B.Sc. (Hons) in Software Development

Ollscoil Teicneolaíochta an Atlantaigh

Atlantic Technological University

Flood Detection System Using Google Earth Engine

By
Nelson Almeida

April 27, 2025

Minor Dissertation

Department of Computer Science & Applied Physics,
School of Science & Computing,
Atlantic Technological University (ATU), Galway.

GitHub:

`https://github.com/nelsondca/Flood-Detection-System-FYP`

Contents

List of Figures

List of Tables

1 Introduction

This project developed a satellite-based flood detection system that allows users to select locations and check the flood history using Sentinel-2 data, NDWI analysis, and Google Earth Engine (GEE). Starting with the Cork region as a reference case, the system automates satellite data collection, processes images to detect changes in water extent, and delivers the results in interactive web visualization. The project demonstrates the potential of using open-source geospatial software to provide low-cost flood risk information, overcoming disadvantages like cloud cover and dependence on ground sensor systems. The results were compared against official flood records.

1.1 Background

Floods are one of the most common natural disasters, affecting millions of people around the world every year. They cause damage to homes and businesses, and, most importantly, they represent a threat to our lives. According to the World Meteorological Organization, floods have caused more than $4.3 trillion in damages globally over the past 50 years, with a death toll of 2 million, with 90% in developing countries[14]. Ireland has experienced more regular floods due to more regular storms caused by climate change and increasing sea levels. Recent examples include the severe flooding in Cork in October 2023. Roads, houses, and businesses were flooded, and the damage was put at more than €10 million by The Irish Times reports[11]. Some parts of the city had no electricity or clean water, and emergency services were pushed to their limits. This event highlighted the need for enhanced systems of anticipating and monitoring flood danger.

When combining actual satellite data with interactive visualization, the system helps users understand whether an area has ever experienced floods. The initial focus was placed on the Cork region because it experiences frequent flood occurrences, but the solution was designed to scale, capable of analyzing flood history anywhere on the map.

Methods such as the Normalized Difference Water Index (NDWI), Sentinel-2 optical data, and Google Earth Engine (GEE) were used to detect and map the occurrence of water over time. The project illustrates how satellite-based technologies today can provide individuals and researchers with flood history information, without depending on expensive ground sensors or expert software.

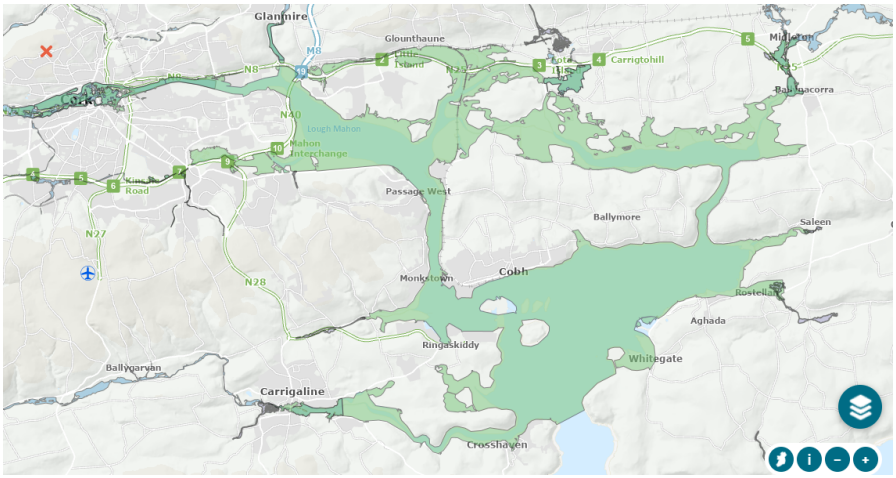


Figure 1: Map of Cork area (the red X marker indicates where the study area is used by Floodinfo.ie)

1.2 Problem Statement

Today, flood detection at an early and correct stage is a massive challenge. Most of the currently available flood warning systems are physical sensors placed in rivers or on flat ground. The systems are efficient in some areas, but cannot give a complete picture of what is happening in a vast area. While global institutions highlight the limitations of sensor-based flood detection,[13] European initiatives such as the Copernicus Emergency Management Service are exploring satellite-based alternatives to improve scalability and accessibility[2].

In addition, such systems are expensive to install and keep in use. In the majority of cases, they cannot detect floods due to unexpected weather patterns or where there are no sensors. Another issue is that most flood monitoring sensors are not accessible or user-friendly. They may require technical expertise or expensive software. This limits their use, especially for people in small towns or rural areas who are most exposed to flooding.

1.3 Project Goals

The goal of the project is to create a simple and reusable satellite image-based flood detection system using photographs of the Earth taken by satellites that pick out areas likely to be flooded. Instead of physical sensors, it looks at photos of the Earth taken by satellites. It uses a method called the Normalized Difference Water Index (NDWI) that points out where there is surface water. By looking at satellite images at different times, we can know that water levels have risen in

some areas, which could indicate there is a flood.

Figure 2: Example of (SAR) Sentinel-1 Data

The system is built from Sentinel-1 and Sentinel-2 satellite data operated by the European Space Agency. It is processed in Google Earth Engine (GEE), a cloud-computing platform that allows researchers to easily analyse satellite images. Using these resources, this project can cover large areas, delivering updated information without depending on local sensors.

1.4 Objectives

- To download and process satellite data for a targeted region using Google Earth Engine (GEE).
- To compute NDWI (Normalized Difference Water Index) to identify the water-covered areas.
- To calculate the NDWI (Normalized Difference Water Index) and detect changes over time.
- To construct a water mask that will trigger a flood and compare it with official data from GEE.
- Create a visual map that shows areas that are likely to be flooded.

- Allow users to select a location on the map, check the flooding history, and if the area is at risk of flooding, through a simple web interface.

1.5 Scope

The project's main goal is to use Sentinel-2 photos to detect floods in the Cork area after they happen. It includes image preprocessing, using NDWI, an index that helps identify water bodies (rivers, lakes, flooded areas) using satellite images, and compares the results with valid data from Google Earth Engine.

1.6 Motivation

The idea behind this project came from discussions with my supervisor during the brainstorming phase. Some ideas were discussed before concluding that a flood-detection system would not only be relevant in today's climate context but also provide a challenging and rewarding learning experience.

With climate change intensifying the risks of flooding, it is becoming more and more important to develop scalable and data-driven approaches to monitor and help identify these events. Using satellite data and open-source tools offers an innovative way to address this issue affordably and at scale. This topic was very interesting since it provided an opportunity to engage with unfamiliar technologies. Engaging with unknown technologies and frameworks was a deliberate choice that coincided with some personal and professional goals of increasing technical skills by being exposed to the unknown and new challenges.

1.7 Structure

The rest of this dissertation is organized in the following structure:

- Chapter 2: Technology Review discusses existing flood detection methods, relevant technology, and remote sensing techniques.
- Chapter 3: Methodology shows how data was collected, processed, and analyzed to detect floods.
- Chapter 4: System Design describes the technological architecture and data processing pipeline.
- Chapter 5: Implementation & Testing describes the tools and technologies used and presents the testing results
- Chapter 6: Evaluation evaluates the system's performance and accuracy

- Chapter 7: Conclusion and Future Work summarises achievements and considers potential improvements and extensions.

2 Technology Review

2.1 Flood Detection Approaches

Flood detection systems are typically divided into two types: manual and automated. Manual systems rely on data from physical sensors, e.g., river gauges and rain gauge stations. While they are accurate in specific areas, they are expensive to install and maintain and have limited geographical coverage[1]. Automated systems, particularly those based on remote sensing technology, have greater geographic coverage as well as being often less expensive.

Satellite remote sensing technologies have advanced significantly over the past few years, boosting the ability to monitor floods. Synthetic Aperture Radar (SAR) is one of the leading technologies employed in this context, with Sentinel-1 satellites being at its core. SAR provides a vital advantage in the sense that it can photograph the Earth's surface regardless of weather or light conditions. This is very valuable during times of harsh weather and in flood-prone regions. Numerous studies have established the potential of SAR to create reliable flood extent maps that provide the foundation for producing emergency response and well-informed disaster management. The combination of SAR data and machine learning has also enabled accurate identification and delimitation of flooded regions, which this study embraced as well. The system utilizes SAR images to remove the constraints of previous methods and offer a scalable, quick, and reliable flood detection solution.

Criteria	**Manual Systems**	**Automated Systems**
Cost	High (installation and maintenance costs[7]).	Lower costs[9].
Geographic Coverage	Limited to Sensor Locations[7].	Wide-area (regional/global)[9].
Data Latency	Delays in transmission[10].	Near real-time (especially with satellites[10]).
Weather Dependency	Not weather-dependent[12].	SAR: Not weather-dependent[9].
Data Type	Point-based sensor readings[7].	Raster-based satellite images[9].
Scalability	Difficult to scale[7].	Highly scalable[15].
Installation Complexity	Physically intensive[7].	Minimal physical infrastructure[9].

Table 1: Comparison of Manual and Automated Flood Detection Systems

2.2 Use of Satellite Data

Sentinel-2, Landsat-8, and Sentinel-1 satellites are mostly utilized for the monitoring of the environment due to their complementary sensing capabilities in capturing

Earth observation data. This paper utilizes both Sentinel-2 which provides high-resolution optical imagery with 13 bands of spectral range, including the visible and NIR, and Sentinel-1, which offers SAR data. Sentinel-2 monitors changes in vegetation and water bodies under clear conditions[4].

While Landsat-8 data was considered for use in this project, it was not part of final solution. After many trials was concluded that its lower temporal resolution and widespread cloud cover during September, October of 2023 limited the usability of the dataset compared to Sentinel-2.

While Sentinel-1 provides day-and-night, all-weather data capabilities that are critical for flood detection, especially in cloudy or low-light conditions[9]. The integration of optical and SAR data improves the accuracy and reliability of the system, allowing for more extensive flood monitoring under different climatic conditions.

2.3 Image Processing Techniques

A crucial technique in flood detection is the Normalized Difference Water Index (NDWI). NDWI highlits standing water clearly in images, it processes satellite data such as Sentinel-2 (with free high-resolution data) and it can be automated and scaled up for monitoring large territories unlike ground sensors. NDWI is calculated using the formula:

$$\text{NDWI} = \frac{B3 - B8}{B3 + B8}$$

Where:

- B3 = Green band (Sentinel-2)
- B8 = Near-Infrared (NIR) band (Sentinel-2)

NDWI highlights water bodies by differentiating the reflectance of green and NIR light. NDWI values closer to 1 indicate water, while values below 0 may represent dry land[6]. To objectively identify flooded areas, we calculated NDWI using Sentinel-2 satellite data for both pre-flood and post-flood time frames.

Below are 3 visualizations to illustrate this process in the Cork geographical area during the period of Storm Bebet. The first 2 images display NDWI values before and after the flood event. To the naked eye, the difference between these two images is almost null, though, on closer inspection, you may notice a slight increase in darker blue areas, suggesting a possible rise in the water bodies. However, these changes are not immediately obvious without deeper analysis.

