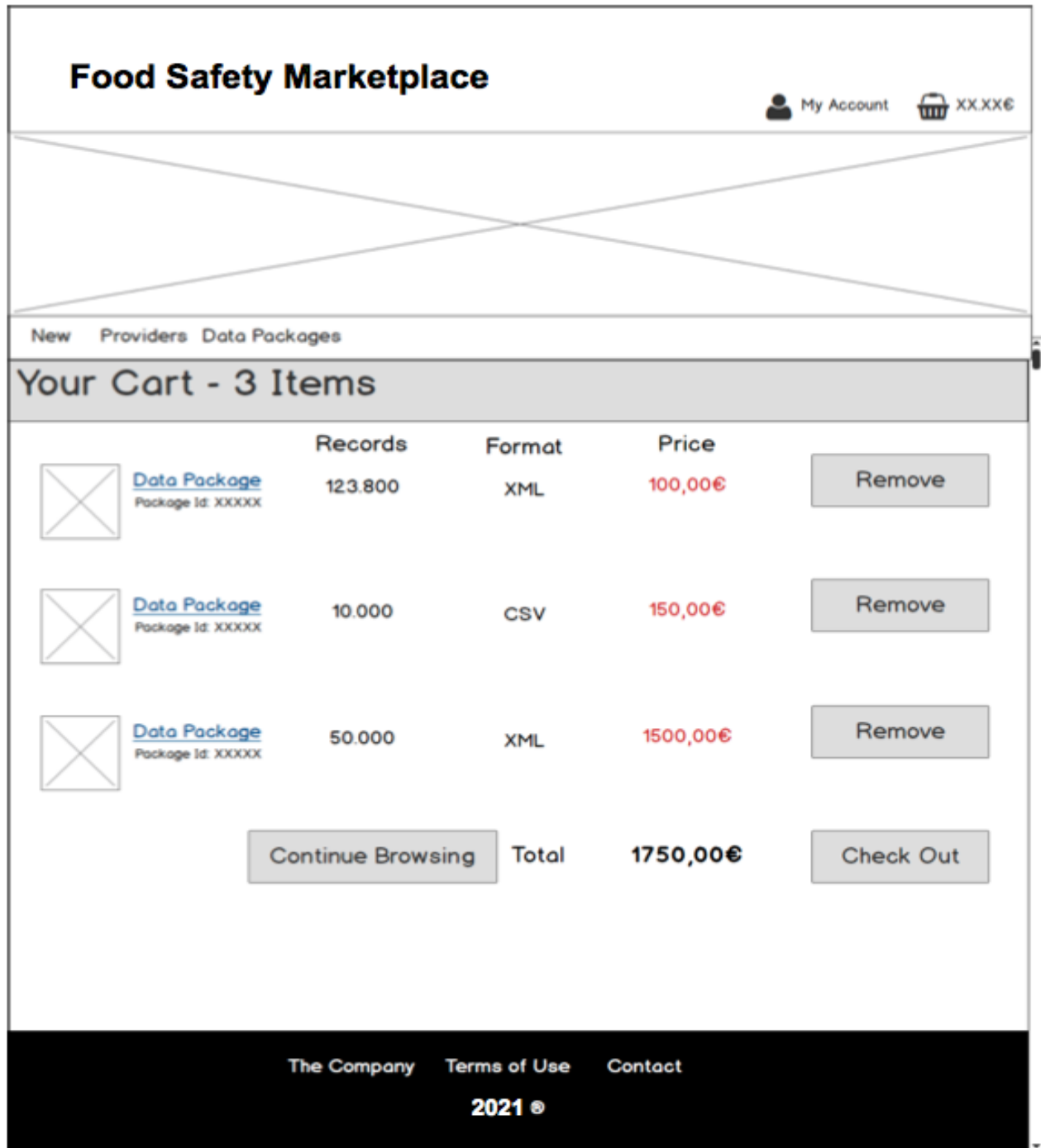



Figure 30: Cart checkout.

This page displays the cart checkout.

