Datenbanken und Web-Techniken

Project: Photovoltaic System Web Application

TABLE OF CONTENTS

TABLE OF FIGURES

1. Introduction

A prototype web application that makes use of local meteorological data and current conditions while concentrating on a chosen manufacturer and product, allowing for more accurate and effective planning for solar installations.

2. Problem Statement

Effective planning and the use of digital technologies have become essential today for both personal and professional endeavours, notably in energy planning. The use of photovoltaic systems (PV/PS systems) is one viable remedy. To precisely assess and determine the usability and effectiveness of a solar system for a particular application, location, and manufacturer, however, presents a substantial difficulty. As a result, it is crucial to provide a comprehensive and simple computerized solution for better solar system planning.

3. Application Overview

The following is a brief overview of the application.

3.1. Enhancing User-Friendliness

The application stresses ease-of-use as a fundamental need to handle the complexity involved with solar system planning. A simple GUI (Graphical User Interface) will be used, guaranteeing a clear and user-friendly design that reduces distractions and promotes a positive user experience. Additionally, the system will have a short learning curve that will make it easy for users of all skill levels to browse and use its features. The goal is to increase accessibility of solar system planning to a larger group of people and enterprises by emphasizing usability.

3.2. Responsiveness

The application's seamless adaptation to different screen sizes and devices gives users a fluid and intuitive experience. The app's responsive design enables excellent operation and simple interaction whether users access it from a desktop computer, laptop, tablet, or smartphone. This careful attention to detail ensures that users may easily access the application's functions and move via its interface without experiencing any difficulties, thus improving overall usability and satisfaction with the application.

3.3. Streamlining Time and Resource Efficiency

Time and resources are extremely valuable in planning, and designing and testing solar systems is no exception. The web-based application has effectively optimized these procedures, resulting in a considerable reduction in the time and resources needed. Users may now quickly design and test solar systems thanks to the power of digital tools and algorithms. This expedited procedure enables both people and companies to take well-informed decisions and maximize their electricity planning without having to put up with lengthy delays or resource-intensive tasks.

3.4. Real-Life Photovoltaic Systems, Manufacturers, and Products

The web-based application excels at seamlessly integrating real-life photovoltaic systems, manufacturers, and products when it comes to the practical requirements of consumers. The ability for users to select their chosen manufacturer and product ensures that calculations and planning are perfectly in line with the equipment that they want to use. With the help of this

special function, customers may get realistic results that are personalized for their needs and correctly reflect the performance of the solar system they have chosen. The system gives consumers the knowledge they need to make wise decisions when installing solar systems in the areas they wish by simulating real-life scenarios.

3.5. Energy Generation Charts

In addition to calculations and planning, the application provides insightful visualizations. Users may visually explore their solar systems' hourly and daily energy generation statistics. Users can monitor patterns in the generation of energy and identify trends owing to these charts' clear and useful representation of the system's performance. Users can pinpoint peak generation periods, observe energy variations, and get an in-depth understanding of their system's overall efficiency by doing a complete analysis of the data. The user is also able to download the data for usage in other visualization tools or applications.

4. System Architecture

The application has both a frontend and backend. The frontend is a SPA (Single Page Application) [1] which renders views and handles user interaction. The backend handles processing of users', projects', products' data, weather API (Application Programming Interface) and database HTTP (Hyper Text Transfer Protocol) calls.

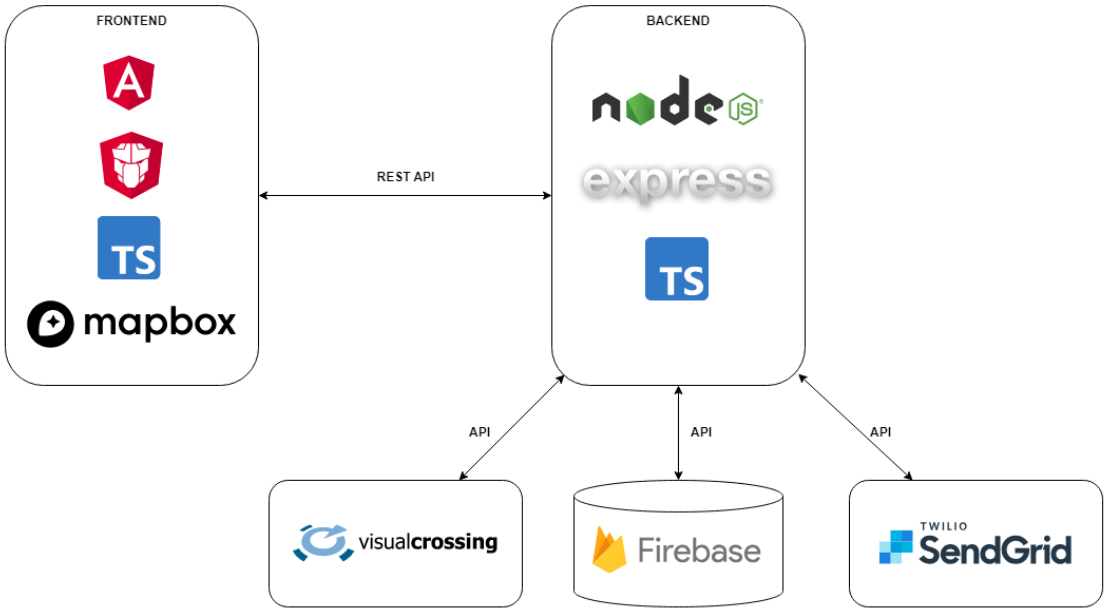


Figure 1 - System Architecture

4.1. Frontend Architecture

4.1.1. Angular, TypeScript, SCSS

The frontend user interface is developed using the Angular framework (v. 15.2) [2], incorporating TypeScript (v. 4.9) [3], and integrating SCSS (Sassy Cascading Style Sheets) [4] for enhanced styling capabilities.

Angular provides a robust and scalable platform suitable for building complex applications. Its extensive toolset, libraries, and pre-configured features accelerate development processes, resulting in enhanced productivity. The integration of TypeScript within Angular [5] brings the

advantages of static typing, improving code quality, facilitating easier maintenance, and enabling efficient refactoring. Furthermore, the component-based architecture of Angular framework promotes code reusability and modularity, allowing developers to create encapsulated and reusable UI components.

4.1.2. RxJS (Reactive Extensions for JavaScript)

RxJS [6], a complete library for reactive programming, is included with Angular [7]. RxJS enables developers to perform asynchronous operations, manage data streams, and design dynamic user interfaces. It provides a diverse set of tools for event handling, data transformation, and state management. The combination of Angular with RxJS allows developers to easily create high-performance and interactive applications.

4.1.3. PrimeNG

PrimeNG (v. 15.3) [8] stands out as an exceptional user interface library specifically crafted for seamless integration with Angular. It offers an extensive collection of UI components and functionalities that empower developers to create visually beautiful and highly professional web applications. With PrimeNG, developers gain access to a diverse range of pre-built components, such as grids, forms, charts, and calendars, removing the need to start from scratch. PrimeNG benefits from an active and supportive community, ensuring reliable updates, comprehensive documentation, and prompt assistance, thereby facilitating a smooth integration experience within Angular projects. The library also allows the developers to fine-tune the appearance and behaviour of components to perfectly align with their unique application requirements.