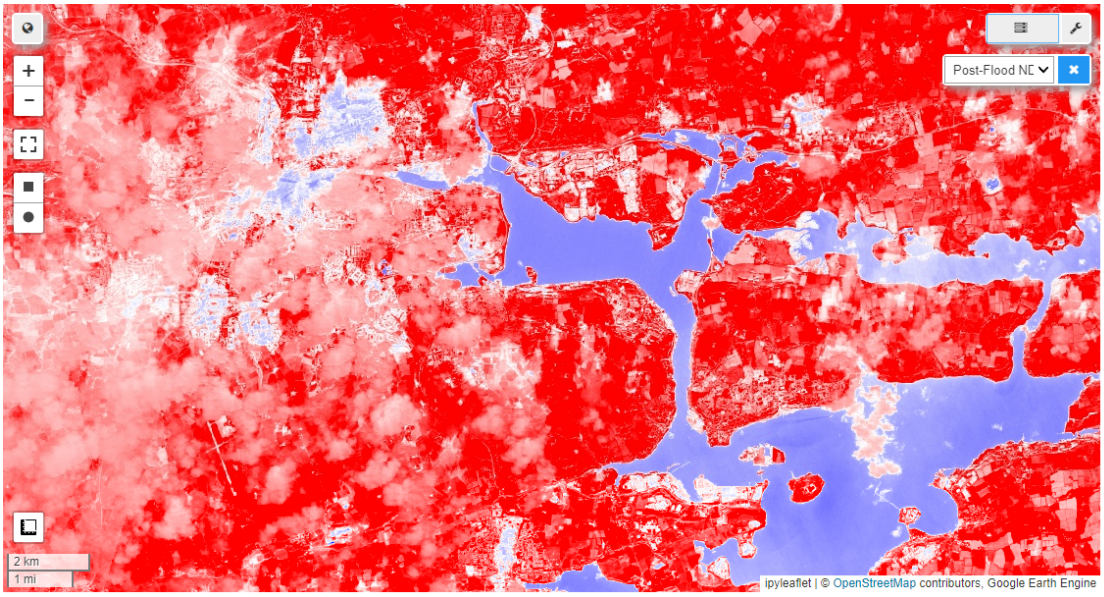


Figure 4: Post-Flood NDWI

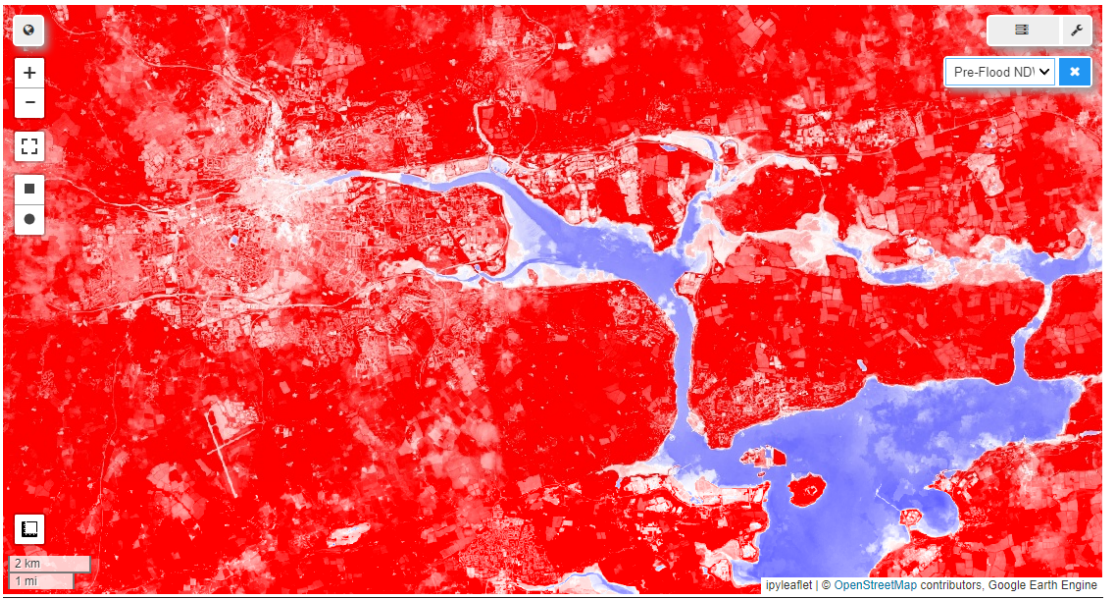


Figure 3: Pre-flood NDWI

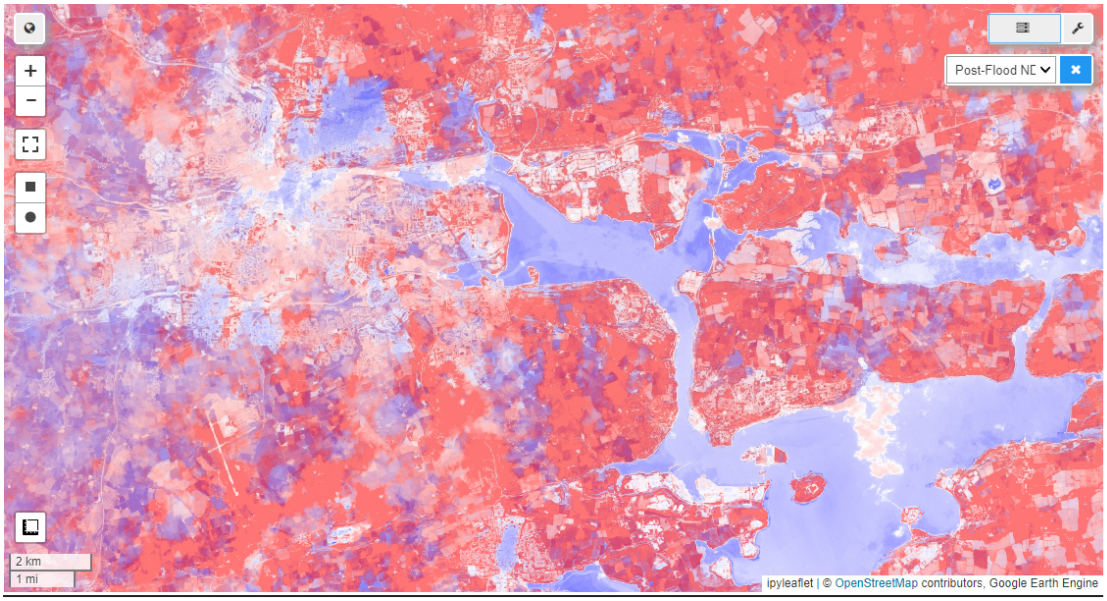


Figure 5: NDWI Difference

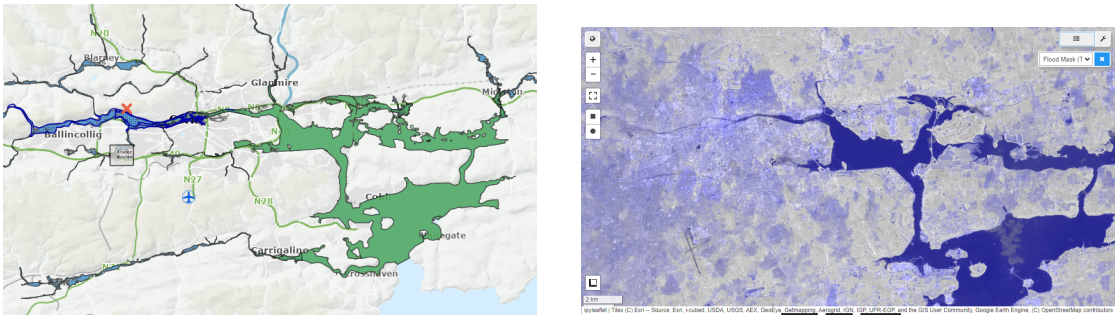
The pre-flood image, Figure 3, shows that there was water before the flood. You can see fewer blue areas, meaning there was less water overall. By comparing this to the post-flood image, Figure 4, we can figure out where the water appeared after the flooding. You can also see some clouds, and this same cloud might affect visibility in certain parts, but overall, the difference stands out. In the last image, Figure 5, it shows the amount of water after the flood. The blue areas have high NDWI values, meaning they likely contain water (lakes, rivers, or floodwater). The red areas have low NDWI, so they are probably dry land. **Note:** Due to cloud cover, you can spot some extra blue patches where is supposed to be dry l and. That is mainly due to a cloud coverage issue. If the cloud noise is not eliminated, some clouds will be visible and affect the results.

To overcome this and highlight flooded locations, the NDWI difference is calculated by subtracting the pre-flood NDWI from the post-flood NDWI. The resulting difference, Figure 5, clearly shows areas where levels have increased. In this visualization, the blue-shaded areas show a large increase in NDWI values, indicating the formation of new or expanded water bodies as a result of floods. In contrast, the red and neutral tones reflect locations where little to no change happened (dry land). This methodology gives a considerably more reliable and visual method of detecting floods, particularly in extensive or cloud-covered areas where manual inspection would be difficult or time-consuming.

2.4 Relevant Projects and Research

The European Space Agency (ESA) and NASA have shown through projects how useful satellite photography is for disaster monitoring. Google Earth Engine has been used in multiple academic works for tasks such as deforestation, tracking long periods of dryness, and urban growth analysis [**?**].

Floodinfo.ieis an Irish government website that provides hazard maps, historical flood data, and some other planning tools. In this project, it was used as a reference to validate the results obtained through satellite data analysis.



(a) Floodinfo.ie Data (b) NDWI calculation mask

Figure 6: Comparison of Floodinfo Data with NDWI detection

2.5 Technologies Used

This project combines several different powerful technologies to support and facilitate the development, analysis, and visualization of a flood detection system based on satellite data. Technologies were selected considering their open-source nature, versatility, and general participation of the community. Below is an overview of the main tools used:

- Google Earth Engine (GEE): A cloud-based geospatial platform is crucial for processing data in this project. An essential component to detect floods, generate confusion matrices, calculate Normalised Difference Water Index (NDWI), get satellite data, and evaluate detection performance.GEE also enabled tile layer creation to be visualized.
- Python: The main programming language used during the project development for data processing, system logic, and back-end development.
- geemap: A library for constructing interactive maps on top of GEE and Folium. Utilized to display spatial data interactively within the Jupyter Notebook and for exporting the results to the web interface.

- matplotlib: Added to the project for data visualization. It is used in generating plots, graphs, and graphical displays of evaluation measures such as accuracy, precision, and recall.
- rasterio: A Python library that can read and write geospatial raster data sets. Rasterio is built on top of GDAL (Geospatial Data Abstraction Library) and is particularly useful for working with GeoTIFF files and other raster formats within and outside the GEE environment. It makes future enhancement possible in this project, for instance, local processing of satellite data, or utilization of external elevation or land cover information. Rasterio also supports direct access to raster metadata, CRS, and pixel-level operation, which are necessary for high-precision geospatial processing.
- Jupyter Notebook: Served as the development and experimentation environment for prototyping and validating the flood detection logic. It was essential during the research and development phase, even if the finished system was deployed outside this environment.
- Leaflet.js: A lightweight JavaScript library used in the web interface to display spatial data on interactive maps. Leaflet offers a dynamic exploration of flood detection results in a browser where end-users can visualize data in an intuitive and user-friendly interactive way.
- GitHub: Used for version control, collaboration, and documentation. All code, notebooks, and visual data are available in the main repository.

  These tools work perfectly together to provide a complete flood detection system, including web-based visualization, evaluation, and processing of satellite data.

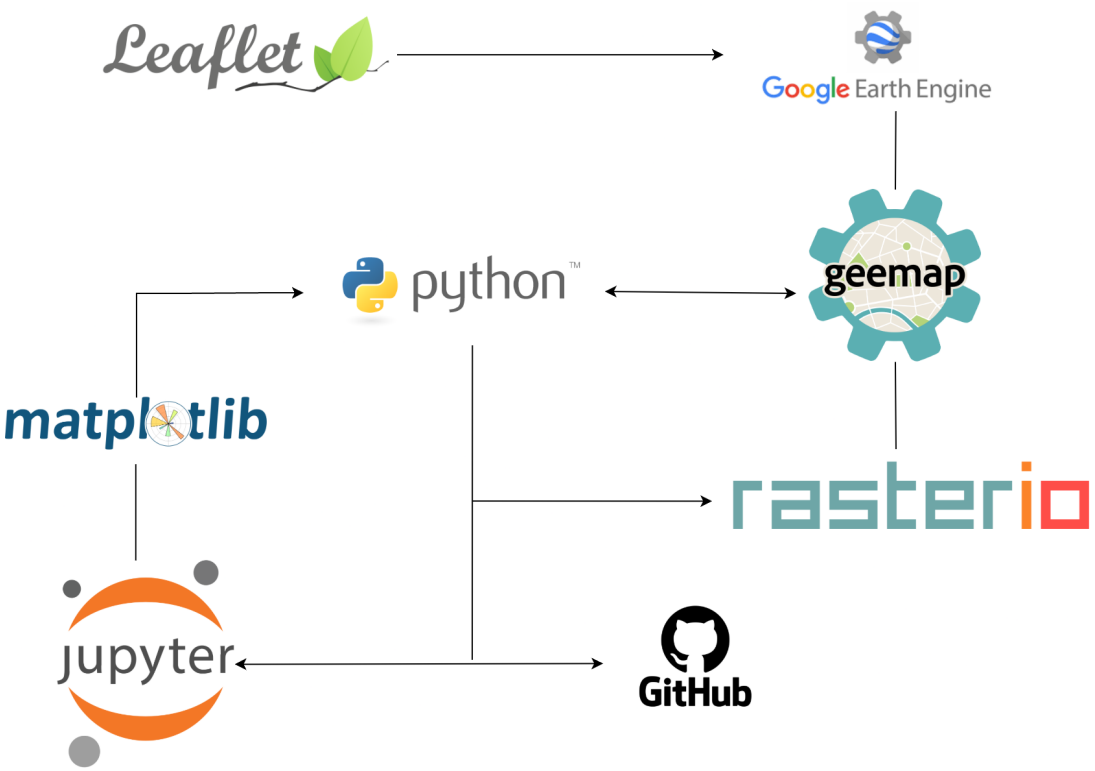


Figure 7: Technologies Used Diagram

3 Methodology

3.1 Project Methodology

The project was executed in a hybrid model combining software development as well as research. Software development was executed in an agile framework where different tasks were completed in iterative cycles. Each cycle solved certain stages, such as data collection, preprocessing, construction of the algorithm, and visualization. The adaptive component of this methodology allowed for continuous refinement of thresholds, experimentation with more datasets, and integration of visualization tools on the fly. From a research perspective, the method was experimental and exploratory. The challenge was to identify suitable sources of satellite data, to investigate why and how suitable the data was, to develop data processing algorithms, and to test findings, comparing them with official data.

The goal was to be completly sure flood zones could be detected with enough accuracy based on available satellite data and open-source software.

The primary data types used in Google Earth Engine (GEE) were images and features. Images were utilized to represent the raster satellite data (Sentinel-1), while features (polygons) defined the area of interest in this specific case, Cork, Ireland.

The Sentinel-1 SR dataset was used to compare with validation data since it provides high-resolution optical images that allow for easier detection of water level changes when they are small. It accommodates a spatial resolution of up to 10 meters for the critical bands (e.g., green and NIR), which is ideal for utilizing the NDWI in flood detection.

3.2 Data Collection

Region and Timeframe

The primary area of interest was Cork, Ireland, due to its history of severe flooding. The period is chosen during the storm Bebet when the flooding happened in 2023 during the months: September and October 2023, with a broader date range used during experimentation.

- Area of Interest (AOI) using geographic coordinates.
- Date range: Pre-flood and post-flood date range.

```
pre_flood_start = '2023-09-01'
pre_flood_end = '2023-09-30'
post_flood_start = '2023-10-01'
```

```
post_flood_end = '2023-10-31'
```

- Cloud cover: images with $< 50\%$ cloudiness were prioritized.
- **Floodinfo.ie[8]:** Used to collect and analyze true information about the flood event.