4.1.13 Login prompt when trying to checkout



The screenshot shows the 'Food Safety Marketplace' website. At the top, there is a navigation bar with the site name, a 'My Account' link with a user icon, and a shopping cart icon showing 'XX.XX€'. Below this is a large banner area with a diagonal cross. Under the banner is a navigation menu with 'New', 'Providers', and 'Data Packages'. The main content area displays a message: 'To proceed with the check out you must first provide your credentials'. Below this is a login form with a 'Log in' button and a 'New to Data Shop' link. The form contains fields for 'Username' and 'Password', a 'Forgot your Password?' link, and a 'Log in' button. Below the form is a 'Back to Cart' button. At the bottom of the main content area is a note: 'Note: Instructions for downloading your data package or registering to the API will be sent via email after the completion of the order and the confirmation of the payment'. The footer contains links for 'The Company', 'Terms of Use', and 'Contact', followed by '2021 ©'.

Food Safety Marketplace

My Account XX.XX€

New Providers Data Packages

To proceed with the check out you must first provide your credentials

Log in New to Data Shop

Username

Password

[Forgot your Password?](#)

Log in

[Back to Cart](#)

Note: Instructions for downloading your data package or registering to the API will be sent via email after the completion of the order and the confirmation of the payment

The Company Terms of Use Contact

2021 ©

Figure 31: Login prompt when trying to checkout (user is not logged in).

This page displays the login prompt shown when user tries to checkout, but they are not yet logged in to the platform.

4.1.14 On the fly registration of unregistered user



Food Safety Marketplace

My Account XX.XX€

New Providers Data Packages

To proceed with the check out you must first provide your credentials

[Log in](#) [New to Data Shop](#)

Username

Email

Password

Repeat Password

[Register](#)

[Back to Cart](#)

Note: Instructions for downloading your data package or registering to the API will be sent via email after the completion of the order and the confirmation of the payment

The Company Terms of Use Contact

2021 ©

Figure 32: On the fly registration of unregistered user during cart checkout.


This page displays the option to immediately register a new user during the checkout process, if they are not already logged in.

