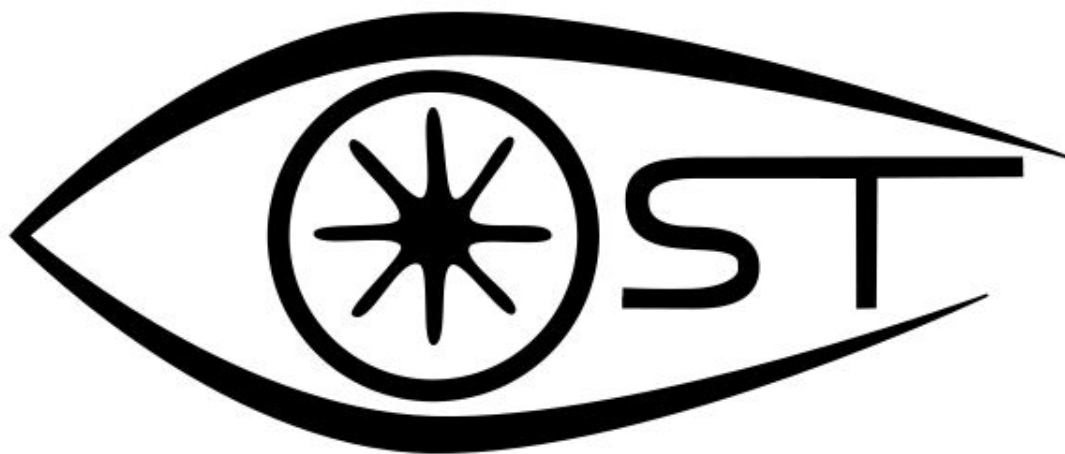


OST documentation



Document Title:	ITTI Orbits- User Manual		
Document Reference:			
Document Version:	0.1	Date:	2020-06-30
Abstract			
Deliverable: User manual – draft version, backend rest api, database, component diagram			

Action	Name	Function	Signature	Date
Prepared by:	Michał Śmieszek	Junior Analyst		

Name	Contact	Description	Date
© COPYRIGHT ITTI, 2019 The copyright of this document is vested in ITTI. This document may only be reproduced in whole or in part, stored in a retrieval system, transmitted in any form, or by any means electronic, mechanical, photocopying, or otherwise, with the prior permission of the ITTI.			

Issue	Date	Description
0.1	2020-06-30	Creation of the document

Table of Content

Table of Content	3
Component diagram	4
User Manual	4
Admin	6
Creating trial with questions	6
Trial list	6
Trial details	7
Stage details	8
Question set details	8
Question details	9
Role details	10
Managing users and sessions	10
Session details	10
Session tracking details	11
User list	11
User details	12
Observer	13
Question Set view	13
Question view	13
Event view	14
Deployment Manual	14
API description	20
Database	32
Environmental variable	33

1. Component diagram

Backend, frontend and database can be launched by one docker compose. Backend can connect with Testbed (Kafka) and with Keycloak (system to authorization).

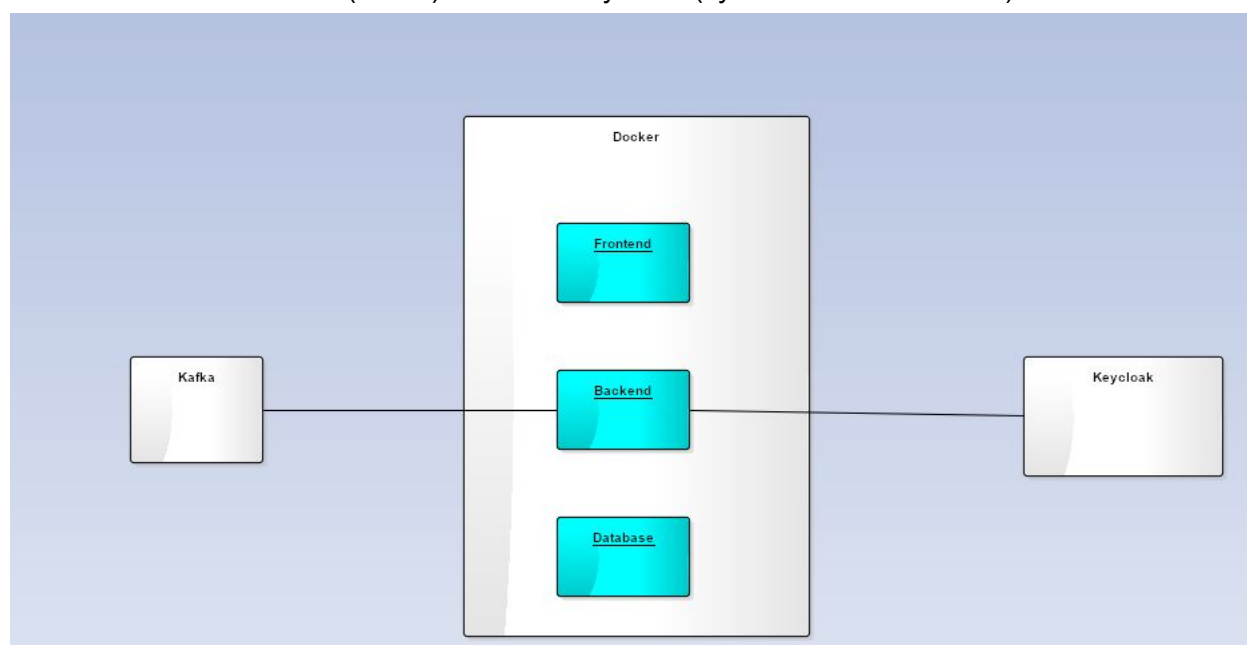


Figure 1. Component diagram of OST.

2. User Manual

User can login to server as admin or observer. Admin can manage trials, session, stages, roles, add question to roles and stages, add and edit users and control active session. Observer can answer on question and comment it.