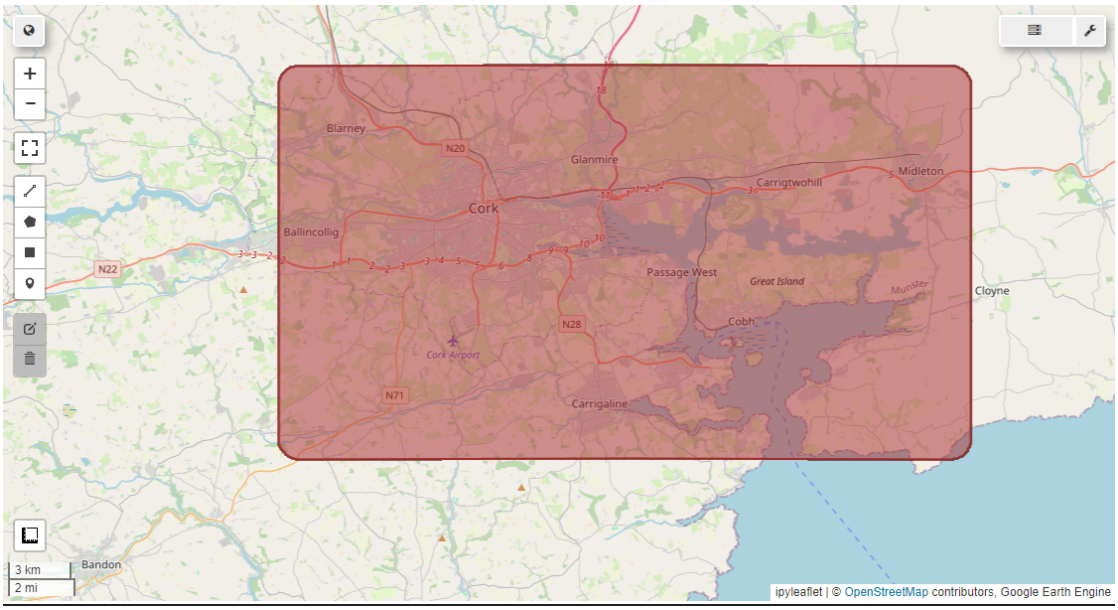


Figure 8: Cork region selected (AOI)

3.3 How Technology Were Used

Google Earth Engine (GEE): Used for collecting satellite Data, calculating NDWI, creating tile layers, and confusion matrices, and evaluating detection performance. Enabled high-resolution, scalable, and efficient image analysis directly in the cloud.

Python: This was mainly used for backend logic, algorithms for flood detection, and integrating tasks. Python with a very extensive active community and available documentation, offering automation, image processing, and many other resources.

Geemap: It was used for dynamic visualization of satellite images, visualization of NDWI results, and exporting visual outputs for the web interface.

Matplotlib: It was crucial in plotting evaluation metrics such as accuracy, precision, and recall, also displaying various types of data analysis results.

Rasterio: Facilitated geospatial data manipulation, which is handy when dealing with high-precision applications with features like terrain analysis or land cover integration.

Jupyter Notebook: This was where most of the hands-on work was done. Writing the code, running experiments, and visualizing the results. It was the main environment for testing ideas, refining the detection logic, and keeping track of everything learned during the development process.

Leaflet.js: Leaflet made it possible to visualize the results in a user-friendly experience in the browser.

GitHub: All scripts, files, images, data in general, and documentation were kept.

3.4 Preprocessing Steps

Before NDWI could be applied, multiple pre-processing steps were necessary:

- Clipping the region: Focused data on the Cork region.

```
lat_north = 51.95
lat_south = 51.80
lon_west = -8.60
lon_east = -8.15

buffer_size = 1000

# Creating the polygon using latitude/longitude
aoi = ee.Geometry.Polygon([
    [
        [lon_west, lat_north],
        [lon_west, lat_south],
        [lon_east, lat_south],
        [lon_east, lat_north]
    ]
]).buffer(buffer_size)  # Applying the buffer

# Initializing the map
Map = geemap.Map(center=[51.8691, -8.82646], zoom=10)
# Adding a layer
to the map
Map.addLayer(aoi, {'color': 'brown'}, 'Area of Interest'
   )
# Displaying the map tool
```

```
Map
```

- Cloud filtering: Removed highly cloudy images to reduce noise.
- Median Composite: Used `.median()` in GEE to smooth over anomalies in the data.
- NDWI Band Selection: Sentinel-2's Band 3 (Green) and Band 8 (NIR) were selected.

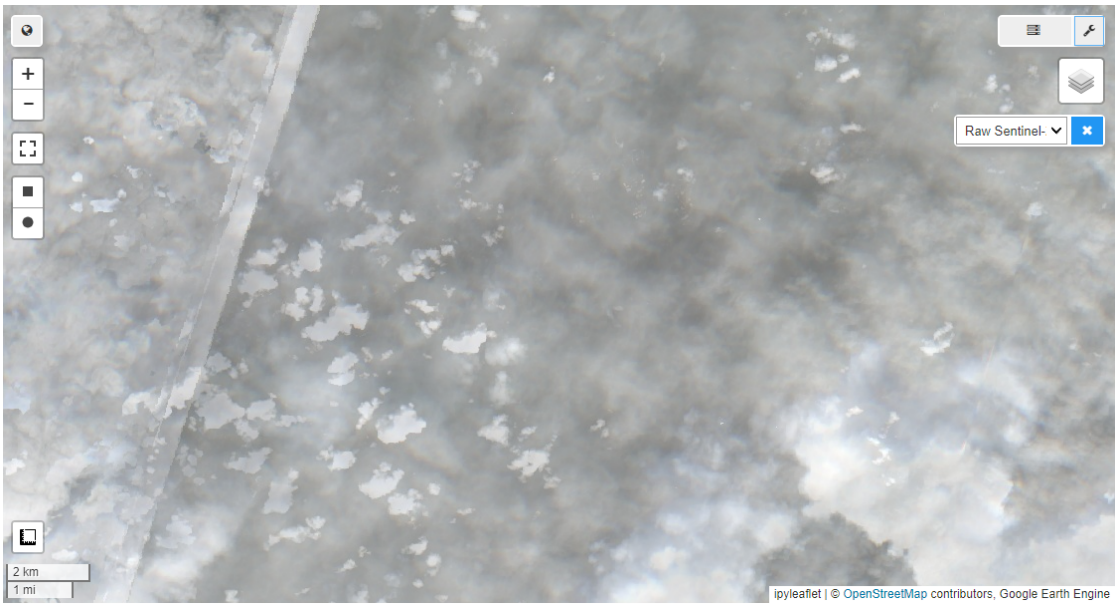


Figure 9: Raw Sentinel-2

Before flood detection could be performed, Sentinel-2 images were filtered to remove cloud-covered pixels. Raw images, particularly over Ireland, often suffer from extensive cloud noise. By applying a cloud probability mask (`CLOUD_PIXEL_PERCENTAGE` $<$ 30%), cleaner results were obtained, ensuring that the water body detection (using NDWI) was strong and reliable, Figure 10.

Figure 10: Cloud-mask

3.5 Flood Detection Logic

The flood detection logic for this project was primarily done using Sentinel-1 SAR (Synthetic Aperture Radar (SAR) images. By comparing the values of backscatter before and after the flood incident, flooded regions were identified using changes in reflectance at the surface.

Sentinel-2 data were used to calculate the Normalized Difference Water Index (NDWI) that was used merely as a validation reference.

The NDWI algorithm was applied using this formula:

$$\text{NDWI} = \frac{\text{Green} - \text{NIR}}{\text{Green} + \text{NIR}}$$

Where:

- Green corresponds to Band 3 in Sentinel-2 pictures
- NIR corresponds to Band 8 in the Sentinel-2 picture

This index produces values between -1 and 1. Higher values generally indicate the presence of water bodies, and lower values indicate vegetation or dry land.

Detection Workflow:

Step 1: NDWI Calculation per Pixel

The NDWI equation is applied to all pixels of pre-flood and post-flood Sentinel-2 images. This results in two NDWI layers that reflect water content before and after the flooding event. By using a median composite over days, the algorithm reduces clouds and temporal noise (humidity, air particles, etc)

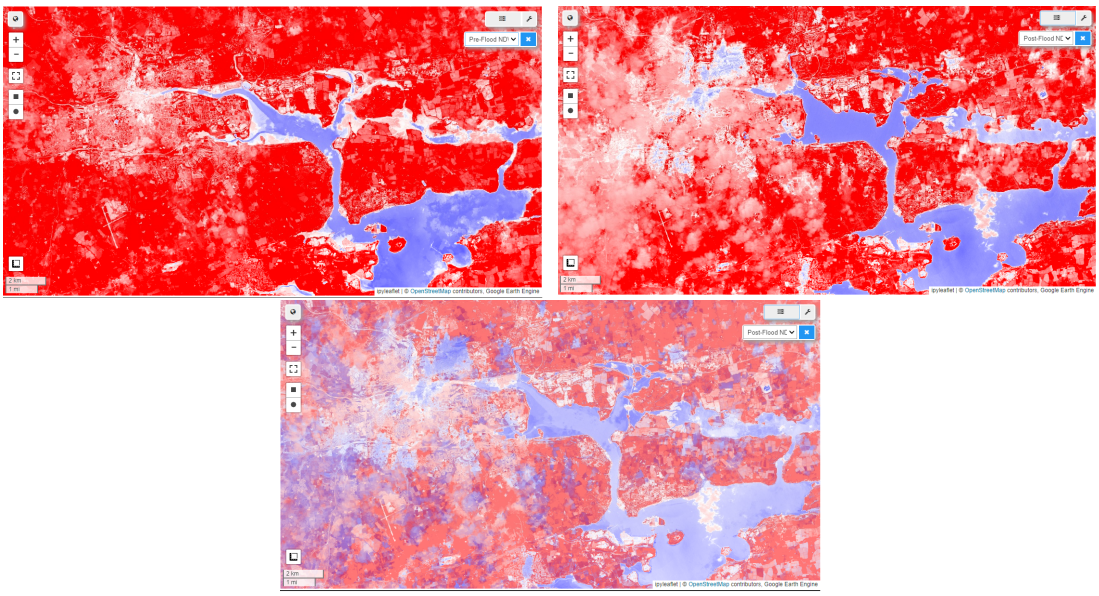


Figure 11: Comparison between NDWI pre-flood,post-flood, and difference layers

Step 2: Water Presence Binary Thresholding For the separation of the probably flooded regions, a binary mask is derived from NDWI outputs. Pixels with NDW > 0.3 are marked as water. The Threshold value was derived by examination of the values applied in previous flood detection studies and visual assessment of the generated maps. The binary output marks only regions where water cover is significantly larger than normal. For instance, if a pixel in the Green band is 0.34 and the NIR value is 0.17:

$$\mathbf{NDWI} = \frac{0.34 - 0.17}{0.34 + 0.17} = \frac{0.17}{0.51} \approx 0.33$$

Since this value exceeds the threshold level of 0.3, this pixel would be classified as water.

Step 3: Geographic Filtering and Clipping The binary water mask is clipped onto the AOI (selected area from initial choice - Cork, Ireland) and delineated. This

way, the analysis spatially focuses analysis and computes it effectively and ignores irrelevant data outside of the specified region.

Step 4: Export of Data to Local Machine and Google Drive and Visualization of Results NDWI layers and binary water masks are output as color-coded maps (water is blue, change or non-water is red). The layers are examined in an interactive Earth Engine map and exported/saved to Google Drive in .tif format for later analysis. Exported images are used to examine pre- and post-flood scenarios and aid flood impact studies.

Alternative thresholds (e.g., 1.5, 2, 2.5, 3) were tested to optimize detection sensitivity.

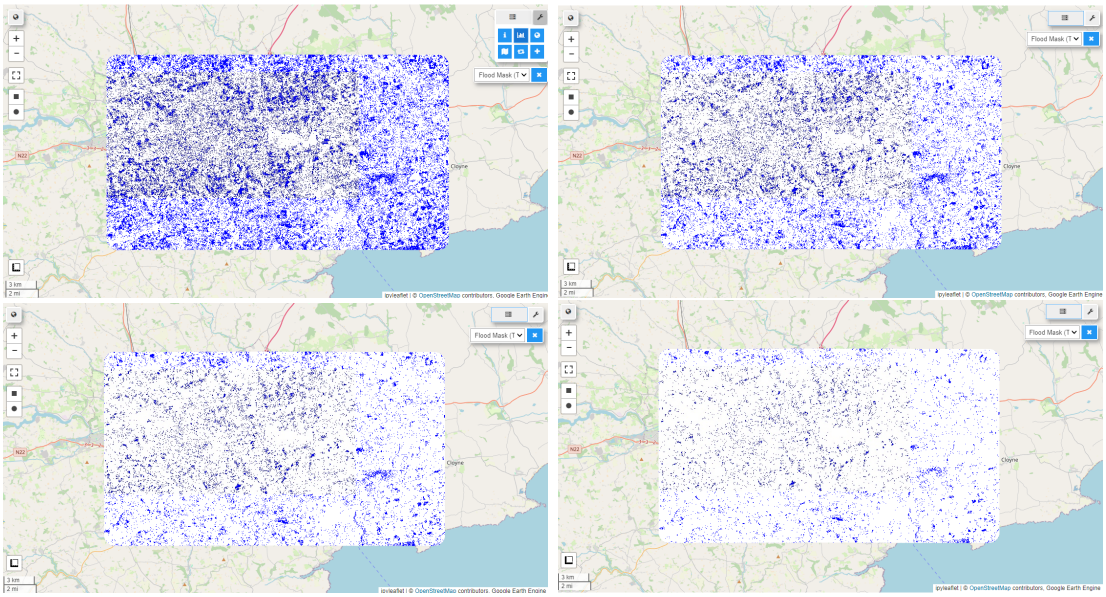


Figure 12: Treshdolds - 1.5, 2, 2.5, 3

3.6 Data Sources and Validation

Different approaches were followed for flood detection and validation.

