Smart Dispatcher For Secure And Controlled Sharing Of Distributed Personal And Industrial Data

smashHit

User Guide

Platform Administrator

This project has received funding from the European Union’s Horizon 2020 research and innovation program under grant agreement No. 871477

Introduction

Purpose

This guide aims to help users with a platform administrator role on how to use the smashHit platform and gives knowledge about the different functionalities available.

Audience

This guide is meant for and solely for users of the smashHit platform with platform administrator role. Other roles can find their own user guides under smashhit.eu.

Scope

The contents of this guide are meant to be taken into consideration only when using the smashHit platform and will only cover functionalities meant to be used by the role stated above.

The smashHit team does not take responsibility for improper use of the application or the data provided when not following the instructions given in this guide.

Troubleshooting

For any questions or inquiries about the use of the smashHit platform web application or the contents of it or this guide, or if you find there is no content in this guide for some functionality, please forward it to: info@smashhit.eu

Contact

smashHit Project website: https://smashhit.eu

smashHit platform support: info@smashhit.eu

Contents

Guide

Basic knowledge

The objective of the smashHit framework is to assure trusted and secure sharing of data streams from both personal and industrial platforms, needed to build sectorial and cross-sectorial services, by establishing a framework to facilitate the processing of data subject consent and legal rules and effective contracting, as well as joint security and privacy preserving mechanisms. While these data streams offer new opportunities to build innovative services, they often require the management of a complex set of consent chains.

The main goal of smashHit is thus to overcome these obstacles by offering a set of methods and tools to facilitate the management of consents, supported by semantic models of consent and legal rules. The new tools include traceability of use of data, data fingerprinting and automatic contracting among the data subjects, data controllers and data processors.

Step by step

The “platform administrator” role is meant for users with elevated privileges to manage most of the resources in the system, there is actually one specific functionality reserved for them, which is the management of organization registration requests. This special role is reserved for smashHit staff, and typically it will be assigned to the administrator user for the smashHit Identity Manager (IdM) itself.

As such, this kind of users are not meant to be created or managed by smashHit at all, but rather manually managed by the entity responsible for hosting the smashHit system. Thus, outside of using its privileges to provide support to other users, the main specific administrator responsibility is to review new organization registration requests and either accept them into the platform or reject them.

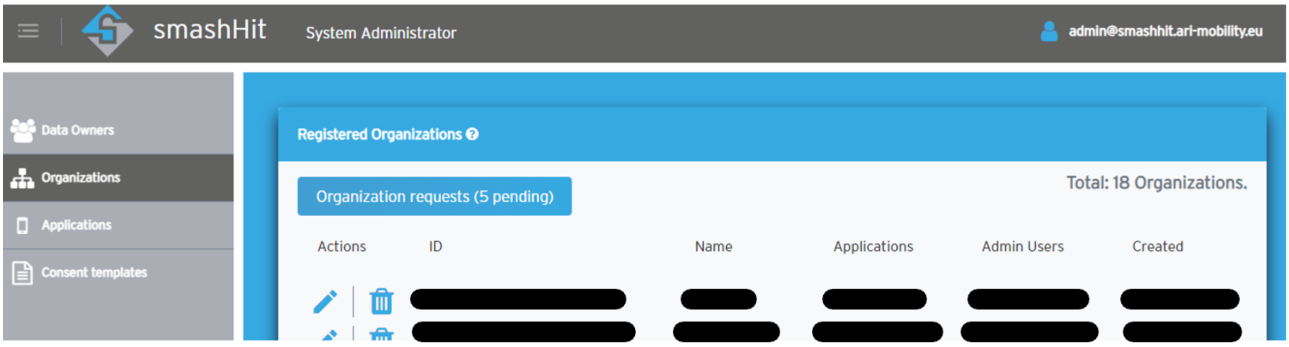
This guide will first offer the administrator a brief guide through the functionalities available of the UI:

- Organization registration (exclusive to the administrator)
- Application management
- Consent template management
- Data owner management

Then, the last 2 sections will offer an insight in some key points concerning the deployment of smashHit and the context-sensitive S&P module specifically, that the platform administrator should be aware of.

Organization Registration

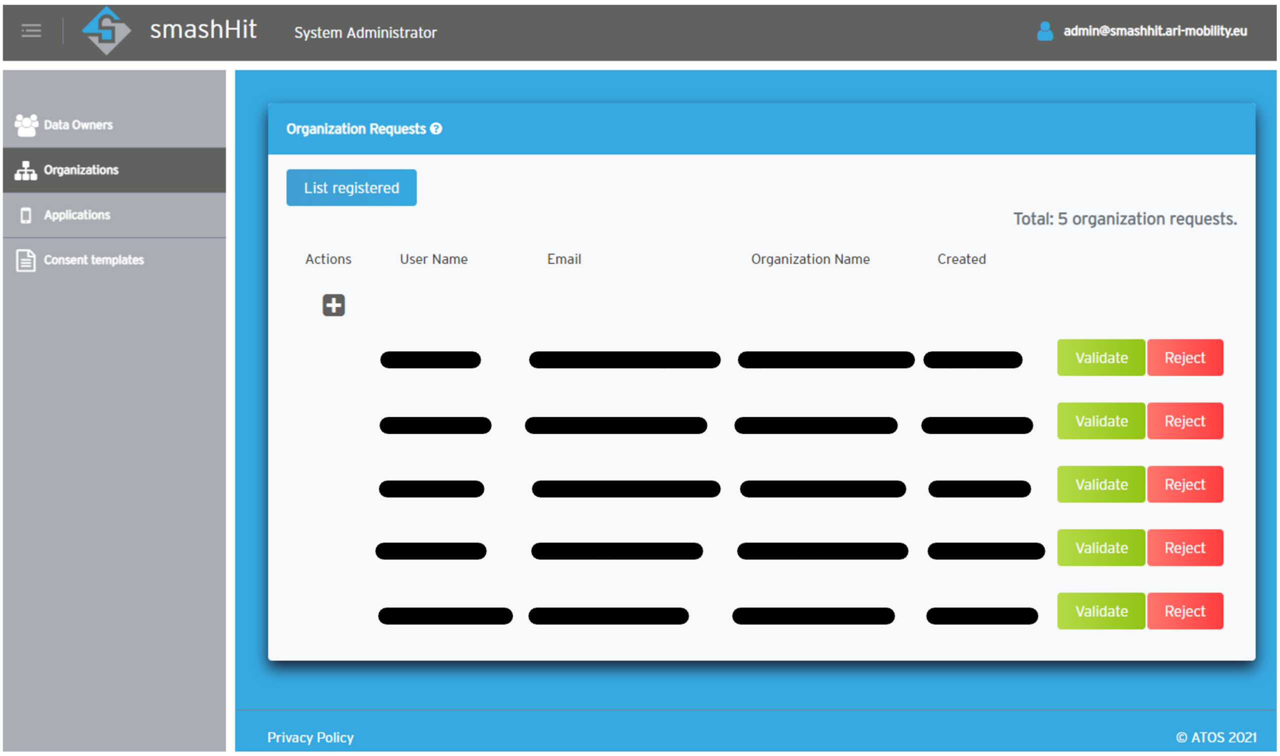
As it can be seen in Figure 1, from the admin panel one can access the organization management interface, where the existing organizations will be listed, but most importantly, administrators can review organization registration requests by clicking the “Organization requests”. The organization registration requests list will be displayed, where the admin can either validate or reject the registration (Figure 2 Organization registration requests).