4.1.15 Billing information during checkout process


Food Safety Marketplace

My Account
XX.XX€


New Providers Data Packages



[Data Package](#)
 Package Id: XXXXX
 Records: 123800
 Format: XML **100,00€**



[Data Package](#)
 Package Id: XXXXX
 Records: 123800
 Format: CSV **150,00€**



[Data Package](#)
 Package Id: XXXXX
 Records: 123800
 Format: CSV **1500,00€**

Total 1750,00€

Billing Invoice

Company Name

Street Address

No

ZIP/ Postal Code

State/ Province

City

Country

Select Country

Delivery Method

Send Download Instructions to
☒ My Registered Email
☐ other Email

Proceed to Payment

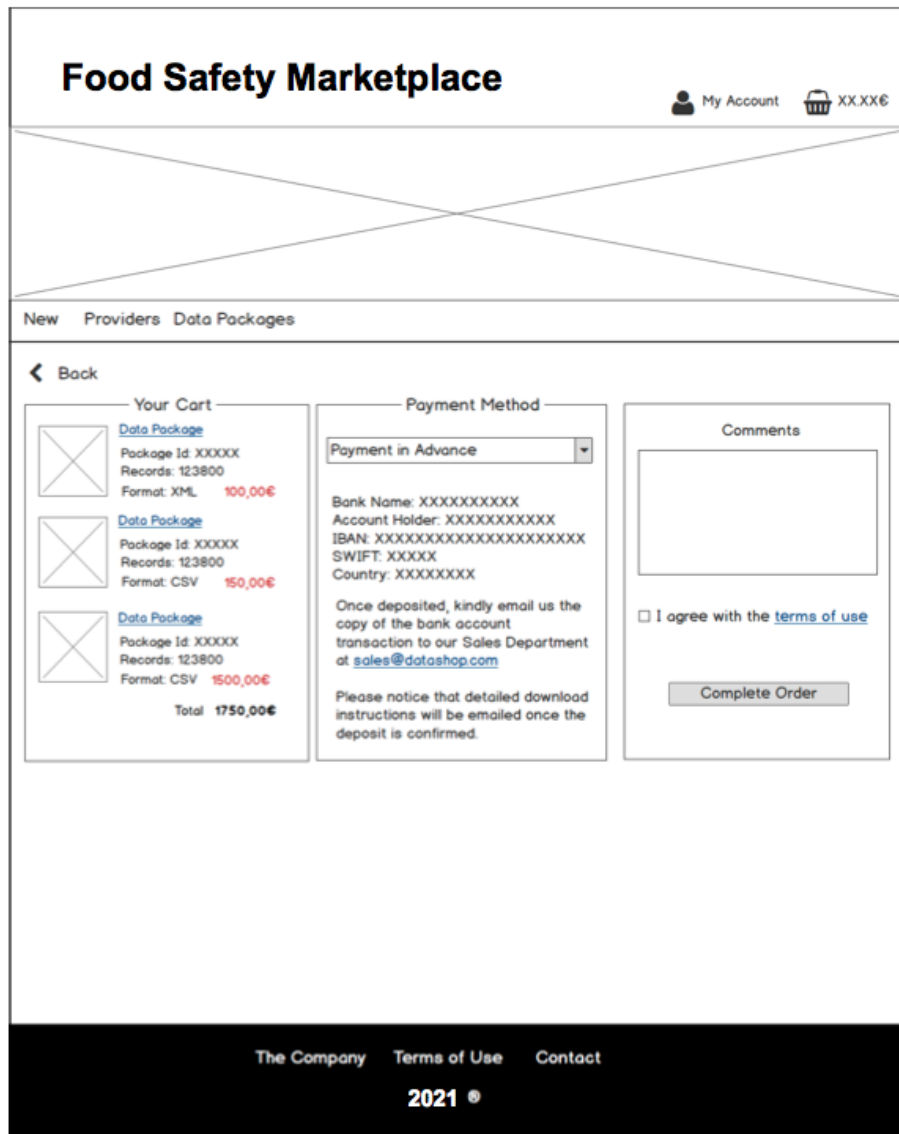
The Company Terms of Use Contact

2021

Figure 33: Billing information during checkout process.

This page displays the billing information input during the checkout process.

4.1.16 User comments and order finalization



Food Safety Marketplace

My Account
 XX.XX€

[New](#)
[Providers](#)
[Data Packages](#)

[Back](#)

Your Cart

[Data Package](#)
 Package Id: XXXXX
 Records: 123800
 Format: XML **100,00€**

[Data Package](#)
 Package Id: XXXXX
 Records: 123800
 Format: CSV **150,00€**

[Data Package](#)
 Package Id: XXXXX
 Records: 123800
 Format: CSV **1500,00€**

Total 1750,00€

Payment Method

Payment in Advance

Bank Name: XXXXXXXXXX
 Account Holder: XXXXXXXXXX
 IBAN: XXXXXXXXXXXXXXXXXXXX
 SWIFT: XXXXX
 Country: XXXXXXXX

Once deposited, kindly email us the copy of the bank account transaction to our Sales Department at sales@datashop.com

Please notice that detailed download instructions will be emailed once the deposit is confirmed.

Comments

☐ I agree with the [terms of use](#)

[Complete Order](#)

[The Company](#)
[Terms of Use](#)
[Contact](#)

2021 ®

Figure 34: User comments and order finalization.

This page displays the order finalization, while also forcing the user to agree to terms of use and provide any additional comments they wish to add.