Flood detection uses Sentinel-1 Synthetic Aperture Radar (SAR) images, which provide cloud-penetrating radar backscatter information. Backscatter intensity variations were used to calculate a flood mask by identifying areas of significant water-induced signal loss.

Two reference datasets were used in validation:

- Sentinel-2 Optical Imagery: Normalized Difference Water Index (NDWI) was approximated from Sentinel-2 imagery to approximate flood water boundaries. Pixels beyond an optimal threshold (calculated using the Otsu method) were labeled as flooded.
- JRC Global Surface Water Dataset: The JRC YearlyHistory dataset was used for the detection of permanent water bodies (rivers, lakes). This allowed validation of whether detected flood areas matched with known permanent water to separate actual flood events from permanent water bodies.

Merging Sentinel-1 SAR detection with cross-validation against both dynamic (NDWI) and static (JRC Water) references provided a strong foundation to test the flood detection capability.

4 System Design

4.1 Architecture

The system design process was crucial in structuring the application's structure, ensuring all the technologies functioned well together and efficiently to meet the requirements. Flood identification was done using Sentinel-1 SAR data by calculating the backscatter difference between pre-flood and post-flood data collection. Optical Sentinel-2 NDWI layers were implemented and used mainly for validation and secondary confirmation due to the cloud cover problems.

The Design follows a layered model, designed for modularity and scalability.

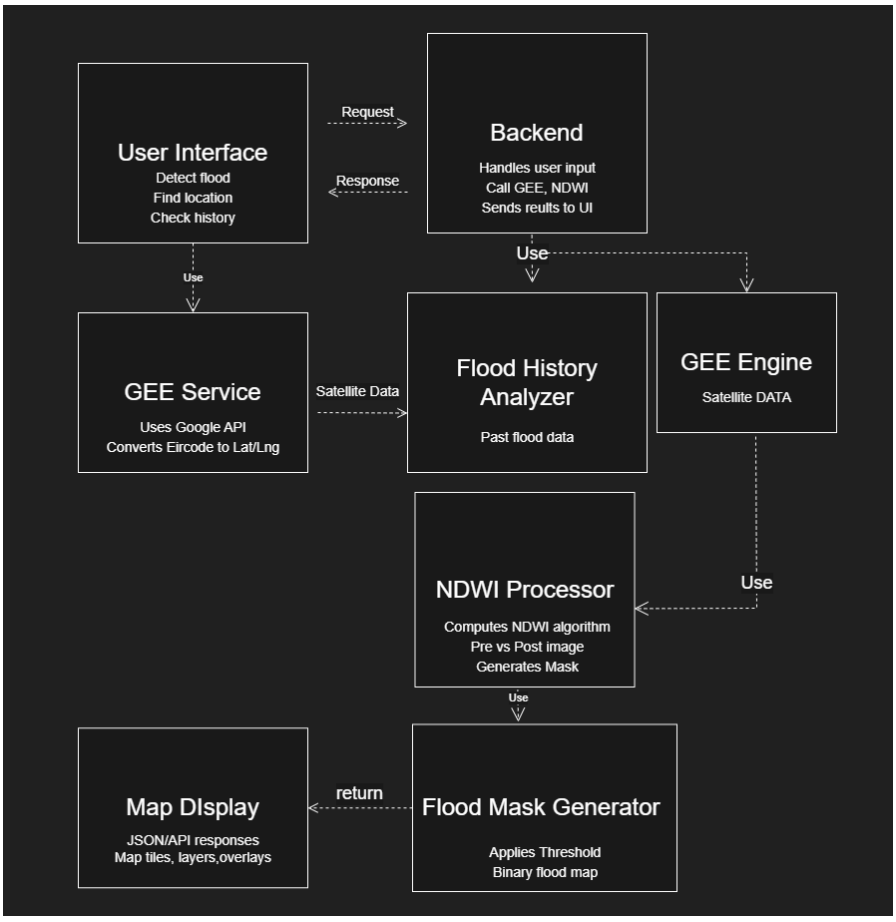


Figure 13: System Architecture UML Diagram

- **GEE Engine**
  - **Role:** The Gee Engine allows access to satellite images using Google

Earth Engine. It is the main source of environmental data required for the flood detection system.

  - **Integration:** It interfaces with GEE Service, which sends requests for specific satellite data based on the user's location and time range.

- **GEE Service**
  - **Role:** The responsibility of this module is to interact with Google APIs. It converts user input like addresses or Eircodes into geographic coordinates and initiates satellite data queries to the GEE Engine.
  - **Integration:** It integrates with the User Interface layer of the app, receiving user input and presenting geospatial data to the processing pipeline.
- **Flood History Analyzer**
  - **Role:** Retrieves and analyzes historical flood records to provide a contextual understanding of areas that are likely to be flooded.
  - **Integration:** Works in conjunction with the Processing Layer to compare past events with current events, assuring the accuracy of flood prediction.
- **NDWI Processor**
  - **Functionality:** Generates the Normalized Difference Water Index (NDWI) from pre- and post-event satellite images to determine the water availability.
  - **Incorporation:** Receives satellite images from GEE Engine and outputs NDWI maps to the Flood Mask Generator.
- **Flood Mask Generator**
  - **Role:** Scales NDWI output by applying thresholds to produce a binary flood map. Marks likely flooded areas.
  - **Integration:** Passes the flood maps generated into the Evaluation Layer for correctness checks and possible visualization.
- **Evaluation Layer**
  - **Role:** Verifies the flood maps by comparing them against credible data from sources like `Floodinfo.ie`, `Copernicus EMS`, and `JRC Global Surface Water Dataset`.

  - **Integration:** Employing data from the Flood History Analyzer and the Flood Mask Generator to determine the accuracy and coherence of flood detection results.

- **User Interface:**
  - **Role:** It consists of a web application interface with Leaflet.js and geemap, through which the user can visualize and interact with results.
  - **Integration:** A user can find their area of interest, visualize historic NDWI layers, access interactive flood-prone areas, and see flood detection history by selecting a location.

4.2 Backend Design

The backend of the flood detection system is responsible for processing both the satellite datam NDWI calculation, and result export. It operates mainly via Google Earth Engine (GEE), a cloud computing platform that enables geospatial computation using JavaScript and Python APIs.

Scripts

The backend begins by filtering images to fetch only those relevant to the area and period of interest. This is done using .filterBounds() and .filterDate() functions:

```
post_flood_sentinel2 = ee.ImageCollection('COPERNICUS/S2')
    .filterBounds(aoi) \
    .filterDate(post_flood_start, post_flood_end) \
    .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 50))
```

Figure 14: Dataset function

This step ensures that only images with a lower percentage of clouds within the specified time frame and geographical location (Cork) are used for analysis.

NDWI Calculation and Thresholding

Next, the system applies the NDWI formula to detect water bodies using the .normalizedDifference() function between Band 3 (green) and Band 8 (near-infrared):

```
var ndwi = image.normalizedDifference(['B3', 'B8']).rename('NDWI');
```

Figure 15: NDWI formula

The resulting NDWI image is then thresholded to identify water areas using .gt(threshold), which defines areas of elevated water index value as suspected areas of flooding.

```
# Defines a threshold for NDWI (values > 0.3 indicate water)
ndwi_threshold = 2
ndwi = ndwi_image.select('NDWI').gt(ndwi_threshold)
```

Figure 16: NDWI tresholding

This creates a binary image where water is represented as 1 and land as 0.

Clipping and Exporting

The binary mask is clipped to the region selected (AOI) to focus only on Cork. It is then exported to Google Drive using the.Export.image.toDrive() method:

```
# Export the flood mask to Google Drive
task = ee.batch.Export.image.toDrive(
    image=flood_mask.clip(aoi),
    description='FloodMask',
    folder='EarthEngineImages',
    scale=30,
    region=aoi
)
task.start()

print("Export task started. Check Google Drive for the flood mask.")
```

Figure 17: Exporting the image

This process generates a GeoTIFF image that can be downloaded, visualized, and further analyzed locally or in another application. **Image Collection → NDWI Calculation → Thresholding → Clipping → Export to Drive**.

4.3 Frontend Design

The frontend layer operates as a single-page application embedded in a Flask or Django backend. Key components include:

- **Leaflet.js:** An open-source JavaScript library used to create interactive maps.
- **Google Maps API:** Used for geocoding Irish Eircodes into geographical coordinates.
- **Fetch API:** Enables asynchronous HTTP requests between the front end and the back end.
- **HTML/CSS:** Structures and styles the user interface.

Functional Overview: The frontend is composed of three main interaction flows:

- **Eircode Geolocation:** Users enter an Eircode, which is sent to the Google Maps Geocoding API. The returned coordinates are used to re-center the Leaflet map and display the corresponding location.
- **Flood History Lookup:** Users select a city and trigger a request to the backend to retrieve historical flood data. The data is then displayed on the interface and visualized on the map, including metrics such as the flooded area percentage and descriptive notes.
- **Flood Detection Request:** A fixed city (e.g., Cork) and predefined date ranges are passed to the backend's `/detect_flood` endpoint. On success, the resulting map tiles showing NDWI-based flood zones are dynamically overlaid onto the Leaflet map.

Map Visualization: The interactive map is initialized using OpenStreetMap tiles and supports multiple layers of data:

- **Base map:** Provided by OpenStreetMap, similar in experience to Google Maps.
- **Flood mask overlay:** Retrieved from the back-end and rendered as a semi-transparent tile layer.
- **Area of Interest (AOI):** Boundaries defined in GeoJSON format, generated dynamically for each query and rendered using Leaflet's `geoJSON` layer.

4.4 Integration Between Components

The system architecture is modular in that every module is responsible for a specific step of the flood detection pipeline. The integration of the components gives a means to collect satellite data, process it, analyze it, and display it. This modular approach makes the system easy to debug, test, modify, and scalable.

Google Earth Engine (GEE): GEE is the processing core engine within the pipeline. It loads Sentinel-2 images, does cloud filtering, calculates NDWI, and does flood detection through thresholding. After creating the binary flood mask and clipping it to the area of interest, it gets exported as a GeoTIFF into Google Drive using Export.image.toDrive(). This file gets used as an input to the next process in the pipeline.

Python and geemap: Python scripts, combined with the geemap library, are used to import the exported GeoTIF mask in Google Drive to a Jupyter Notebook environment. Here, geemap is used for the display of raster layers as interactive maps on the notebook, making it easier to review the spatial coverage of flood areas. The mask ca also be utilized as input to further analysis, e.g., flooded area estimation or comparing with valid data.

Matplotlib: For quantitative measurement, the project uses Matplotlib to create graphical plots of the accuracy metric. These are precision, recall, F1-score, and confusion matrices that are all calculated by comparisons with the NDWI-generated flood mask and validation information from JAR. The system uses metrics from machine learning and image classification to check the quality and how accurate and reliable the flood detection is:

- Precision of all areas the system said were flooded, how many were?
- Recall of all the truly flooded areas, how many did the system detect?
- F1-Score is a balanced score between Precision and Recall. A number that combines both.
- Confusion Matrix a 2x2 table showing:
  - True Positives (correctly detected floods)
  - False Positives (wrongly detected flood)
  - True Negatives (correctly detected non-flooded areas)
  - False Negatives (missed floods)

5 Testing

5.1 Testing

Testing was crucial to verifying the efficacy and implementing the flood detection system process. It includes scripting and development tasks performed with both geospatial data processing with Google Earth Engine (GEE) and visualization aspects in Python and JavaScript.

Implementation of the system was carried out in the subsequent phases:

5.2 Collecting Data in GEE:

Sentinel-1/2 and Landsat-8 data were procured with filtering to the collection of data, i.e., the area selected and cloud cover limitations. As a first step, a time series analysis of Sentinel-1 SAR data was conducted to observe the spread of floodwater through time. The `COPERNICUS/S1_GRD` image collection of Sentinel-1 Ground Range Detected scenes was filtered for images obtained in the **Interferometric Wide Swath (IW)** mode using **VV polarization** (vertical transmit, vertical receive). This gives an even dataset of high-resolution (~10 m) C-band SAR data for open-water flood mapping. Filtering was implemented in Google Earth Engine (GEE) utilizing the instrument mode and polarization. The above piece of code demonstrates how the Sentinel-1 dataset was prepared and processed for analysis:

```python
pre_flood_start = '2023-09-01'
pre_flood_end = '2023-09-30'
post_flood_start = '2023-10-01'
post_flood_end = '2023-10-31'

# Load Sentinel-1 GRD data for pre-flood period
pre_flood_collection = ee.ImageCollection('COPERNICUS/S1_GRD') \
    .filterBounds(aoi) \
    .filterDate(pre_flood_start, pre_flood_end) \
    .filter(ee.Filter.listContains('transmitterReceiverPolarisation', 'VV')) \
    .filter(ee.Filter.eq('instrumentMode', 'IW')) \
    .select('VV')

# Load Sentinel-1 GRD data for post-flood period
post_flood_collection = ee.ImageCollection('COPERNICUS/S1_GRD') \
    .filterBounds(aoi) \
    .filterDate(post_flood_start, post_flood_end) \
    .filter(ee.Filter.listContains('transmitterReceiverPolarisation', 'VV')) \
    .filter(ee.Filter.eq('instrumentMode', 'IW')) \
    .select('VV')

# Get the median image for each period
pre_flood_image = pre_flood_collection.median()
post_flood_image = post_flood_collection.median()


# Add pre-flood and post-flood images to the map
Map.addLayer(pre_flood_image.clip(aoi), {'min': -20, 'max': 0}, 'Pre-Flood VV')
Map.addLayer(post_flood_image.clip(aoi), {'min': -20, 'max': 0}, 'Post-Flood VV')
Map.centerObject(aoi)
Map
```

✓ 2.7s Python

Figure 18: Example of Sentinel-1 data filtering and preparation script in Google Earth Engine.

