D3.6 Territory Monitoring System

Release Notes

Project acronym:	FLOOD-serv
Project full title:	Public FLOOD Emergency and Awareness SERvice
Grant agreement no.:	693599
Responsible:	ANO
Contributors:	Pedro Leite, Sergio Almeida
Document Reference:	D3.6
Dissemination Level:	PU
Version:	1.9
Date:	02/12/19

History

Version	*Date*	*Modification reason*	*Modified by*
0.1	03/01/2017	Initial draft: structure of contents	Pedro Leite
0.2	29/01/2017	Final content and revision	Pedro Leite
1.1	18/11/2019	Revision after feedback from EC	Pedro Leite
1.2	19/11/2019	Chapters 3, 4 and 5 – reviewed and updated	Pedro Leite
1.3	20/11/2019	Chapter 6 – update Test cases and Results	Pedro Leite
1.4	21/11/2019	Chapter 7 – Updated Release notes and links	Pedro Leite
1.6	22/11/2019	Chapter 6 – update Test cases and Results	Pedro Leite
1.6	29/11/2019	Chapter 4 – tech specs added Chapter 9 and appendix I added Total Number of chapters changed 1.3 Changes in Response to Final Review added	Pedro Leite
1.7	29/11/2019	1.3 Changes in Response to Final Review updated	Pedro Leite
1.8	29/11/2019	1.3 Changes in Response to Final Review updated	Pedro Leite
1.9	02/11/2019	Added references to user guide in Chapter 11 – Appendix II	Pedro Leite

Table of contents

1 Introduction 8

1.1 Purpose of the Document 8

1.2 Structure of the Document 8

1.3 Changes in Response to Final Review 8

3 Overview of user stories implemented 13

4 Technical Specifications 15

5 System Test Case Repository 17

5.1 System Tests 17

5.2 Baseline for System Tests - Preconditions 17

6 Test cases – overview 18

7 Test Cases and Results 19

7.1 Data Collection – Test Cases and Results 19

7.1.1 Test Case – Receive image or pair of images for analysis 19

7.1.2 Results – Receive image or pair of images for analysis 19

7.1.3 Test Case – Visualization of analysis report for flood detection 21

7.1.4 Results – Visualization of analysis report for flood detection 21

7.1.5 Test Case – Visualization of analysis report for flood prevention 22

7.1.6 Results – Visualization of analysis report for flood prevention 22

7.1.7 Test Case – Visualization of analysis report for flood impact 24

7.1.8 Results – Visualization of analysis report for flood impact 24

8 TMS Release Notes 37

8.1 System access requirements 37

8.2 Features 37

8.3 Installation guide 37

9 Work Developed and Conclusions 38

List of tables

Table 1 Naming Conventions Industry Standards..6

List of abbreviations

<Abbreviation>	**<Explanation>**
CDF	Citizen Direct Feedback
EMC	Emergency Management Console
SW	Semantic Wiki
TMS	Territory Monitoring System
DM	Decision Maker
F	Facilitator
FE	Flood Emergency Expert
ER	Emergency Responder
SMC	Social Media Component
JSON	JavaScript Object Notation

Naming Conventions and Terminology

Naming Domain	*Standard*	*Examples*	*Link*
Agile Development	SCRUM	Epic, theme, user story	http://www.scrumguides.org/
Requirements	IREB	Use case, non-functional requirements	https://www.ireb.org/content/downloads/1-cpre-glossary/ireb_cpre_glossary_16_en.pdf
QA	ISTQB	System test, unit test, integration test, defect	http://www.istqb.org/downloads/ glossary.html

Table 1 Naming Conventions Industry Standards

Executive summary

D3.6 Territory Monitoring System is the sixth deliverable from WP3. This is a software deliverable about development of the TMS component.

The TMS development is based on the D3.1 and D3.2 documents, which deals with the functional specifications through the definition of the user stories for each component composing the FLOOD-serv platform and the description of the technical specification of each component: structure, data model, interfaces, services, data providers and implementation environment.

D3.6 is being developed at the same time as D3.3, D3.4, D3.5, and D3.7, which are reporting the development of the rest of components composing the FLOOD-serv platform, i.e.: the Social media component (SMC), the Emergency Management Console (EMC), the Semantic Wiki (SW), and the Citizen Direct Feedback (CDF), respectively.

1 Introduction

1.1 Purpose of the Document

The goal of this document is to provide the release note for the Territory Monitory System. The document is based on D31 and D3.2.

1.2 Structure of the Document

The document is organized as in the following:

- Chapter one: Introduction
- Chapter two: Overall approach and methodology
- Chapter three: Overview of user stories implemented
- Chapter four: Technical specifications
- Chapter five: System test case repository
- Chapter six: Test cases overview
- Chapter seven: Test cases and results
- Chapter eight: Release notes
- Chapter nine. Work developed and Conclusions
- Chapter ten: Appendix I – API Doc

For the reference, before the *Changes in Response to Final Review*, the document was organized:

- Chapter one: Introduction
- Chapter two: Overall approach and methodology
- Chapter three: Overview of user stories implemented
- Chapter four: System test case repository
- Chapter five: Test cases overview
- Chapter six: Test cases and results
- Chapter seven: Release notes

1.3 Changes in Response to Final Review

Observations	***Actions and Changes***
It is unclear what exactly was achieved under the WP3 in the areas of the Territory Monitoring System and Citizen Direct Feedback components.	**All the user stories previously identified were implemented with success and with the alterations mentioned and the component is up and running and fully integrated in the system.** To underline this, the chapters with the user stories implemented – chapter 3 was updated, Chapter 4 – Tech Specs was added, Chapter 6 and Chapter 7 for the test cases and results was updated, chapter 8 with release notes was

	updated with the currently working links for all 5 pilots, chapter 9 with work developed and conclusions was added, Appendix I was added. **Important to clarify that TMS offers data related to water level based on image analysis, as an alternative source to sensors (for places where sensors do not exist).**
more substantial description of the content of listed components should be provided in the Deliverables D3.3, D3.5 - D3.7 accordingly to the remarks of the present report and its Annex 1 - Deliverables due for the Period 2/Final review.	**The whole document was revised and more details was added**: the chapters with the user stories implemented – chapter 3 was updated, Chapter 4 – Tech Specs was added, Chapter 6 and Chapter 7 for the test cases and results was updated, chapter 8 with release notes was updated with the currently working links for all 5 pilots, chapter 9 with work developed and conclusions was added, Appendix I was added.
Crucial to the overall applicability of TMS is USTMS4, which analysis the output report and cross checks information on potential flood occurrence threat. Is an information on a water depth is available from other sources ? How cross-validation and fusion of data is performed ?	**The whole document was revised and more details was added – more relevant for this point the updated user stories – USTMS4**: the chapters with the user stories implemented – chapter 3 was updated, Chapter 4 – Tech Specs was added, Chapter 6 and Chapter 7 for the test cases and results was updated, chapter 8 with release notes was updated with the currently working links for all 5 pilots, chapter 9 with work developed and conclusions was added, Appendix I was added. In chapter 7 among other information, explains how this was originally designed. The TMS analysis the images and and outputs the analysis/depth. This info via API can them be crossed checked with the other data presented in the EMC/Portal – the main data silo crawlers of the system.
As such, procedures for pattern recognition of images and decision process must be described in more details as well as the testing procedure which was applied to this component of the FLOOD-serv platform.	**The whole document was revised and more details was added – more relevant for this point chapter 7 was fully updated and chapter 4 with the tech specs added.**

The access link to this API is not working. .	The demo environment, originally referenced in the document was out of commission. The updated links are provided in Chapter 8 – release notes, chapter 4 added and appendix I with the API Docs Updated links: For Decision Makers and/or Operators you can access to the TMS directly in the FLOOD-Serv Platform or directly by typing in: https://bilbao-floodserv-saas.ano.pt/ https://bratislava-floodserv-saas.ano.pt/ https://genova-floodserv-saas.ano.pt/ https://tulcea-floodserv-saas.ano.pt/ https://vnfamalicao-floodserv-saas.ano.pt/ The credentials to access are: • User: salmeida • Password: 123 Choose TMS >> Calculate Depth on the main menu (top left corner). The legacy "Calculate Differences" method is also available, via the same option path.
D4.2 The document still lacks information on proposed algorithms for the image/video analysis. In particular, on calibration of different types of cameras, monoscopic and stereoscopic measurements, filtering, segmentation, classification, detection of shape and contours etc as well on image correction, motion estimation. What is error /uncertainty magnitude associated with such type of sensor?	**The whole document was revised and more details was added – more relevant for this point chapter 7 was fully updated and chapter 4 with the tech specs added.**
D4.4 (…) while TMS is limited to a camera reading of a water level gauge at a given location, without a possibility of horizontal assessment of the flood extent.	**The whole document was revised and more details was added – more relevant for this point chapter 7 was fully updated and chapter 4 with the tech specs added.**

2 Overall approach and methodology

The tests described in this document demonstrate that the TMS component (D3.6) has been successfully implemented and the new features identified in D3.1 and D3.2 have been implemented, with the appropriate shift in the original strategy, as stated in the rest results.

As the next steps integration tests will be implemented as part of WP4. A Portal UI and integration with the other components will be also be implemented in WP4.

WP5 is about user acceptance tests, which is the last phase of a software testing process. During UAT (User Acceptance Testing), actual software users test the software to make sure that it works in real-world scenarios, according to specifications.

The overall approach applied is SCRUM, consequently the results and documentation the software delivered in D3.6 was following the sprints and user stories implemented.

In SCRUM the tasks are divided into time boxes (small time frames) to deliver specific features in the release so that the working software build can be delivered after each iteration. Builds are incremental in terms of features; the final build of D3.6 has all the features.

