DATA PREPARATION PLAN

Part of milestone 2

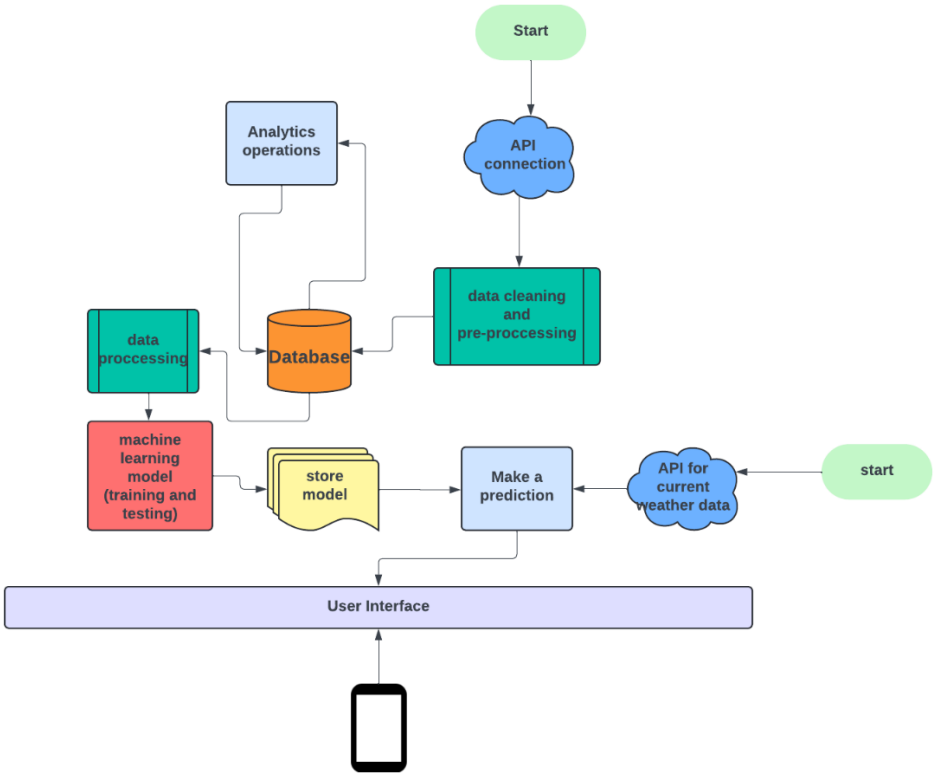
Student Name	Surname	Student Number
Thandeka	Mavunda	577949
Nosipho Precious	Donkrag	577354
Nontsikelelo Sharon	Buhlungu	577878
Owethu	Moyo	577051
Omphile	Tladi	577776
Oratile	Hlashwayo	577279

1. System Overview

The system design integrates satellite images, weather data, image processing and machine learning to track and predict the location of the hyacinth mat on the Hartbeespoort dam. It utilises a centralised database to ensure consistency of data across the different functionalities, such as analytics operations, data processing and model training. The system architecture highlights the key components required for the success of the project.

System architecture

The overall system architecture for the Hyacinth project is given below:



Functional Description

- API connection for historic data
  The initial stage is to collect the historic weather and image data through an API connection to google earth engine. The image data will be collected using the Landsat 8 satellite.
- Data cleaning and preprocessing
  The raw data will go through a cleaning and processing step before it is stored in the database. This is to ensure a cohesive database with relevant data as opposed to directly storing the data in the database without processing it.
- Data in the database
  The available data in the database will be suitable for analytics required by the Rhodes team, from here the data will undergo processing to make it suitable for the machine learning algorithm.
- ML model
  The ml model will be trained on the processed data and stored.

- Making a prediction
  To make a prediction the current weather data for the day will be fetched using an API and loaded into the stored model for a prediction.
  The model will then render a prediction of the location and spread of the Hyacinth mat across the Hartbeespoort dam.

Data Architecture:

The machine learning algorithm will be fed the weather information and is expected to predict the target y variable which is the pixels value (binary) of each image, based on the input weather data.