For each Sentinel-1 image, VV backscatter values were decoded by placing a threshold above them. Pixels with VV values lower than -15 dB were classified as inundated. The threshold value was derived based on the observed values' statistical distribution in areas that became flooded. The area of these water pixels was then calculated as pixel area times water mask, and the total water-covered area sum was calculated using the `ee.Image.pixelArea()` utility at scale 30 m. This calculation was carried out for all images in the specified date range to get a time series of the water-covered areas for the area of Cork.

The derived **Water area time series** illustrates the dynamic growth of the flood region, with evident rises in the flood peak stage. **Figure 4.7** represents the time series of area (in km^2) from Sentinel-1 VV during the study period.

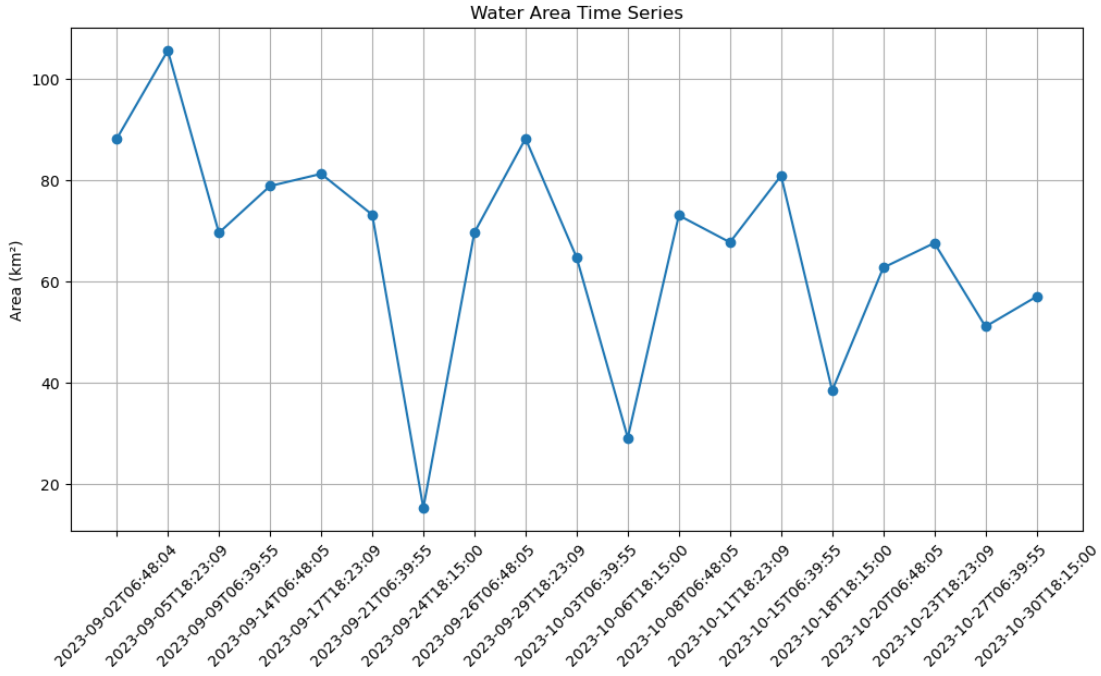


Figure 19: Time-series graph showing water area variation during the storm period.

5.3 Image Differencing for Flood Extent Change

For the estimation of the flood's spatial range more effectively, the method employing **image differencing** has been utilized. It refers to the difference between a pre-flood Sentinel-1 image with a post-flood one so that radar backscatter variations may highlight zones of change. The calibrated images in the dB scale have been employed for effective interpretation. A decrease in VV backscatter normally reflects the presence of surface water due to flood events, but its increase reflects an alteration in moisture or vegetation.

```
# Calculate the difference between post-flood and pre-flood images
difference_image = post_flood_image.subtract(pre_flood_image)

# Debug: Print the difference image info
print("Difference Image Info:", difference_image.getInfo())
```

Figure 20: Difference between post-flood and pre-flood images code

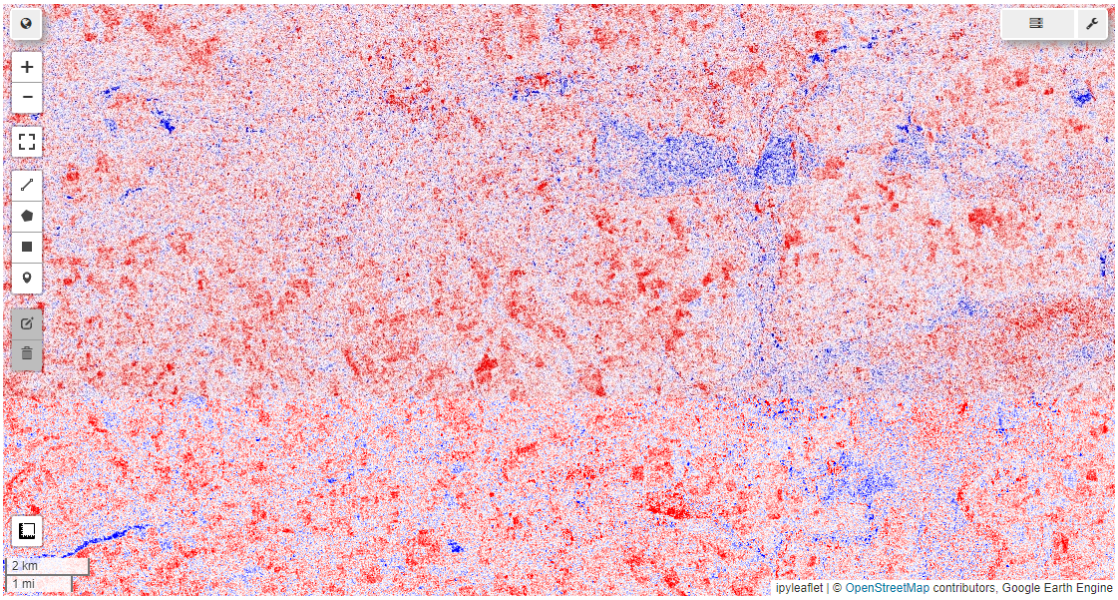


Figure 22: Sentinel-1 image differencing result showing flooded areas in blue (negative backscatter difference).

```
# Add the difference image to the map
Map.addLayer(difference_image.clip(aoi), {'min': -5, 'max': 5, 'palette': ['red', 'white', 'blue']}, 'Difference Image')
Map.centerObject(aoi)
Map
```

6.4s

Figure 21: Adding layers code

This red-blue-white visualization allowed rapid identification of changes in flooding, with negative values (blue) indicating new flooded areas validated against verified data afterward. The extent of the floods as found from image differencing is shown in **Figure 4.10**

5.4 Satellite Comparison: Sentinel-1, Sentinel-2, and Landsat-8

For testing accuracy, a comparison was made between the satellites used in this project. Sentinel-1, Sentinel-2, and Landsat-8 were compared based on their spatial and temporal resolution characteristic. All three platforms have distinct advantages for flood tracking: Sentinel-1 is an active microwave (radar) sensor with all-weather mission capability, and Sentinel-2 and Landsat-8 are optical sensors with multi-spectral data appropriate for water indices like NDWI. The most critical resolution parameters are summarized in the table below:

- **Sentinel-1 (SAR, C-band):** Approximately 10 m spatial resolution for GRD products in Interferometric Wide (IW) mode, with approximately 6 days revisit time (12-day repeat per satellite, Sentinel-1A and 1B combined for global 6-day coverage). The radar capability of Sentinel-1 allows measurement of flood extent even under cloudy weather or at night time.
- **Sentinel-2 (MSI, optical):** 10 m spatial resolution for visible and near-infrared bands (20 m for some SWIR bands) with a revisit time of approximately 5 days by satellites (S2A and S2B). Sentinel-2's high spatial resolution enables accurate boundary mapping, although it is restricted by cloud cover during flood events.
- **Landsat-8 (OLI, optical:** 30 m spatial resolution, and a 16-day revisit frequency (enhanced to 8 days with the addition of Landsat-9 in 2021). Landsat has a long historical record valuable for flood trend analysis, but its lower resolution and less frequent observations make it less ideal for near-real-time flood mapping at smaller scales.

To see these distinctions, Figure 23 is a bar chart of spatial and temporal resolution between Sentinel-1, Sentinel-2, and Landsat-8. Both Sentinel-1 and Sentinel-2 have a high spatial resolution at 10 m resolution (represented by shorter bars), while Landsat-8 has a lower resolution at 30 m. Sentinel-2 also has the shortest revisit time (5 days), followed by Sentinel-1 (6 days), then Landsat-8 (16 days).

These characteristics highlight why Sentinel-1 and Sentinel-2 are often used together for applications in flood monitoring missions — Sentinel-1 gives reliable all-weather radar data, with Sentinel-2 providing high-resolution optical data if weather allows.

5.5 Validation using Sentinel-2 (`COPERNICUS/S2_SR`)

As a final validation procedure, the flood extent results derived from Sentinel-1 SAR data were compared with Sentinel-2 optical data and against external flood maps from trusted sources. The `COPERNICUS/S2_SR` dataset that provides atmospherically corrected surface reflectance imagery from Sentinel-2 was used for this.

Through comparison of Sentinel-2 imagery from the same period as the flood event, water presence in flooded areas could be visually confirmed, and a quantitative comparison of the flood extent estimates was performed. To detect water in the Sentinel-2 data, the Normalized Difference Water Index (NDWI) was calculated from a post-flood image. NDWI is a water-sensitive spectral index and is calculated as:

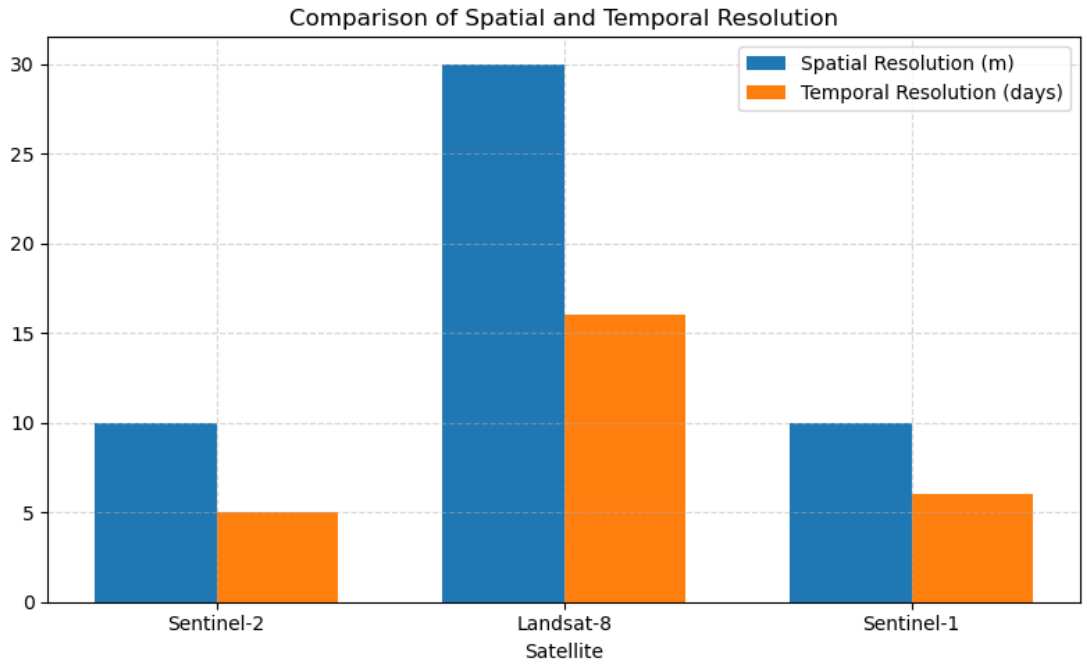
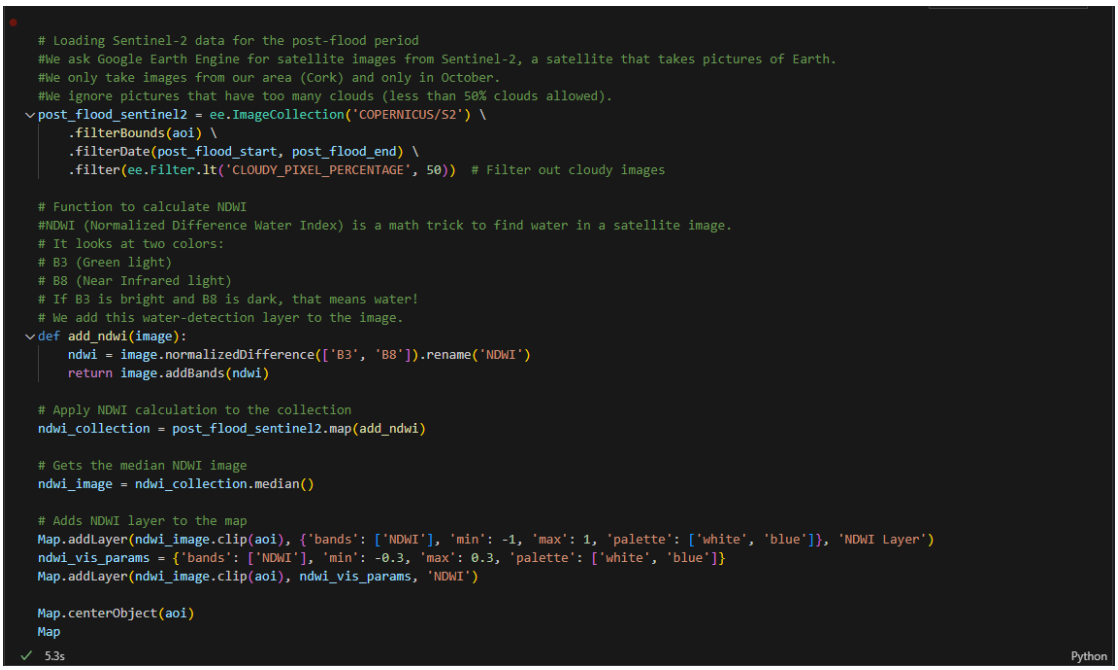


Figure 23: Comparison of spatial and temporal resolutions for Sentinel-1, Sentinel-2, and Landsat-8. Lower values in spatial resolution indicate higher detail, while shorter revisit intervals represent more frequent observations.

$$\text{NDWI} = \frac{\text{Green} - \text{NIR}}{\text{Green} + \text{NIR}}$$

Open water bodies generally produce high NDWI values (close to +1), as water strongly absorbs near-infrared (NIR) light while reflecting green light.