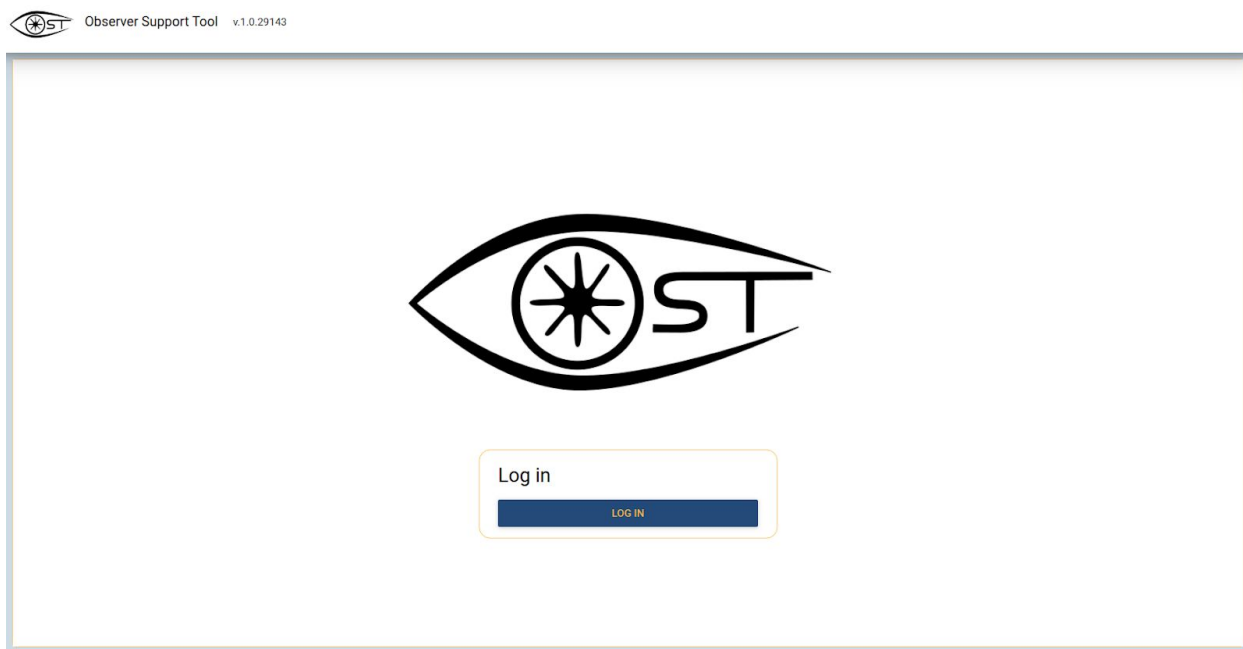


Figure 2. Starting page

After clicking “LOG IN” user is redirected to keycloak. If user log in to admin account, then he will be redirected to main view of admin (look at Figure 4), else user will be redirected to observer view.

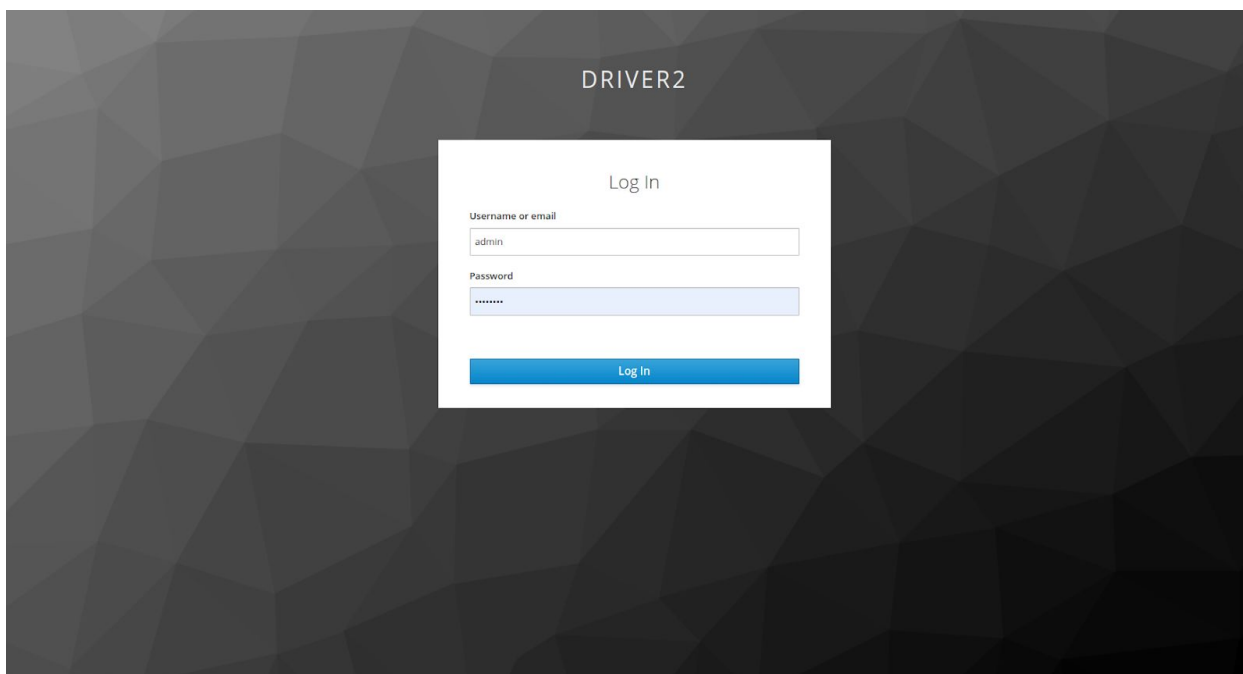


Figure 3. Keycloak log in page

Admin

Goal of the admin is creating a working session for observers. This process contains three stages: creating trial with questions, creating roles and assigning them to question sets and preparing sessions with users. To create questions admin should create trial, stages in trial, question sets in stages and question in question sets. Description can be found in the subsection named „Creating trial with questions”. Second part is creating roles and assigning them to a question set (which already are assigned to stage). It means that questions, which are available for the observer, depend on the stage and on the role of the logged observer. Last step is creating session with observers. Observers in other sessions can have other roles, so assignment user to role in session. Admin needn't create new users, if observers have their account in system. Description of this step can be found in subsection „Managing users and sessions”.