Total: 18 Organizations.

Actions	ID	Name	Applications	Admin Users	Created
	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]

Figure 1 Administrator’s Organizations view



Total: 5 organization requests.

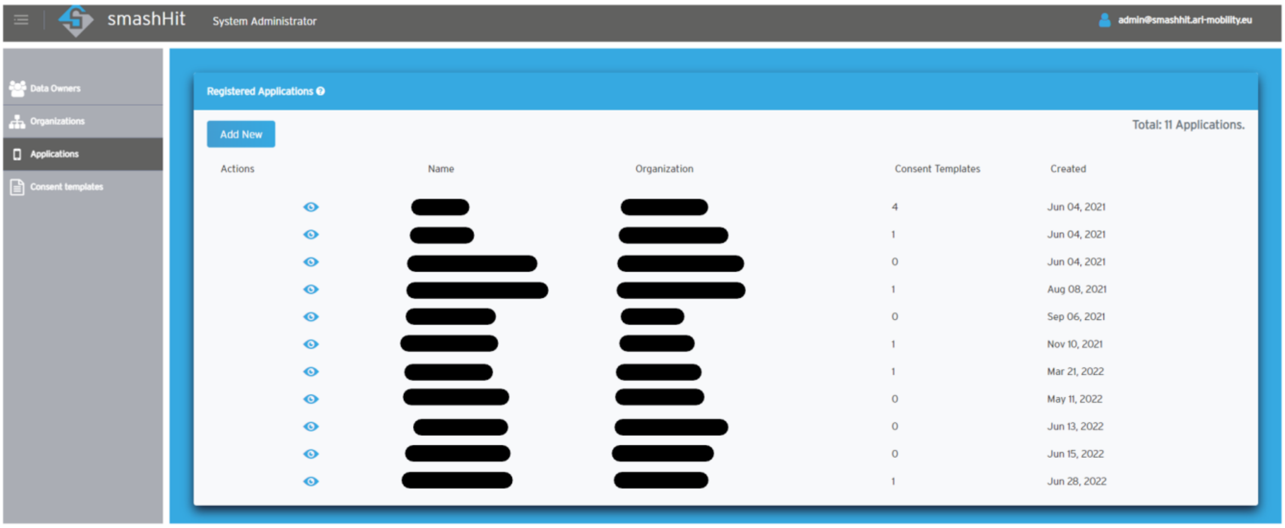
Actions	User Name	Email	Organization Name	Created	
	[redacted]	[redacted]	[redacted]	[redacted]	Validate Reject
	[redacted]	[redacted]	[redacted]	[redacted]	Validate Reject
	[redacted]	[redacted]	[redacted]	[redacted]	Validate Reject
	[redacted]	[redacted]	[redacted]	[redacted]	Validate Reject
	[redacted]	[redacted]	[redacted]	[redacted]	Validate Reject

Figure 2 Organization registration requests

Once an organization request is accepted, the system will automatically register an account for the organization admin user, and then notify him by email with the news that the organization has been approved and next steps to access smashHit.

Application management

As explained, administrators have access to most resources in the platform, which is very helpful to provide support. From the “Application” tab, the administrator can review and manage the existing applications, as it can be seen in Figure 3.

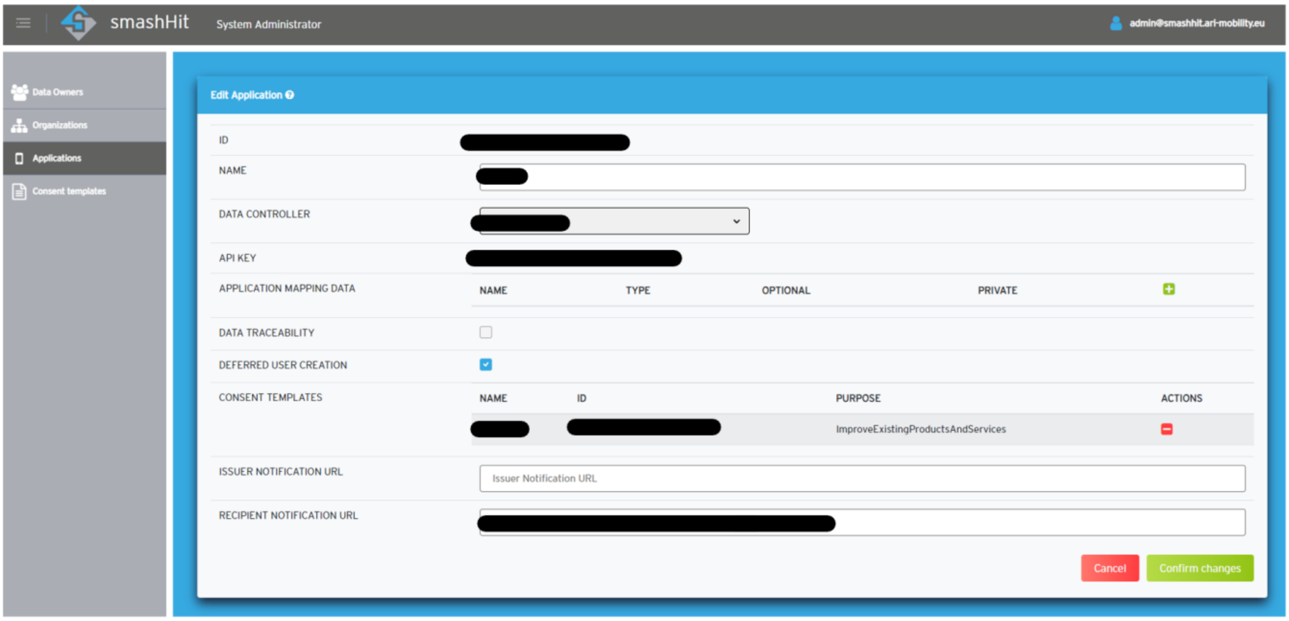


Total: 11 Applications.