Role of the tester

- Ensure End-user satisfaction through delivery of high-quality software.
- Engagement is early during the project from sprint planning.
- Discuss and understand each user story and then decide on acceptance criteria for the same.
- Define activities for themselves to estimate time, updating test cases as and when changes appear, complete testing within the sprint time etc.
- Develop test cases as per the story acceptance criteria and change whenever there is a change in story.
- Deliver high quality software iteratively from a couple of weeks to a couple of months.
- Ensure user stories get clarified where there is insufficient information.
- Break user stories into different testing tasks.
- Decide each story test coverage

3 Overview of user stories implemented

The original user stories to be implemented to accomplish with the non-functional, functional and technical requirements of the TMS are collected in the following table:

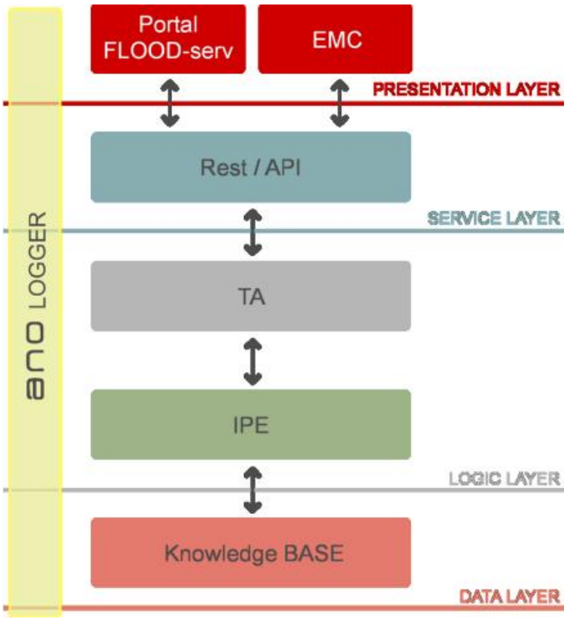
ID	Summary	Description
USTMS1	Identify relevant occurrences for flood prevention	As a facilitator or flood emergency expert I want to analyse a geographical area, and search for occurrences that might prove relevant for flood prevention or measure the level of a particular body of water. Acceptance criteria: 1.The user can upload one or two images; 2.The user is notified by the TMS the analysis is processing;
USTMS2	Identify relevant occurrences for flood detection	As a facilitator or flood emergency expert I want to analyze a geographical area, and search for occurrences that might prove relevant for flood detection or measure the level of a particular body of water. The relevant differences can range from alterations in constructions, to flooded areas and others. Acceptance criteria: 1. The user can upload one or two images; 2. The user is notified by the TMS the analysis is processing;
USTMS3	Identify relevant occurrences for post flood impact analysis	As a facilitator or flood emergency expert I want to analyze a geographical area, and search for occurrences that might prove relevant for post flood impact analysis or measure the level of a particular body of water. Acceptance criteria: 1. The user can upload one or two images; 2. The user is notified by the TMS the analysis is processing;

USTMS4	Analyse the output report and cross check information	As a facilitator or flood emergency expert I want to analyze and cross check the TMS report of a specific area. This functionality aims to provide end users with functionality to check each one of the occurrences and cross relate them with other relevant information in the FLOOD-serv project. Each occurrence will have specific information about: • Area; • Geo coordinates; • Type of occurrence; • Trusted interval. or Water level Acceptance criteria: Depth Calculation 1. The user can visualize the outputted water level Differences Calculation (legacy) 1. The user can visualize the list of occurrences of a specific report and check its details, or check the calculated water level; 2. Cross check the occurrence with information from the FLOOD-serv system, via integration with overall system/other components

Originally, the user stories to be implement in the TMS, were based on a single main method, that focused on calculation differences between two images. Development and testing proved that this method, as the bellow results show, proved to be below our expected threshold. For this, a second method, focused on vertical images and on calculating depth of bodies of water was implemented specially for the project and thus the user stories and consequent results adapted. The legacy method of “Calculation Differences” remains available.

4 Technical Specifications

The TMS component is a service-oriented application with a multi-layer perspective. The architecture of the TMS was designed taking in account the industries best practices allowing scalability, modularity and code efficiency. The various layers allow that data, logic, API and interfaces are separated and that can be managed separately.



The TMS, originally was planned do not have a FO graphical user interface but, it deemed necessary to implemented it, to allow it to work independently from other components.
Its main access point is the Service Layer that provides the API access to the data and business logic of the TMS component, via a RESTful Web Service described in Appendix I.

The TMS has the following base characteristics for the imaging processing algorithm/pattern recognition for the newly developed “Depth Calculation” method:

- The water level measurement will be performed by identifying a predetermined standard and measuring the number of equal sized measurements at the surface
- The pattern similar to the “reference bar/marker”, visible in the input image, oriented vertically and facing the image plane, must have the pattern for recognition with a width of 64 pixels or greater. Measurement marks, squares, must be at least 30 pixels sideways.
- The measurement resolution and its error are associated with the resolution, in effective pixels (description below) and the minimum reading error will be 3 pixels of the image if it is monochrome and 5 pixels if it is colour. If atmospheric factors or optical distortions are added other errors add up.
- The measurement resolution and its error are associated with the resolution, in effective pixels (description below) and the minimum reading error will be 3 pixels of the image if it is monochrome and 5 pixels if it is colour. If atmospheric factors or optical distortions are added other errors add up.
- Effective pixels: Resolution at which the transition from white to black occurs only 3 pixels if the image is monochrome and 5 if it is colour

- The algorithm will consist of a pre-trained pattern recognition component to detect the pattern (top of the scale element) and a second detection and measurement component (squares).

The dedicated and newly developed UI for TMS is accessible via web for:

- Decision Makers
- Facilitator/Flood Expert
- Emergency Responders
- System Operator

5 System Test Case Repository

This chapter provides an overview of the current system tests.

The tests described in this document demonstrates that the Territory Monitoring System is implemented and the features identified in D3.1 and D3.2 have been successfully implemented as part of the task T3.8, with the adapted strategy.

5.1 System Tests

The core focus of the system tests is to test the Territory Monitoring System - without integration with the other modules - as a black box as seen by the user. This test level is being performed by dedicated experts (testers, test manager).

5.2 Baseline for System Tests - Preconditions

In order to extract reproduceable and consistent results from executing the system tests they must be performed in a defined environment. Besides system level requirements (database is up and running), there are other internal settings that must be set accordingly. These settings are called preconditions and this chapter lists some of the most relevant ones which are present in most of the verification tests.

Id	Title	Setup
PRE 1	Activated Facilitator	client exists user with facilitator role has been added by the FLOOD-serv platform password for facilitator has been set
PRE 2	Activated Flood Expert	client exists user with flood expert role has been added by the FLOOD-serv platform password for flood expert has been set
PRE 3	Activated Decision Maker	client exists user with decision maker role has been added by the FLOOD-serv platform password for decision maker has been set
PRE 4	Activated Emergency Responder	client exists user with emergency responder role has been added by the system operator password for emergency responder has been set

6 Test cases – overview

We have grouped the tests carried out into clusters: such related to data collection, data visualization and emergency management.

	Italian Version	Portuguese Version	Romanian Version	Slovakian Version	Spanish Version
Data Collection					
Receive pair of images for analysis	X	X	X	X	X
Data Visualization					
Visualization of analysis report for flood detection	X	X	X	X	X
Visualization of analysis report for flood prevention	X	X	X	X	X
Visualization of analysis report for flood impact	X	X	X	X	X

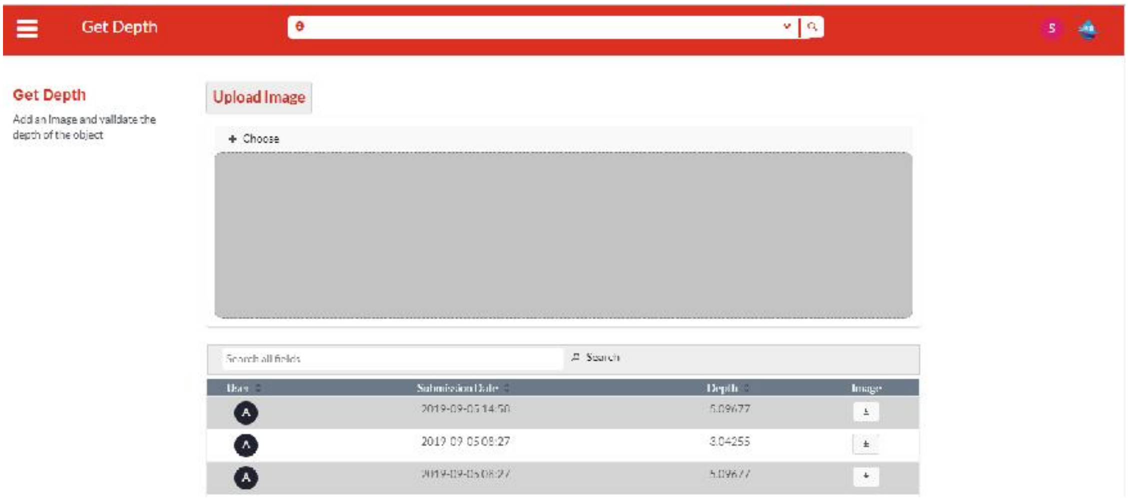
7 Test Cases and Results

7.1 Data Collection – Test Cases and Results

7.1.1 Test Case – Receive image or pair of images for analysis

Receive pair of images for analysis	
Test Type:	Manual
Status:	Final
Preconditions:	) PRE 1 - Activated Facilitator (Platform operator)
Steps to complete:	1. Upload Images to TMS
Expected Outcome:	1. Module receives images for processing 2. Images are validated and ready for processing

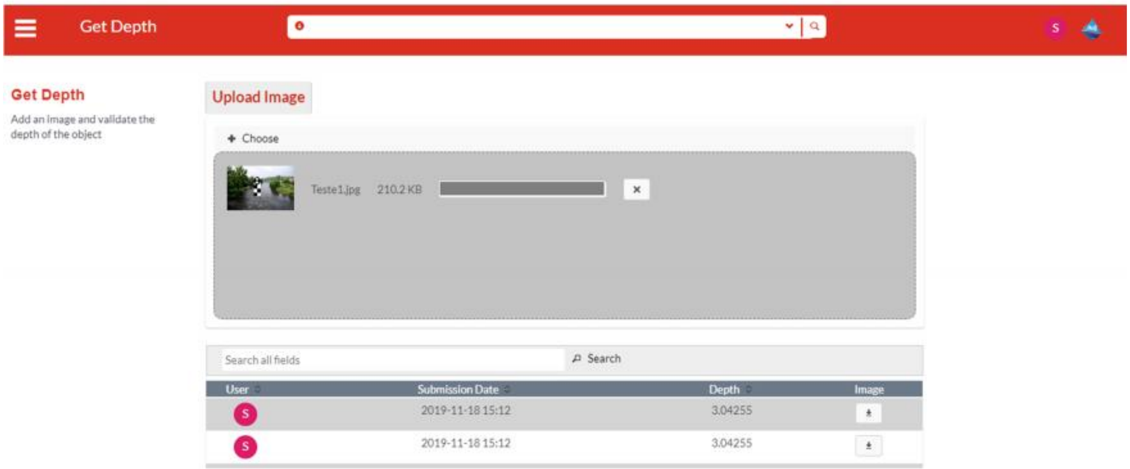
7.1.2 Results – Receive image or pair of images for analysis



Add an Image and validate the depth of the object

User	Submission Date	Depth	Image
A	2019-09-05 14:50	5.09677	⭳
A	2019 09 05 08:27	3.04255	⭳
A	2019-09-05 08:27	5.09677	⭳

Option for uploading image for depth calculation. This can be accessed via main menu » TMS » Calculate Depth