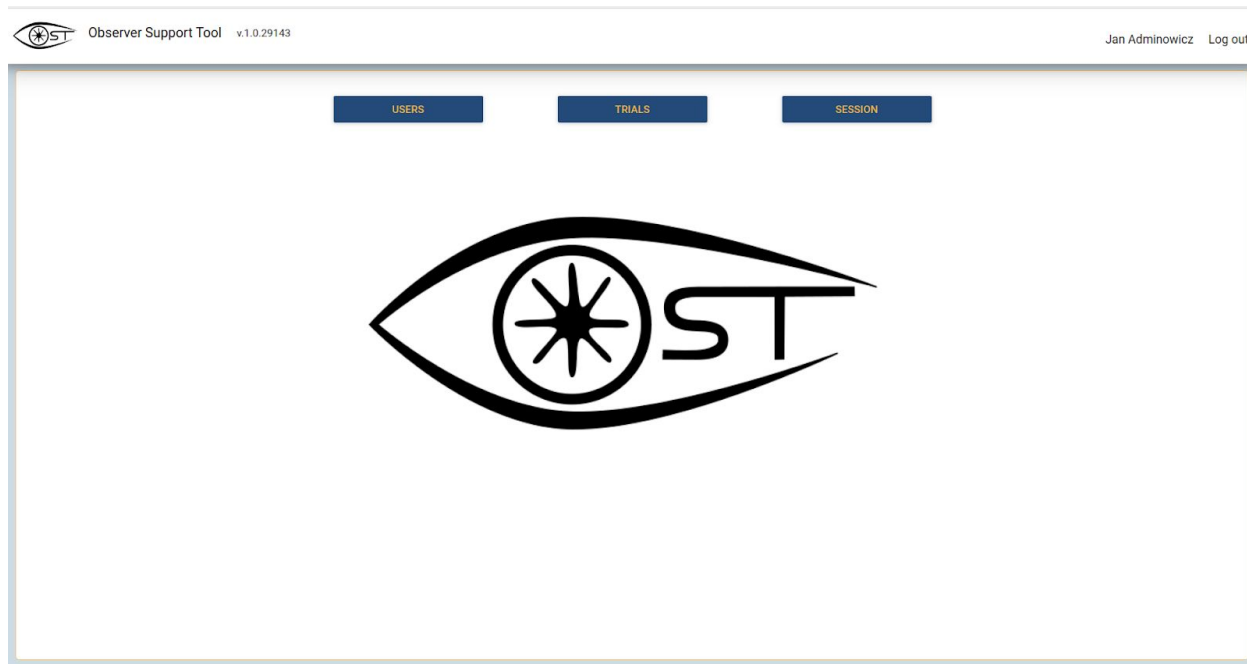


Figure 4. Page after log in as admin

Creating trial with questions

Trial list

If we choose “TRIALS” from the launch window we will be redirected to the Trial List. There is a list of trials, which can be managed also we can upload after choosing the “Import trial” button.

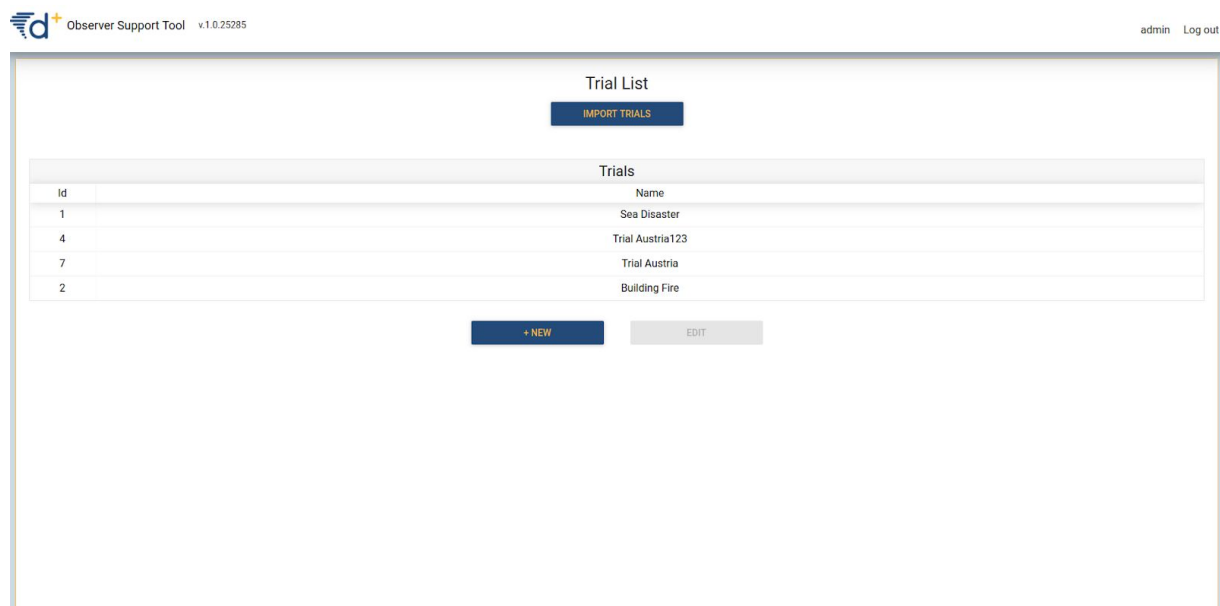


Figure 5. Trial list view

Trial details

After choosing trial from trial list and click edit or after double clicking the left mouse button on trial from trial list, system redirects to trial details. There are basic details about trial and data about sessions, stages and roles in trial. Also trial can be edited and removed there.

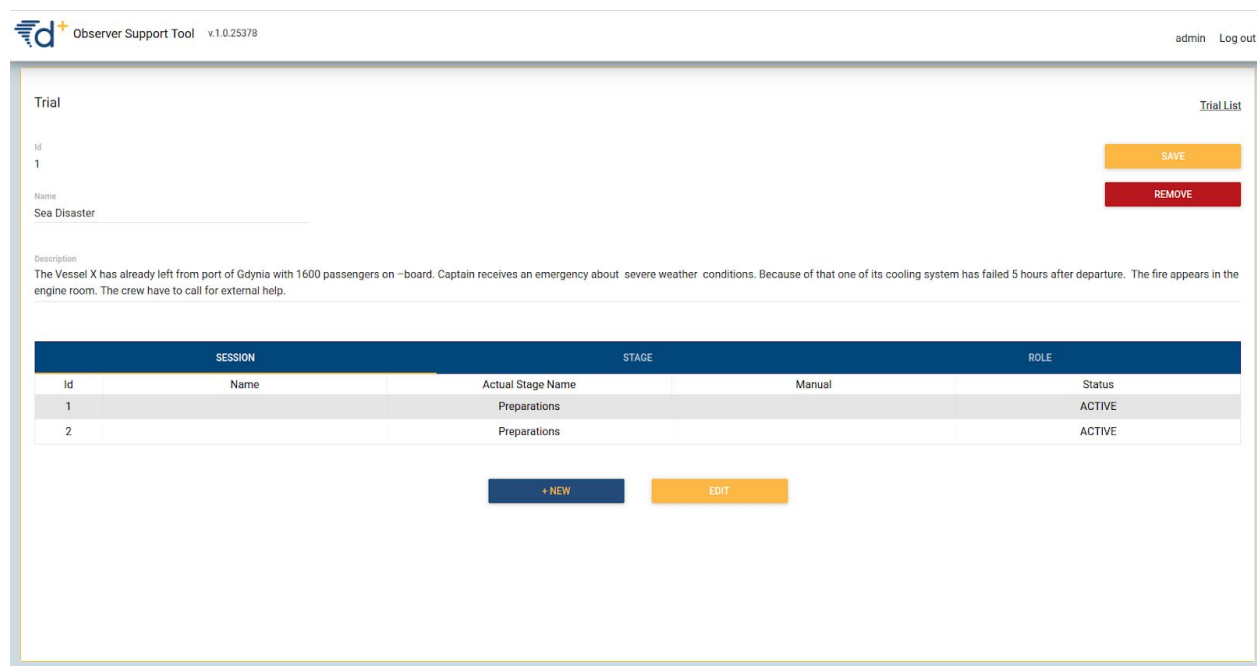


Figure 6. Trial detail view

This project has received funding from the European Union's 7th Framework Programme for Research, Technological Development and Demonstration under Grant Agreement (GA) N° #607798

Stage details

After choosing stage from trial details and click edit or after double clicking left mouse button on stage from trial details, system redirects to stage details. There are basic details about stage, which can be edited. Also stage can be deleted by admin. Question set list of chosen stage is below basic stage details. You can redirect to trial by clicking trial name above save button.