Actions	Name	Organization	Consent Templates	Created
👁	[redacted]	[redacted]	4	Jun 04, 2021
👁	[redacted]	[redacted]	1	Jun 04, 2021
👁	[redacted]	[redacted]	0	Jun 04, 2021
👁	[redacted]	[redacted]	1	Aug 08, 2021
👁	[redacted]	[redacted]	0	Sep 06, 2021
👁	[redacted]	[redacted]	1	Nov 10, 2021
👁	[redacted]	[redacted]	1	Mar 21, 2022
👁	[redacted]	[redacted]	0	May 11, 2022
👁	[redacted]	[redacted]	0	Jun 13, 2022
👁	[redacted]	[redacted]	0	Jun 15, 2022
👁	[redacted]	[redacted]	1	Jun 28, 2022

Figure 3 Application management

Clicking the “eye” icon next to the application name will allow the admin to access the detailed view of the application and even edit its configuration in case it is necessary (Figure 4). This might be helpful in case some application must be migrated across organizations or simply to provide general support to organizations.

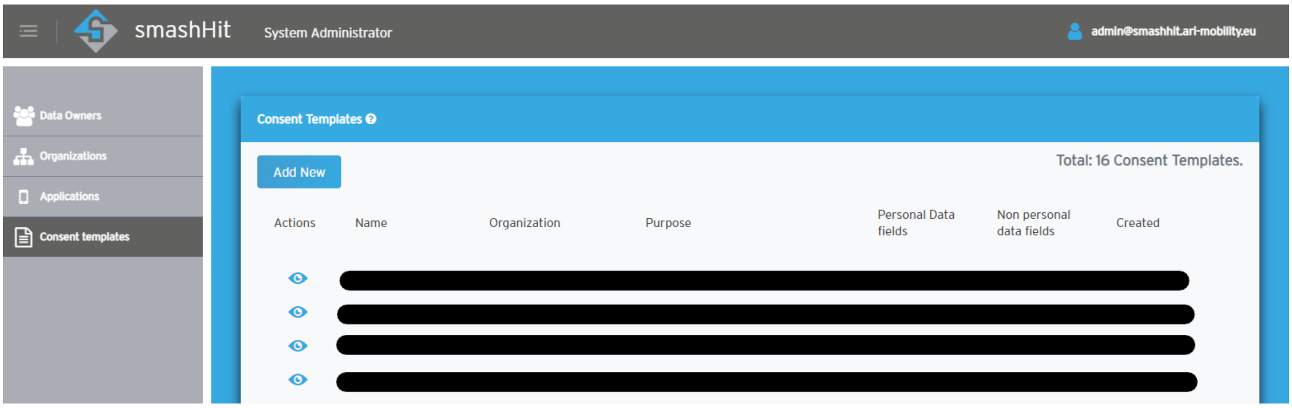


ID	[redacted]
NAME	[redacted]
DATA CONTROLLER	[redacted]
API KEY	[redacted]
APPLICATION MAPPING DATA	NAME TYPE OPTIONAL PRIVATE
DATA TRACEABILITY	☐
DEFERRED USER CREATION	☑
CONSENT TEMPLATES	NAME ID PURPOSE ACTIONS [redacted] [redacted] ImproveExistingProductsAndServices
ISSUER NOTIFICATION URL	Issuer Notification URL
RECIPIENT NOTIFICATION URL	[redacted]

Figure 4 Application support

Consent template management

Similar to applications, an admin user can list and review any consent template defined in the platform by clicking the “Consent templates” option in the GUI, as seen in Figure 5.



smashHit System Administrator admin@smashhit.ari-mobility.eu

- Data Owners
- Organizations
- Applications
- Consent templates

Consent Templates

Add New

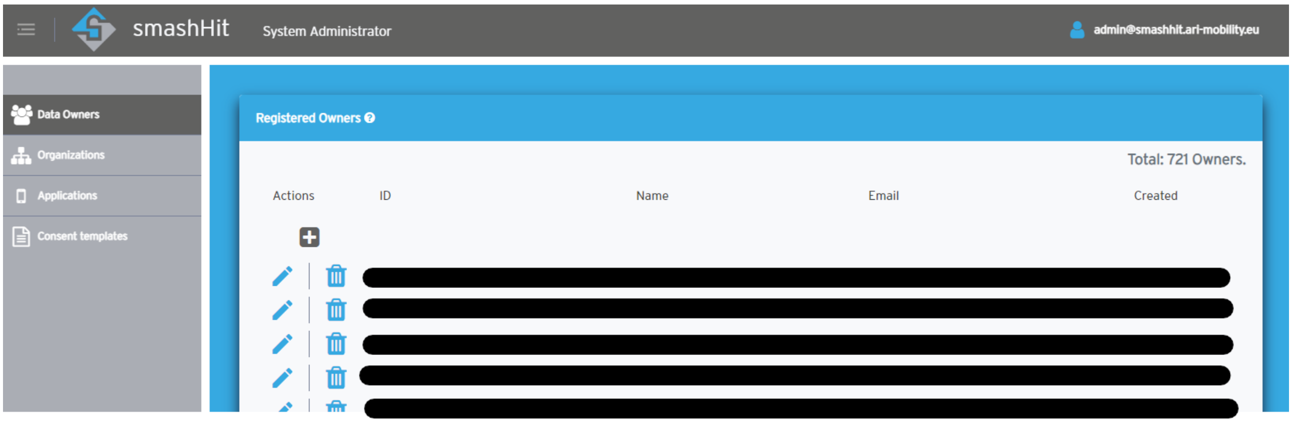
Total: 16 Consent Templates.

Actions	Name	Organization	Purpose	Personal Data fields	Non personal data fields	Created
👁	[redacted]					
👁	[redacted]					
👁	[redacted]					
👁	[redacted]					

Figure 5 Consent template management

Data Owner management

Finally, registered data owners can be listed from the administrator panel (Figure 6), with the additional (recommended) option to access the Keyrock IdM[1] itself directly to manage advanced user settings, such as manually review and adjust roles or even reset a user password to provide support, as seen in Figure 7 and Figure 8. For more details on Keyrock administration, please refer to the official FIWARE documentation[2].