Data structure (data frame):

The high-level design of the dataset is given below

Input X data					**Target Y data**
Wind speed	**Wind direction**	**Temperature**	**Gust**	**Season**	**Pixels**
Numeric	**categorical**	**Numeric**	**Numeric**	**Categorical**	**binary**

The dataset that will be utilised is the Landsat 8 surface reflectance Tier 1 dataset.

- Landsat 8 images cover 185 km x 185 km of the earth's surface area.
- Spatial resolution (measure of the smallest object that can be distinguished)
  - Visible, near-infrared (invisible to the naked eye) waves: 30 meters
  - 15 meters for panchromatic waves
  - 100 meters for thermal infrared

Number of pixels that will be predicted for each image:

Horizontal and vertical pixels size:

$$\frac{158000\ meters}{30\ meters/pixel} = 6167\ pixels$$

Total pixels:

$$6167 * 6167 = 38\ 067\ 689$$

Target variable data:

	0	1	2	…	38 067 688
Image 1	1	0	0	..	1
Image 2	1	0	0	1	0
Image n	0	1	0	1	…

The target variables will be binary; 0 will indicate the absence of the hyacinth while 1 will indicate the presence of hyacinth.

Design Constraints

- Data design constraints
  - Only satellite images from Landsat 8 will be utilised to ensure data consistency.
  - Only Electromagnetic waves related to green vegetation will be considered, no infrared radiation will be considered (heat radiation).
  - Rainfall will not be a feature considered for training the ml models, as it will include additional considerations such as water depth and other nearby water bodies that may influence the dam. However, to account for the rainfall season an additional feature called 'Season' will be included.
  - Only data within the region of interest (ROI) that being the Hartbeespoort dam will be considered.

1. Detailed Design

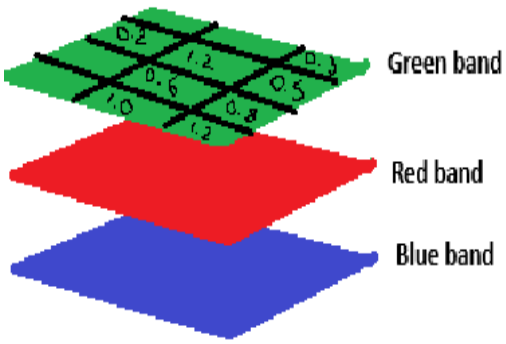
1.1. Data Design

1.1.1. Structure of satellite images

The satellite images for the Hartbeespoort dam will be collected form the google earth engine. The structure of the images from the earth engine is discussed below:

Each image is represented using multiple bands and these bands represent different wavelengths in the Electromagnetic spectrum (ArcGIS Pro, 2023).

Each of these bands are represented by 2D matrices of pixels; and each pixel has a pixel value corresponding to the intensity of a signal in that specific wavelength. These bands will be used to identify if a specific region on the image (in pixel form) has the Hyacinth mat or not.



The high-level structure of each image object is shown below:

```
image_object_1 = {
    "bands": {
        "B1": [ [ pixel_value_1, pixel_value_2, ... ], [ pixel_value_n1, pixel_value_n2,
        "B2": [ [ pixel_value_1, pixel_value_2, ... ], [ pixel_value_n1, pixel_value_n2,
        "B3": [ [ pixel_value_1, pixel_value_2, ... ], [ pixel_value_n1, pixel_value_n2,
        "B4": [ [ pixel_value_1, pixel_value_2, ... ], [ pixel_value_n1, pixel_value_n2,
        "B5": [ [ pixel_value_1, pixel_value_2, ... ], [ pixel_value_n1, pixel_value_n2,
        // Additional spectral bands as needed
    },
    "metadata": {
        "system:time_start": "YYYY-MM-DDTHH:MM:SSZ",  // Start time of the image acquisiti
        "system:time_end": "YYYY-MM-DDTHH:MM:SSZ",    // End time of the image acquisitio
        "system:asset_size": "size_in_bytes",          // Size of the image asset
        "system:version": "version_number",            // Processing version of the image
        "SATELLITE_NAME": "Satellite Name",            // Name of the satellite that capt
        "system:footprint": {                         // Geospatial bounds of the image
            "type": "Polygon",
            "coordinates": [ [ [ lon1, lat1 ], [ lon2, lat2 ], ... ] ]
        },
        // Additional properties and metadata as needed
    },
    "spatial_resolution": "resolution_in_meters",   // Spatial resolution of each band
    "projection": {                               // Projection information
        "crs": "EPSG:xxxx",                       // Coordinate Reference System
        "transform": [ x_scale, y_scale, x_translation, ... ]  // Transformation parameter
    }
}
```

