Infovis group 2: Global Wind Power Tracker

Contents

[1 Introduction 1](#_Toc167049069)

[2 Data preprocessing 2](#_Toc167049070)

[3 Validation (RUTH) 4](#_Toc167049071)

[4 Product 4](#_Toc167049072)

[5 Evaluation (MISHKAT) 6](#_Toc167049073)

[6 Team organization 7](#_Toc167049076)

[7 Authors 7](#_Toc167049077)

[8 NOTES TO CREATE THE REPORT (remove in final version) 7](#_Toc167049078)

[Appendix A Project plan 8](#_Toc167049079)

[Appendix B Map visualization comparison table 9](#_Toc167049080)

[Appendix C How to deploy? 10](#_Toc167049081)

[Appendix D Repository structure 11](#_Toc167049082)

# Introduction

## Dataset selection

The Global Wind Power Tracker (GWPT)[[1]](#footnote-1) is a worldwide dataset of utility-scale, on and offshore wind facilities. It includes wind farm phases with capacities of 10 megawatts (MW) or more. A wind project phase is generally defined as a group of one or more wind turbines that are installed under one permit, one power purchase agreement, and typically come online at the same time. The GWPT dataset catalogs every wind farm phase at this capacity threshold of any status, including operating, announced, pre-construction, under construction, shelved, cancelled, mothballed, or retired. The dataset itself is subdivided into a ‘large’ (wind farms larger than 10MW) and ‘small’ (wind farms between 1 and 10MW) subset. The most recent release of this data was in December 2023.

## Dataset license

All Global Energy Monitor data are freely available under a [Creative Commons Attribution 4.0 International Public License](https://globalenergymonitor.org/creative-commons-public-license/) unless otherwise noted. Under this license, you are free to:

* Share: Copy and redistribute data in any medium or format
* Adapt: remix, transform, and build upon the data for any purpose

 Under the following terms:

* Attribution: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
* No Additional Restrictions: You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

## Target user

Our target user is anyone interested in wind power, with at least a high school level understanding of STEM subjects. They should be familiar with the concept of electrical power measured in Watts, the prefix "M" for million, and that windmills generate electrical energy. They should also know that wind farms can be located both onshore and offshore, and have a basic knowledge of world geography, including continents, regions, and countries.

## Goal

The goal is to provide our target users with an interactive web dashboard to help them explore various wind power-related questions, such as:

1. How is offshore wind currently distributed across different continents?
2. In future years, will the world invest more in onshore or offshore wind?
3. Can you list the top 20 largest operating wind farms in India commissioned between 2010 and 2020?
4. Can you locate the 3rd largest onshore wind farm in South Africa and retrieve its capacity value?

The dashboard is designed as an educational tool for exploring wind power data.

# Data preprocessing

## Exploratory data analysis

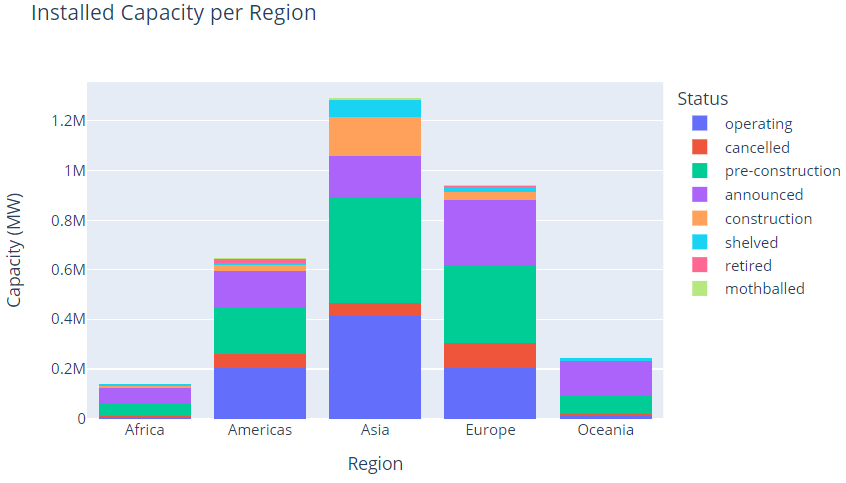
To gain a better understanding of the data, exploratory data analysis (EDA) was performed using both a Python Jupyter notebook (with pandas) and Excel workbooks. During EDA, an initial assessment of the raw data was conducted to understand its structure. Here’s what we found:

* The ‘large’ subset contains 26,523 wind farm phases.
* The ‘small’ subset contains 899 additional wind farm phases.
* Each subset has 29 features.
* The dataset contains wind farms from 155 distinct countries.
* Features are expressed either as strings (e.g., project name, country, owner) or floats (e.g., capacity, latitude, longitude).

Additionally, during EDA, missing values, outliers, duplicates, and inconsistencies within key features were identified:

* There were no missing data in any key features except for the start year. 9,919 observations (36.17% of the dataset) did not have a start year.
* The distinct values for 'Installation Type' were 'onshore', 'unknown', 'offshore mount unknown', 'offshore hard mount', and 'offshore floating', which are more detailed than necessary for our target users.
* Similarly, the distinct values for 'Status' were 'operating', 'cancelled', 'pre-construction', 'announced', 'construction', 'shelved', 'retired', and 'mothballed', which are also too detailed for our target users.

Finally, some visuals, like the one in INSERT REF were created to get a feeling for the distribution of the data.



## Data wrangling

Following EDA, data wrangling techniques were used to improve the quality of the data and create clean data ready for use in the dashboard .

**Simplification of categorical features:** The type column was reduced to ‘onshore’ and ‘offshore’. ‘Unkown’ units were visualised and put under ‘onshore’ as they were all located on land. The status column was reduced to ‘operating’, ‘future’ and ‘retired’. Cancelled and shelved projects were removed from the dataset as they provide no value for our target user.

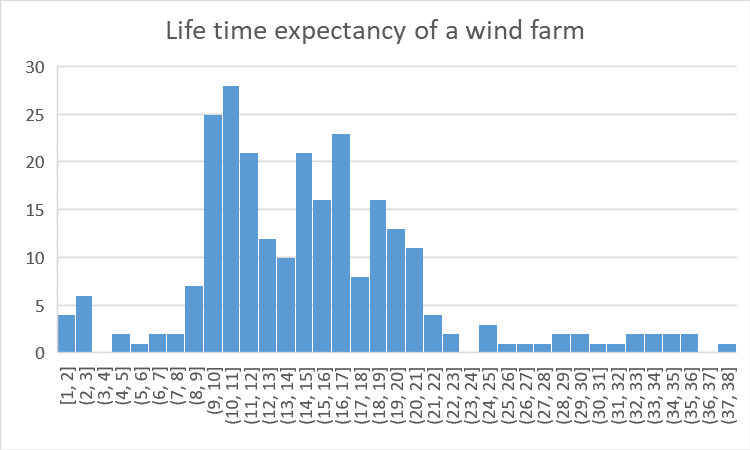
**Handling missing values:** Missing values were handled through imputation[[2]](#footnote-2) based on the nature of the data. Analysis on complete records has shown that the average life span of a wind farm is around 15 years. This knowledge, combined with the ‘status’ feature, which is always filled in, was used to impute the missing ‘start year’ data. INSERT REF

**Abbreviation:** The project names were shortened to avoid visual clutter in the dashboard. Common occurences like ‘wind farm’, ‘wind project’, ... were removed as this is clear from the context. This reduced the average project name length from 28 to 18 characters.

**Handling outliers:** Records with outliers in terms of installed capacity were removed. Based on business expertise[[3]](#footnote-3) and research on a sample of these records we deduced that units with a capacity of more than 10GW are likely unrealistic and should be removed.

**Data type harmonization.** Data types were converted into the desired format (datetime, integer, ...) before storing the final cleaned records in ‘.parquet‘ files.