smashHit System Administrator admin@smashhit.ari-mobility.eu

- Data Owners
- Organizations
- Applications
- Consent templates

Registered Owners

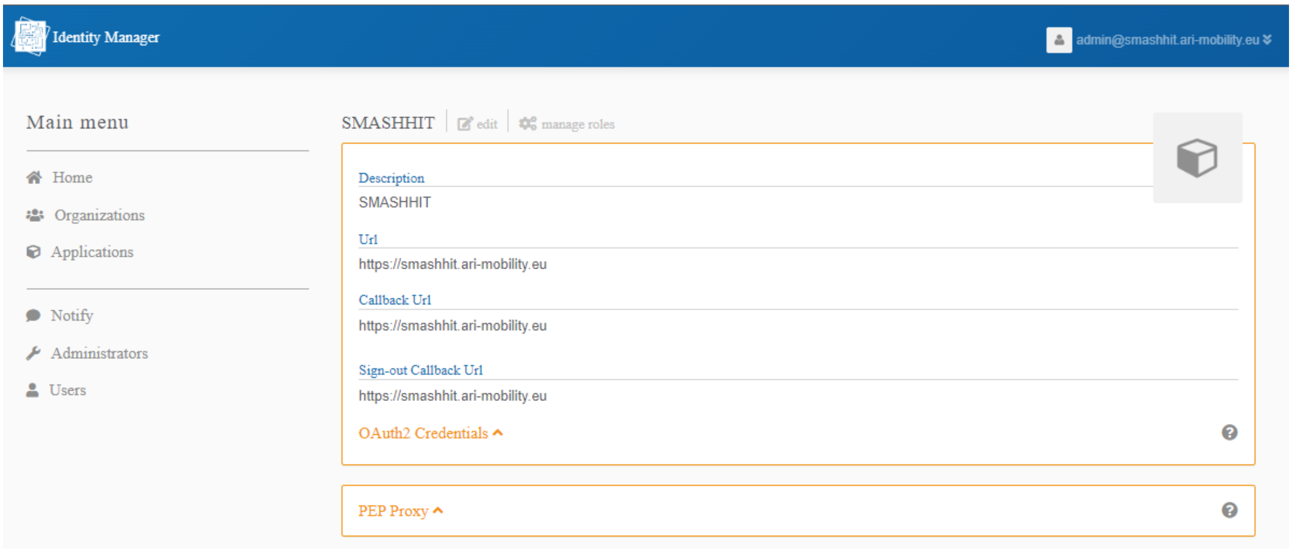
Total: 721 Owners.

Actions	ID	Name	Email	Created
➕				
✎ | 🗑	[redacted]			
✎ | 🗑	[redacted]			
✎ | 🗑	[redacted]			
✎ | 🗑	[redacted]			
✎ | 🗑	[redacted]			

Figure 6 Data Owner management (I)

[1] https://idm-smashhit.ari-mobility.eu

[2] https://fiware-idm.readthedocs.io/en/latest/



SMASHHIT

Description
SMASHHIT

Url
https://smashhit.ari-mobility.eu

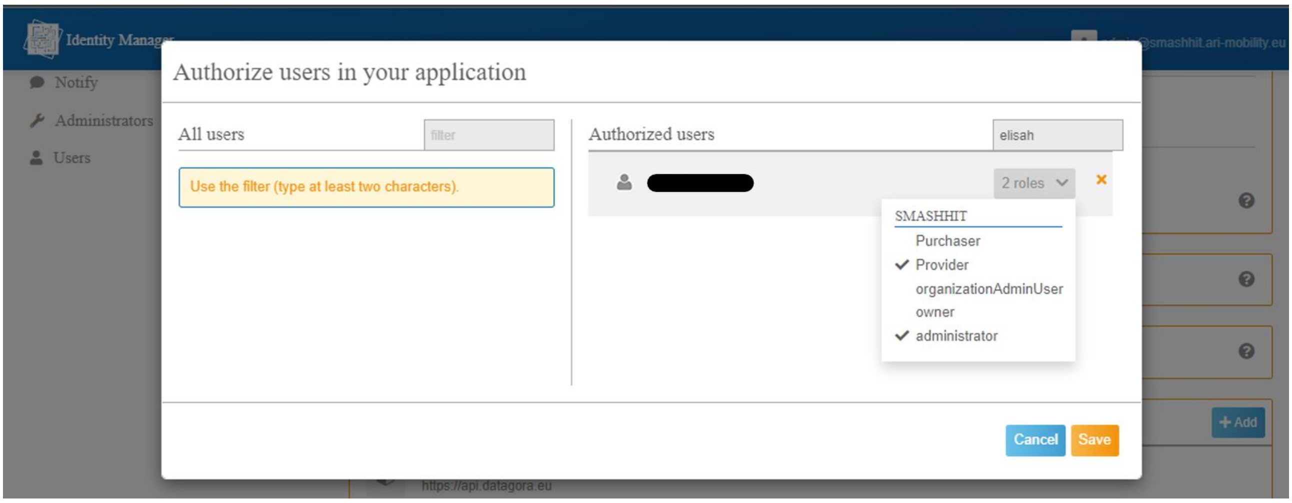
Callback Url
https://smashhit.ari-mobility.eu

Sign-out Callback Url
https://smashhit.ari-mobility.eu

OAuth2 Credentials

PEP Proxy

Figure 7 Data Owner management (II) IdM



Authorize users in your application

All users filter

Use the filter (type at least two characters).

Authorized users elisah

[redacted] 2 roles

SMASHHIT
Purchaser
✓ Provider
organizationAdminUser
owner
✓ administrator

https://api.datagora.eu

Figure 8 Data Owner management (III) IdM

smashHit deployment administration

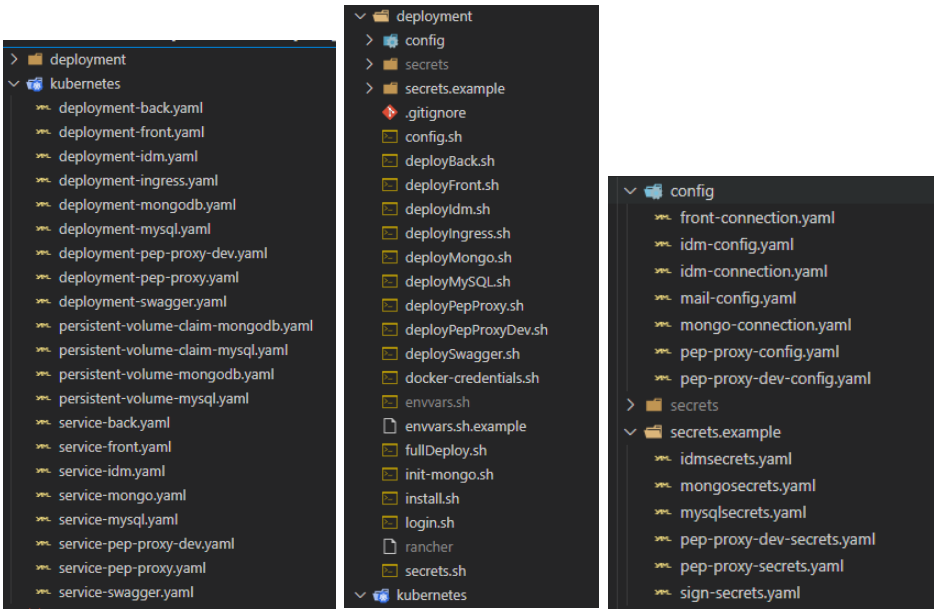
This section aims to present the key aspects of the main smashHit consent manager components. Following modern industry standards, the different components of smashHit are distributed as docker images, and all the parametrizations can be done through environment variables.