Each image object comes with its corresponding metadata stating the start date and end date of which the image was taken; satellite images can be taken over several periods (time lapses) hence it is uncommon to have a start and end date for an image. For the projects, the end date will be used as the date allocated to the images. To reconstruct these images, certain bands of importance will be combined and mashed together. These will be discussed in one of the final steps.

1.1.2. Image processing steps

The following section of the project will be dedicated towards preparing the satellite images and weather data for the machine learning algorithms. This section aims to ensure that the data is in a consistent state to be used to train the machine learning algorithms.

Step 1: Image segmentation

Isolate the region of interest; this being the Hartbeespoort dam using Masking or Clipping. This step involves creating a binary mask where the pixels within the desired region are retained and those outside this region are set to zero.

Step 2: Atmospheric correction

The images captured by satellite sensors have been affected by atmospheric factors such as gasses, water vapor, and clouds. Correcting these images will ensure that the true surface reflectance is gathered.

- Importance of atmospheric correction:
  Consistency of images: These images will be captured at different times under different atmospheric conditions. Atmospheric correction will standardise these images to ensure that these images can be compared at different seasons and times.

- Minimizing environmental noise: The atmosphere can introduce noise into the data such as distortion or obscurity of images. To improve the accuracy of the models when detecting, noise must be removed or minimised.

- Detection and classification: Preparing the images for training includes extracting vegetation indices, these are a combination of surface reflectance's at different wavelengths. These are used to amplify a particular property of vegetation such as green, vegetation features and water (Geospatial Software, 2024). Atmospheric correction ensures that these numeric values reflect the actual conditions thus leading to more accurate results.

Step 3: Cloud Masking

To improve data quality and consistency; the images will then undergo cloud masking; cloud masking is the process of identifying and removing or minimising the impact of clouds and cloud shadows on images.

Clouds reflect light differently across different wavelengths; this distinct spectral signature will allow Fmask (an algorithm) to detect or mask the clouds in the images. Fmask will flag an image's pixels as either containing a cloud or not.

If cloud masking is performed before atmospheric correction, the uncorrected data could lead to inaccuracies when detecting the clouds as the images will likely still have noise.

1.1.3. Feature Extraction

The purpose of this section is to distinguish between the Hyacinth mat, water and land. This section will calculate vegetation indices and create additional features, related to the mat, that the algorithm can use to train on.

Step 1: Extract the NIR (near-infrared) and red bands of each image

To calculate the required indices; the NIR bands and red bands need to be extracted from the satellite images.

The NIR band is a specific range of wavelengths in an EM spectrum (0.7 μm - 1.1μm); and this band is helpful for quantitative analysis of data (Larkin, 2018).

In this project the NIR will be used for vegetation analysis. Vegetations reflect more NIR in comparison to other surfaces; hence it will be a useful measure in calculating the vegetation indices.

The Red band is in the visible part of the EM spectrum; it captures visible red light reflected by the surface of the earth. Vegetation absorbs most of this redlight for photosynthesis and reflects only a small amount; hence it will be a useful measure in identifying the location of the mat.

Check if the NIR and red bands have the same alignment; that is ensuring that each pixel in the NIR corresponds to the same location on the ground.

Python library to do this: **gdal**; gdal will also correct any alignment issues.

The last part of this function is to append the NIR and red band values to a panda's data frame.

Step 2: Normalised Difference vegetation Index (NDVI)

This function will calculate the NDVI which is a commonly used vegetation index that measures the density and health of vegetation. This index will be used to measure the density of the Hyacinth mat on the dam. The NDVI distinguishes between vegetation and not vegetative surfaces such as water and land. The NDVI is calculated as following:

$$NDVI = \frac{NIR - RED}{NIR + RED}$$

The value of the NDVI will give the vegetation's density; these values must also be appended to the padas data frame.