4.1.4. PrimeIcons

The icon library PrimeIcons (v. 6.0) [9] provides a huge selection of premium icons that have been tailored for web applications. Because of its adaptability, developers can effortlessly enhance the aesthetics and user experience of their applications by integrating a broad variety of icons into different components. A consistent and coherent icon experience across all platforms is made possible by PrimeIcons' easy connection with PrimeNG. For developers looking to improve the design of their projects, this library makes it simple to add high quality icons to web applications.

4.1.5. Mapbox GL JS

Mapbox GL JS (v. 2.15) [10] [11] is an advanced mapping library written in JavaScript that provides developers the tools and resources they require to build dynamic and interactive maps for mobile and web applications. It has several features, such as customized map styles, data visualization, and interactive map interactions. Developers can utilize Mapbox GL JS to effortlessly incorporate maps into their applications, change the appearance and behaviour of maps to meet their individual needs, and take use of advanced geospatial functions such as routing and geocoding. Mapbox GL enables developers to create engaging mapping experiences, making it an invaluable library for designing aesthetically appealing applications.

4.2. Backend Architecture

4.2.1. Node.js, TypeScript

Node.js (v. 16.18) [12] and TypeScript (v. 5.0) [3] form a powerful combination for server-side development. Node.js, a widely embraced JavaScript runtime environment, provides a scalable platform to build robust server applications and APIs. TypeScript, a superset of JavaScript, elevates the development experience with static typing and advanced features. When utilized together, developers can leverage the benefits of TypeScript's strong typing, improved code maintainability, and enhanced tooling support within the Node.js ecosystem. This seamless integration enables the creation of efficient and reliable server-side applications, empowering developers to deliver high-quality solutions.

4.2.2. Express

Express (v. 4.18) [13] is a Node.js-specific web application framework. It provides a lightweight and fast approach to web application development, offering developers with a seamless experience via user-friendly APIs and minimalist design. Express enables effortless management of routing, middleware integration, and server-side logic. Its modular and adaptable architecture allows for easy customization and expansion. With its wide adoption in the Node.js community, Express has established itself as a reliable and performant framework, enriched by a diverse ecosystem of plugins and middleware. From small-scale APIs to large-scale projects, Express empowers developers to create exceptional web applications with confidence and efficiency.

4.2.3. REST API

The REST API (Representational State Transfer Application Programming Interface) [14] is a popular architectural approach for developing web services. It provides a standardized method for client and server applications to communicate that is not complicated. REST APIs use HTTP methods for performing resource operations, GET for getting, POST for creating, PUT for updating and DELETE for deleting a resource. These APIs use unique URIs to identify resources and return data in forms such as JSON or XML. REST API is a popular solution for developing scalable and interoperable systems that can interface with a variety of technologies. It plays a vital role in modern web development, enabling the exchange of data and facilitating the creation of versatile applications.

4.2.4. Firebase Authentication

Firebase's authentication [15] system ensures the secure transmission of user credentials by employing industry-standard protocols. It has completed the ISO 27001, 27017, 27018 , SOC 1, SOC 2, and SOC 3 process [16]. This cryptographic mechanism protects the confidentiality and integrity of data exchanged during sign-in or sign-up processes.

4.2.5. JWT Session Management

JWT (JSON Web Token) (v. 9.0) [17] session management involves authentication, token storage, token verification, and session expiration. During authentication, user credentials are verified, resulting in a JWT. This token is stored by the client and sent to the server in subsequent requests. The server verifies the token's authenticity and extracts user details. JWTs can include an expiration time for session regulation. If a session expires, users need to

re-authenticate. Proper management of these components ensures a secure JWT-based session management system.

4.2.6. Cloud Firestore

To manage and store data, the application conveniently incorporates Cloud Firestore [18] database. It is a robust and flexible cloud-based NoSQL database provided by Google Cloud [19]. The benefits of cloud databases are scalability, dependability, and accessibility. Data may be swiftly accessed by users from any place with an internet connection, enabling collaboration and data synchronization between various devices. By utilizing a cloud database's capabilities, the application enhances data management efficiency and offers a reliable and user-friendly experience. It is being used to store the application's data which includes information about the real-life photovoltaic systems, users' projects and sessions, and the weather data.

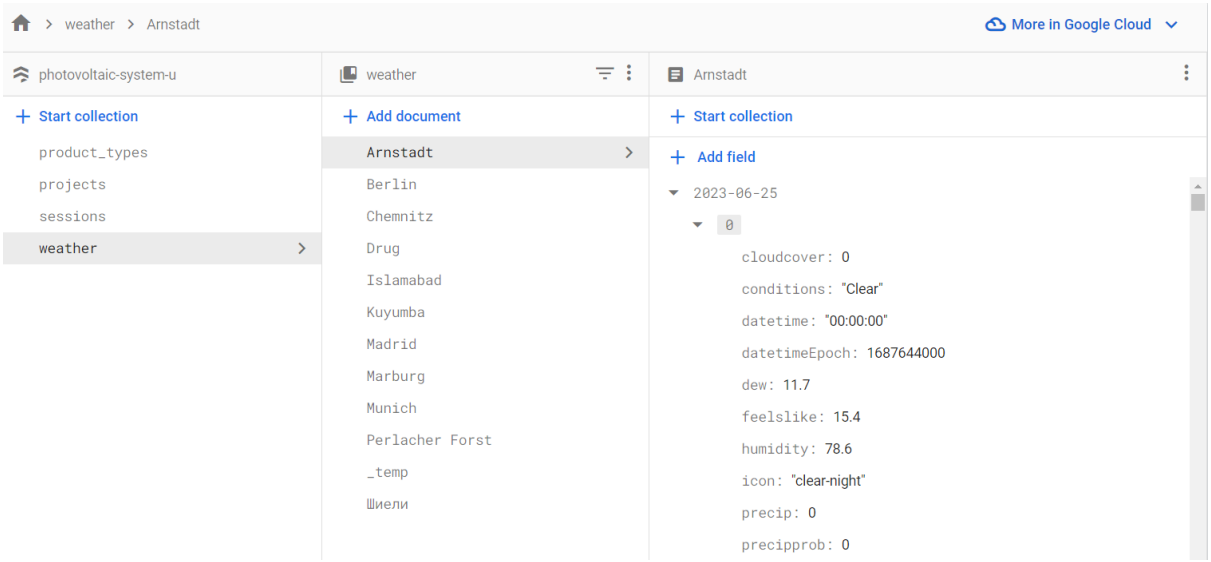


Figure 2 - Database Structure

4.2.7. Cron Job

Cron [20] job, a powerful feature for task automation and scheduling, serves as an invaluable tool within applications. The seamless integration of cron job functionality with Node.js empowers developers to automate repetitive tasks, improving application functionality, and streamlining overall operations. The combination of Node.js and cron job functionality provides an efficient approach to task automation, enhancing productivity and enabling smoother application management.