While the deployment can be of course adapted to different orchestration frameworks, the consent manager platform is currently deployed in a Rancher[3]-managed Kubernetes cluster, and a set of scripts and ".yaml" files are included with the source code so the deployment can be easily migrated or updated.

The following is a short description of the current deployment of smashHit, including the configuration, workloads & services, load balancing, and of course volumes to persist the data.

Configuration

This section aims to describe the deployment strategy and files utilized for the smashHit platform in a Kubernetes cluster. The deployment files' structure can be seen in the Figure 9:



```
> deployment
v kubernetes
    deployment-back.yaml
    deployment-front.yaml
    deployment-idm.yaml
    deployment-ingress.yaml
    deployment-mongodb.yaml
    deployment-mysql.yaml
    deployment-pep-proxy-dev.yaml
    deployment-pep-proxy.yaml
    deployment-swagger.yaml
    persistent-volume-claim-mongodb.yaml
    persistent-volume-claim-mysql.yaml
    persistent-volume-mongodb.yaml
    persistent-volume-mysql.yaml
    service-back.yaml
    service-front.yaml
    service-idm.yaml
    service-mongo.yaml
    service-mysql.yaml
    service-pep-proxy-dev.yaml
    service-pep-proxy.yaml
    service-swagger.yaml
```

```
v deployment
  > config
  > secrets
  > secrets.example
    .gitignore
    config.sh
    deployBack.sh
    deployFront.sh
    deployIdm.sh
    deployIngress.sh
    deployMongo.sh
    deployMySQL.sh
    deployPepProxy.sh
    deployPepProxyDev.sh
    deploySwagger.sh
    docker-credentials.sh
    envvars.sh
    envvars.sh.example
    fullDeploy.sh
    init-mongo.sh
    install.sh
    login.sh
    rancher
    secrets.sh
v kubernetes
```

```
v config
    front-connection.yaml
    idm-config.yaml
    idm-connection.yaml
    mail-config.yaml
    mongo-connection.yaml
    pep-proxy-config.yaml
    pep-proxy-dev-config.yaml
> secrets
v secrets.example
    idmsecrets.yaml
    mongosecrets.yaml
    mysqlsecrets.yaml
    pep-proxy-dev-secrets.yaml
    pep-proxy-secrets.yaml
    sign-secrets.yaml
```

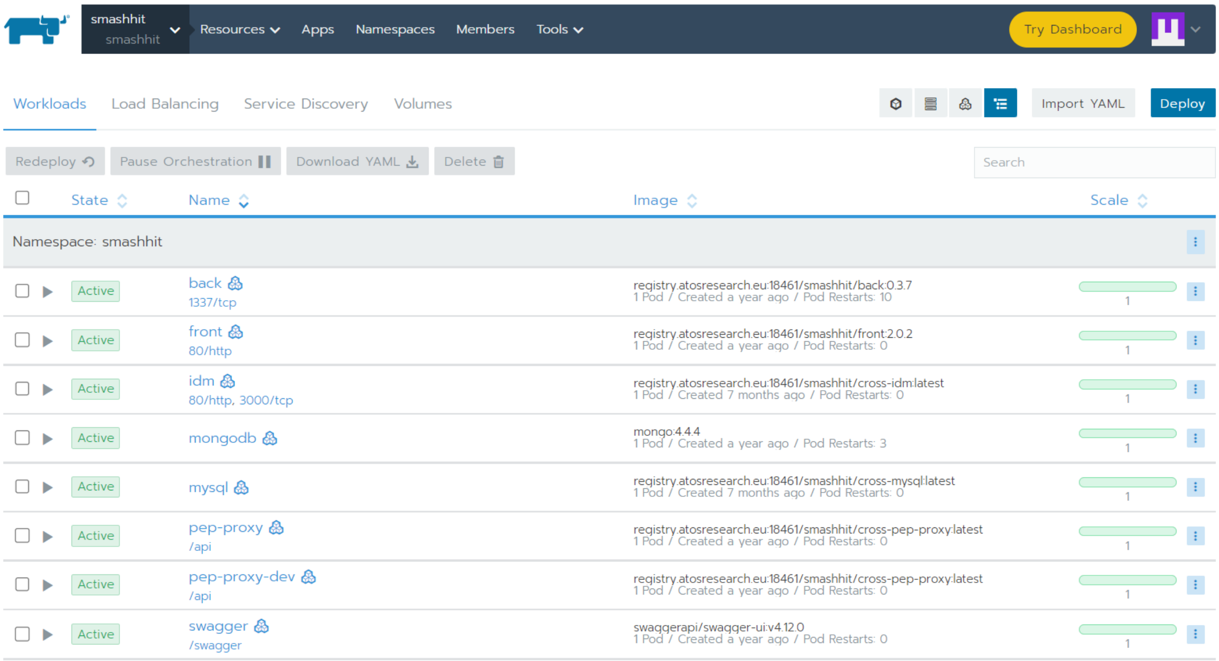
Figure 9: Deployment files structure

- The kubernetes folder holds the deployment, service and volume claim yaml files
- The deployment folder holds the config and secret folders, with the definition of the configMaps (configurable parameters such as URLs) and secrets (passwords, client credentials, certificates…)
- Finally, also in the deployment folder, a few scripts to automatize the deployment are provided:
    - Some general scripts such as envvars.sh, install.sh and login.sh allow you to first setup the necessary parameters, such as the registry credentials, rancher URL or token, then install the rancher-cli, and finally perform a login with the cluster
    - Then, for a full deployment, one can just run the fullDeploy.sh script, which automatically deploys all the components in order, configs, volumes, deployments, ingress…
    - Alternatively, the individual "deploy*.sh" scripts are provided to deploy a single specific component

[3] https://www.rancher.com/

Workloads

A workload is an application running on Kubernetes, inside a set of pods. In Kubernetes, a Pod represents a set of running containers on the cluster. Workloads allow rules to be defined for application scheduling, scaling, and upgrading. The figure below shows the deployed workloads for smashHit, the status and the scale.