By using a Jupyter notebook, this ETL process is completely self-documenting and maintainable.



# Validation (RUTH)

something about UEQ+ here??? WPO4

## Domain

lorem ipsum

## Abstraction

lorem ipsum

* make the problem statement more generic? lecture 04

## Visual encoding

lorem ipsum

chose visuals based on human capabilities – make the link lecture 04

## Algorithm

lorem ipsum

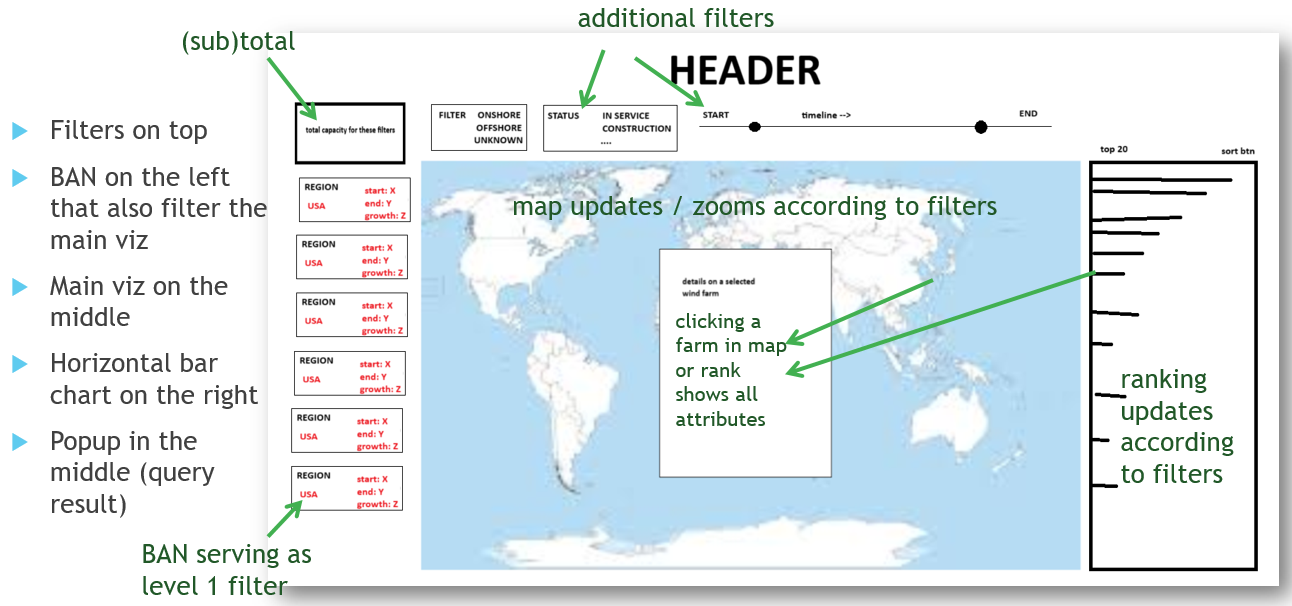
# Product

## Mockups

For inspiration we drew from the John Hopkins Covid dashboard Course example. During a team call, a mockup was created to agree on a target layout. From this sessions we concluded:

* Big aggregate numbers (BANs) should be used to show aggregate results, and be placed on the left.
* The filters should be on the top of the screen. They should impact the BAN, map and bar chart.
* The map should be the central piece of the dashboard and take up sufficient space.
* A ranked bar chart should be placed on the right and allow the user to explore the largest units.

Having made this mockup allowed us to collect feedback before creating the first prototype (e.g. some filters are missing).





## Framework choices

The interactive visualisation is built using the Dash and Plotly graphics.

Dash[[4]](#footnote-4) is a web framework for building interactive, web-based data visualization applications using Python. Dash is stateless, meaning that each user interaction with the application does not rely on the previous interactions, allowing for scalable and consistent performance regardless of the number of users. Dash is popular because it simplifies the process of building interactive, web-based data visualizations and applications with Python, making it accessible even for those with limited web development experience.