The cron job is being run at 12.05 AM (UTC +0200) every night. It first gets a list of all products in all projects per user. It then loops over every product and checks its and its project's status. If they are of active status, then it checks whether 30 days have passed since the creation of the product. If yes, it generates an electricity report for the product for the last 30 days, sends the CSV report to the user, and sets the product's status as inactive. If all the products in the project are now of inactive state, then it sets the status of the project as inactive too. If no, then it gets the weather forecast for the current date for the region of the product (if not already present) and saves it in the database.

4.2.8. Visual Crossing (Weather API)

Visual Crossing [21] is a Weather API that provides access to a wealth of weather data. It allows developers to retrieve weather information, including solar radiance, temperature, precipitation, wind speed, humidity, etc.

A service is created on the backend which provides an abstraction to use this API with ease according to the application requirements. A GET request is sent to the API to get the forecasted weather of current data or get the historical weather for the last 30 days. The response from the API also has per hour weather data. A request for a date is only sent when the weather data for the region for that date is already not present in the database.

Figure 3 - Visual Crossing API Response

4.2.9. TWILIO SendGrid

TWILIO SendGrid is a cloud-based email delivery platform that enables developers to programmatically send emails. It provides a reliable and scalable infrastructure for delivering large quantities of emails while guaranteeing that messages are delivered to their intended recipients.

5. User Interface

The application has the following components:

1. Sign Up Page
2. Sign In Page
3. Profile Page
4. Projects Page
5. Help Dialog
6. Dashboard

When a project is chosen, the project's name is shown in the header. The users can also sign out from here.

5.1. Sign Up

The users can sign up using their email address and a strong password for registering themselves into the web application.

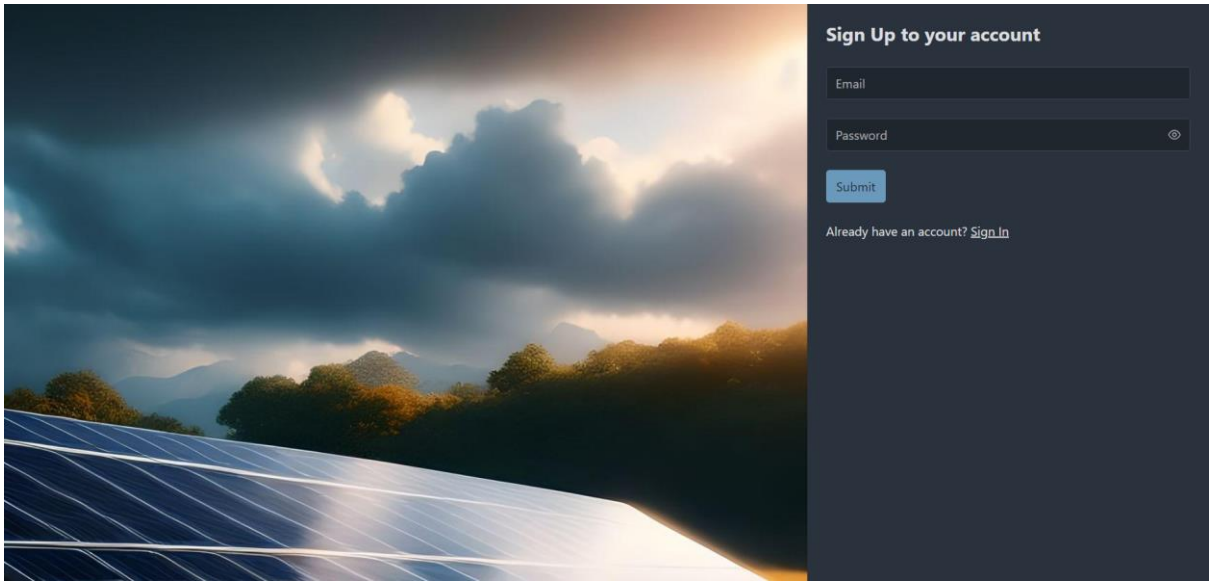


Figure 4 - Sign Up Page

5.2. Sign In

The users can sign in using the email address and password they set up when registering.

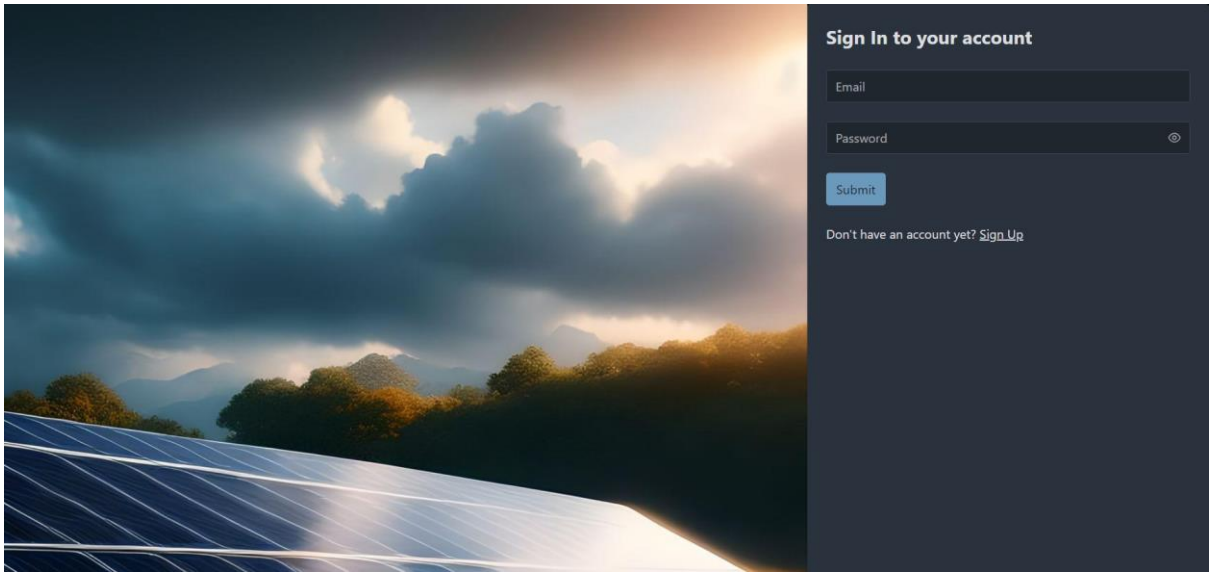


Figure 5 - Sign In Page

5.3. Profile

The profile page allows users to manage their personal information efficiently.