The screenshot displays the 'Stage detail view' interface. At the top left, the title 'Stage' is shown. Below it, the 'Id' is '1' and the 'Name' is 'Preparations'. To the right, there is a 'See Disaster' link and two buttons: 'SAVE' (orange) and 'REMOVE' (red). Below the form is a table titled 'Question Set' with columns 'Id', 'Name', and 'Position'. The table contains two rows: one with 'Id' 1 and 'Name' 'Situational awareness', and another with 'Id' 2 and 'Name' 'Communication'. At the bottom of the interface, there are two buttons: '+ NEW' (blue) and 'EDIT' (orange).

Question Set		
Id	Name	Position
1	Situational awareness	
2	Communication	

Figure 7. Stage detail view

Question set details

In the question set details windows admin can change name or description also list of questions of this question set is shown. Users can redirect to the chosen question by double clicking on question or clicking question and edit. You can redirect to trial or stage by clicking the trial name or stage name above save button.

Observer Support Tool v.1.0.25681 admin Log out

Question Set [Building Fire](#) [yyy.stage.building Fire](#)

Id
3

Name
Quality of Connection

Description
Questions collect data about the connection. Remind every problem you noticed.

Position
1

[SAVE](#)
[REMOVE](#)

Question			
Id	Name	Position	Type
14	What problems they had while connecting with db?	1	TEXT_FIELD
16	How long took connection with database?	1	SLIDER
15	How do you feel about quality of connection?	1	SLIDER

[+ NEW](#)
[EDIT](#)

Figure 8. Question set detail view

Question details

In question details, the admin can modify the question, description or type of answer. Also admin can add new options of choice, show them or remove. In question details parameters like commented and required can be set. You can redirect to trial, stage or question set by clicking trial name, stage name or question set name above the save button. Question options can be added after saving the question.

Observer Support Tool v.1.0.25681 admin Log out

Question [Trial Austria](#) [Block1_Emergent_Groups_Telegram](#) [Block1_Emergent_Groups_Telegram](#)

Id
17

Name
How many years of professional experience do you have

Description

Answer type
radio button

Question position
1

☐ Commented
☐ Required

[SAVE](#)
[REMOVE](#)

Option		
Id	Value	Position
27	0-2	0
34	3-5	1
35	6-10	2
36	11-15	3
37	15+	4

[+ NEW](#)
[REMOVE](#)

Figure 9. Question detail view

This project has received funding from the European Union's 7th Framework Programme for Research, Technological Development and Demonstration under Grant Agreement (GA) N° #607798

Creating roles and assign it to question set

Role details

After choosing the role from trial details and clicking edit or after double clicking the left mouse button on role from trial details, system redirects to role details. There are basic details about role, which can be edited. Also in role details we can manage user assignment to role and question set, which should be shown, when the right stage is chosen. You can redirect to trial by clicking trial name above save button.

Observer Support Tool v.1.0.29143 Jan Adminowicz Log out

Role [MichalTrial](#)

Id: 120 SAVE

Name: Observer REMOVE

Question Set		
Id	Name	Assigned
1693	QS2	<input checked="" type="checkbox"/>
1718	QS1	<input type="checkbox"/>

User Assignment		
Id	User name	Role name
No rows found		

Figure 10. Role detail view

Managing users and sessions

Session details

New session can be created by clicking +New, when session window in trial detail is active. After choose session from trial details and click edit or after double clicking left mouse button on session from trial details, system redirects to session details. There are basic details about session, which can be edited or deleted, also data (answer of observers, date of trial, date of server etc) from session can be downloaded. In the user role table admin can assign role to user in this session. User can answer on question only if session is active.

Observer Support Tool v.1.0.25378 admin Log out

Session

Id: 2

Name: Sea Disaster

Status: Active

Current Stage: Preparations

Sea Disaster

SAVE

REMOVE

DOWNLOAD OBSERVATIONS

Id	User name	User Role	Role name
No rows found			

+ NEW REMOVE

Figure 11. Session details view

Session tracking details

After clicking the session from the launch window, the system redirects to session tracking detail. In session tracking details admin can control active sessions. There are information, which trial is currently used, if the session is controlled by testbed or manually (manual column) and which stage is active at this moment.


Observer Support Tool v.1.0.25794 admin Log out

Active Sessions				
Id	Name	Actual Stage Name	Manual	Status
18	Demo	Stage1	no	ACTIVE

Figure 12. Session tracking details view

User list

If we choose "USERS" in the launch window we will be redirected to the User list. There is some basic data about users there.


Observer Support Tool v1.0.29143
Jan Adminowicz Log out

User List

Users				
Id	Login	FirstName	LastName	Deleted
91f11262...	admin	Jan	Adminowicz	No
b11b67a...	admin2	Admin2	Adminowicz2	No
fd775ad0...	finaltest	Final	Test	No
cc37e33...	karo	John	Red	No
1ad95d6...	msmi	jhkgk	gkkgkg	No
fc108bbc...	nieruszaj	Piotr	Nieruszaj	No
9a0fd937...	observer	Jan	Obserwerowicz	No
c62bd10...	pawel	pawel	pawel	No
eb915a7...	robert	Kate	Yellow	No
35d3618...	robert_obs	Emmy	Blue	No

Previous
Page 1 of 1
10 rows
Next

+ NEW
EDIT

Figure 13. User list view

User details

After choosing a user from user list and clicking edit or after double clicking left mouse button on the user from the user list, system redirects to user details. There are basic details about user and data about sessions, stages and roles in trial.

Also user can be edited and removed there.

If we want to add a new user, we should click +New on User list. Then the system redirects to user detail and after filling data, the system creates a new user. All fields are required.

The screenshot shows the 'Observer Support Tool' interface. At the top, it says 'v.1.0.29143' and 'Jan Adminowicz Log out'. The main section is titled 'User List'. A modal window titled 'User details' is open, showing the following information:

- ID: 91f12620-70cb-4c42-b105-b556608aeaf3 ☒ Admin ☐ Deleted [SAVE](#)
- Password:
- Confirm Password:
- First Name: Jan
- Last Name: Adminowicz
- email: admin@itti.com.pl
- Contact(Phone): 123123123
- [CANCEL](#)

In the background, a table lists users with columns 'Id' and 'Login'.

Id	Login
91f1262...	admin
b11b67a...	admin2
fd775ad0...	finaltest
cc37e33...	karo
1ad95d6...	msml
fc108bbc...	nieruszaj
9a0fd937...	observer
c62bd10...	pawel
eb915a7...	robert
35d3618...	robert_obs

Buttons at the bottom include '+ NEW' and 'EDIT'.