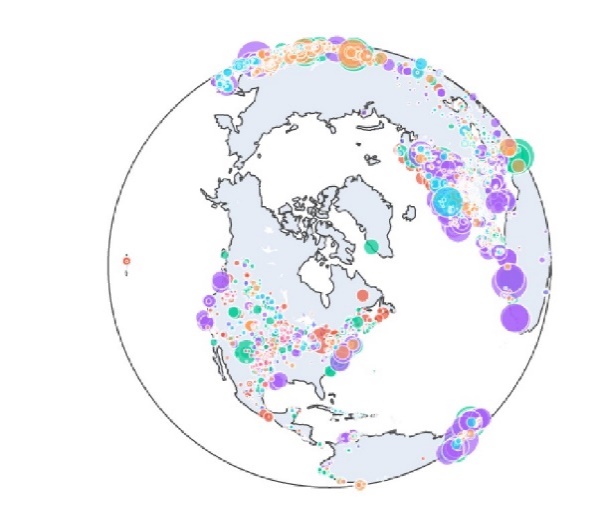
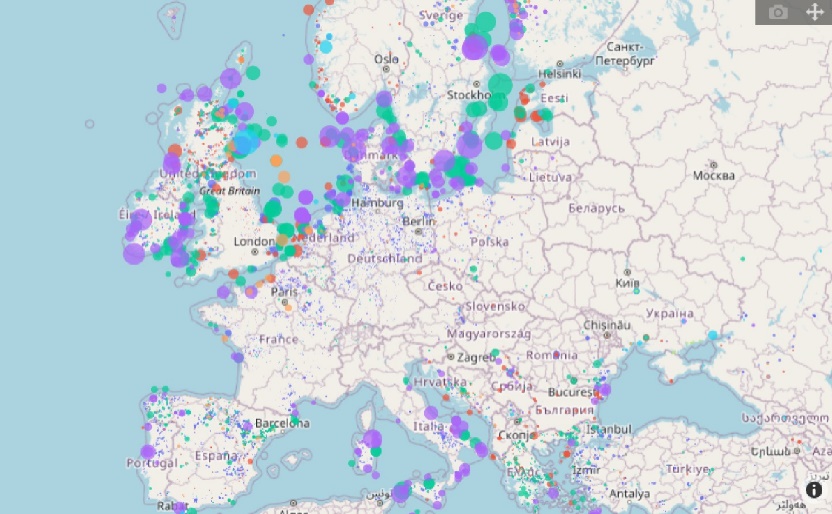
Plotly[[5]](#footnote-5) is an open-source graphing library that enables the creation of interactive, web-based visualizations with ease and flexibility. Plotly is versatile, offering numerous ways to create a wide range of interactive visualizations. Given its many bindings, Python developers can use Plotly without needing to know JavaScript.

Plotly and Dash integrate seamlessly.