Get Depth

Add an image and validate the depth of the object

Upload Image

+ Choose

Teste1.jpg 210.2 KB ×

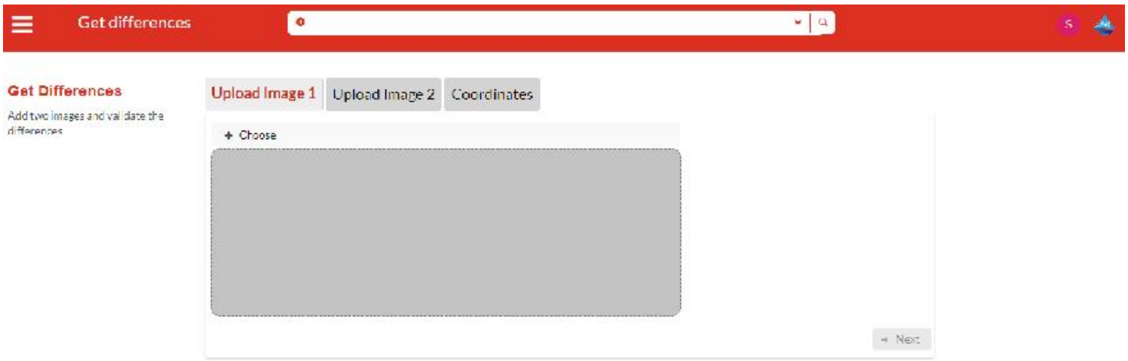
Search all fields

Search

User	Submission Date	Depth	Image
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓

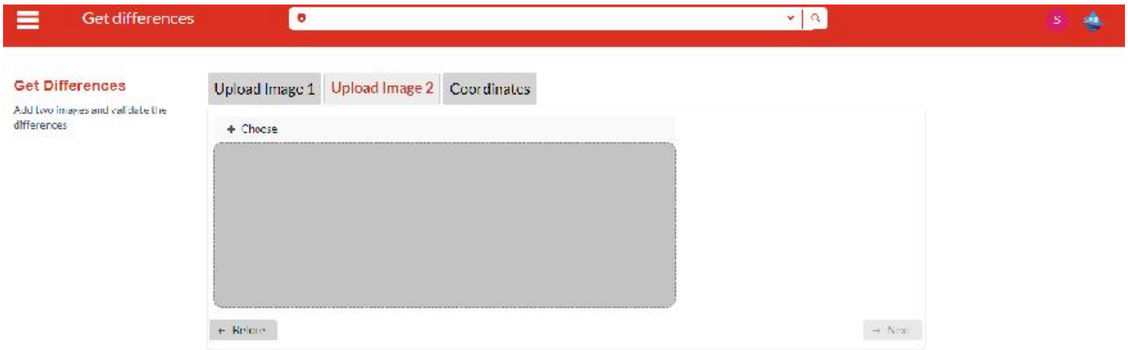
After choosing an image, the TMS will upload and assesses if the image complies with the base parameters.

For the legacy method “Get Differences”:



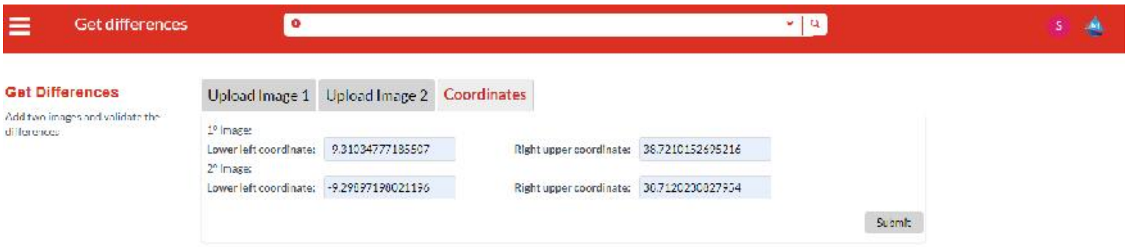
Get Differences

Add two images and validate the differences



Get Differences

Add two images and validate the differences



Get Differences

Add two images and validate the differences

1° Image:

Lower left coordinate: 9.31034777185507 Right upper coordinate: 38.7210152695216

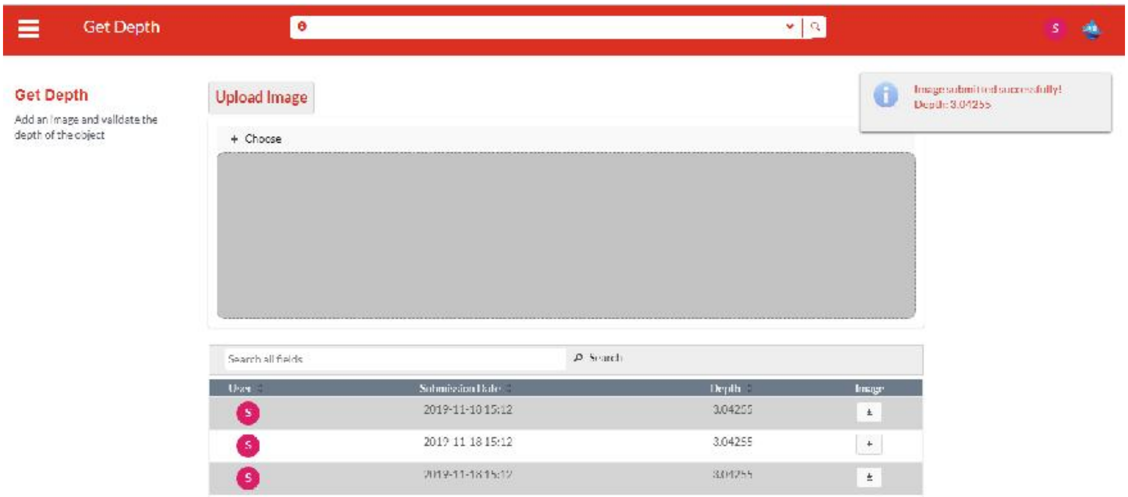
2° Image:

Lower left coordinate: -9.29897190021196 Right upper coordinate: 30.7120230027934

7.1.3 Test Case – Visualization of analysis report for flood detection

Visualization of analysis report for flood detection	
Test Type:	Manual
Status:	Final
Preconditions:	- PRE 1 - Activated Facilitator (Platform operator) - Analysis correctly operated
Steps to complete:	1. Start analysis of image or pair of images 2. Expected analysis report
Expected Outcome:	1. Receive analysis report of images

7.1.4 Results – Visualization of analysis report for flood detection



Get Depth

Add an image and validate the depth of the object

Image submitted successfully!
Depth: 3.04255

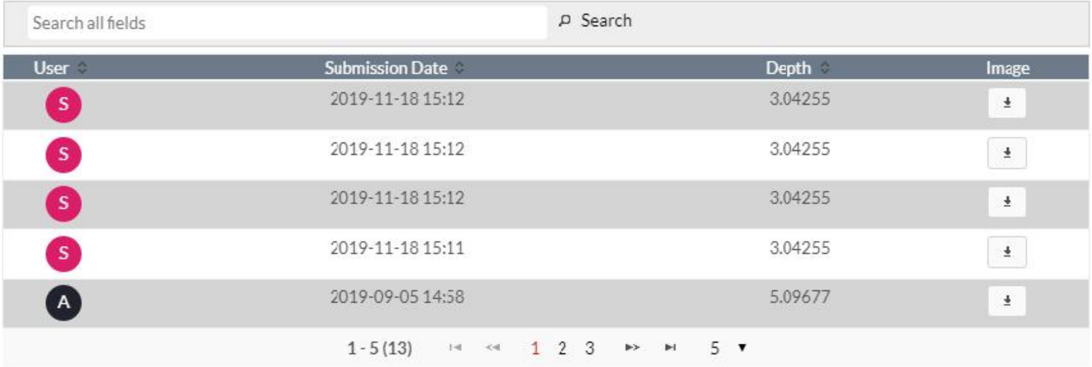
Search all fields

User	Submission Date	Depth	Image
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓

After uploading image for depth calculation, the depth calculate will appear as:

- Popup in the screen

- Added resulted in the list below, where we can consult the complete history of analysis.



Search all fields | Search

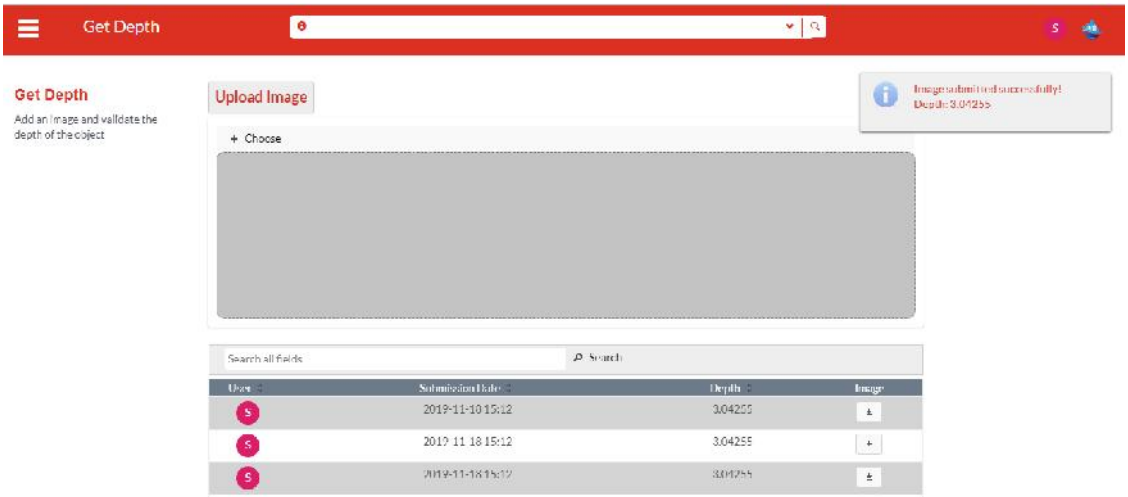
User	Submission Date	Depth	Image
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:11	3.04255	⤓
A	2019-09-05 14:58	5.09677	⤓

1 - 5 (13) 1 2 3 5 ▾

7.1.5 Test Case – Visualization of analysis report for flood prevention

Visualization of analysis report for flood prevention	
Test Type:	Manual
Status:	Final
Preconditions:	- PRE 1 - Activated Facilitator (Platform operator) - Analysis correctly operated
Steps to complete:	1. Start analysis of pair of images 2. Expected analysis report
Expected Outcome:	1. Receive analysis report of pair of images 2. Identify issues on river bank