Step 3: Extract the Enhanced Vegetation Index (EVI)

This index is created to improve the detection of the vegetation in satellite images. It will be used as a safety net to train the ml models in case the NDVI is inadequate. It involves the extraction of the blue band value, a gain factor, coefficients of aerosol resistance (6 and 7.5) and canopy background adjustment value.

- Blue band
  This value for each image will be retrieved from the satellite's metadata of the images. The blue bands accounts for atmospheric effects such as tiny particles suspended in the air (aerosols). This will be used to further support atmospheric correction.

- Gain factor
  The gain factor is a constant usually set to 2.5 and enhances contrast between the vegetation and non-vegetation areas. This value enhances the sensitivity to the vegetation. It ranges between 1 and 3. The standard value will be used as the data source is uniform (G = 2.5) (Pat S. Chavez, 1988).

- Coefficients for aerosol resistance
  Hartbeespoort becomes dusty at certain periods of the year; to enhance the accuracy of the mat's detection considering the dust aspect; the C1 and C2 coefficients will be tuned first. Dust introduces aerosol effects hence tuning these coefficients will be an attempt in improving accuracy readings. The range values for C1 and C2 are given below:

  $$5.0 \leq C1 \leq 7.0$$

  $$7.0 \leq C2 \leq 8.5$$

  The standard values are C1 = 6 and C2 = 7.5.

- Canopy background adjustment factor (L)
  This value reduces the impact of non-vegetative elements; considering that the project is only centred around a dam and that there is not a lot of elements to consider that may look like the mat but are not; tuning this value will not be required. The standard value is 1.

  All these values come together to form the following formula:

$$EVI = G * \frac{NIR - RED}{NIR + (C1 * RED) - (C2 + BLUE) + L}$$

The EVI values will be appended to the data frame.

Step 4: Add feature to account for different seasons

Seasonal changes affect the vegetation density and lifecycle; to account for this, additional features must be created that will be used to train the model. This function will be dedicated to creating two additional features the first one is a categorical feature called season, that will hold the season of each data entry.

- **Extract season**
  Each season will be extracted from either the weather data's date or the image data's data for each entry. The specific element of date that will be extracted will be the month. The mapping is given below:

Months	Season	Entry into the data frame
December, January and February	Summer	S
March, April and May	Autumn	A
June, July and August	Winter	W
September, October and November	Spring	Sp

  The new feature called 'Season' will be added to the data frame.

- **Seasonal composite**

This feature will be created by averaging the NDVI or the EVI values for each season. Then based on the 'Season' value the data entry will be allocated an averaged value corresponding to its season's value.
Averaging these values will reduce the impact of noise and smooth out temporal noises (inconsistencies in images due to time lapses).

1.1.4. Normalise Indices

To ensure consistency across the dataset and optimise the model's performance's; the indices will be scaled down to a range of [0,1]. This function will be dedicated to scaling down the NDVI and EVI values in the data frame.

1.1.5. Label the target pixels and assign values

The last step to ensure that the images are fully prepared for the ml models is to assign a threshold value, for the NDVI and EVI indices; this threshold will be used to assign a binary 1 or 0 to each pixel on the image. These pixels are the target variable.

Assign 0 to pixel: If NDVI and EVI values are below the threshold, this indicates the absence of the hyacinth.

Assign 1 to pixel: If NDVI and EVI values are above the threshold, this indicates the presence of the hyacinth.

Deciding on the threshold value:

- **Using statistical analysis (NDVI and EVI)**
  Within the region of interest
  - Water will have a low or negative NDVI and EVI values.
  - Non-vegetative surfaces (land) will have a low or closer to 0 NDVI and EVI values.
  - Vegetation (hyacinth) will have a moderate to high (0.2 to 0.8) NDVI and EVI values.

- **Manually setting the threshold**
  During winter months the water hyacinth levels are at their lowest (Bega, 2023), Using this fact the NDVI and EVI values of the year 2023 during the winter months will be read and their range recorded.

  This will then be double checked by viewing the images in this period to ensure that the mat is not present; once this is confirmed this range can be used as a threshold. However, the following considerations need to be placed:
  - Only NDVI and EVI pixel values from areas where the vegetation is not present needs to recorded.
  - The land EVI and NDVI values must be the same or approximately the same throughout the years, hence verification of this will be required.