## Map visualizations

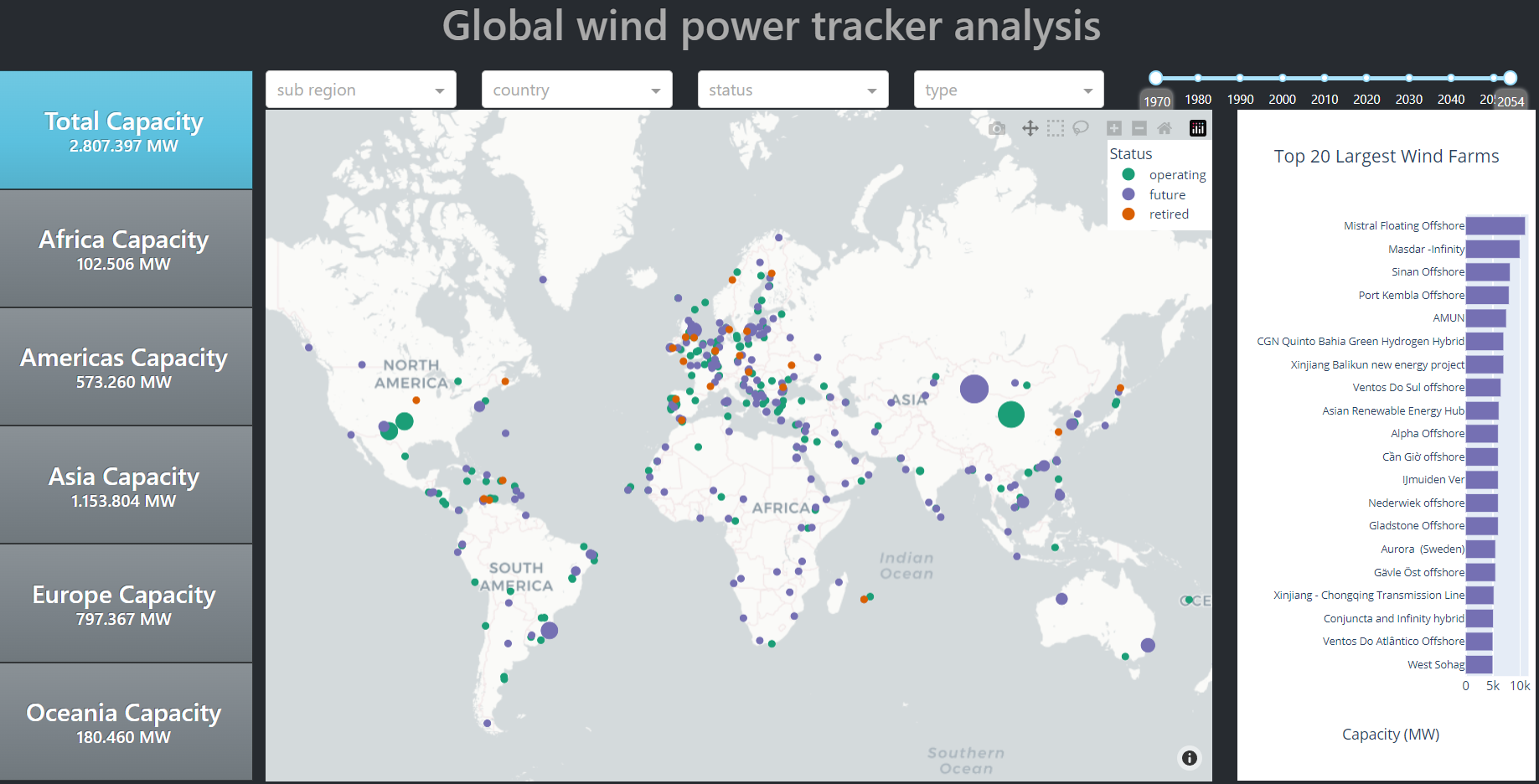
With a map as the centerpiece of the app, some research time was devoted to finding and testing the optimal map visualization framework. While a 3D representation may give off a modern first impression, it is inferior to a 2D representation due to occlusion. E.g. in image INSERT REF it is impossible to see what countries the circles on the north relate to.

A comparative table was found on Plotly’s website and copied in **appendix B.** Special attention was given to built-in semantic zooming support. In the end the choice was made to go forward with Plotly’s built-in support for map visualisations based on mapbox. Mapbox is a platform that provides tools and APIs for creating custom, interactive maps and geospatial visualizations for web and mobile applications. Two semantic zoom levels were implemented manually, to increase performance and maintain fine control over the location of the aggregate circles.

## Final product

some screenshots & clarification



# Evaluation (MISHKAT)

interview framework

results on the interviews

something about our learnings

like

# Team organization

The team established collaboration practices within the first week. This section outlines those agreements.

## Communication

For asynchronous communication, the team relied on a WhatsApp group. For synchronous communication, the team organized eight meetings via Microsoft Teams over the semester. Minutes from these meetings were recorded in OneNote and distributed by email.