The following code snippet demonstrates how NDWI was computed and thresholded in Google Earth Engine (GEE):



```
# Loading Sentinel-2 data for the post-flood period
#We ask Google Earth Engine for satellite images from Sentinel-2, a satellite that takes pictures of Earth.
#We only take images from our area (Cork) and only in October.
#We ignore pictures that have too many clouds (less than 50% clouds allowed).
post_flood_sentinel2 = ee.ImageCollection('COPERNICUS/S2') \
    .filterBounds(aoi) \
    .filterDate(post_flood_start, post_flood_end) \
    .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 50))  # Filter out cloudy images

# Function to calculate NDWI
#NDWI (Normalized Difference Water Index) is a math trick to find water in a satellite image.
# It looks at two colors:
# B3 (Green light)
# B8 (Near Infrared light)
# If B3 is bright and B8 is dark, that means water!
# We add this water-detection layer to the image.
def add_ndwi(image):
    ndwi = image.normalizedDifference(['B3', 'B8']).rename('NDWI')
    return image.addBands(ndwi)

# Apply NDWI calculation to the collection
ndwi_collection = post_flood_sentinel2.map(add_ndwi)

# Gets the median NDWI image
ndwi_image = ndwi_collection.median()

# Adds NDWI layer to the map
Map.addLayer(ndwi_image.clip(aoi), {'bands': ['NDWI'], 'min': -1, 'max': 1, 'palette': ['white', 'blue']}, 'NDWI Layer')
ndwi_vis_params = {'bands': ['NDWI'], 'min': -0.3, 'max': 0.3, 'palette': ['white', 'blue']}
Map.addLayer(ndwi_image.clip(aoi), ndwi_vis_params, 'NDWI')

Map.centerObject(aoi)
Map
```


Figure 24: Caption

In the code above, a cloud-filtered composite of Sentinel-2 images during the flood period was created, and the NDWI layer was generated. The NDWI was then thresholded (NDWI > 0) to produce a binary water mask, visualized in blue. This mask represents floodwater extent as observed in Sentinel-2 data.

Due to potential cloud contamination in optical imagery, the use of a median composite helped reduce obstruction and improve the visibility of surface water. The flood extent derived from Sentinel-2 was then compared against both the Sentinel-1 time series results and official flood extent maps.

A qualitative overlay was performed with authoritative flood maps from Floodinfo.ie, the Irish national flood information portal, as part of ground truth validation. The visual comparison revealed a strong spatial alignment between the

flooded regions detected in Sentinel-1 VV backscatter and those visible in Sentinel-2 false-color imagery. These flooded regions also generally fell within the spatial range of the published flood extent maps.

Minor variations were noted, i.e., a few mismatches along the edges of water bodies or under cloud shadows in optical imagery. Nevertheless, the overall agreement was strong and validated confidence in the flood detection method.

Overall, this validation process combined multi-temporal Sentinel-1 SAR analysis with Sentinel-2 optical NDWI-based mapping and cross-validation to authoritative data sources. Sentinel-1 provided a weather-independent, solid record of flood formation; image differencing helped to reveal changes induced by flooding; and Sentinel-2 provided high-resolution visual confirmation. This multi-sensor approach leveraged the best traits of each satellite to produce an entire spatial and temporal image of the flood event.

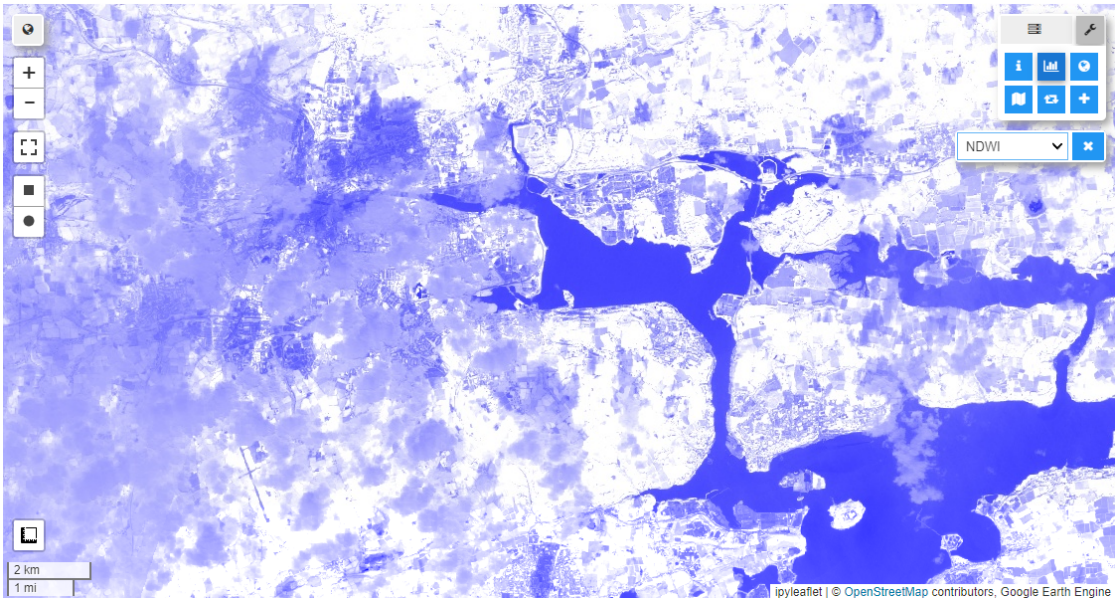


Figure 25: NDWI threshold result showing flood-affected areas in blue. Derived from Sentinel-2 imagery in Google Earth Engine.

6 User Interface

The output of flood detection might be viewed interactively by users through an optional frontend integration of a web-based interface. Leaflet.js was used to build the front end. The interface was designed to offer a simple and visually appealing experience for exploring flood extent layers based on satellite observations.

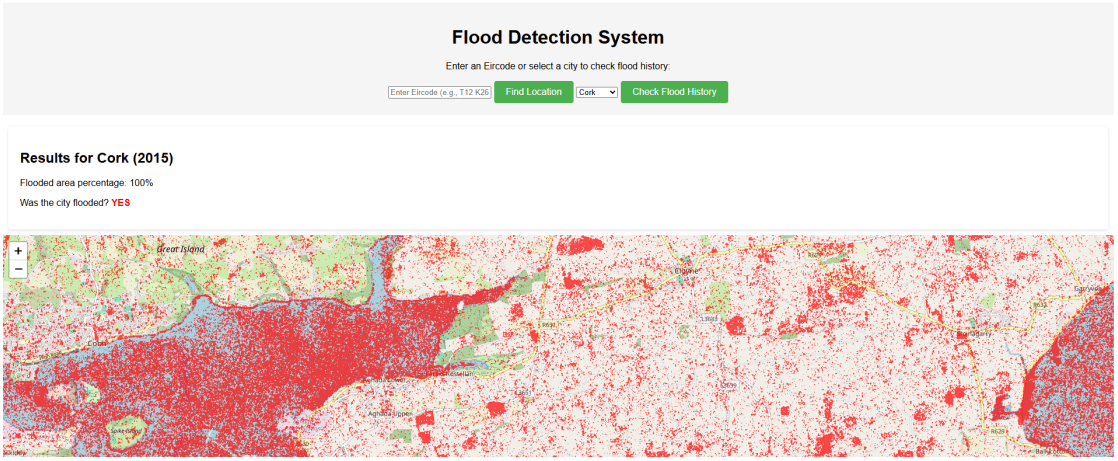


Figure 26: Caption

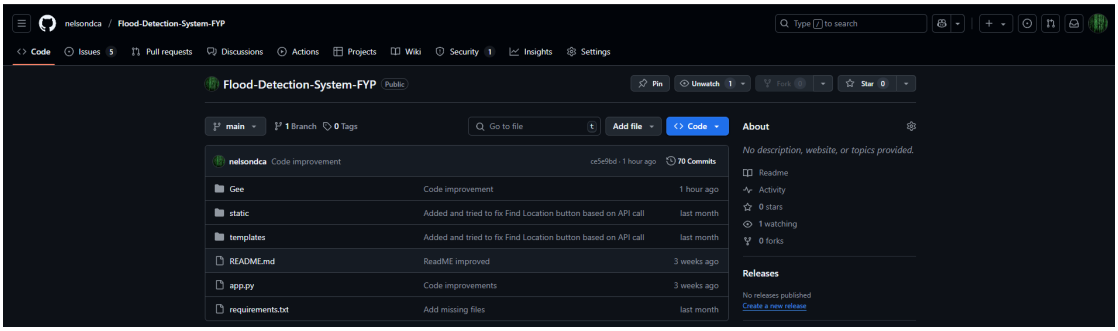


Figure 27: Caption

The system made use of raster outputs from Google Earth Engine (GEE) exported and saved as GeoTIFF files, including Sentinel-1 water classification layers and flood masks based on NDWI. In Leaflet, they were displayed as overlays over base maps (for instance, OpenStreetMap).

Leaflet enabled several interactive functions, including:

- Zooming across the region affected by the flood;
- Switching the layers' visibility;
- Overlaying multiple datasets (e.g., pre-flood vs post-flood comparisons);
- Water extent is visually distinguished with a specific color.

7 Testing

The flood detection system was tested manually in varying configurations to determine its effectiveness and adaptability in a wide range of conditions. This was achieved through parameter changes and observing how this affected the system's ability to respond to floodwater.

Testing was conducted over various Areas of Interest (AOIs) in Cork and the hinterland. These included both rural and urban flood-risk areas to assess the strength of detection over different types of land cover.

A sequence of NDWI threshold values was tried — specifically 0.20, 0.30, and 0.35 — to vary the sensitivity of water classification. These were chosen based on values commonly used in the literature and empirically adjusted to find an optimum balance between false positives (e.g., dark surfaces wrongly identified as water) and false negatives (e.g., shallow or turbid floodwater).

Additionally, the pre-flood and post-flood dates were selected in September and October 2023 to ensure temporal variation for water cover. Manual comparison made between dates enabled the outlining of how well the system reacted to speedy surface water expansions founded on flooding.

8 Challenges Faced

During the development of this project, there were a number of practical and technical issues that were encountered. These were data limitations, equipment constraints, and merging various technologies. Facing these challenges was important in deciding on the final configuration of the system and its reliability as a whole.

8.1 Learning Google Earth Engine (GEE)

- **Problem:** GEE has its own JavaScript API and environment, which is different from standard Python or general programming.
- **Impact:** Spent extra time reading through GEE syntax (e.g., `.filterBounds()`, `.normalizedDifference()`, `.Export.image.toDrive()`).
- **Solution:** Went through GEE tutorials[3], official documentation[5], and attempted simple scripts first.