7.1.6 Results – Visualization of analysis report for flood prevention



Get Depth

Add an image and validate the depth of the object

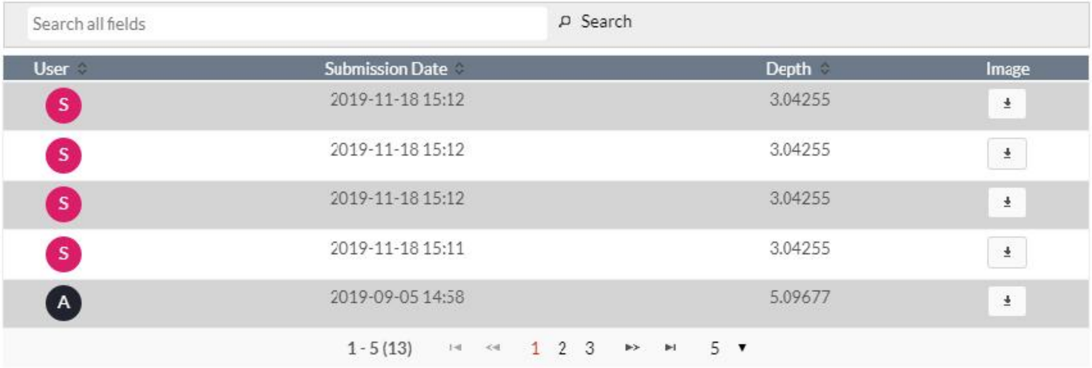
Image submitted successfully!
Depth: 3.04255

Search all fields | Search

User	Submission Date	Depth	Image
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓

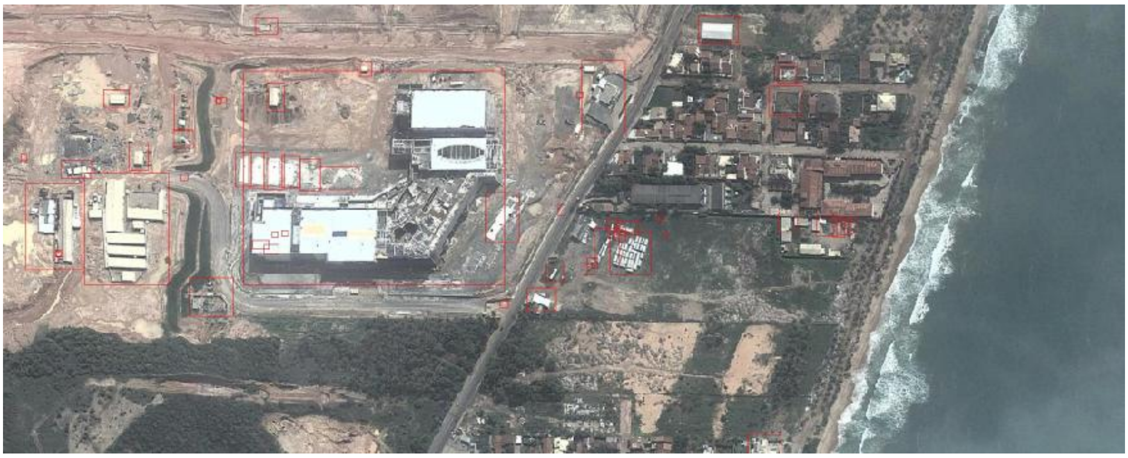
After uploading image for depth calculation, the depth calculate will appear as:

- Popup in the screen
- Added resulted in the list below, where we can consult the complete history of analysis.

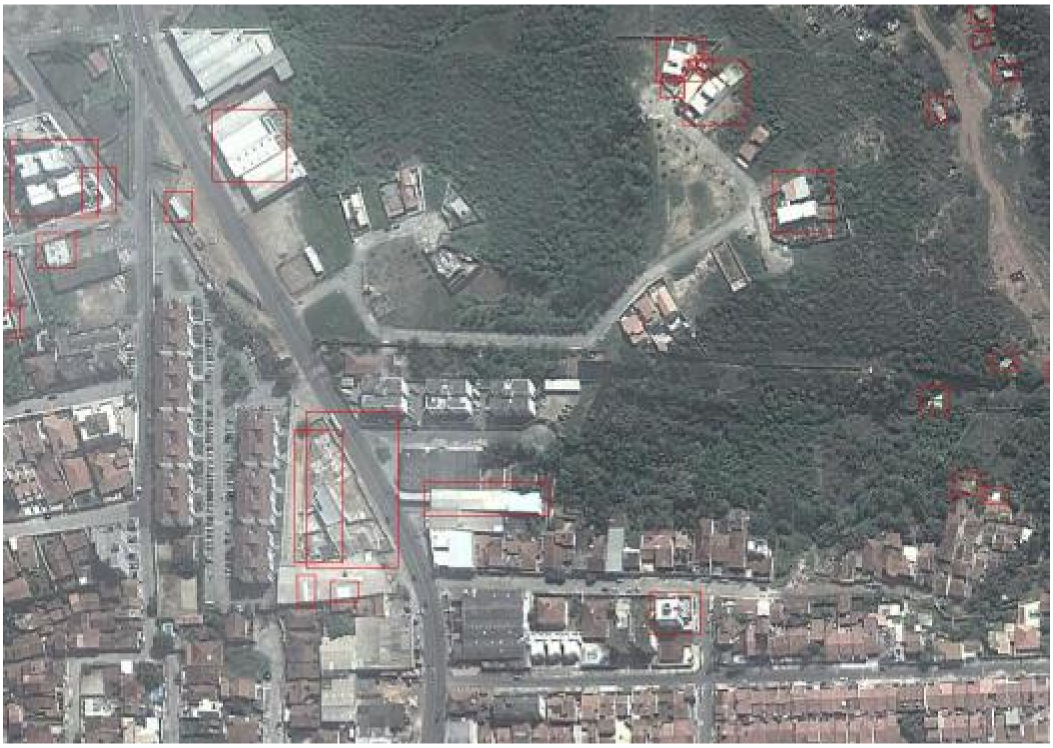


User	Submission Date	Depth	Image
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:11	3.04255	⤓
A	2019-09-05 14:58	5.09677	⤓

Another Analysis #2:



Another Analysis #3:



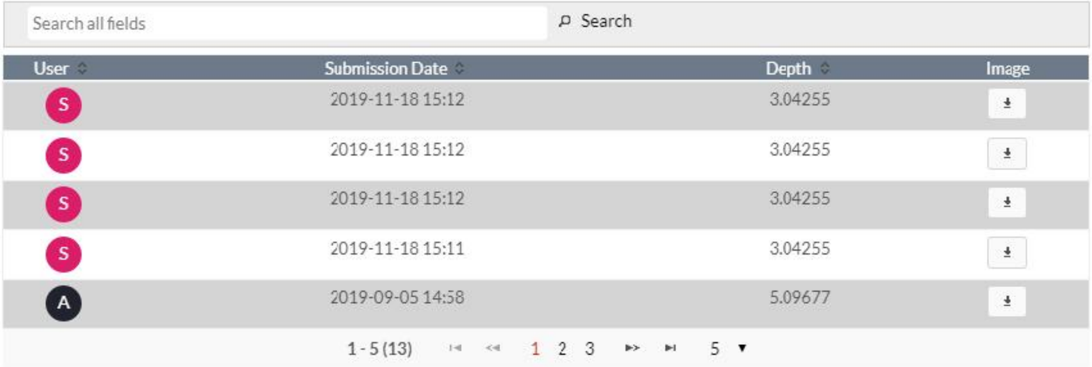
7.1.7 Test Case – Visualization of analysis report for flood impact

Visualization of analysis report for flood impact	
Test Type:	Manual
Status:	Final
Preconditions:	- PRE 1 - Activated Facilitator (Platform operator) - Analysis correctly operated
Steps to complete:	1. Start analysis of image(s) 2. Expected analysis report
Expected Outcome:	1. Receive analysis report of image(s) 2. Identify changes and/or water levels

7.1.8 Results – Visualization of analysis report for flood impact

After uploading image for depth calculation, the depth calculate will appear as:

- Popup in the screen
- Added resulted in the list below, where we can consult the complete history of analysis.

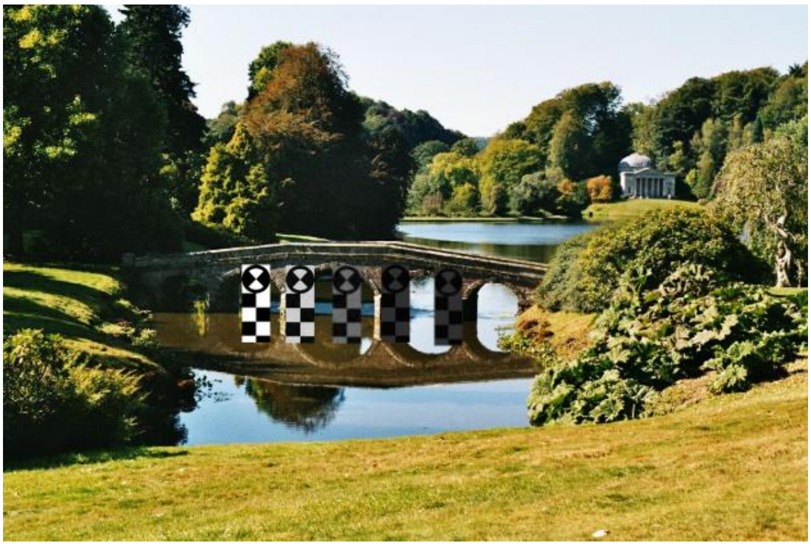


User	Submission Date	Depth	Image
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:12	3.04255	⤓
S	2019-11-18 15:11	3.04255	⤓
A	2019-09-05 14:58	5.09677	⤓

These results are also available via Rest API, and can then be integrated with EMC/Portal for cross check analysis