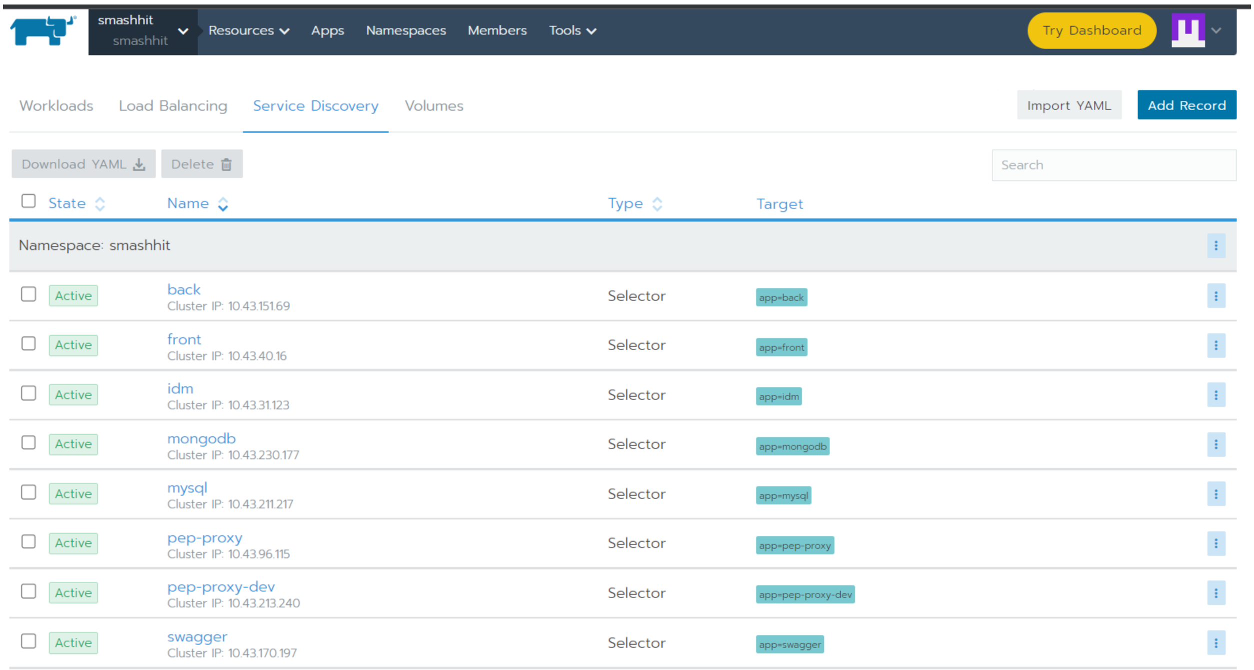
Namespace: smashhit

State	Name	Image	Scale
Active	back 1337/tcp	registry.atosresearch.eu:18461/smashhit/back:0.3.7 1 Pod / Created a year ago / Pod Restarts: 10	1
Active	front 80/http	registry.atosresearch.eu:18461/smashhit/front:2.0.2 1 Pod / Created a year ago / Pod Restarts: 0	1
Active	idm 80/http, 3000/tcp	registry.atosresearch.eu:18461/smashhit/cross-idm:latest 1 Pod / Created 7 months ago / Pod Restarts: 0	1
Active	mongodb	mongo:4.4.4 1 Pod / Created a year ago / Pod Restarts: 3	1
Active	mysql	registry.atosresearch.eu:18461/smashhit/cross-mysql:latest 1 Pod / Created 7 months ago / Pod Restarts: 0	1
Active	pep-proxy /api	registry.atosresearch.eu:18461/smashhit/cross-pep-proxy:latest 1 Pod / Created a year ago / Pod Restarts: 0	1
Active	pep-proxy-dev /api	registry.atosresearch.eu:18461/smashhit/cross-pep-proxy:latest 1 Pod / Created a year ago / Pod Restarts: 0	1
Active	swagger /swagger	swaggerapi/swagger-ui:v4.12.0 1 Pod / Created a year ago / Pod Restarts: 0	1

Figure 10: smashHit Deployed Workloads

Service discovery

Each workload must define a service discovery entry which configures the networking of the pods (usually the ports to use and the network driver). The figure below shows the service configuration for smashHit. The strategy used in this deployment is to only expose components internally, so for example, the databases are not reachable from the public internet but rather by other smashHit components in the cluster only. Instead, only selected workloads are exposed through an ingress controller, as described in the next section.

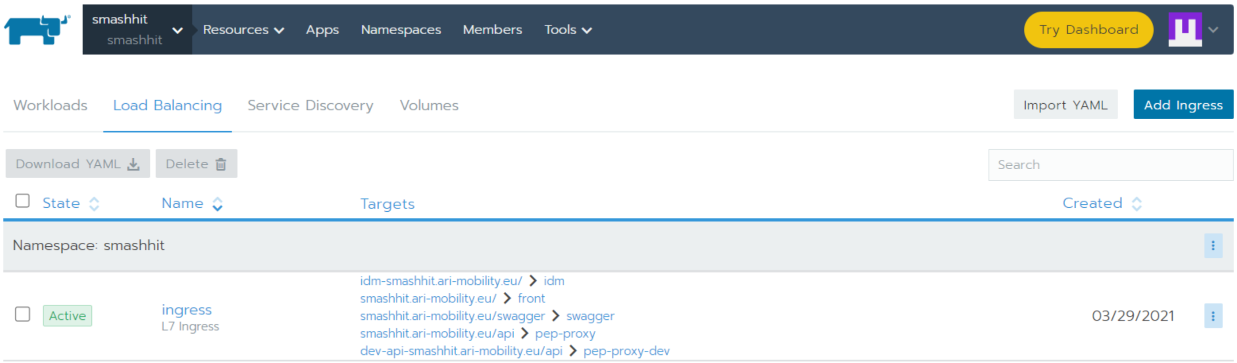


□	State	Name	Type	Target
Namespace: smashhit				
□	Active	back Cluster IP: 10.43.151.69	Selector	app=back
□	Active	front Cluster IP: 10.43.40.16	Selector	app=front
□	Active	idm Cluster IP: 10.43.31.123	Selector	app=idm
□	Active	mongodb Cluster IP: 10.43.230.177	Selector	app=mongodb
□	Active	mysql Cluster IP: 10.43.211.217	Selector	app=mysql
□	Active	pep-proxy Cluster IP: 10.43.96.115	Selector	app=pep-proxy
□	Active	pep-proxy-dev Cluster IP: 10.43.213.240	Selector	app=pep-proxy-dev
□	Active	swagger Cluster IP: 10.43.170.197	Selector	app=swagger

Figure 11: smashHit Service Discovery configuration

Load balancing

Another component included in the architecture of the smashHit framework is an Ingress[4] load balancer to expose HTTPS routes from outside the cluster to services within the cluster, allowing traffic routing, traffic load balancing, SSL / TLS termination, and offer name-based virtual hosting. Figure below shows the Ingress configuration exposing the IdM service, the smashHit web application, the smashHit API (through a pep-proxy) and the swagger server of the smashHit API specification.