  This will be an iterative process as it involves fine tuning.

1.1.6. Weather data processing steps

1.1.6.1. Correlate the images with the weather data

The Landsat satellites take images of the earth's surface once every 2 weeks (Google Earth Engine, 2024); this means that the correct weather data must be corresponded to the correct image of the dam.

This can be achieved by indexing the weather data using the date (down to the day), then joining the two datasets using the date index. This will ensure if there are any missing images or weather data, they will not be added.

Inner join two tables:

Date	Wind Speed (km/h)	Wind direction	Temperature (°C)	Gust (mph)
2020-02-13	...	NW	21	...
2021-03-03	5	...	...	11.2
2022-12-15	...	N	25	...
2023-05-18	7	...	5	10.8

⊓

Date	Pixel 1	....	Pixel n
2021-03-03	0	...	1
2024-01-01	1	1	0
2022-12-15	0	...	...

=

Date	Pixel 1	....	Pixel n
2021-03-03	0	...	1
2022-12-15	0	...	...

1.1.6.2. Check for missing values

At this point there must be no missing values in the resulting date column, however there may be missing values in the other columns. Below is a design for handling missing values in the other weather columns:

- **Numeric columns (wind speed, temperature and gust)**
  Normally missing numeric values would be replaced with the mean, mode or median of that specific column (depending on the % of missing values); however careful consideration needs to be placed regarding weather data.
  Instead of replacing missing values with the average for the whole specific column, missing values will be replaced with the average (mean) of each season.

- **Categorical column (Wind direction)**
  If there are any missing values in the categorical column (wind direction) these missing values will be replaced with the mode of each season. That is, if a missing value occurs in June that value will be replaced with the mode of the season W (winter). Possible values for wind direction:

Name	Cardinal Direction	In degrees
North	N	0 or 360
North-Northeast	NNE	22.5
Northeast	NE	45
East-Northeast	ENE	67.5

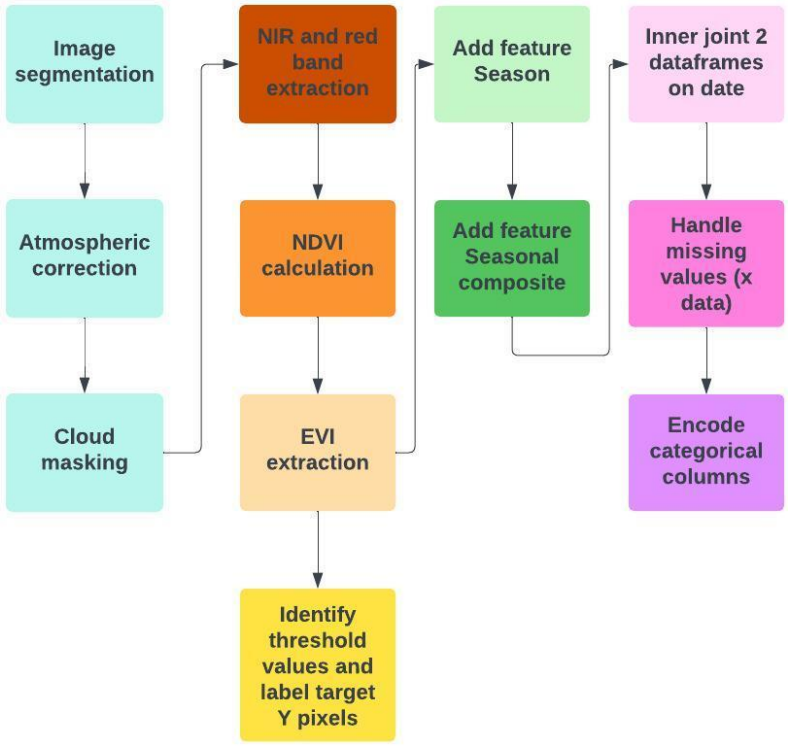
East	E	90
East-Southeast	ESE	112.5
Southeast	SE	135
South-Southeast	SSE	157.5
South	S	180
South-Southwest	SSW	202.5
Southwest	SW	225
West-Southwest	WSW	247.5
West	W	270
West-Northwest	WNW	292.5
Northwest	NW	315
North-Northwest	NNW	337.5

- **Drop date column**
  The date column and any existing index column will only hold unique values, this after the inner join these columns must be drooped as they will not provide any useful information in training the model.