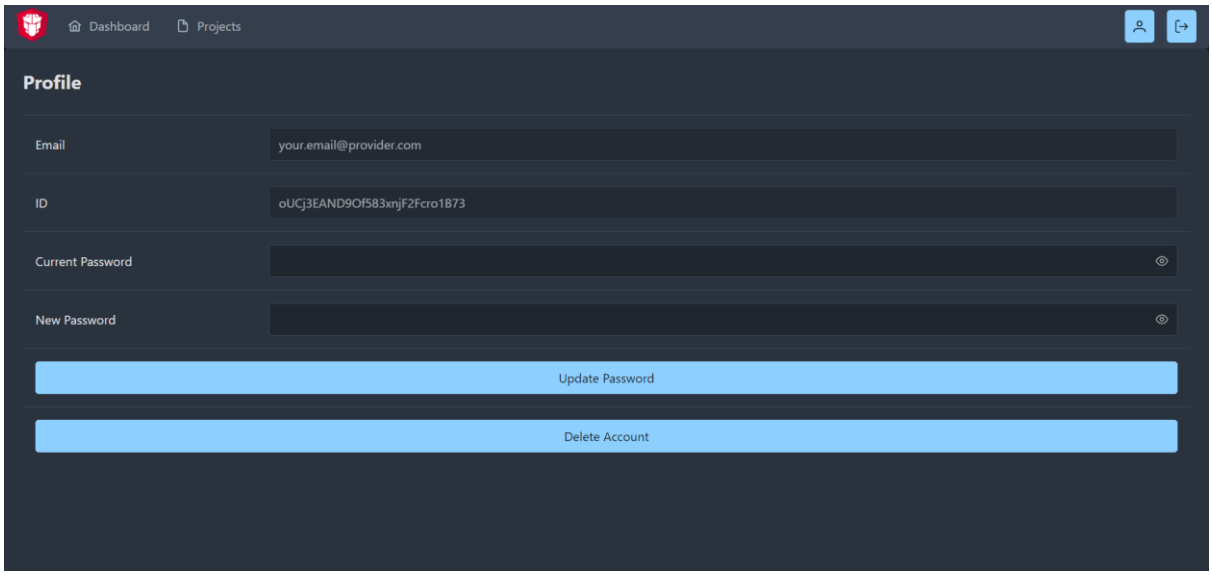


Figure 6 - Profile Page

- **Update Profile:** The users can update their profile by changing their password.
- **Delete Profile:** The users can delete their profile.

5.4. Projects

The projects page enables users to manage, organize and view their projects efficiently.

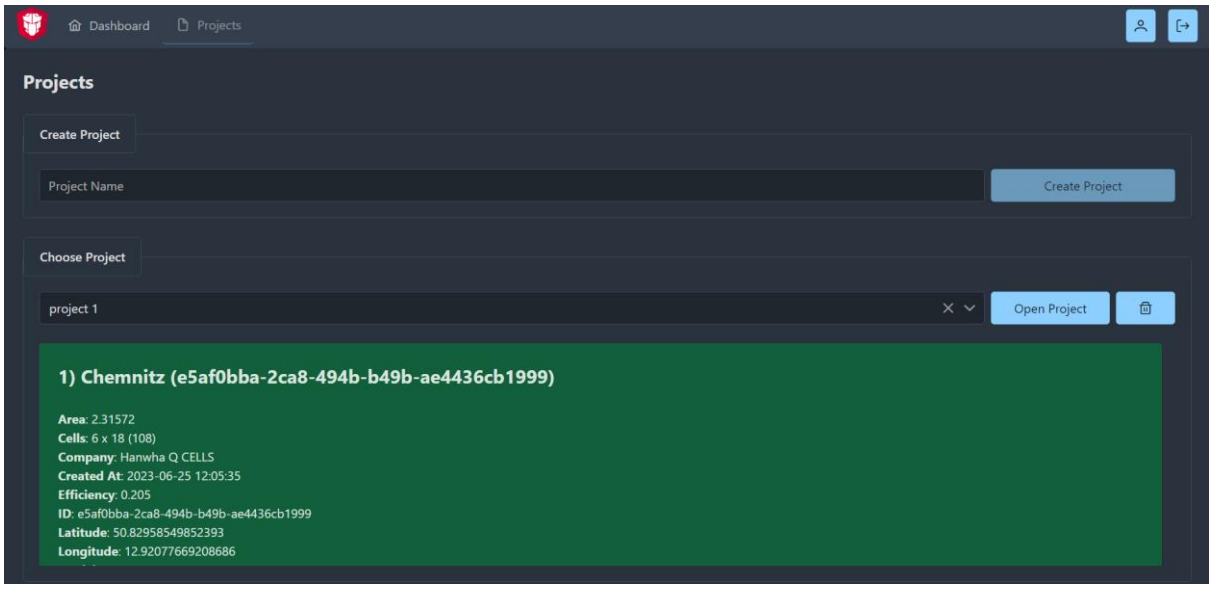


Figure 7 - Projects Page

- **Create Project:** The users can create a new project. The name of the project should be unique, otherwise an error is shown.
- **Choose Project:** The users can choose an already created project. The projects are grouped based on their status, i.e., whether they are active or inactive. An overview of the products in the project is also provided when a project is chosen. A visual representation of the status of the products is also present. Products with green background indicate an active status and those with red background indicate an inactive one.
- **Delete Project:** The users can delete their projects by clicking on the trash icon.

5.5. Help Dialog

On clicking on the “Help” icon in the header, a dialog is displayed which shows two components.

5.5.1. Usage

This component informs the user how to use the application. It lists downs the steps in a concise and user-friendly manner.

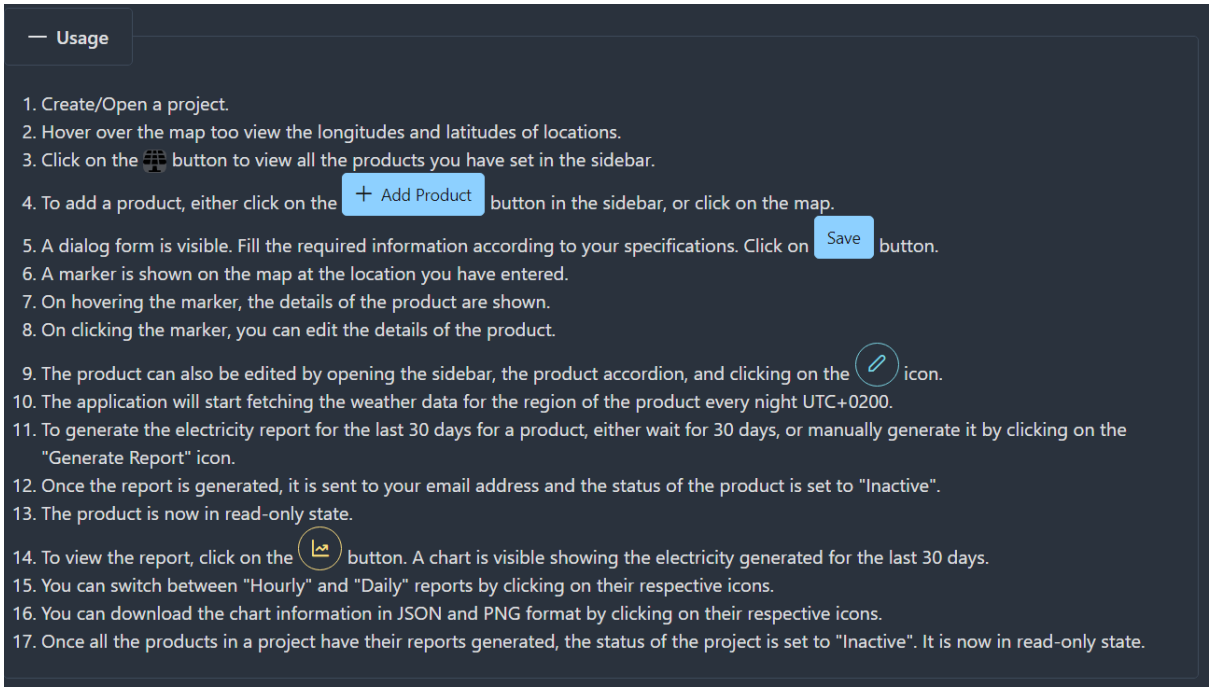


Figure 8 - Application Usage Details

5.5.2. Attribution

This component is used to provide references to the libraries, technologies and web services used the application.