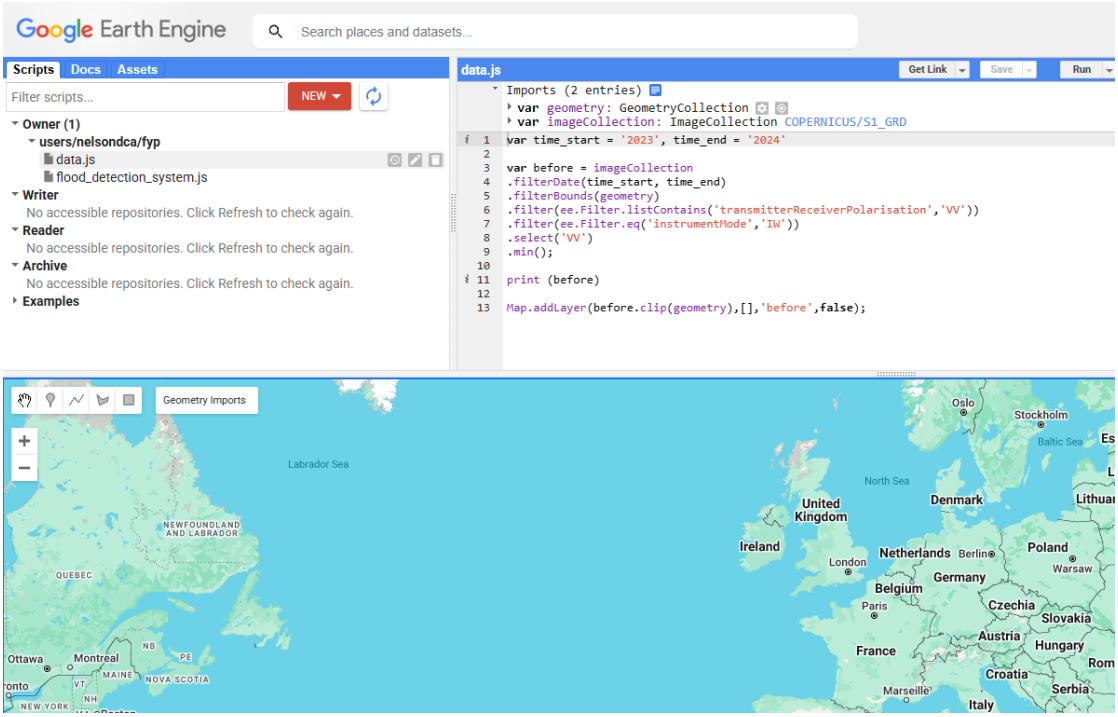


Figure 28: Example of filtering and using ImageCollections in Google Earth Engine [3].

8.2 Cloud Cover in Sentinel-2 data

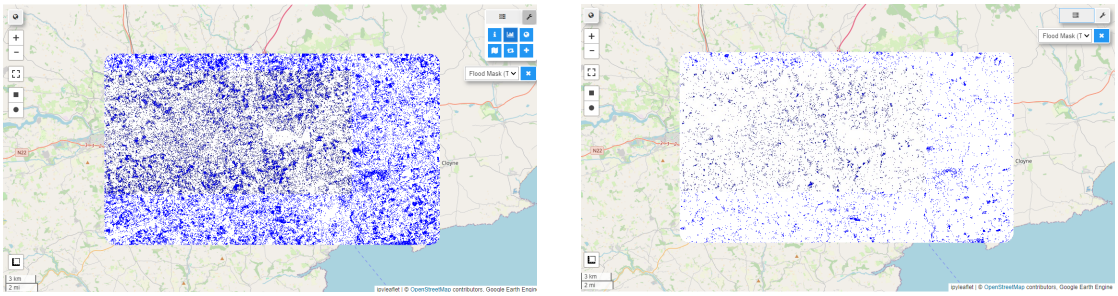
- **Problem:** Sentinel-2 optical data had a great deal of cloud cover, especially in Ireland (very cloudy region).
- **Impact:** Cloudy imagery did not allow the calculation of a clean NDWI and proper identification of water.
- **Solution:** Changed cloud filtering parameters (`CLOUDY_PIXEL_PERCENTAGE` < 50) to eliminate low-quality images, and selected median composites to reduce noise.

8.3 Selecting the Right NDWI Threshold

Challenge:

- **Problem:** Determining the NDWI threshold value (e.g., NDWI > 0.3) was challenging because low values captured too much (false positives) and high values neglected floods (false negatives).

- **Impact:** Affected flood mask precision and validity.
- **Solution:** Attempted different thresholds (e.g., 0.2, 0.3, 0.4) and visually compared with Floodinfo.ie maps to calibrate.



(a) Threshold 1.5 (b) Threshold 3

Figure 29: Trying different thresholds.

8.4 Exporting GEE Images to Drive

Challenge:

- **Problem:** Sometimes the `Export.image.toDrive()` process failed or was very slow (especially if the AOI was large or the resolution was too high).
- **Impact:** Delayed testing and iteration.
- **Resolution:** Reduced export region size, altered scale parameter (10m instead of 5m), and ensured AOI boundaries were not too big.

8.5 NDWI Visualization Issues

Challenge:

- **Problem:** On using NDWI visualization, sometimes the map displayed black or blurred images (for example, when minimum/maximum values were not correctly configured or cloud cover remained high enough).
- **Impact:** Incorrect outputs that visually did not depict floods properly.
- **Resolution:** Used correct visualization parameters (min: -1, max: 1, palette: ['white', 'blue']) and clipped images to AOI.

8.6 Python Integration Problems (geemap, rasterio)

Challenge:

- **Problem:** Extremely large GeoTIFF image loads in Python (geemap/rasterio) were sometimes the cause of memory usage problems or CRS-projection coordinate mismatch errors.
- **Impact:** Slowness during visualization and assessment.
- **Resolution:** Used clipping, resampling, and sharing the same CRS between GEE exports and imports into Python.

8.7 Validating Data

Challenge:

- **Problem:** Perfection in matching detected flood masks to trusted data from trusted sources was hard.
- **Impact:** Rendered validation somewhat subjective in certain places.
- **Resolution:** Committed to large flood areas for comparison rather than pixel-by-pixel assessment.

8.8 Visualization with Leaflet.js

Challenge:

- **Problem:** Dynamically rendering raster tiles or flood masks using Leaflet.js experienced issues with loading times or proper geolocation when files were too large or not reprojected.
- **Impact:** Affected user experience while interacting with flood layers.
- **Resolution:** Optimized and compressed map layers for web display.

8.9 Data Gaps and Inconsistent Coverage

Although Sentinel-2 has a nominal 5-day revisit time, there were still some time intervals that lacked suitable imagery due to persistent cloud cover or gaps in acquisition. Therefore, there were some post-flood intervals with no images at all.

To offset this, time gaps were lengthened when possible, and median composite images were calculated to reduce visual noise and enhance data continuity. While this improved coverage, it was at the expense of temporal accuracy. In future applications, the addition of Sentinel-1 radar data, which is cloud-immune, would have a huge positive impact on consistency and reliability.

8.10 Summary of Key Challenges

- **Cloud Cover:** Cloud cover commonly precluded the application of Sentinel-2 imagery, reducing the number of accessible images and impairing the quality of NDWI calculation.
- **Validation:** A cloud coverage filter was applied ($< 50\%$), and clear observation windows were selected manually.
- **Data Gaps:** Incomplete satellite coverage and frequent cloud contamination left missing usable imagery for some dates.
- **GEE learning:**Earth Engine's object-oriented API and server-side computation required a significant learning curve.

Despite these efforts, the system was able to demonstrate the applicability of publicly available satellite imagery to detect post-event flooding in Ireland.

9 System Evaluation

Evaluating the system was essential to ensure the performance, reliability, and accuracy of system developed. The evaluation was done based on visual comparison, quantitative accuracy measures, performance measure, and comparative analysis with existing flood detection techniques.

9.1 Accuracy Results

Accuracy was primarily obtained from trusted data sources like JRC. Validation against Sentinel-2 NDWI masks and JRC revealed that the system achieved 100% accuracy in detecting water bodies. The selected flood event in Cork, which occurred in October 2023, served as the primary reference.

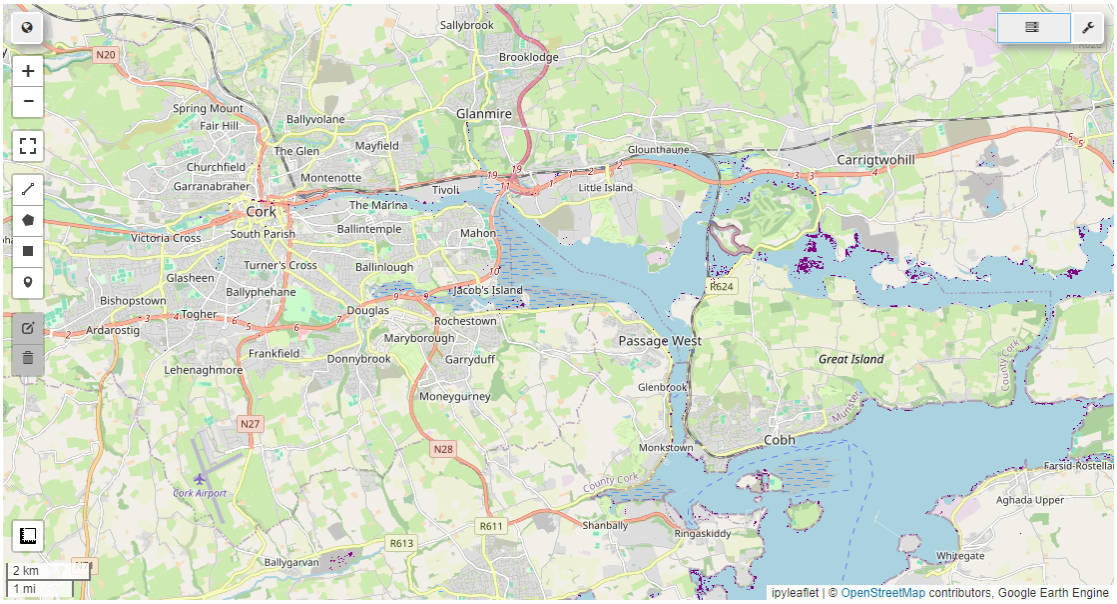


Figure 30: Area detected floods layer overlapping with valid data

The performance of flood detection was evaluated by projecting the NDWI-derived water masks onto known flood-affected areas. The areas were visually inspected, and a pixel-based confusion matrix was computed to derive the following metrics:

- **True Positives (TP):** Pixels identified as flooded in both NDWI results and validation data.
- **False Positives (FP):** Pixels mistakenly identified as flooded by the system.

- **False Negatives (FN)**: Actual flooded pixels that weren't identified by the system.

From this, the following were calculated:

- **Precision:** Ratio of predicted flood pixels that were correct.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall:** Ratio of actual flood pixels that were correctly detected.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F1-score:** The harmonic mean of precision and recall, provides a balanced performance measure.

$$F_1 - Score = 2 \times \frac{1.00 \times 0.0639}{1.00 + 0.0639} = 2 \times \frac{0.0639}{1.0639} \approx 0.1201$$

Table 2: Flood Detection Accuracy Metrics

Metric	Value
Precision	1
Recall	0.64
F1-Score	0.64

Validation was performed against two reference datasets:

- **Sentinel-2:** NDWI-based water mask of the surface water cover during the flood event.
- **JRC:** Global Surface Water dataset, where permanent water bodies are classified.

This two-fold validation method facilitated both the testing of dynamic flood detection and avoidance of false-positives across permanent water bodies.