1.1.7. Encoding

The final step in data processing is to encode any categorical columns. If there is an intrinsic order in the data then the specific column will be one-hot encoded; and if there isn't then the specific column will be label encoded.

1.1.8. Data processing flow diagram



Final Data frame structure before sending it to ML model:

Wind speed	Wind Direction (NW)	…	Wind Direction (SWS)	Temperature	Gust	Season (W)	…	Season (Sp)	Season composite (averages)	Pixel 0	…	Pixel 38 067 689
6	…	1	0	28.5	11.2	0	1	1	12.5	0	0	1
…	1	0	1	….	…	1	0	1	34.8	0	0	0
5	…	0	0	15.2	10.8	…	0	0	…	1	0	1

2. Testing and Validation

2.1. Data Processing

The chosen satellite (Landsat 8) images come already atmospherically corrected hence atmospheric correction will not be required.

2.1.1. Test case 1.1: Image segmentation

Input:	Satellite images with identified boundary.
Expected output:	A binary mask where the dam (ROI) is retained, and the surrounding area is set to zero (blanked out).
Method:	Print 3 randomly selected images before and after binary masking to test if masking is performed correctly.
Validation:	Visually plot the region of interest and check if the surrounding area is masked out.

2.1.2. Test case 3.1: Cloud masking

Input:	Satellite images containing clouds and shadows.
Expected output:	Images with clouds and shadows masked.
Method:	Use Fmask to remove clouds and shadows.
Validate:	Visualise the top 3 cloudy images from the dataset before and after applying the Fmask algorithm.

2.1.3. Test cases 4: Indices extractions

2.1.3.1. Test case 4.1: Extract NIR, blue and red bands

Input:	Raw satellite images.
Expected output:	Correctly extracted indices of each pixel in each image.
Method:	Locate the relevant bands in the dataset and extract their pixel values.
Validate:	Verify the ranges of values retired for each band with known standard values.

2.1.3.2. Test case 4.2: NDVI and EVI calculations

Input:	Extracted indices values
Expected output:	Accurate NDVI and EVI calculations
Method:	Use already existing python libraries to calculate these measurements.
Validate:	Validate these values by writing a function to calculate these values, then cross check these

	calculated values with the ones generated by the python library.

2.1.3.3. Test case 4.3: Seasonal feature creation

Input:	NDVI and EVI values
Expected output:	Correctly calculated seasonal averages
Method:	Group the NDVI and EVI values based on their seasons (from dates column); then average each season's indices values.
Validate:	Compare these averages NDVI and EVI values with the standard values.

2.1.4. Test case 5.1: Binary labelling assignment

Input:	NDVI, EVI and threshold values
Expected output:	Binary labelled pixel values
Method:	Place a 1 or 0 based on the value of the NDVI and EVI for each pixel relative to the threshold value.
Validate:	Visualise an image after cloud masking, for each season, with the pixels labelled 1 in red, and the rest in the normal colour. This will emphasis and missed vegetation as it will be in red. Visualize these very same images before binary labelling.

2.1.5. Weather data processing

Test Case 6.1: Weather data correlation

Input:	Weather data and corresponding satellite images data
Expected output:	Properly matched weather data for each image
Method:	Indexing and joining two data frames on the date column, standardize date column to remove time.
Validate:	Cross check dates and accuracy values, print the data at the quartiles of the data frame.

Test case 6.2: Handling Missing values

Input:	Entire data frame with weather and images.
Expected output:	Correctly imputed missing values.
Method:	Calculate missing values of the data frame and impute them accordingly.
Validate:	Calculate the missing values after imputation and check that there are no longer any missing values.

Test case 6.3: Encoding of Categorical Data

Input:	Categorical data columns
Expected output:	binary values with some additional new columns for one-hot encoded columns.

Method:	Use Sk-learn library to encode both categorical and numerical data.
Validate:	Check that the values are of binary nature.