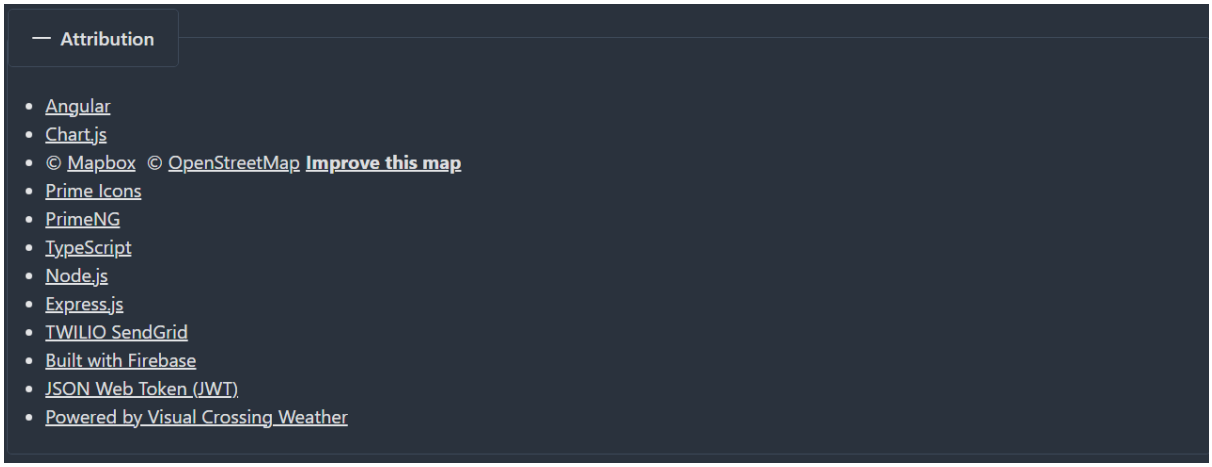


Figure 9 - Attribution

5.6. Dashboard

The dashboard provides the major functionality of the application.

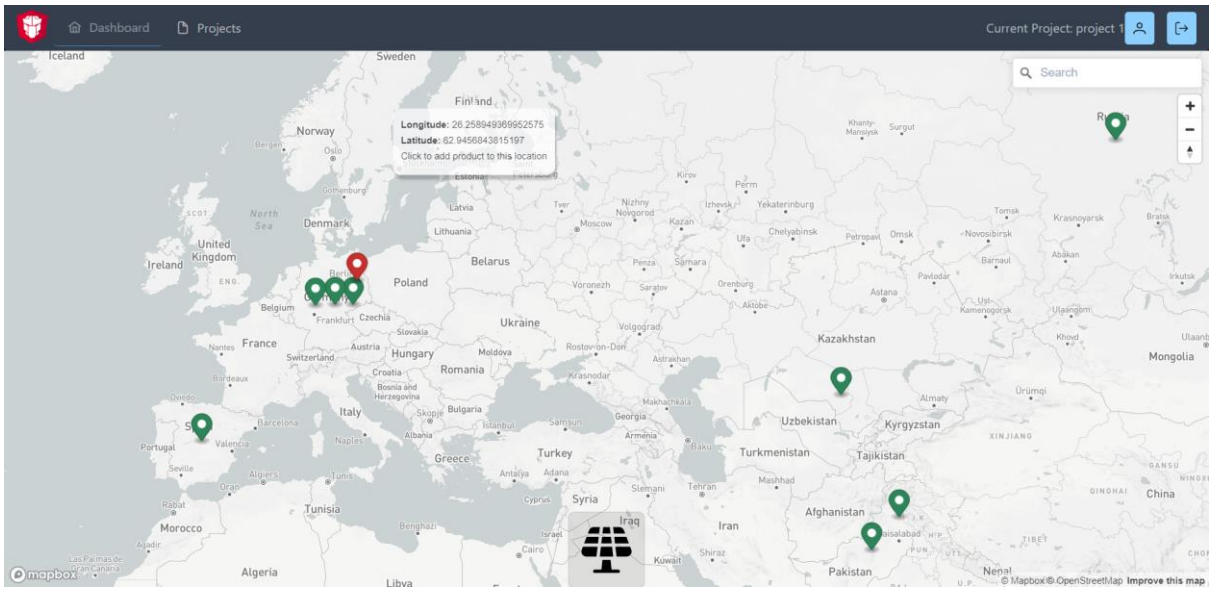


Figure 10 - Dashboard Page

5.6.1. Map

5.6.1.1. Hover

The users can view the latitude and longitude of the location their mouse pointer is hovering on the map.

5.6.1.2. Click

The users can add a product at the latitude and longitude of the location their mouse point is clicked on the map.

5.6.1.3. Map Markers

The users can see the markers of the products on the map. A green marker means an active product and a red marker shows an inactive one. On hovering the marker, user can see the details of the product.

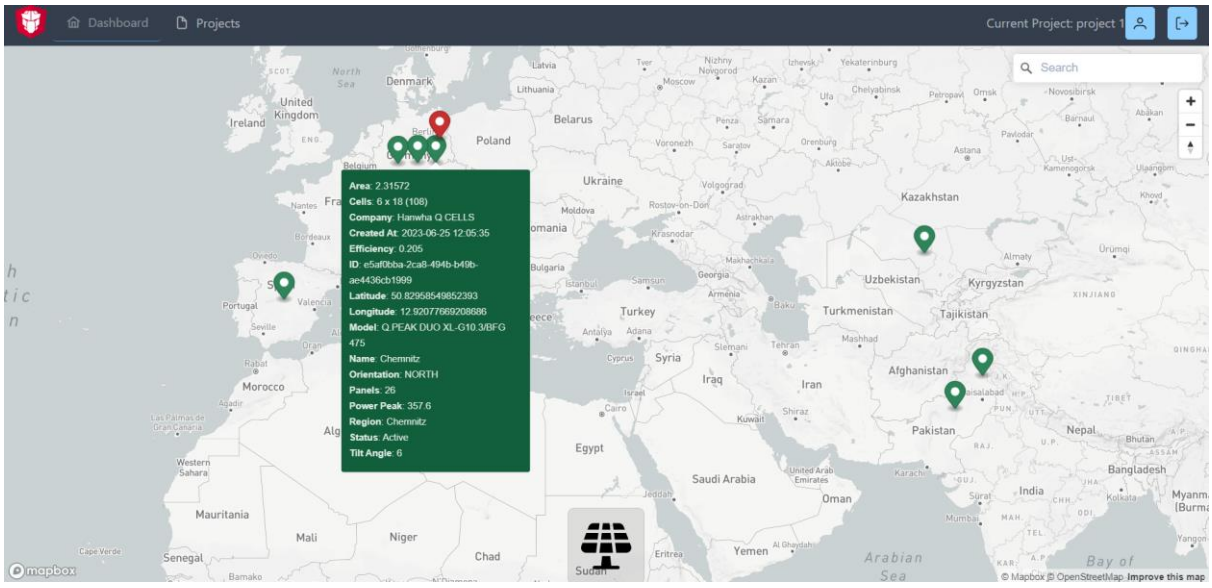


Figure 11 - Map Marker Hover

On clicking the marker, user can edit, delete, and generate the electricity report for the product. If the product has status active, the user can edit the product, else the user can only view the product details and view the electricity report.