## Collaboration

The team decided to use Git for version control and collaborated on the codebase through a public GitHub repository (https://github.com/jorritvm/infovis). Mainline development was performed without specific branching rules, and branches were only created for specific purposes, such as in-depth refactoring or proof-of-concept work for map visualization.

## Planning

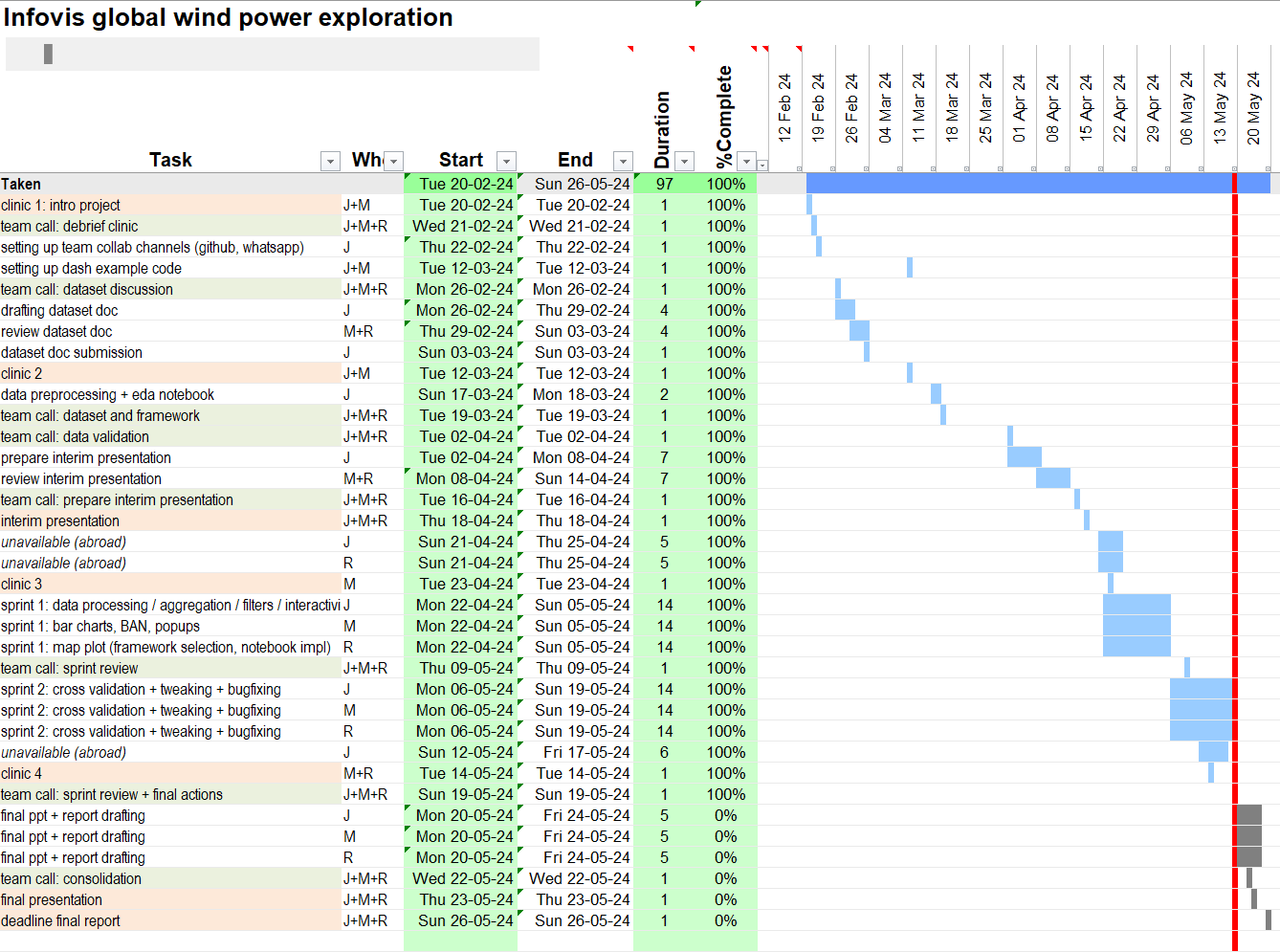
The team established and executed a project plan, which is visualized as a Gantt chart in Appendix A. After the mid-term presentation, the team adopted an agile methodology, dividing the remaining four weeks into two sprints. Each sprint included planning and review sessions. During the sprint reviews, bugs were discussed and assigned.