Figure 14. User details view

Observer

Observer can login to the server, when admin sends him his login and password. After it he can login to the system.

Question Set view

Observers after login to the system can choose question set from the question set list from active sessions (if question sets, during present stage in active session, are available to logged user).

The screenshot shows the 'Observer Support Tool' interface with version 'v.1.0.25536' and user 'test10 Log out'. The main section is titled 'New entry'. There is a button '< BACK TO LIST OF EVENTS'. Below, there are two yellow boxes labeled 'Question Set 1' and 'Question Set 2'.

Figure 15. Question set view

Question view

Observer after choosing the question set is redirected to questions from the chosen question set. Observers can answer questions from this question set by writing text (textfield) or choosing option (checkbox, radiobutton, slider).

This project has received funding from the European Union's 7th Framework Programme for Research, Technological Development and Demonstration under Grant Agreement (GA) N° #607798

Observer Support Tool v.1.0.25536 test10 Log out

< BACK TO LIST OF OBSERVATIONS

Real Time: 06/02/2020 10:01:28
Trial Time: 06/02/2020 09:01:28

Question Set 1 INFO

What:

How much time did the participant need to respond to the new flooding inject?

Did the situation report get updated?

☐ yes
☐ no

BACK SUBMIT

Figure 16. Question view

Event view

Event view is a page, where an observer can look at a set of his/her comment. After choosing question set observer can comment his answer, view his answer or delete his answer.

Trial: Wlnter trial

List of events 22/06/2020 16:32:23

12 26/05/2020 14:47:00	▼
2 20/05/2020 14:14:00	▲

COMMENT VIEW REMOVE

GO TO QUESTIONS

Figure 17. Event view

Deployment Manual

1. Download docker from <https://www.docker.com/>.
2. Create file and name it *docker-compose.yml*

This project has received funding from the European Union's 7th Framework Programme for Research, Technological Development and Demonstration under Grant Agreement (GA) N° #607798

3. Open this file and paste to this file text below:

```

version: '3'
services:
  zookeeper:
    image: confluentinc/cp-zookeeper:latest
    hostname: zookeeper
    ports:
      - "3500:3500"
    environment:
      ZOOKEEPER_CLIENT_PORT: 3500
      ZOOKEEPER_TICK_TIME: 2000
  broker:
    image: confluentinc/cp-kafka:latest
    hostname: broker
    depends_on:
      - zookeeper
    ports:
      - "3501:3501"
    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_ZOOKEEPER_CONNECT: 'zookeeper:3500'
      KAFKA_ADVERTISED_LISTENERS: 'EXTERNAL://localhost:3501,PLAINTEXT://broker:9092'
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: 'EXTERNAL:PLAINTEXT,PLAINTEXT:PLAINTEXT'
      KAFKA_LISTENERS: 'EXTERNAL://0.0.0.0:3501,PLAINTEXT://0.0.0.0:9092'
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1
      KAFKA_DEFAULT_REPLICATION_FACTOR: 1
      KAFKA_MESSAGE_MAX_BYTES: 100000000
      KAFKA_REPLICA_FETCH_MAX_BYTES: 100000000
  schema_registry:
    image: confluentinc/cp-schema-registry:latest
    hostname: schema_registry
    depends_on:
      - zookeeper
      - broker
    ports:
      - "3502:3502"
    environment:
      SCHEMA_REGISTRY_HOST_NAME: schema_registry
      SCHEMA_REGISTRY_LISTENERS: 'http://0.0.0.0:3502'
      SCHEMA_REGISTRY_KAFKASTORE_CONNECTION_URL: 'zookeeper:3500'
      SCHEMA_REGISTRY_KAFKASTORE_BOOTSTRAP_SERVERS: 'PLAINTEXT://broker:9092'

  kafka_rest:
    image: confluentinc/cp-kafka-rest:latest
    hostname: kafka_rest
    depends_on:
      - zookeeper
      - schema_registry
      - broker

```

```

ports:
  - "8082:8082"
environment:
  KAFKA_REST_HOST_NAME: kafka_rest
  KAFKA_REST_BOOTSTRAP_SERVERS: 'PLAINTEXT://broker:9092'
  KAFKA_REST_ZOOKEEPER_CONNECT: 'zookeeper:3500'
  KAFKA_REST_LISTENERS: 'http://0.0.0.0:8082'
  KAFKA_REST_SCHEMA_REGISTRY_URL: 'http://schema_registry:3502'
  KAFKA_CONSUMER_REQUEST_TIMEOUT_MS: 30000
  KAFKA_REST_ACCESS_CONTROL_ALLOW_METHODS:
'GET,POST,PUT,DELETE,OPTIONS'
  KAFKA_REST_ACCESS_CONTROL_ALLOW_ORIGIN: '*'
kafka_topics_ui:
  image: landoop/kafka-topics-ui:latest
  hostname: kafka_topics_ui
  depends_on:
    - kafka_rest
  ports:
    - "3600:8000"
  environment:
    KAFKA_REST_PROXY_URL: 'http://kafka_rest:8082'
    PROXY: 'true'

kafka_schema_registry_ui:
  image: landoop/schema-registry-ui:latest
  hostname: kafka_schema_registry_ui
  depends_on:
    - schema_registry
  ports:
    - "3601:8000"
  environment:
    SCHEMAREGISTRY_URL: 'http://schema_registry:3502'
    PROXY: 'true'

postgres:
  image: postgres:9.6
  hostname: postgres
  environment:
    POSTGRES_USER: postgres
    POSTGRES_PASSWORD: postgres
    POSTGRES_DB: TRIAL_ADMIN
  volumes:
    - postgres-data:/var/lib/postgresql/data
# restart: unless-stopped

admintool:
  image: drivereu/test-bed-admin:latest
  depends_on:
    - postgres
    - broker

```



```

- schema_registry
ports:
- "8090:8090"
volumes:
- ./admintool-config/gateways.json:/opt/application/config/gateways.json
- ./admintool-config/solutions.json:/opt/application/config/solutions.json
- ./admintool-config/topics.json:/opt/application/config/topics.json
- ./admintool-config/standards.json:/opt/application/config/standards.json
- ./admintool-config/testbed-solutions.json:/opt/application/config/testbed-solutions.json
- ./admintool-config/testbed-topics.json:/opt/application/config/testbed-topics.json
- ./admintool-config/configurations.json:/opt/application/config/configurations.json
environment:
KAFKA_BROKER_URL: broker:9092
SCHEMA_REGISTRY_URL: http://schema_registry:3502
zookeeper_host: zookeeper
zookeeper_port: 3500
schema_registry_url: http://schema_registry:3502
testbed_secure_mode: 'DEVELOP'
testbed_init_auto: 'false'
management_ca_cert_path: http://localhost:9090
cert_handler_url: https://localhost:8443
cert_pem_handler_url: https://localhost:8443
security_rest_path_group: https://localhost:9443
security_rest_path_topic: https://localhost:9443

afteractionreview:
image: drivereu/after-action-review:latest
depends_on:
- postgres
- broker
- schema_registry
ports:
- "8095:8095"
environment:
KAFKA_BROKER_URL: broker:9092
SCHEMA_REGISTRY_URL: http://schema_registry:3502
zookeeper_host: zookeeper
zookeeper_port: 3500
schema_registry_url: http://schema_registry:3502

pgadmin:
image: fenglc/pgadmin4
depends_on:
- postgres
ports:
- "5050:5050"
# restart: unless-stopped

time_service:
image: drivereu/test-bed-time-service:latest