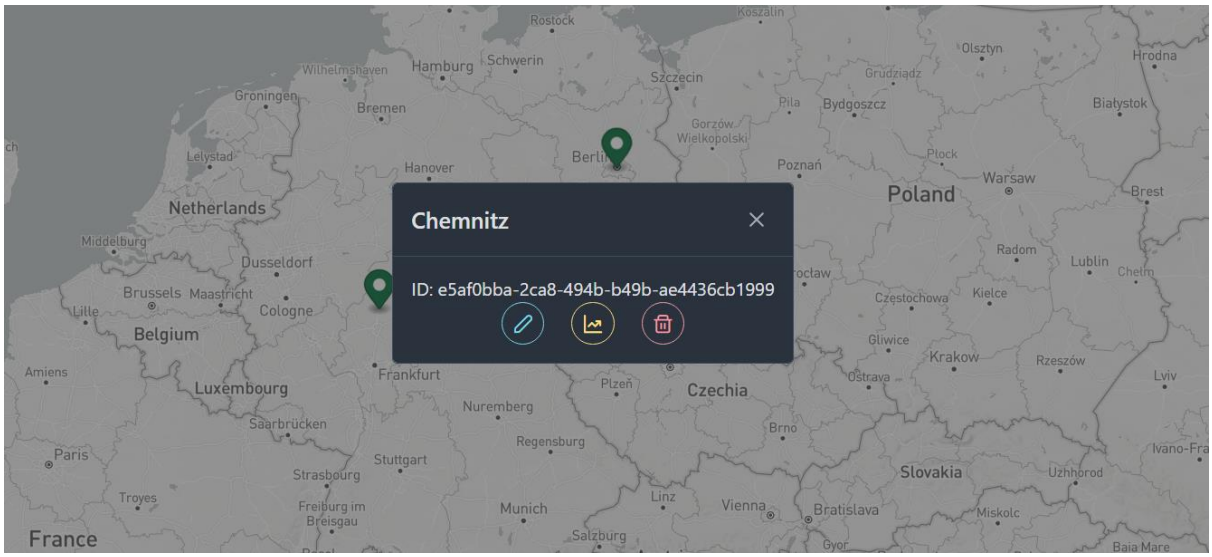


Figure 12 - Map Marker Click (Product Active)

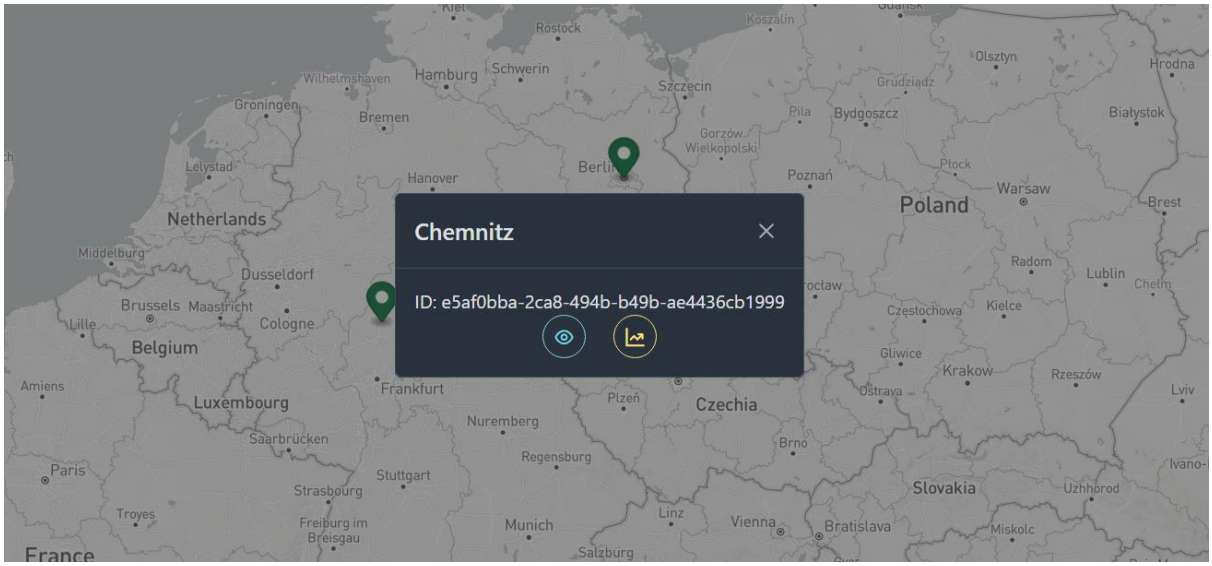


Figure 13 - Map Marker Click (Product Inactive)

5.6.1.4. Map Controls

On the right side of the map, map controls are located.

5.6.1.4.1. Geocoder Control

The users can use the geocoder to search the name of the place on which they want to set up a product.

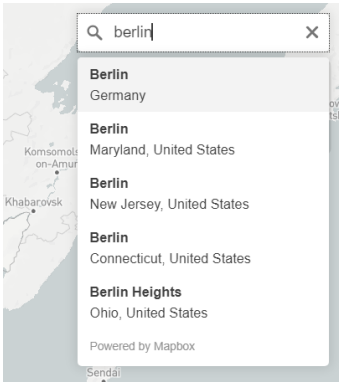


Figure 14 - Geocoder Control

5.6.1.4.2. Navigation Control

The users can use the navigation controls to zoom in, zoom out and visualize the current compass bearing. Mouse scroll wheel can also be used to zoom in and out of the map.

Figure 15 - Navigation Control

5.6.1.4.3. Geolocate Control

The users can view their current location on the map using the capabilities of GPS (Global Positioning System).

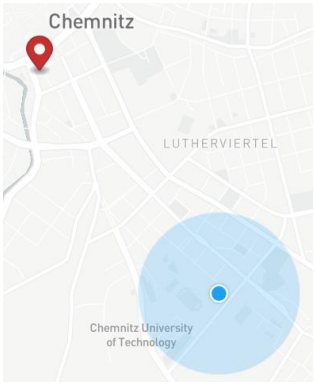


Figure 16 - Geolocate Control

5.6.1.4.4. Scale Control

The users can we the map scale (metric units) on the bottom right of the map.

Figure 17 - Scale Control

5.6.2. Products

On clicking the products icon in the bottom of the dashboard, a sidebar is toggled.

The sidebar has a button to add a product and lists down all the products in the project in an accordion format. The according header has the name of the product, a status badge and colour indicating the status of the product. A green text colour represents an active product, and a red colour represents an inactive one.