# Authors

|  |  |  |
| --- | --- | --- |
| Jorrit Vander Mynsbrugge | 0606134 | jorrit.jules.vander.mynsbrugge@vub.be |
| Ruth Vandeputte |  | ruth.vandeputte@vub.be |
| Mishkat Haider Chowdhury |  | mishkat.haider.chowdhury@vub.be |

# NOTES TO CREATE THE REPORT (remove in final version)

# Appendix A Project plan



# Appendix B Map visualization comparison table



# Appendix C How to deploy?

## Setting up your local copy

1. Clone the repo

git clone https://github.com/jorritvm/infovis.git

or unzip the provided archive

1. Make sure you're using python 3.11
2. Navigate to your project directory

cd infovis

1. Set up a virtual environment

python -m venv venv

1. Restore the dependencies in that virtual environment

pip install -r requirements.txt

1. Copy .env.template to .env and configure the required secrets

(optional if you started from the zip archive)

## Running the dashboard from the CLI

1. Activate the venv

venv\Scripts\activate

1. Navigate to the app directory:

cd app

1. Run the dash app:

python app.py

1. Explore the app using your web browser.

# Appendix D Repository structure

|  |  |
| --- | --- |
| infovis  ¦ .env  ¦ .env.template  ¦ README.md  ¦ requirements.txt  ¦  +---app  ¦ ¦ app.py  ¦ ¦  ¦ +---assets  ¦ ¦ +---css  ¦ ¦ ¦ styles.css  ¦ ¦  ¦ +---callbacks  ¦ ¦ ¦ cb\_bar\_chart.py  ¦ ¦ ¦ cb\_continent.py  ¦ ¦ ¦ cb\_country\_filter.py  ¦ ¦ ¦ cb\_map.py  ¦ ¦ ¦ cb\_status\_type.py  ¦ ¦ ¦ cb\_sub\_region.py  ¦ ¦  ¦ +---components  ¦ ¦ ¦ filters.py  ¦ ¦ ¦ visualisations.py  ¦ ¦  ¦ +---utils  ¦ ¦ utils.py  ¦  +---data  ¦ +---analysis  ¦ ¦ ...  ¦ ¦  ¦ +---clean  ¦ ¦ ...  ¦ ¦  ¦ +---etl  ¦ ¦ process\_gwpt\_data.ipynb  ¦ ¦  ¦ +---raw  ¦ Global-Wind-Power-Tracker-December-2023.xlsx  ¦  +---doc  ¦ +---assignment  ¦ +---dataset\_selection  ¦ +---final\_presentation  ¦ +---midterm\_presentation  ¦ +---mockup  ¦ +---planning  ¦ +---report  ¦ +---video  ¦  +---experimentation  ¦ ... | .env file read by app  .env template file  main dashboard app file  custom css definitions  callback handlers grouped by output  visual components of the dashboard like bootstrap dropdown filters and visuals  utility functions like filtering  folder containing manual data analysis  folder containing cleaned data in parquet & xlsx  folder containing the ETL pipeline in jupyter  folder containing the raw input data  self explanatory documentation folder  folder containing some map experiments |

1. https://globalenergymonitor.org/projects/global-wind-power-tracker/ [↑](#footnote-ref-1)
2. As discussed during the mid-term presentation, we did not simply remove these records but found a way to guess a reasonable value. [↑](#footnote-ref-2)
3. Two of our team members are active in the energy industry [↑](#footnote-ref-3)
4. https://dash.plotly.com/ [↑](#footnote-ref-4)
5. https://plotly.com/python/ [↑](#footnote-ref-5)