```

```

depends_on:
- broker
- schema_registry
ports:
- "8100:8100"
environment:
KAFKA_BROKER_URL: broker:9092
SCHEMA_REGISTRY_URL: http://schema_registry:3502
AUTO_REGISTER_SCHEMAS: 'true'

large_file_service:
image: drivereu/large-file-service:latest
ports:
- "9090:9090"
environment:
HOST: localhost
PORT: 9090

# wms_service:
# image: drivereu/test-bed-wms-service:1.0.10
# hostname: wmservice
# environment:
# URL: http://localhost/wms/
# WMS_PORT: 5101
# WMS_HOST: wmservice
# WMS_FOLDER: /server/data
# WMS_EXTERNAL_PORT: 5101
# WMS_EXTERNAL_HOST: http://localhost/wms/
# WMS_EXTERNAL_ADDRESS: http://localhost/wms/
# WMS_KAFKA_HOST: broker:9092
# WMS_SCHEMA_REGISTRY: http://schema_registry:3501
# WMS_CORS_ORIGIN: https://oefen-veiligheidsregio.lcms.nl
# WMS_CORS_HEADERS: 'authorization, content-type'
# ports:
# - 8101:5101
# depends_on:
# - broker
# - schema_registry

trial_management_tool:
image: drivereu/trial-management-tool:latest
depends_on:
- broker
- schema_registry
ports:
- '3210:3210'
environment:
CLIENT_ID: TB-TrialMgmt
KAFKA_HOST: broker:9092
SCHEMA_REGISTRY: http://schema_registry:3502

```

```

    TRIAL_MANAGER_SERVER_PORT: 3210
    PRODUCE:
system_request_change_of_trial_stage,system_tm_phase_message,system_tm_role_player,system_tm_session_mgmt
    SSL: 'false'
    SSL_PFX: certs/TB-TrialMgmt.p12
    SSL_PASSPHRASE: changeit
    SSL_CA: certs/test-ca.pem
    volumes:
    - trial-data:/app/trials

ost_db:
    image: drivereu/ost_database:1.0.29143
    ports:
    - "5437:5432"
    volumes:- database-OST:/var/lib/postgresql/data
# restart: always

ost_web:
    image: drivereu/ost_frontend:1.0.29143
    links:
    - ost_api
    ports:
    - "127.0.0.1:85:80"
    - "127.0.0.1:445:443"
# restart: always

ost_api:
    image: drivereu/ost_backend:1.0.29143
    links:
    - ost_db
    ports:
    - "8080:8080"
# restart: always

silent-producer:
    image: silent-producer
    depends_on:
    - broker
    - schema_registry
    environment:
    KAFKA_HOST: broker:9092
    SCHEMA_REGISTRY: http://schema_registry:3502
    PRODUCE_TOPICS:
simulation_request_unittransport,simulation_request_startinject,simulation_entity_item,sumo_AffectedArea,standard_cap,system_timing,system_topic_access_invite
    volumes:
    database-OST:
    postgres-data:
    trial-data:

```

4. Go back to the required folder.
5. If you are windows user
 - 5.1. Press **Shift + right click** mouse button anywhere on the folder window (this folder must have file docker-compose.yml).
 - 5.2. Choose open powershell
 - 5.3. Write in terminal **docker-compose up** and press enter
6. If you are linux/ubuntu user
 - 6.1. Press **right click** mouse button anywhere on the folder window (this folder must have file docker-compose.yml).
 - 6.2. Choose open terminal
 - 6.3. Write there **docker-compose up** and press enter
7. Congratulation, your application has been just turned on.

API description

1. Get /api/answers

Api let us see answer of trial session including some text inside.

@RequestParam("trialsession_id") long trialSessionId,

@RequestParam("search") String text

Produces Json

TrialUserDTO.ListItem user

String Name

String observationTypeName

String observationTypeDescription

2. Get /api/answers/csv-file

Api let us load CSV file with answers of session.

@RequestParam(value = "trialsession_id") long trialSessionId

Return null

3. Delete /api/answers/{answer_id:\\d+}/remove

Api let us delete answer.

@PathVariable(value = "answer_id") long answerId,

@RequestParam("comment") String comment

Return null

4. GET /api/answers-events

Api let us see all answers and events in session of current user.

@RequestParam(value = "trialsession_id") long trialSessionId

Produces JSON

long id;

long observationTypeId;

String name;

String description;

ZonedDateTime time;

ZonedDateTime trialTime;

String type;

5. Get /api/event/search

Api returns list of events in session.

@RequestParam(value = "trialsession_id",) long trialSessionId,Pageable pageable

Produces JSON

public String firstName;

public String lastName;

public String trialRoleName;

6. GET /api/observationtypes

Api returns list of question

sets

@RequestParam("trialsession_id") Long trialSessionId

Long answersId

String name

String description

7. GET /api/observationtypes/form

Api returns roles, which can answer on question set.

@RequestParam("observationtype_id") Long observationTypeId,

@RequestParam("trialsession_id") Long trialSessionId

Return json

List<TrialRoleDTO.ListItem> roles

JsonNode jsonSchema,

where in roles are two variables:

long id,

String name

8. /api/observationtypes/admin/addNewQuestionSet

@RequestBody

```

String name
String description
long trailStageld;
long trailId;
boolean multiplicity;
boolean withUsers;
int position;
List<AdminQuestionDTO.ListItem> questions = new ArrayList<>();
Return response with http status and json
long id
String name
String description
long trailStageld;
long trailId;
boolean multiplicity;
boolean withUsers;
int position;
List<AdminQuestionDTO.ListItem> questions = new ArrayList<>();

```

9. GET /api/observationtypes/admin/getNewQuestionSet

Api let us get question set from chosen stage.

@RequestParam(value = "id") long id

Return response with http status and json

```

long id
String name
String description
long trailStageld;
long trailId;
boolean multiplicity;
boolean withUsers;
int position;
List<AdminQuestionDTO.ListItem> questions = new ArrayList<>();

```

10. PUT /api/observationtypes/admin/updateQuestionSet

Api let admin update question set.