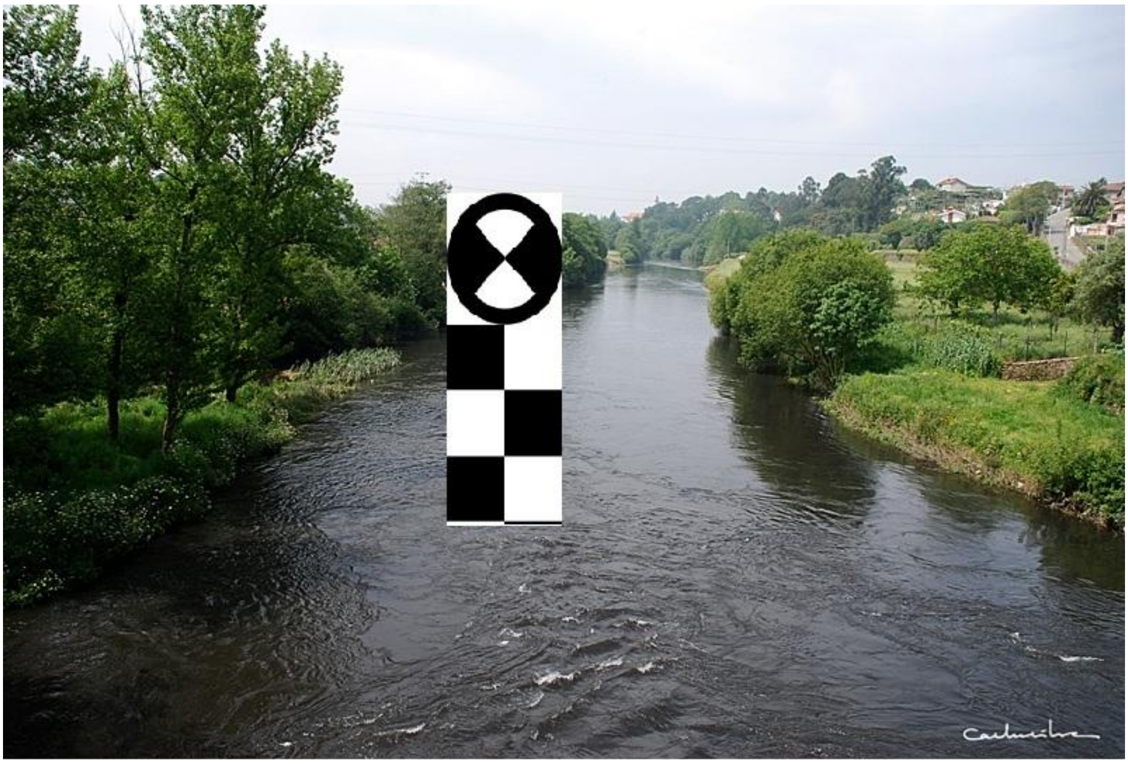
Several images were used for testing the capability of the image processing algorithm. Some examples:

Test 01:

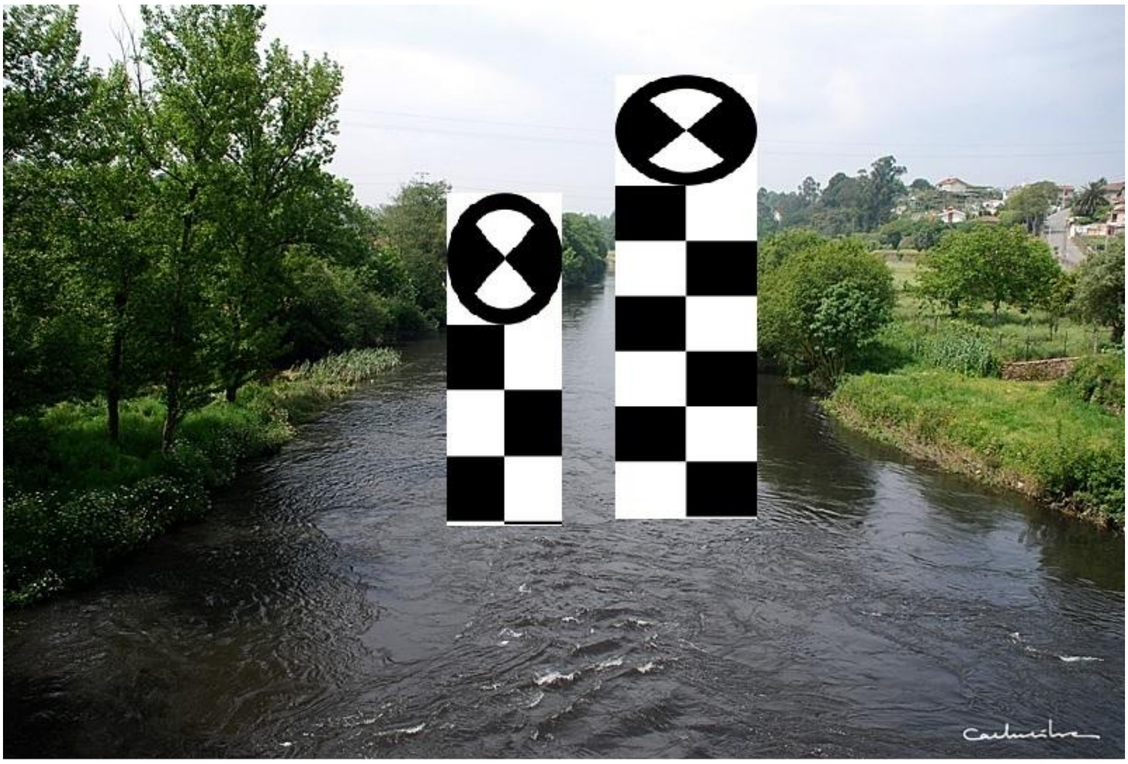


Test 02:

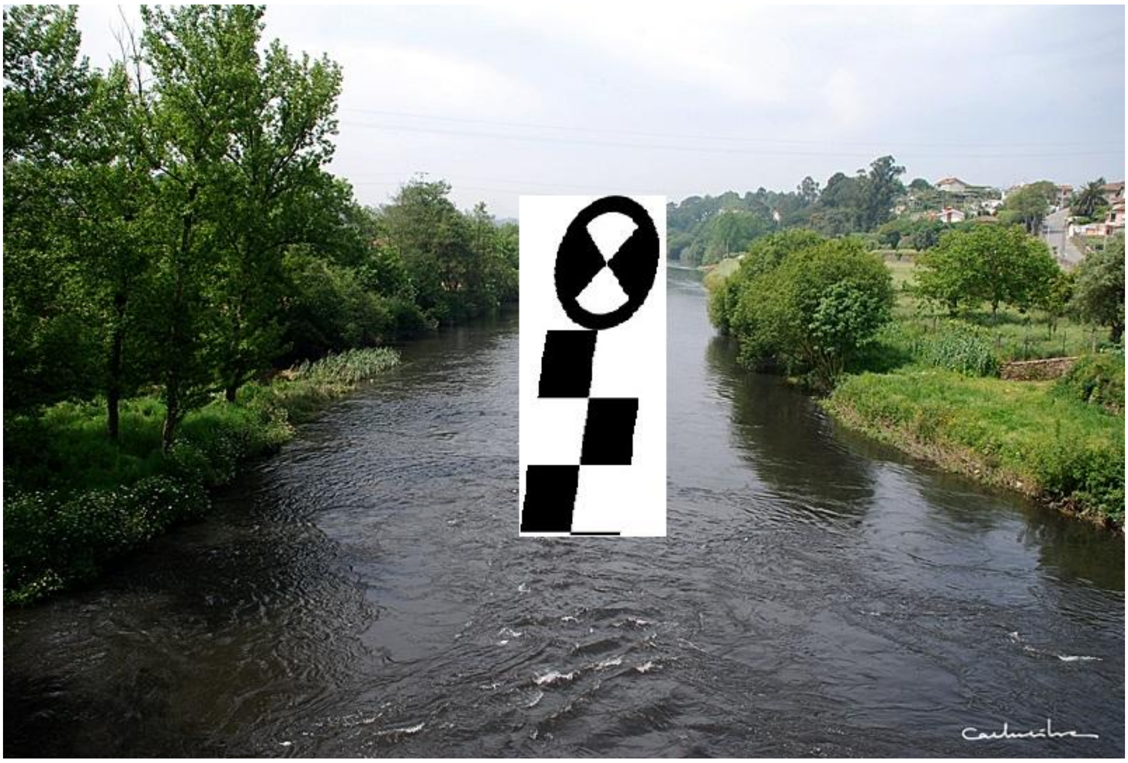
Test 03:



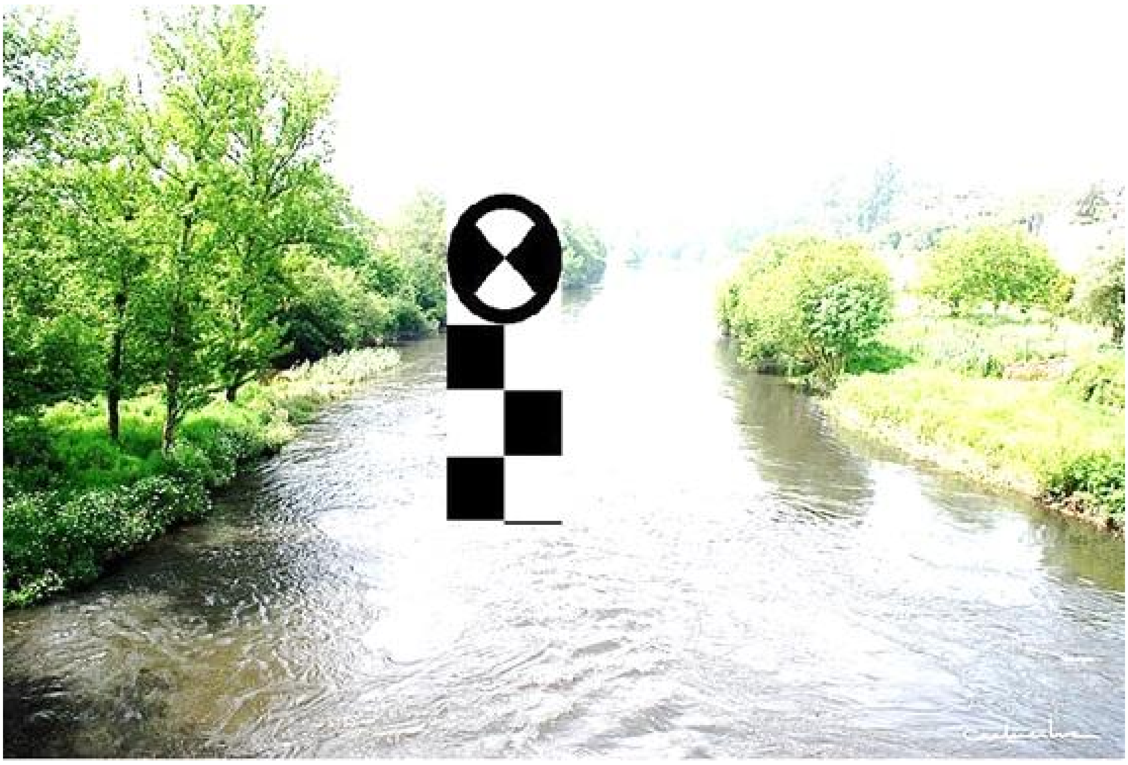
Test 04:



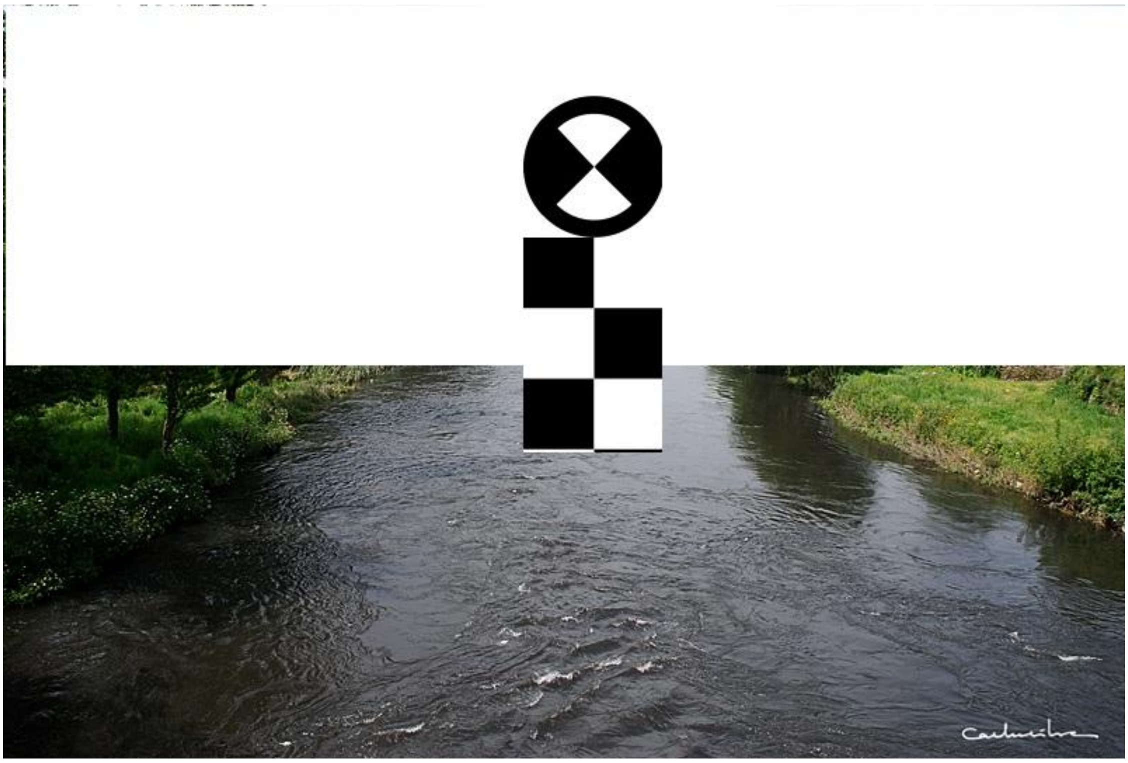
Test 05:



Test 06:



Test 07:



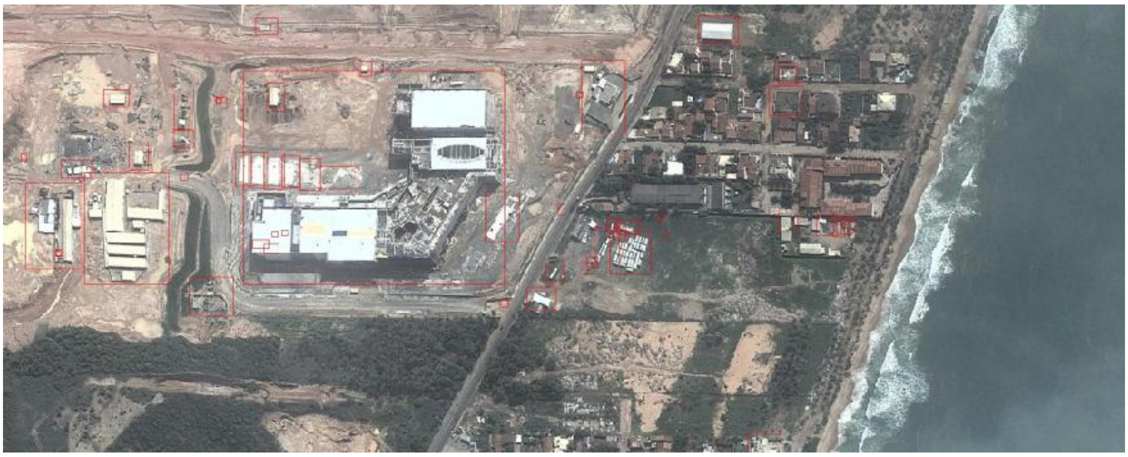
Test 08:

Test 09:

Test 10:

For depth analysis, the algorithm proved to be effective in detecting the water level in 95% of the vertical images tested. For aerial photographs, including from drones, the algorithm proved ineffective for water level calculation, and thus the attention of the component shifted to this method.

Examples of aerial pictures used for water level calculation:

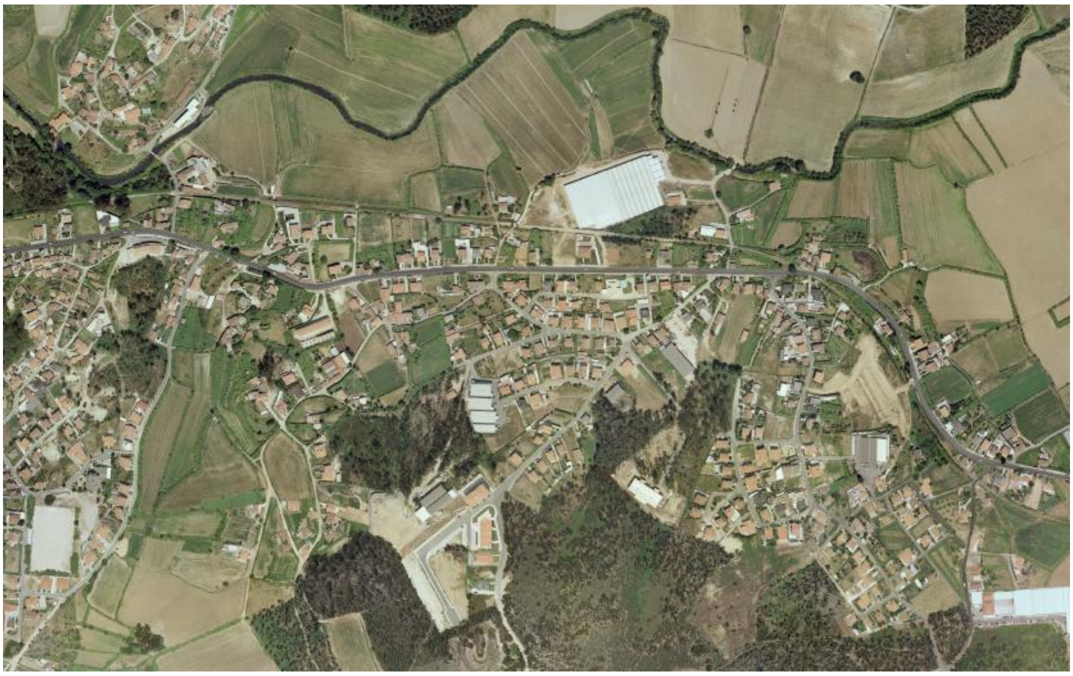


Regarding the method for asserting differences (new buildings for example) in urban areas, the algorithm proved effective:

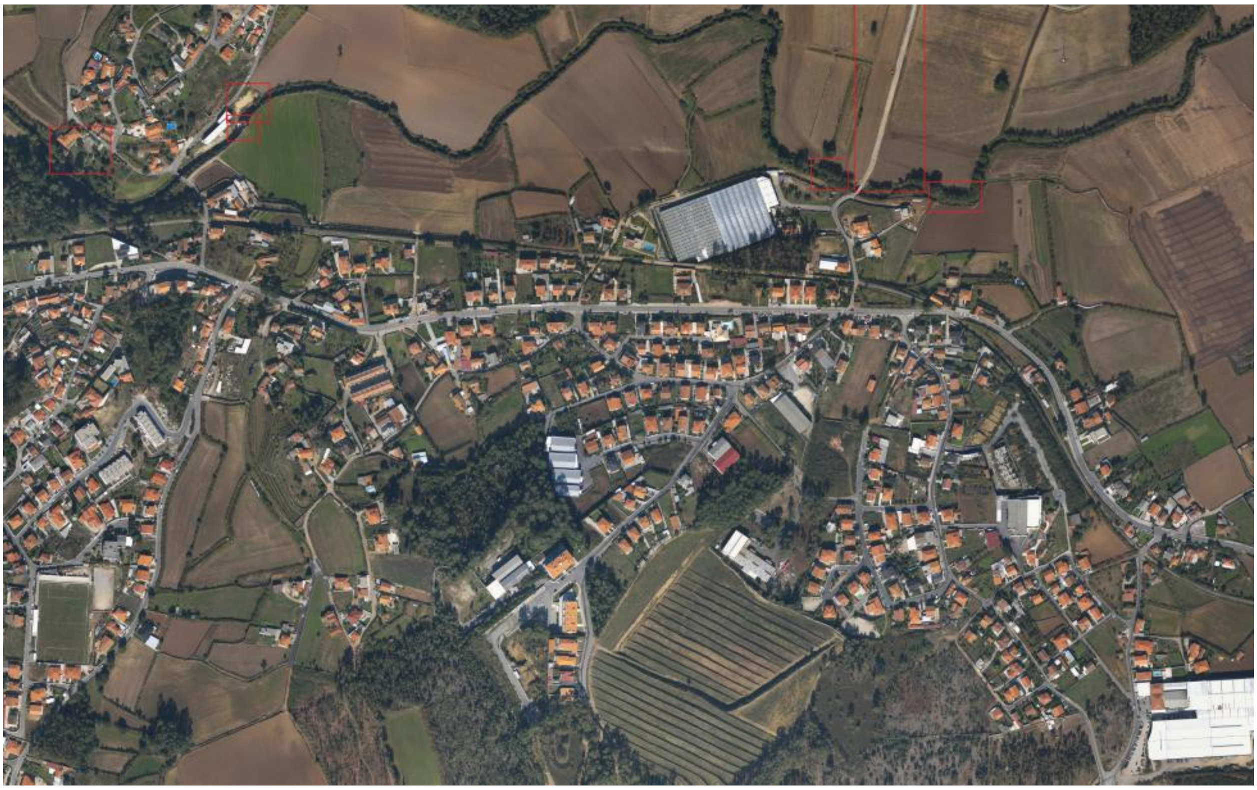
Test 1:

After:

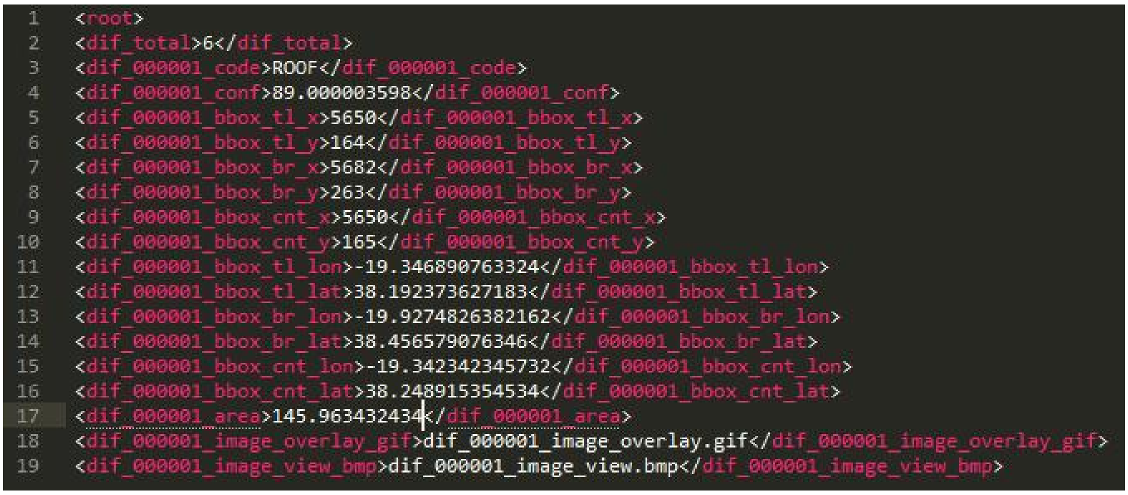
Before:



Result:

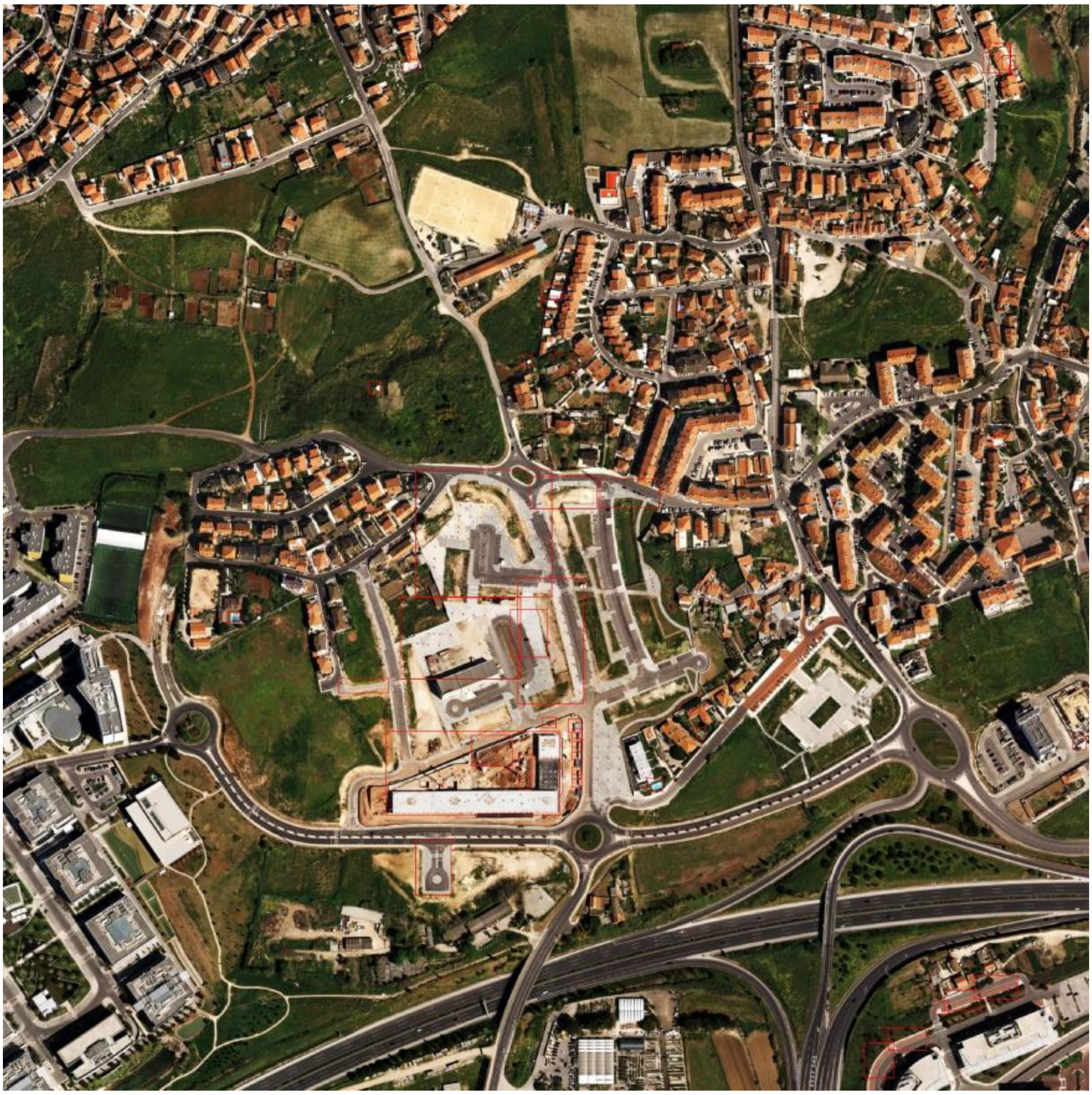


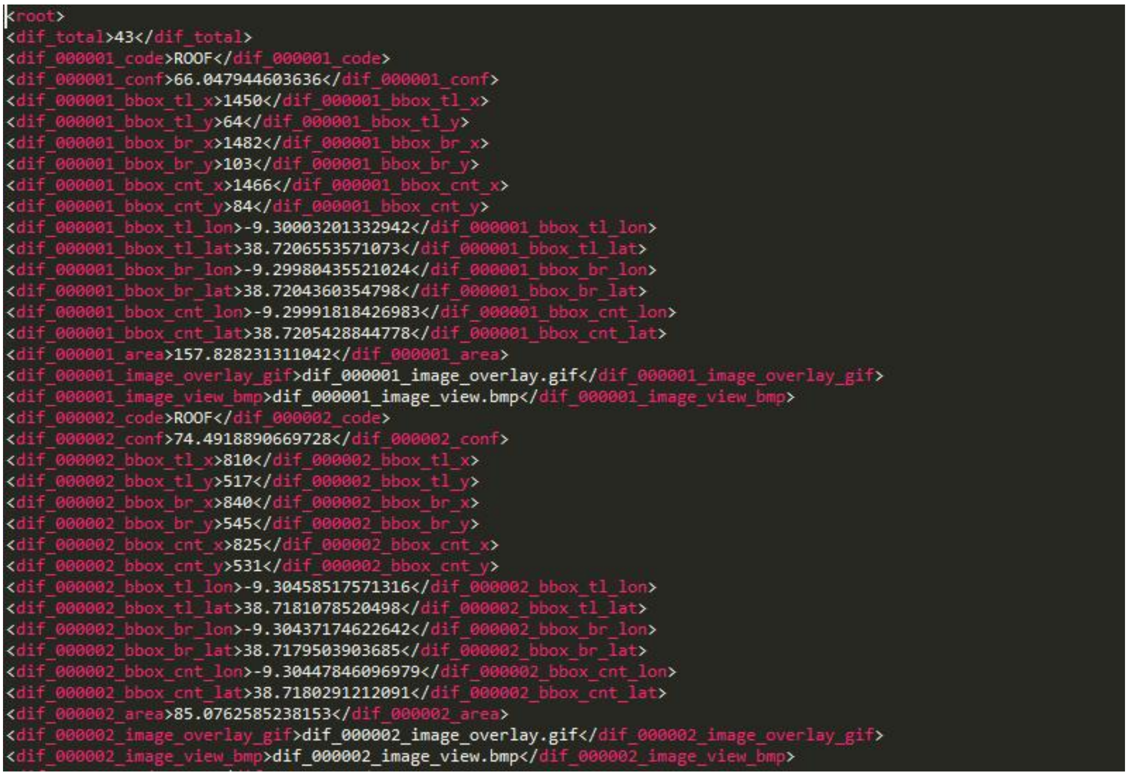
Report – this report is automatically downloaded via zip from the proof of concept UI or reported via the REST API and can then be integrated with EMC/Portal for cross check analysis:



```
<root>
<dif_total>6</dif_total>
<dif_000001_code>ROOF</dif_000001_code>
<dif_000001_conf>89.000003598</dif_000001_conf>
<dif_000001_bbox_tl_x>5650</dif_000001_bbox_tl_x>
<dif_000001_bbox_tl_y>164</dif_000001_bbox_tl_y>
<dif_000001_bbox_br_x>5682</dif_000001_bbox_br_x>
<dif_000001_bbox_br_y>263</dif_000001_bbox_br_y>
<dif_000001_bbox_cnt_x>5650</dif_000001_bbox_cnt_x>
<dif_000001_bbox_cnt_y>165</dif_000001_bbox_cnt_y>
<dif_000001_bbox_tl_lon>-19.346890763324</dif_000001_bbox_tl_lon>
<dif_000001_bbox_tl_lat>38.192373627183</dif_000001_bbox_tl_lat>
<dif_000001_bbox_br_lon>-19.9274826382162</dif_000001_bbox_br_lon>
<dif_000001_bbox_br_lat>38.456579076346</dif_000001_bbox_br_lat>
<dif_000001_bbox_cnt_lon>-19.342342345732</dif_000001_bbox_cnt_lon>
<dif_000001_bbox_cnt_lat>38.248915354534</dif_000001_bbox_cnt_lat>
<dif_000001_area>145.963432434</dif_000001_area>
<dif_000001_image_overlay_gif>dif_000001_image_overlay.gif</dif_000001_image_overlay_gif>
<dif_000001_image_view_bmp>dif_000001_image_view.bmp</dif_000001_image_view_bmp>
```

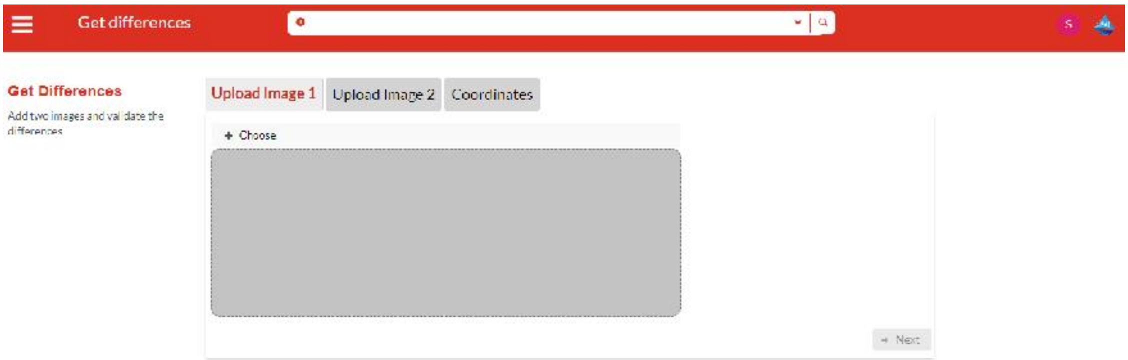
Test 02:





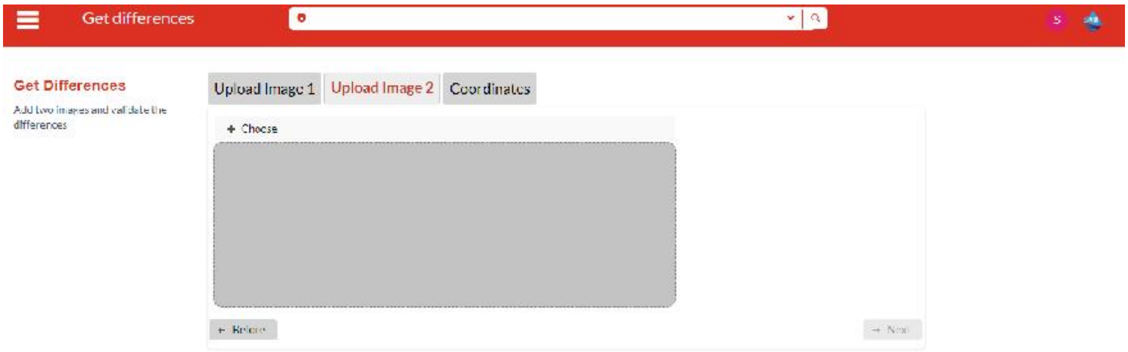
```
<root>
<dif_total>43</dif_total>
<dif_000001_code>ROOF</dif_000001_code>
<dif_000001_conf>66.047944603636</dif_000001_conf>
<dif_000001_bbox_tl_x>1450</dif_000001_bbox_tl_x>
<dif_000001_bbox_tl_y>64</dif_000001_bbox_tl_y>
<dif_000001_bbox_br_x>1482</dif_000001_bbox_br_x>
<dif_000001_bbox_br_y>103</dif_000001_bbox_br_y>
<dif_000001_bbox_cnt_x>1466</dif_000001_bbox_cnt_x>
<dif_000001_bbox_cnt_y>84</dif_000001_bbox_cnt_y>
<dif_000001_bbox_tl_lon>-9.30003201332942</dif_000001_bbox_tl_lon>
<dif_000001_bbox_tl_lat>38.7206553571073</dif_000001_bbox_tl_lat>
<dif_000001_bbox_br_lon>-9.29980435521024</dif_000001_bbox_br_lon>
<dif_000001_bbox_br_lat>38.7204360354798</dif_000001_bbox_br_lat>
<dif_000001_bbox_cnt_lon>-9.29991818426983</dif_000001_bbox_cnt_lon>
<dif_000001_bbox_cnt_lat>38.7205428844778</dif_000001_bbox_cnt_lat>
<dif_000001_area>157.828231311042</dif_000001_area>
<dif_000001_image_overlay_gif>dif_000001_image_overlay.gif</dif_000001_image_overlay_gif>
<dif_000001_image_view_bmp>dif_000001_image_view.bmp</dif_000001_image_view_bmp>
<dif_000002_code>ROOF</dif_000002_code>
<dif_000002_conf>74.4918890669728</dif_000002_conf>
<dif_000002_bbox_tl_x>810</dif_000002_bbox_tl_x>
<dif_000002_bbox_tl_y>517</dif_000002_bbox_tl_y>
<dif_000002_bbox_br_x>840</dif_000002_bbox_br_x>
<dif_000002_bbox_br_y>545</dif_000002_bbox_br_y>
<dif_000002_bbox_cnt_x>825</dif_000002_bbox_cnt_x>
<dif_000002_bbox_cnt_y>531</dif_000002_bbox_cnt_y>
<dif_000002_bbox_tl_lon>-9.30458517571316</dif_000002_bbox_tl_lon>
<dif_000002_bbox_tl_lat>38.7181078520498</dif_000002_bbox_tl_lat>
<dif_000002_bbox_br_lon>-9.30437174622642</dif_000002_bbox_br_lon>
<dif_000002_bbox_br_lat>38.7179503903685</dif_000002_bbox_br_lat>
<dif_000002_bbox_cnt_lon>-9.30447846096979</dif_000002_bbox_cnt_lon>
<dif_000002_bbox_cnt_lat>38.7180291212091</dif_000002_bbox_cnt_lat>
<dif_000002_area>85.0762585238153</dif_000002_area>
<dif_000002_image_overlay_gif>dif_000002_image_overlay.gif</dif_000002_image_overlay_gif>
<dif_000002_image_view_bmp>dif_000002_image_view.bmp</dif_000002_image_view_bmp>
```

To access this method via the proof of concept UI:



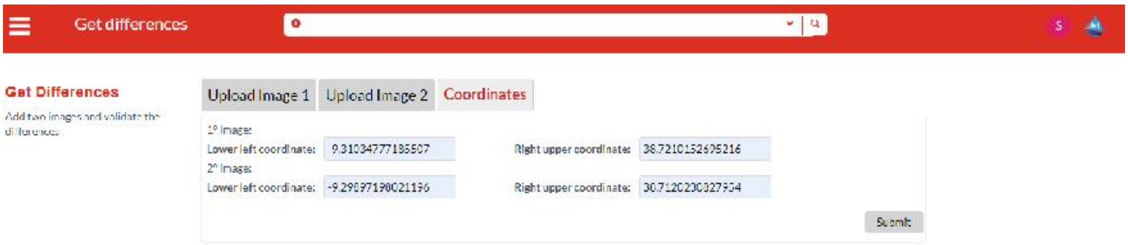
Get Differences

Add two images and validate the differences



Get Differences

Add two images and validate the differences



Get Differences

Add two images and validate the differences

1º Image:

Lower left coordinate: 9.31034777185507 Right upper coordinate: 38.7210152695216

2º Image:

Lower left coordinate: -9.29897190021196 Right upper coordinate: 30.7120230027954

The method can, again, be accessed via the REST API.

8 TMS Release Notes

This section contains the TMS release notes.

8.1 System access requirements

The TMS is a technical oriented component to be used by the central platform and the EMC for analysis of remote areas and is available on REST Api and File based operation. A proof of concept UI as developed within CDF for testing purposes.

8.2 Features

The TMS component is used to:

- Detected water level in vertical images;
- Originally designed to:
  - Detected changes on the terrain using a pair of images of different period in time;
  - Report those changes on a consolidated report with GEO Coordinates;

8.3 Installation guide

For Decision Makers and/or Operators you can access to the TMS directly in the FLOOD-Serv Platform or directly by typing in:

https://bilbao-floodserv-saas.ano.pt/

https://bratislava-floodserv-saas.ano.pt/

https://genova-floodserv-saas.ano.pt/

https://tulcea-floodserv-saas.ano.pt/

https://vnfamalicao-floodserv-saas.ano.pt/

The credentials to access are:

- User: salmeida
- Password: 123

Note: Since the PA can delete users, this user can at any point be deleted by the PA. If so, please refer directly to the FLOOD-Serv platform to gain access.

Choose TMS >> Calculate Depth on the main menu (top left corner).

The legacy “Calculate Differences” method is also available, via the same option path.

9 Work Developed and Conclusions

Regarding the TMS component, the 4 user stories were implemented, with the already presented alterations: Originally, the user stories to be implement in the TMS, were based on a single main method, that focused on calculation differences between two images. Development and testing proved that this method, as the bellow results show, proved to be below our expected threshold. For this, a second method, focused on vertical images and on calculating depth of bodies of water was implemented specially for the project and thus the user stories and consequent results adapted. The legacy method of “Calculation Differences” remains available. For this, the development consisted on (macro level):

- Implementing the dedicated web UI;
- Developing the Business Works layer to implement the new underline logic;
- Developing the Data layer to accommodate the new data structures.

The previous technology stack deemed fit for the changes and new modules incorporated.

Under the work developed in WP3, the TMS is ready to be integrated with the FLOOD-Serv system.

10 APPENDIX I: API Documentation

10.1.1 Introduction

The following chapters identify the methods present in the three main areas of the TMS API. For the URL, each pilot has its own TMS instance:

https://bilbao-floodserv-saas.ano.pt/

https://bratislava-floodserv-saas.ano.pt/

https://genova-floodserv-saas.ano.pt/

https://tulcea-floodserv-saas.ano.pt/

https://vnfamalicao-floodserv-saas.ano.pt/

For the API link, they obey the same logic:

https://{pilot_instance_name}/{pilot_contextroot_name}/services/api/records/

Pilot	{pilot_instance_name}	{pilot_contextroot_name}
Bilbao	bilbao	bilbao
Bratislava	bratislava	bratislava
Genova	genova	genova
Tulcea	tulcea	tulcea
Vila Nova de Famalicão	vnfamalicao	vnfamalicao

10.1.2 Depth Analysis

10.1.2.1 Get a list of previous analysis filtered by date of creation

Link	https:// {pilot_instance_name}-floodserv-saas.ano.pt/{pilot_contextroot_name}/services/api/floodserv/
Path	getDepthByDate/{date}
Method	GET
Produces	application/json

Parameters from headers	Authorization => “Basic VVNFUIdTOmZsb29kc2VydjEyMw==”
	token => the Oauth2 access token

Parameters from path	date => date in milliseconds
Return	[

	{ "id": < identifier>, "depth": <depth>, "creationDate": "<date created in milliseconds>", "processedBy": "<username>", }, …]

10.1.2.2 Create Analysis

Link	https:// {pilot_instance_name}-floodserv-saas.ano.pt/{pilot_contextroot_name}/services/api/floodserv/
Path	calculateDepth
Method	POST
Consumes	multipart/form-data
Produces	application/json

Parameters from headers	Authorization => “Basic VVNFUldTOmZ1dHVyZWRvYw==”
	token => the Oauth2 access token

Parameters from body	image => image

Return	{
	“newId”: “<internal ID of the created analysis >”,
	“depth”: “<calculated-depth>”
	}

11 APPENDIX I: User Guide

The user guide, in Powerpoint and Video format are available in each instance of the TMS of each pilot in the following links:

- https://bilbao-floodserv-saas.ano.pt/bilbao/images/TMS_Quick_guide.pptx
- https://bratislava-floodserv-saas.ano.pt/bratislava/images/TMS_Quick_guide.pptx
- https://genova-floodserv-saas.ano.pt/genova/images/TMS_Quick_guide.pptx
- https://tulcea-floodserv-saas.ano.pt/tulcea/images/TMS_Quick_guide.pptx
- https://vnfamalicao-floodserv-saas.ano.pt/vnfamalicao/images/TMS_Quick_guide.pptx