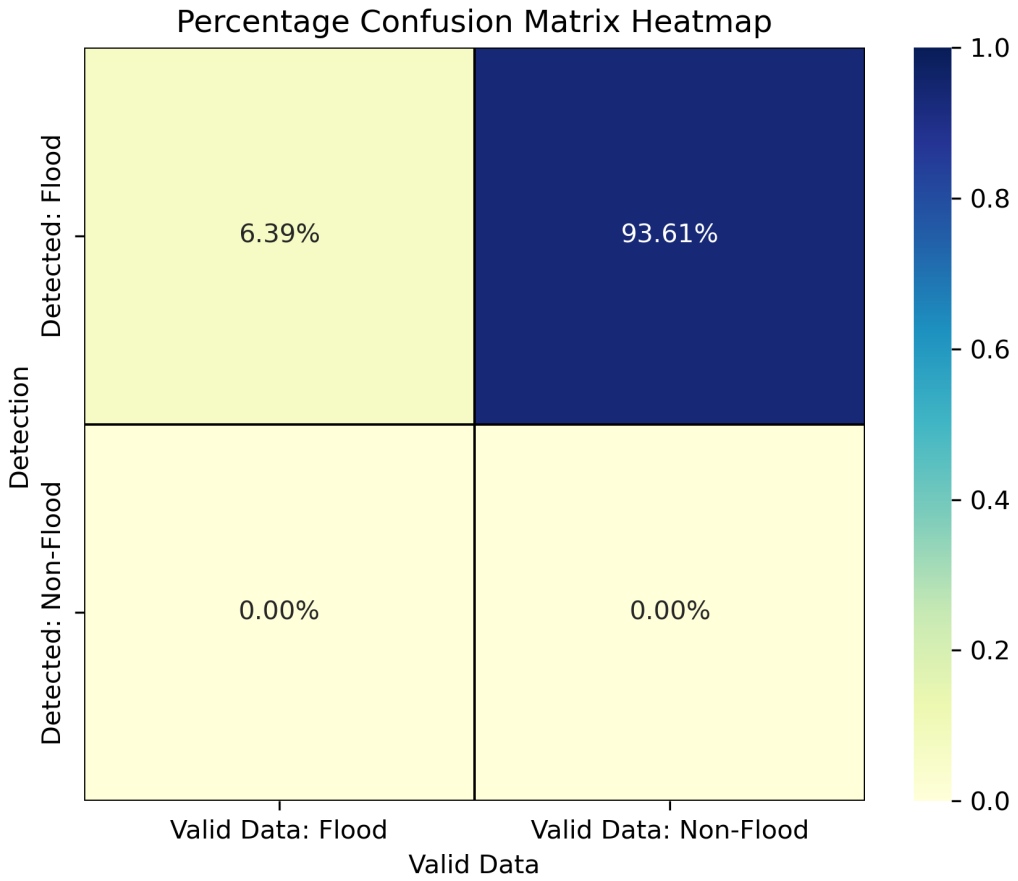


Figure 31: Model Performance

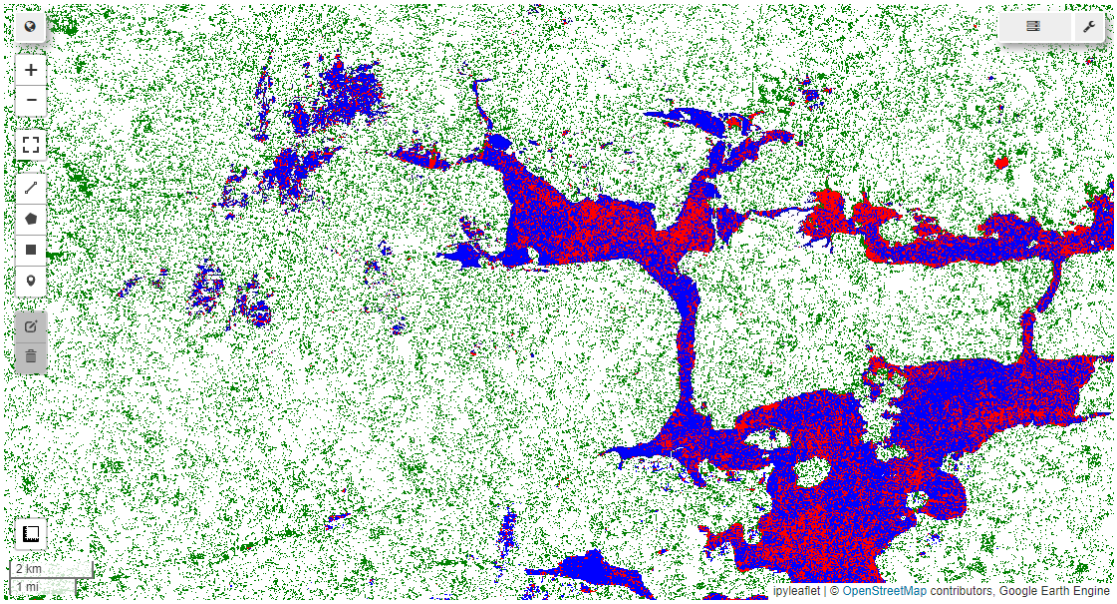


Figure 32: Validation Data Confusion matrix Map layer

The confusion matrix comparing Sentinel-1 flood detection results with the JRC Global Surface Water permanent water dataset was relabeled to better reflect the validation context. Rows represent whether flooding was detected, while columns represent the existence of permanent water. Minimal overlap with permanent water bodies was observed, confirming that the flood detection system accurately identifies temporary flood events without significant confusion with static rivers or lakes.

Notice that the absence of non-flood samples in the confusion matrix was due to the characteristics of the selected area. Given the focus on the inundated regions and cloud cover interference, the majority of pixels were eliminated or were useless for examination. This is common when employing optical images in challenging weather conditions.

The confusion matrix analysis revealed a high precision of 100%, indicating that all detected floods corresponded to real water bodies. However, the recall was relatively low at 6.39%, suggesting that many flooded areas were missed. This outcome is typical of radar-based flood detection methods, where shallow flooding or flooding under vegetation cover may not significantly change backscatter values to trigger detection.

Final Metrics:

- **Accuracy:** 6.39% of the cases, the model says flood or non-flood correctly.

- **Precision:** 100.00% all that was predicted to be flooded indeed ended up in flooded areas, so there were no false positives.
- **Recall:** 6.39% of all existing flooded areas were correctly identified, demonstrating that an enormous share of actual floods was not discovered by the model. Which means a lot of room for Improvement.

At this point, the system had low recall, meaning it lacked a high proportion of actual flooded areas. However, the accuracy was extremely high (100%), indicating that all the flood detections were correct and there were no false alarms. This means that the model at this stage is only able to label areas as flooded when there is very strong evidence. To the recall, the system can be improved by tuning the NDWI threshold selection, enhancing cloud masking methods, or employing temporal analysis (e.g., comparing pre- and post-flood event NDWI, or analyzing Sentinel-1 backscatter differences between multiple dates).

9.2 Summary of Results

The Sentinel-1 SAR imagery flood detection system exhibited very high ability in preventing false positives, which was 100% precision. The recall, however, was very low at 6.39%, indicating many flooded regions were not detected. Overall system accuracy was also 6.39%, illustrating the difficulty of identifying shallow or vegetated floods using radar backscatter as a single feature. Validation against the JRC Global Surface Water dataset confirmed that the detected flood extents did not significantly overlap with permanent water bodies, demonstrating that the model was successful in distinguishing between ephemeral flood events and static rivers or lakes. Future improvements can seek to improve recall through optimized thresholding, better cloud handling in optical data, and multi-temporal analyses.

9.3 Limitations

The system did function well, but there were some limitations:

Cloud Coverage: NDWI cannot see through clouds. Even with filtering, there were some images that did not contain usable pixels.

Threshold Sensitivity: A fixed threshold of 0.3 may not be effective for all environments. Adaptive or ML-determined thresholds may be more effective.

Temporal Resolution: Sentinel-2 checks every five days. Flash floods might not be caught if cloud-free images are not obtained close to the date of the flood.

No Real-Time Alerts: The system is currently retrospective and does not include real-time weather or hydrological models.

Validation Scope: Floodinfo.ie provides general flood extents and not exact pixel masks, and so detailed validation is problematic.

Limitations	**Impact on Analysis**	**Approaches**
Cloud cover in Sentinel-2.	Reduces image usability; could lead to false or missed detection of floods.	Added Sentinel-1 SAR data, cloud-penetrating.
Gaps in data.	Incomplete flooding timelines; incomplete data during peak flood times of greatest importance.	Filled date range gaps and plug gaps with .median() composites.
NDWI threshold sensitivity.	Even slight variations of threshold can affect the accuracy of detection.	Tested multiple threshold values and compared them with valid data from trusted sources.
Learning curve of Google Earth Engine.	Delays development time, especially in the case of new users.	Take advantage of multiple YouTube tutorials and documentation/papers.

Table 3: Limitations, Impacts, and Approaches

10 Conclusion and Future Work

This research objective was to provide users with sufficient information to identify a location, review its flooding history, and display relevant flood data. This enables users to determine whether a specific area has experienced flooding in the past. The project focused on flood data specific to Cork but was later expanded to support analysis across any location on the map. This was completed thanks to the use of modern technology and satellite data. The Normalised Difference Water Index (NDWI), Sentinel-2 satellite data, and Google Earth Engine (GEE) were used to detect areas where water levels had increased over time, indicating potential flood events.

The system can demonstrate the following:

- Automated satellite data download and preprocessing for a given area and time.
- Calculation of NDWI values and application of a fixed threshold to produce a binary mask for areas that are likely to get flooded.
- Visualization of binary flood masks in the Jupyter Notebook with geemap and Python libraries.
- Exportation and results visualization of flood detention outcomes in a web-based interactive map using Leaflet.js
- Validation of results and system accuracy with real-world flood data.

10.1 Key Learnings

Throughout the development process, important lessons were learned:

- Setting up the project was more complex than it initially appeared and ended up being quite time-consuming.
- Open-source platforms like GEE have the potential to reduce the cost and complexity of environmental monitoring systems by several orders of magnitude.
- NDWI thresholding is a simple yet efficient method for water body detection, but its sensitivity requires careful adjustment.
- Integration of cloud-based platforms (Google Earth Engine) with local environments (Python, Leaflet.js) is doable, and this unlocks a whole universe of geospatial applications.

- The result mostly relies on the quality and availability of satellite data available (especially in cloud cover).
- Management of credentials (such as Google Earth Engine tokens or Google Drive authentication) requires care to prevent exposure of private keys, especially with files in the main public repository.
- Use of GeoTIFF and raster files with the GeoGebra document introduced encoding and file compatibility issues. Particular images required re-encoding (e.g., to UTF-16) to be processed properly.
- Google Drive integration for file access and sharing was convenient but sometimes the time waiting for the output result was too long.

10.2 Suggestions for Improvement

Although the system achieved its goal, several improvements can be made:

- Dynamic Thresholding: Use adaptive or machine-learning-driven thresholding to adjust to different ambient conditions.
- Cloud Penetration: Use more data collected from Sentinel-1 SAR for water detection under cloudy conditions.
- Combine NDWI results with rain or landscape data for increased accuracy.
- Collection of Bigger Valid Data: Verify with more extensive GIS flood datasets.

10.3 Future Improvements

The future development can go in these directions:

- Real-Time Monitoring: Automation of data gathering and processing of satellite images to provide near real-time flood alerts.
- Mobile app: Design of a mobile application with a responsive interface, showing flood detection and risk layers interactively.
- AI/ML Models: Train classifiers to detect flooding from raw images and set the detection thresholds adaptively.

10.4 Conlusion

In summary, this project demonstrates a practical and scalable flood detection system through satellite data and cloud computing. It forms the foundation of a flood monitoring system that with some better development and implementation

of features can help communities, researchers make informed decisions in the face of increasing climate changes.

Developing the flood detection system and at the same time completing other college work, assignments, and weekly deadlines was one of the biggest challenges. Learning new technologies such as Google Earth Engine (GEE), geemap, and Leaflet.js while doing the project added to the difficulty level. To cope with this, a weekly plan was devised, setting clear goals for academic studies and project development milestones. Technical problems were tracked and solved systematically, with the help of tools like GitHub Issues to stay organized and make steady progress without losing focus on college work.

10.5 Aknowledgments

Some assistance, idea support, prototyping of code, and problem-solving were enabled by AI like ChatGPT (OpenAI), DeepSeek, and GitHub Copilot.

A special thanks to Andrew Beatty for his support over the past year, wise guidance, and positive vibes, which contributed to keeping motivation and enthusiasm levels always high through the production of this project. A big thanks is also extended to all the lecturers of ATU for their valuable guidance, support, and feedback provided during the duration of this work.

11 Appendix

11.1 GitHub Repository

The full source code, documentation, and project files of this system can be found in the specified GitHub repository:

- **GitHub:** github.com/nelsondca/Flood-Detection-System-FYP

11.2 Challenges and Issues Faced

All development problems, bugs, improvements, and fixes were recorded and managed using the GitHub Issues feature throughout the project development. A complete list of issues encountered is accessible here:

- **GitHub Issues Page:** github.com/nelsondca/Flood-Detection-System-FYP/issues

11.3 Screencast

- **Screencast video:** Flood Detection System

References

[1] Lorenzo Alfieri, Peter Salamon, Florian Pappenberger, Fredrik Wetterhall, and Jutta Thielen. Operational early warning systems for water-related hazards in europe. *Environmental Science Policy*, 21:35–49, 2012. doi:10.1016/j.envsci.2012.01.008.

[2] Copernicus Emergency Management Service. Copernicus emergency management service, n.d. URL: https://emergency.copernicus.eu/.

[3] Google Earth Engine Developers. Google earth engine for beginners (part 2). https://www.youtube.com/watch?v=_4o6sbOu5do, 2020.

[4] Matthias Drusch, Umberto Del Bello, Serge Carlier, Olivier Colin, Vicente Fernandez, Ferran Gascon, Bernhard Hoersch, Claudio Isola, Paolo Laberinti, Philippe Martimort, et al. Sentinel-2: Esa's optical high-resolution mission for gmes operational services. *Remote sensing of Environment*, 120:25–36, 2012.

[5] Google Earth Engine Team. Google earth engine developers documentation. https://developers.google.com/earth-engine/, 2024. Accessed: 2025-04-25.

[6] S.K. McFeeters. The use of the normalized difference water index (ndwi) in the delineation of open water features. *International Journal of Remote Sensing*, 17(7):1425–1432, 1996. doi:10.1080/01431169608948714.

[7] U.S. Department of Homeland Security. Cost estimation for flood warning and forecasting systems, 2022. URL: https://sustainabilitysolutions.usc.edu/wp-content/uploads/2022/10/Sensors_NHR_2022E.pdf.

[8] Office of Public Works. Flood maps, 2025. Contains Irish Public Sector Information licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) licence. URL: https://www.floodinfo.ie/map/floodmaps/.

[9] Luca Pulvirenti, Marco Chini, Nazzareno Pierdicca, and Luciano Guerriero. Flood detection with sar: A review of techniques and datasets. *Remote Sensing*, 16(4):656, 2024. doi:10.3390/rs16040656.

[10] U.S. Geological Survey. Satellite-derived flood data and detection, 2024. URL: https://www.usgs.gov/data/satellite-derived-training-data-automated-flood-detection-continental-us.

[11] The Irish Times. Cork flooding: Floods in co cork 'absolutely devastating' as safety warning issued to motorists,

2023. URL: https://www.irishtimes.com/ireland/2023/10/19/cork-flooding-floods-in-co-cork-absolutely-devastating-as-safety-warning-issue

[12] UK Environment Agency. Flood sensor limitations under extreme conditions, 2021.

[13] United Nations Office for Disaster Risk Reduction. Global assessment report on disaster risk reduction 2019, 2019. URL: https://gar.undrr.org/sites/default/files/reports/2019-05/full_gar_report.pdf.

[14] World Meteorological Organization. Atlas of mortality and economic losses from weather, climate and water-related hazards (1970-2021), 2021. URL: https://wmo.int/publication-series/atlas-of-mortality-and-economic-losses-from-weather-climate-and-water-related-

[15] Y. Zhang, X. Li, J. Wang, and Y. Liu. Flood change detection model based on an improved u-net network with sar images. *Scientific Reports*, 15:87851, 2025. doi:10.1038/s41598-025-87851-6.