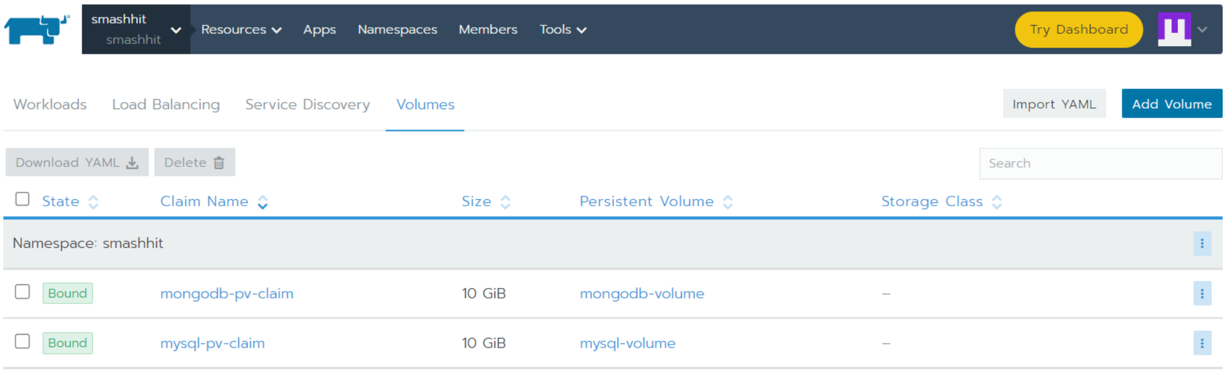
□	State	Name	Targets	Created
Namespace: smashhit				
□	Active	ingress L7 Ingress	idm-smashhit.ari-mobility.eu/ > idm smashhit.ari-mobility.eu/ > front smashhit.ari-mobility.eu/swagger > swagger smashhit.ari-mobility.eu/api > pep-proxy dev-api-smashhit.ari-mobility.eu/api > pep-proxy-dev	03/29/2021

Figure 12: smashHit Load Balancer Configuration

Database Volumes

In order to maintain data between workload restart, rescheduling or even upgrades, all databases are backed by persistent volumes, as it can be seen in the figure below.

[4] https://www.nginx.com/resources/glossary/kubernetes-ingress-controller/



Namespace: smashhit

	State	Claim Name	Size	Persistent Volume	Storage Class
□	Bound	mongodb-pv-claim	10 GiB	mongodb-volume	–
□	Bound	mysql-pv-claim	10 GiB	mysql-volume	–

Figure 13: smashHit Volumes

Context-Sensitive Security and Privacy administration

This guide to the smashHit context-sensitive security & privacy (S&P) policies and mechanisms is for platform administrators. The S&P mechanisms consist of three main components: Policy Tool, Policy Server and Event Processing Point (EPP), the latter running as part of the Policy Server. These components provide support for the creation, testing, deployment in the smashHit platform, and support for the enforcement of S&P policies in the smashHit ecosystem. The primary use of the S&P mechanisms is by other components of the smashHit platform, such as the Automatic Contracting Tool. Developers of smashHit components determine how the S&P policies and mechanisms are to be used, and when and how S&P is to be run within the smashHit platform. As an internally used component of the smashHit platform the S&P mechanisms do not have their own GUI though they can and do provide support to applications that do have a GUI.

Initialization of local definitions

Beyond the organizing principles and syntax of the Declarative Policy Language for Privacy (DPLP) that is provided by the S&P mechanisms, a considerable portion of an S&P policy is dependent on domain-specific concepts, vocabulary and relationships from the operational environment. The smashHit Core ontology identifies and defines such information for the smashHit context for the business cases currently addressed.

The S&P system does not have the definition of ontology concept instances built-in but allows these to be provided as a “definitions policy” which is a parameter of a DPLP policy. It is the joint responsibility of the developers and the platform administrator to assure that the appropriate definitions policy is loaded into the Policy Server before policies needing the definitions are constructed.

The developers should determine the appropriate set of terms to be included in the definitions policy and the platform administrator should assure that the definitions policy is loaded. The loading could be part of the initialization script for the S&P mechanisms. First it is necessary to assure that the appropriate definitions policy is available and has been rendered from the smashHit Core ontology in the necessary format. If necessary, the definitions policy could be hand coded but ideally it would be generated automatically from the authoritative ontology. In smashHit responsibility has been assigned to the ontology/context system to generate these definitions in the policy syntax required by the S&P system.

For smashHit S&P policies the definitions policy identifies the elements of the purpose hierarchy, data processing operation hierarchy, and personal data category hierarchy. Each hierarchy may have a “flat” structure where each of the members is assigned only to the category name, e.g. “data processing operation”. Instances of the category may then be compared with the equality relation.

Alternatively, some elements may be assigned to the category name, and the remaining elements assigned in turn to these in a hierarchical fashion.

Instances of the category may then be compared with a “dominates” relation that evaluates to true if there is a sequence of one or more assigns that leads from the second argument of the relation to the first argument. Otherwise, the relation evaluates to false.

The resulting definition policy that defines the members and structure of purpose, data processing operation, and personal data category sets can be loaded in one of several ways into the Policy Server to be available for reference when ordinary DPLP policies are defined:

- built in to the policies.pl file that resides in the server’s source code,
- placed in a distinct file that is dynamically loaded during the system startup sequence by invocation of the paapi/load endpoint,

- loaded dynamically as an argument to the paapi/loadi endpoint,
- built up dynamically and incrementally by a sequence of invocations of the paapi/add endpoint.

The chosen method will depend on how policies are generally managed in the operational environment.

Operational policy management, backup and recovery

Policy management is an issue that must be considered relative to how the S&P mechanisms are used within a system. S&P does not itself provide policy persistence.

If, on the one hand, the security policies that are to be used for policy decisions within S&P are relatively static and determined by processes outside of S&P then the policies can be loaded when S&P is initialized. If the system is shut down and restarted the policies will need to be reloaded as part of system initialization or as part of a restart of the S&P mechanisms. If, on the other hand, the security policies used within S&P are dynamically created and are actively changing during a run of the S&P system, such that S&P holds the primary, or sole, copy of the policy then a strategy is needed to carry over the policy state from one execution session to another.

One way of doing this would be to save the policy from the running server, store it in a file, and use the file to restore the policy after the S&P server is started. Another way of doing this is to have the system components that use the S&P server to keep track of the state of the information needed to build the policy in their own persistent store, and to rebuild the policy to that state after the S&P server is started.

In smashHit, the information that comprises policy elements originates with other smashHit components, nor does S&P provide the primary storage of that information. Policies are built and referenced as needed by other smashHit components to provide low-level policy comparisons and compliance decisions. When the smashHit platform is started, and with it the S&P Policy Server, the administrative startup procedures must be designed to work together with the smashHit components to establish the desired state of the S&P policy from information in the ACT’s graph database, the Context system, and the smashHit ontology tools.