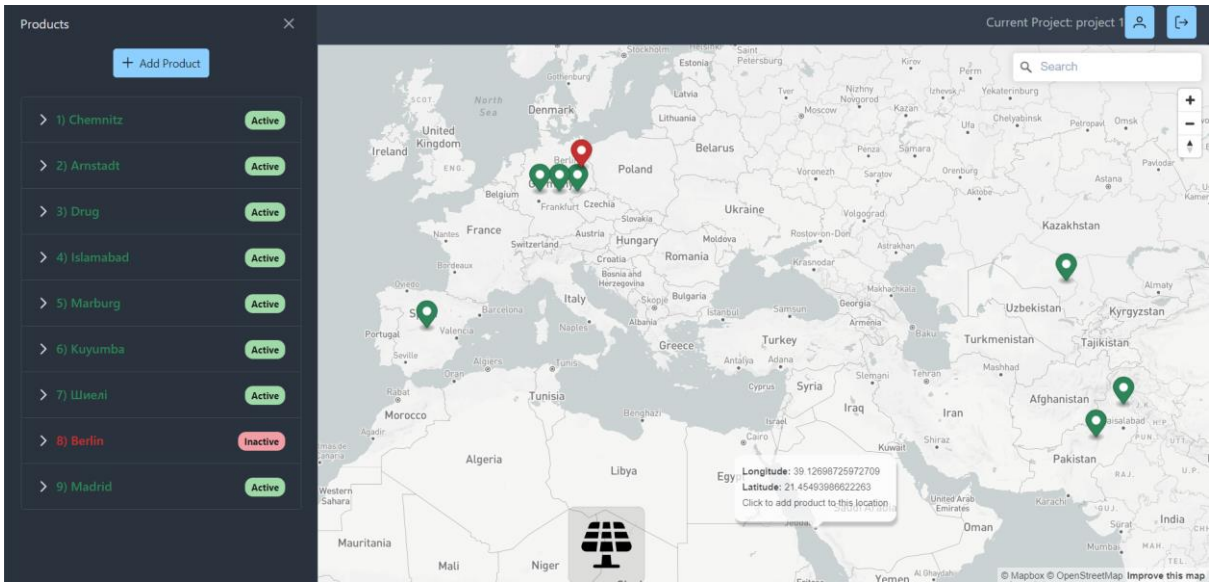
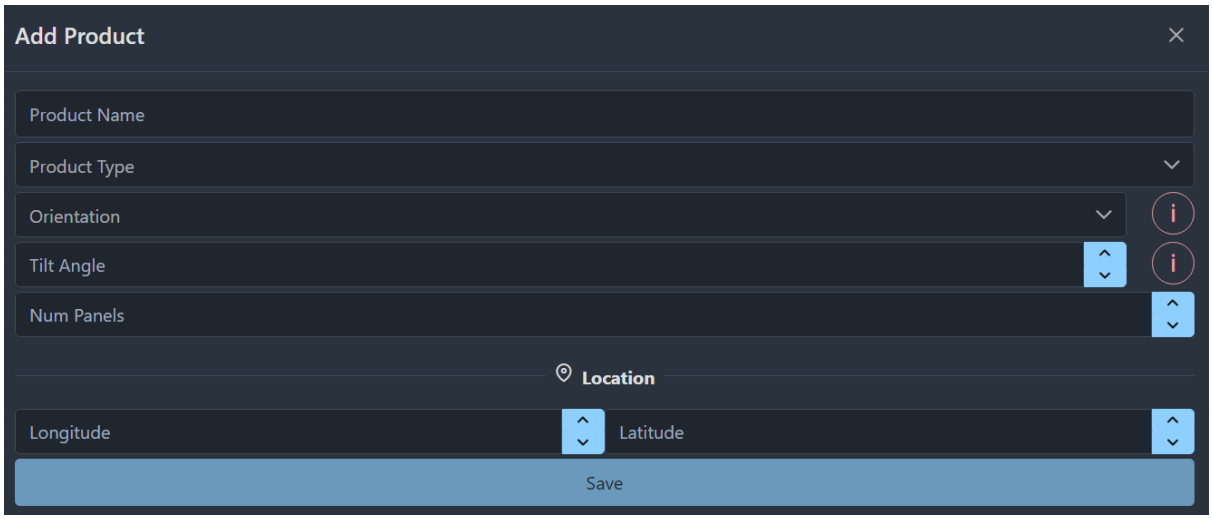


Figure 18 - Products Sidebar

5.6.2.1. Add Product

On clicking the “Add Product” button, a dialog is shown in which user can enter configuration of the product.



The user is also able to view the different orientation and tilt angle factors on clicking the respective info buttons. If the user has entered some value in these fields, then the respective row will also be highlighted.

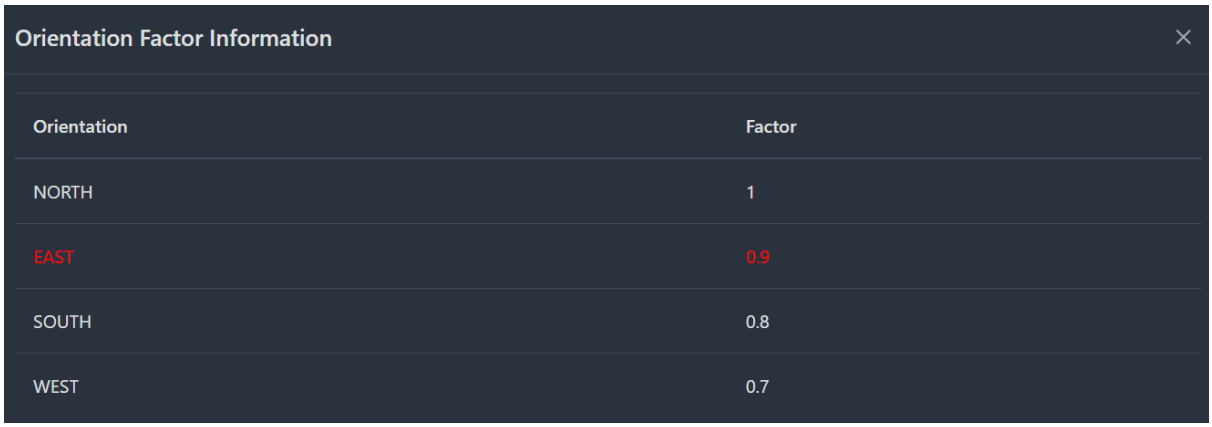


Figure 19 - Orientation Factor Information

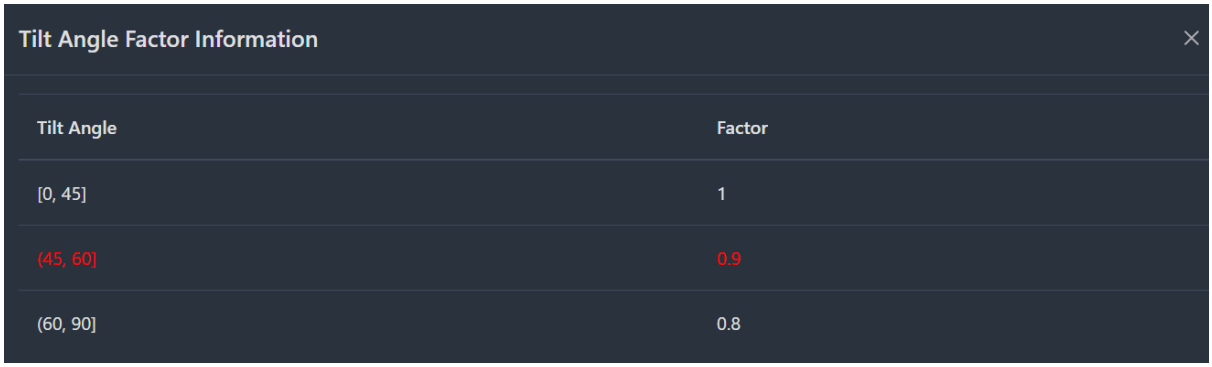


Figure 20 - Tilt Angle Factor Information

5.6.2.2. Product Accordion

On opening the product's accordion, the map is zoomed into the location and the product's marker has a glowing animation which lasts for 5.5 seconds. On closing it, the map is zoomed out. The details of the product are listed down in a tabular format.