@RequestBody

```

String name
String description
long trailStageld;
long trailId;
boolean multiplicity;
boolean withUsers;
int position;

```

```
List<AdminQuestionDTO.ListItem> questions = new ArrayList<>();
```

Return response with http status and json

```
long id
```

```
String name
```

```
String description
```

```
long trailStageId;
```

```
long trailId;
```

```
boolean multiplicity;
```

```
boolean withUsers;
```

```
int position;
```

```
List<AdminQuestionDTO.ListItem> questions = new ArrayList<>();
```

11. DELETE /api/observationtypes/admin/deleteQuestionSet

Api let admin delete question set from a stage.

```
@RequestParam(value = "id") long id
```

Return response with http status and string

12. GET /api/questions-answers

Api returns data about answer and question.

```
@RequestParam(value = "answer_id") long answerId
```

```
@Produces JSON
```

```
long answerId
```

```
String name (of observatory type)
```

```
String description (of observatory type)
```

```
String time
```

```
String trialTime
```

```
JsonNode questionSchema
```

```
JsonNode formData
```

```
JsonNode trialRoles
```

13. POST /api/questions-answers/{answer_id:\d+}/comment

Api let observer add comment to answer.

```
@PathVariable(value = "answer_id") long answerId,
```

```
@RequestParam("comment") String comment
```

Return null

14. Get /api/trial-time

Api let us know server time.

Return String with date

15. Get /api/time-elapsed

Api let us know local time.

Return String with date

16. **GET /api/role**
 Api shows all roles in chosen trial.
Return list of roles with:
 long id
 String name
 String roleType

17. **GET api/stages**
 Api shows all stages
 @RequestParam(value = "trial_id") long trialId, Pageable pageable
Return list of stages with:
 long trialId
 String name
 LocalDateTime simulationTime

18. **POST api/stages/admin/addNewTrialStage**
 Api let us add stage to chosen trial.
 @RequestBody(
 long id
 long trialId
 String name)
Return response with http request status and in body are:
 Long trialId
 Long id
 String name

19. **GET api/stages/admin/trialStageWithQuestions**
 Api returns stage with id
 @RequestParam(value = "id") long id
Return response with http request status and in body are:
 Long trialId
 Long id
 String name
 LocalDateTime simulationTime;
 List<AdminObservationTypeDTO.ListItem> questions = new ArrayList<>()

20. **DELETE api/stages/admin/deleteTrialStage**
 Api deletes a trial's stage.
 @RequestParam(value = "id") long id
Return response with a http request status and a String
"Trial Stage id =" + id + " is deleted".

21. **PUT api/stages/admin/updateTrialStage**
 Api update a trial's stage.
 long id
 long trialId
 String name
Return
 long id
 long trialId
 String name

22. **GET /api/role**
 Api shows all roles in the chosen trial.
Return list of roles with:
 long id
 String name
 String roleType

23. **GET /api/trialsessions/{trialsession_id:\d+}**
 Api shows data about chosen trial session.
 @PathVariable(value = "trialsession_id") long answerId
Return json with:
 long trialId
 String trialName
 String trialDescription
 String lastTrialStage

24. **GET /api/trialsessions**
 Api shows list of trial session.
 Api shows all trials, which are available for logged user.
Return list of trials with:
 long trialId
 String trialName
 String trialDescription
 String lastTrialStage

25. **GET /api/trialsessions/active**
 Api shows list of active trial session available from logged user.
Return list of trials:
 long trialId
 String trialName

String trialDescription
 String lastTrialStage
 long initId
 Boolean initHasAnswer
 String name

26. PUT /api/trialsessions/{trialsession_id:\d+}/changeStatus

Api let us change the status of the trial session.

@PathVariable(value = "trialsession_id") long trialSessionId,

@RequestParam(value = "status") String status

Return null

27. GET /api/trialsessions/manual/{trialsession_id}/{is_manual}

Api let us choose if stages of trial will be changed manually or automatically.

@PathVariable long trialsession_id,

@PathVariable boolean is_manual

Return String

"current stage in session " +trialSession.getId() + " is: "

+trialSession.getLastTrialStage().getId() +

"/"+trialSession.getLastTrialStage().getName()

+ " manual mode is " +isManual

28. PUT /api/trialsessions

Api let change stage of trial session

@PathVariable(value = "id") Long trialSessionId,

@RequestBody @Validated TrialStageDTO.MinimalItem minimalItem,

Where minimalItem is json {id:value}

Return json

long trialId;

Long lastTrialStageId;

LocalDateTime startTime;

LocalDateTime pausedTime;

29. POST /api/trialsessionscreateNewSessionFile

Api lets administrators create new session users and create files with their usernames and passwords.

@RequestBody NewSessionForm newSessionForm,

Where newSessionForm is a json

{long trialId

String initialStage

String prefix

String status

List<UserForm> users}, where

UserForm is a json
 {String email
 List<String> role}

Return null

30. GET /api/trialsessions/trials

Api shows all trails, which current user is a session manager.

Return Map(Id : Long, name : String)

31. POST /api/trialsessionsnewSessionValues

Api returns data about trial.

@RequestParam(value = "trial_id") long trialId

Return trial node, in trial node we can find
 trialStages, trialRoles, authUsers

32. POST /api/trialsession/admin/addNewUserRoleSession

Api let us add user to role in trial's session.

@RequestBody

long trialUserId

long trialRoleId

long trialSessionId

Return adminUserRoleDTO

33. DELETE /admin/deleteUserRoleSession

Api let us delete user from the trial's session.

@RequestParam(value = "trialRoleId") long trialRoleId,

@RequestParam(value = "trialUserId") long trialUserId,

@RequestParam(value = "trialSessionId") long trialSessionId

Return Response with http status and string "Trial user session is deleted"

34. GET /api/ostAllQuestionsForMobile

Api returns data about questions set.

Return List<ObservationTypeDTO.SchemaItem>

List<Long> answersId

String name

String description

long id

List<TrialRoleDTO.ListItem> roles

JsonNode jsonSchema, where jsonSchema is a schema of all questions in the question set.

35. GET api/ostTrialId

Api returns id of Trial.

Return Long

36. GET api/ostTrialSessionId

Api returns id of Trial Session.

Return Long

37. GET api/ostTrialStageId

Api returns id of Trial Stage.

Return Long

38. POST api/admin/addNewTrial

Api let user create trial with name and default parameters

```
private long trialIdString
trialName
String trialDescription
String lastTrialStage
private Boolean archived
```

Return json

```
long trialIdString
trialName
String trialDescription
String lastTrialStage
Boolean archived
```

39. POST api/admin/updateTrial

Api let edit trial.

Request Body AdminTrialDTO.ListItem

```
long trialId;String trialName
```

```
String trialDescription
```

```
String lastTrialStage
```

```
Boolean archived
```

Return json

```
long trialId
```

```
String trialName
```

String trialDescription
 String lastTrialStage
 Boolean archived

40. GET api/ostTrail

Api returns active Sessions and their actual stages.

RequestParam(value = "trial_name") String trialName

Return String of actual sessions and stages.

41. GET api/stages

Api returns stages of the chosen session.