4.1.17 Order confirmation prompt

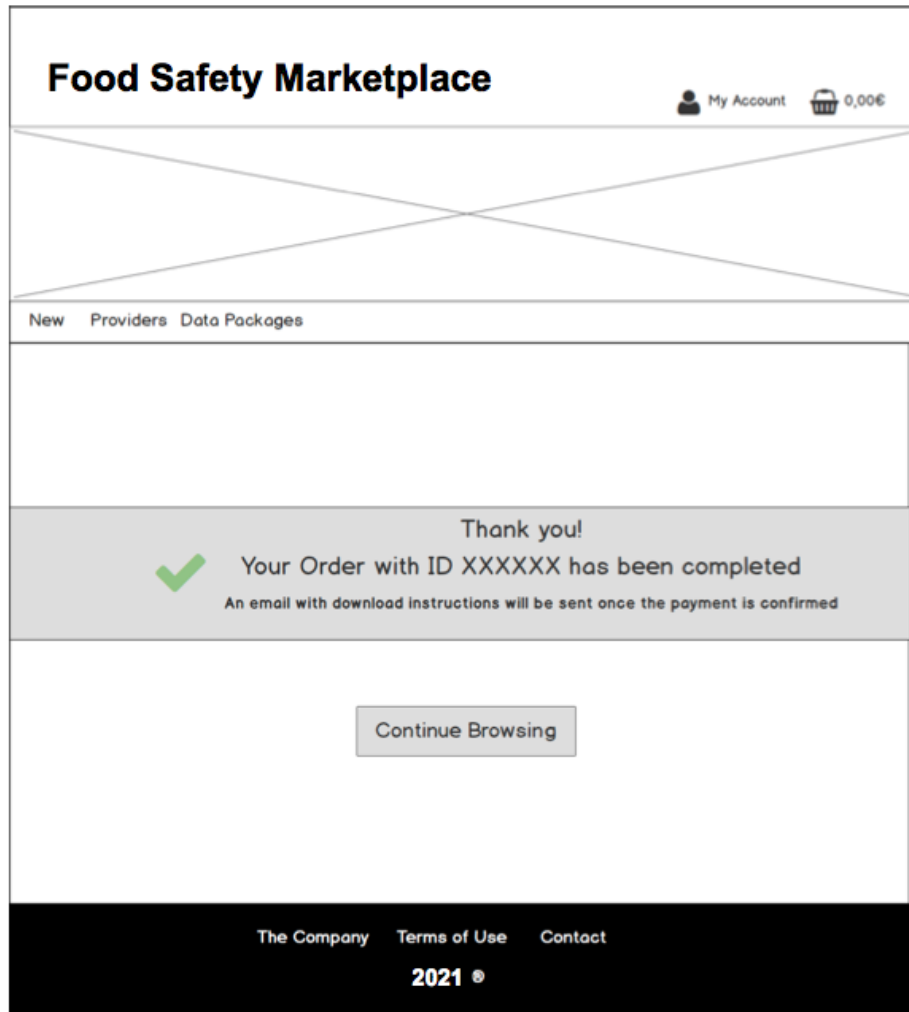


Figure 35: Order confirmation prompt.

This page displays visual feedback, notifying the user that their order has been granted a tracking ID and has been successfully completed.

4.1.18 Representative technological stack

The following is a representative list of technologies which we aim to select for the development of the Data Marketplace:

- **Node JS** for the development of the web API that will be used by the front end
- **elasticsearch** for indexing the information of the datasets that will be published in the Food Safety Market

- **JavaScript** for the development of the front end functionalities
- **Php** for the development of the front end functionalities
- **Wordpress** for the generic content management

5 CONCLUSIONS

In the current demonstrator report we presented the technological stack and infrastructure for the first version of TheFSM platform. The first version of TheFSM Platform covers mainly the backend security and data management mechanisms. More analytically it includes: **access control, user authentication and authorization** and **data security at transit and at rest** and **decentralized identity management** (T3.2, T3.3, T3.4), as well as, **the integration with the backend data curation services** (D2.2, M12) provided the main steps of the data asset upload, mapping, policies definition, storage and encryption-decryption. The UI developed for the demonstrator will be updated and replaced with the final UI adopted by TheFSM Platform, developed within the tasks of TheFSM Data Market (T3.5).

We delved into critical concepts involving the security handling and policy management, since they are critical if we are to ensure the integrity and GDPR principles when protecting the users' data. Explaining conventions and design decisions, we presented the prototype version of the platform we developed both textually and visually. Each functional page and navigation step was documented, while we provided examples of integration points and operations taking place under the hood. Next, we provided the mockups of the way we envision the final iteration of the platform to be, briefly explaining what they represent and their potential functionality. We will keep documenting our progress in the upcoming deliverables, providing detailed sequence diagrams and interaction flows that prove the promised functionality as more interaction flows and use cases are covered.

6 REFERENCES

[1] Amit Sahai, Brent Waters, *Fuzzy Identity-Based Encryption*, 2005.