Authorization to use S&P interfaces

The S&P Policy Server offers unprotected interfaces (such as the policy query APIs) for operations that have no side-effects on the state of the Policy Server (though they may leave entries in the security audit trail). Protected interfaces (such as the policy administration APIs) are provided for operations that can change the configuration state of the Server including its policy store or options that control its operation (such as the policy administration APIs).

The protected interfaces require that a token be presented with each protected API call. The token is a secret chosen by the administrator for one or more execution sessions of the Policy Server and shared with the components using the Server that are authorized to call the protected APIs. The administrator assigns the chosen token to the Policy Server as a command line option when the Server is started and this token will be in force for the duration of this execution session. It is then provided to the applications needing it for the current execution. It is recommended that the using applications can accept the value of this token as a startup argument. The token may be changed on a schedule chosen by the administrator and given to the initialization procedures to give to the Server and applications.

Starting the Policy Server

The S&P Policy Server is started as part of the smashHit platform startup procedure. The S&P Event Processing Point runs as part of the Policy Server. The policy server accepts certain command line options when it is invoked. These options are described in D4.4 Section 5.5.2.1.1. The default values for certain options has been changed for simplicity, viz., --epp and –jsonresp are True by default. As noted previously, the --token option is likely to be used. Other options that are likely to be used are those specifying the URL of the Context system (--context) and the port numbers for the policy query interface (--pqport), the policy administration interface (--paport), or both query and admin the same port (--port). The --load or --policy option is one way of loading a single policy, such as a definitions policy, at startup.

F.A.Q.

smashHit platform

Q: What is the smashHit platform?

A: smashHit is a platform that provides a generic, transparent way to view and manage consent supported by disrupting technologies such as traceability of use of data, data fingerprinting and automatic contracting among the data owner, data provider, and service providers.

Q: Can administrators manually register new users?

A: On the one hand, organization users have their own dedicated registration request flow as described in the “Organization registration” section.

Data owner registration, on the other hand, is not open to the public, but is instead meant to be integrated through the smashHit API by the organizations’ applications themselves (see the data provider/processor developer guide for more details).

That being said, while it is not directly supported or intended through the GUI, administrator users are authorized to utilize the necessary smashHit API endpoints to manually manage data owner accounts, in case it is necessary to provide advanced support.

Q: How can an administrator supervise and manage consents?

A: Consents are only directly visible to the data subjects and the organization(s) directly involved in the processing of data (please refer to the appropriate data owner/developer guidelines for details).

That being said, in the event that support actions require the administrator to review a specific item, the administrator user could use its privileged role to make specific queries to the smashHit APIs to review consent certificates involving a given data subject, organizations, dates, etc.

Glossary

Agent: Entity that bears some form of responsibility in the context of a consent

Administrator: smashHit platform system administrator

Consent: As per Article 4(11) of the GDPR, ‘consent’ of the data subject means any freely given, specific, informed and unambiguous indication of the data subject’s wishes by which he or she, by a statement or by a clear affirmative action, signifies agreement to the processing of personal data relating to him or her

Consent template: A template defining the details of a data processing by an Agent (smashHit organization user), such as the purpose of the processing or the personal data to be processed

CSS: Context sensitivity solution

Data owner: smashHit user role which acts as the data subject according to GDPR

DPL: Declarative Policy Language

DUT: Data Use Traceability

ERL: Event response language

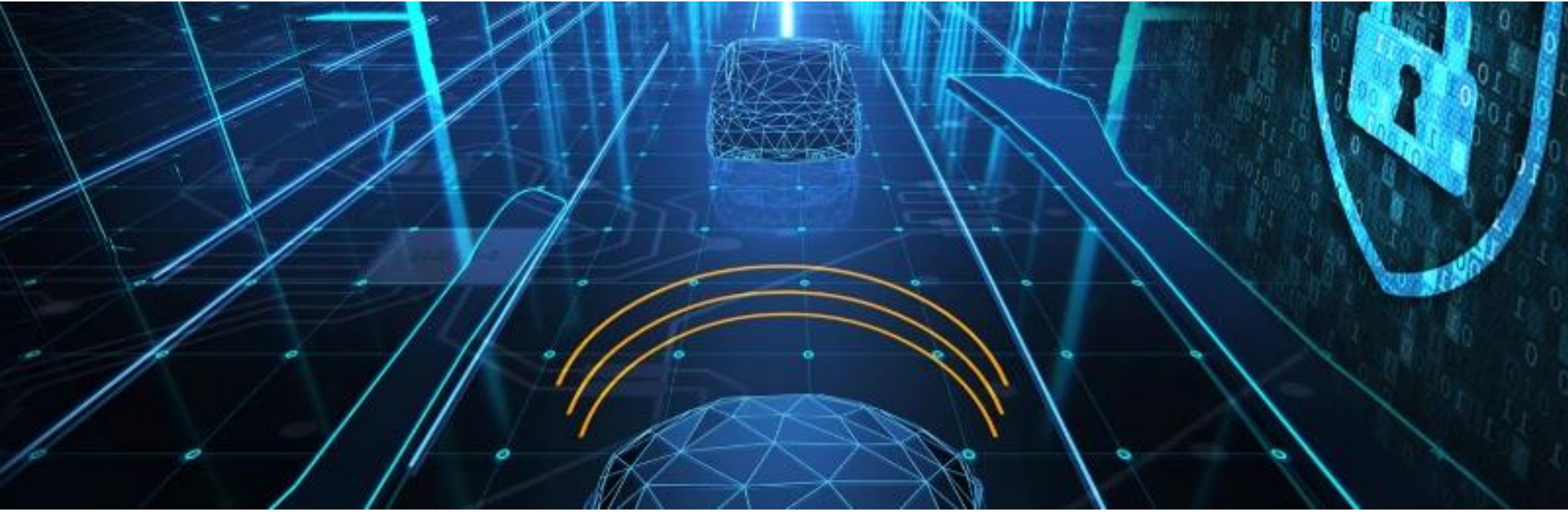
GDPR: Abbreviation for ‘General Data Protection Regulation’, a legal norm on EU level adopted in 2016, which is directly applicable within its scope and lays down rules for the processing of personal data so as to protect natural persons’ fundamental rights and freedoms, in particular their right to the protection of personal data

Personal data: Any information which are related to an identified or identifiable natural person (Art. 4 (1) GDPR), e.g. a name or a home address

Purpose: The purpose of Data Handling/Processing

S&P: Security and privacy module

Figures



Our vision - Solving Consumer Consent & Data Security for Connected Car and Smart City

Further information

This document is part of the smashHit Methodology. The complete set of documents, including other user/developer guides as well as white papers created within this scope can be found on our website:

https://smashhit.eu/publications

Our consortium

Funded by the Horizon 2020 Framework Programme of the European Union

Every effort has been made to ensure that all statements and information contained herein are accurate, however the smashHit Project Partners accept no liability for any error or omission in the same.