@RequestParam(value = "trial_id") long trialId, Pageable pageable)

Return PageDTO<TrialStageDTO.ListItem>.

In the json are:

long trialId

String name

LocalDateTime simulationTime,

long id.

42. GET api/user

Api shows users in the chosen trial session.

RequestParam(value = "trialsession_id") long trialSessionId, Pageable pageable

Return PageDTO<TrialUserDTO.ListItem>

In the json are:

long id

String firstName

String lastName

43. GET api/user/version

Api shows the actual version of the application.

Return String

44. POST api/questions/admin/addNewQuestion

Api let add question to question set.

@RequestBody

String name

String description

long observationTypeId

AnswerType answerType
 int position
 String jsonSchema
 boolean commented

Return response with http status and json

long id;
 String name;
 String description;
 long observationTypeId;
 AnswerType answerType
 int position
 String jsonSchema
 boolean commented

45. GET api/questions/admin/getFullQuestion

Api let us get details about the question.

@RequestParam(value = "id") long id

Return response with http status and json

long id;
 String name;
 String description;
 long observationTypeId;
 AnswerType answerType
 int position
 String jsonSchema
 boolean commented

46. DELETE api/questions/admin/deleteQuestion

Api let us delete question from the question set.

(@RequestParam(value = "id") long id)

Return response with http status and string

"Question id =" + id + " is deleted"

47. PUT api/questions/admin/updateQuestion

Api let us update the question.

```
@RequestBody
String name
String description
long observationTypeId
AnswerType answerType
int position
String jsonSchema
boolean commented
```

Return response with http status and json

```
long id;
String name;
String description;
long observationTypeId;
AnswerType answerType
int position
String jsonSchema
boolean commented
```

48. /api/user

Api let user see a list of users in trial session.

@RequestParam(value = "trialsession_id") long trialSessionId, Pageable pageable

Return DTO with

```
long id;
String login;
```

49. /api/user/version

Api returns version

Return application version

Database

This database schema is used in OST system.

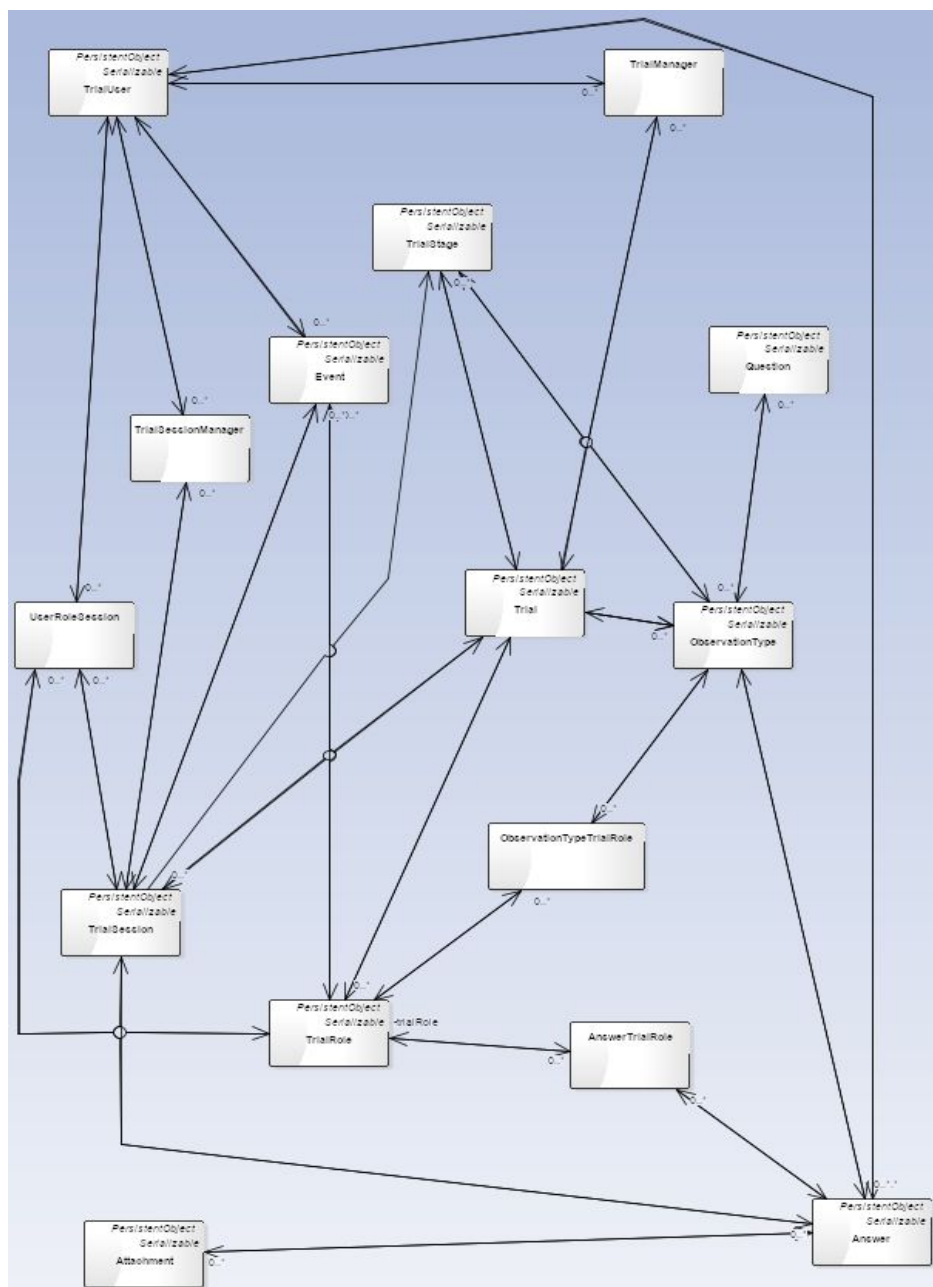


Figure 14. Schema of database

This project has received funding from the European Union's 7th Framework Programme for Research, Technological Development and Demonstration under Grant Agreement (GA) N° #607798

In this model ObservationType=Question Set.

Environmental variable

Table 1 Table of example environmental variables.

Type of environmental variable	Example Value
KEYCLOAK_HOST	127.0.0.1
KEYCLOAK_PORT	8070
KEYCLOAK_REALM	driver2
KEYCLOAK_RESOURCE	ost_app
DRIVER_KEYCLOAK_ADMIN_CLIENT	ost_admin
DRIVER_KEYCLOAK_ADMIN_SECRET	7b170e34-327d-4e1c-9a64-125456aa4ffe
KEYCLOAK_PROXY_UR	http://192.168.1.21:8070/auth
DRIVER_IS_TESTBED_ON	false
OST_DB_HOST	127.0.0.1
OST_DB_PORT	8080
OST_DB_NAME	ost
OST_DB_USER	ost_user
OST_DB_USER_PASS	ost_user_pass