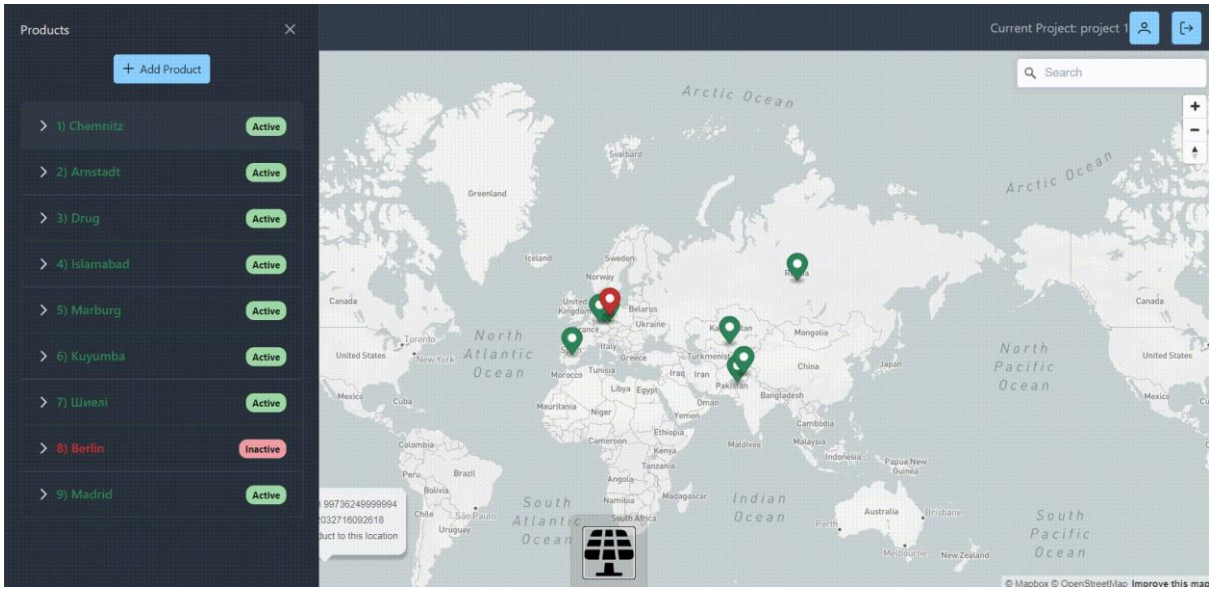


Figure 21 - Product's Accordion Opening, Closing

5.6.2.3. Buttons

If the product is in active state, the buttons can be used to view the product on the map, edit and delete the product. A button can also be used to generate the report for the electricity generated using the product details and the location's weather data for the last 30 days.

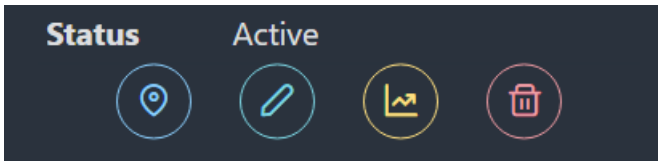


Figure 22 - Active Product's Buttons

If the product is in inactive state, the buttons can only be used to view the product on the map, viewing the details of the product and the already generated report.

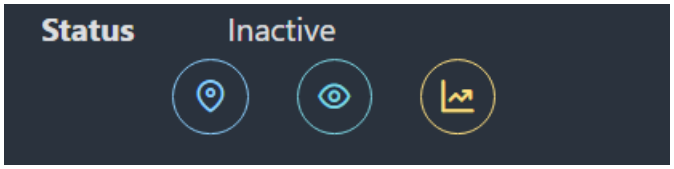


Figure 23 - Inactive Product's Buttons

5.6.2.4. Electricity Generation Report

On clicking the "Generate Report" button, a report is generated for the electricity generated using the product details and the location's weather data for the last 30 days. If 30 days have already passed since the creation of the product in the project, the server automatically generates the report and sets the status of the product to inactive. If all the products in the projects are of inactive status, then the status of the project is also set to inactive.

The electricity is generated using the following formula [22] [23] [24]:

```
/**
 * Calculates the orientation factor based on the orientation of the solar panel.
 * @param orientation - Orientation of the solar panel ("NORTH", "EAST", "SOUTH", "WEST")
 * @returns The orientation factor as a decimal value.
 */
private calculateOrientationFactor(orientation: string): number {
    let orientationFactor = 1.0;

    switch (orientation) {
        case "NORTH":
            orientationFactor = 1.0;
            break;
        case "EAST":
            orientationFactor = 0.9;
            break;
        case "SOUTH":
            orientationFactor = 0.8;
            break;
        case "WEST":
            orientationFactor = 0.7;
            break;
        default:
            break;
    }
    return orientationFactor;
}
```

Figure 24 - Orientation Factor Calculator

```
/**
 * Calculates the inclination factor based on the tilt angle of the solar panel.
 * @param tiltAngle - Tilt angle of the solar panel in degrees.
 * @returns The inclination factor as a decimal value.
 */
private calculateTiltAngleFactor(tiltAngle: number): number {
    let tiltAngleFactor = 1.0;

    if (tiltAngle >= 0 && tiltAngle <= 45) {
        tiltAngleFactor = 1.0;
    }
    else if (tiltAngle > 45 && tiltAngle <= 60) {
        tiltAngleFactor = 0.9;
    }
    else if (tiltAngle > 60) {
        tiltAngleFactor = 0.8;
    }
    return tiltAngleFactor;
}
```

Figure 25 - Tilt Angle Factor Calculator

```
/**
 * Calculates the electricity generated by solar panels.
 * @param solarIrradiance - Solar irradiance in watts per square meter (W/m²).
 * @param powerPeak - Maximum power output of the solar panel in watts (W).
 * @param orientation - Orientation of the solar panel ("NORTH", "SOUTH", "EAST", "WEST").
 * @param tiltAngle - Tilt angle of the solar panel in degrees.
 * @param areaPerPanel - Area per cell of the solar panel in square meters (m²).
 * @param numPanels - Total numbers of panels being used.
 * @param powerConversionEfficiency - Power conversion efficiency of the solar panel as a decimal value.
 * @returns The electricity generated in kilowatt-hours (kWh).
 */
public calculateElectricityProduced(
    solarIrradiance: number,
    powerPeak: number,
    orientation: string,
    tiltAngle: number,
    areaPerPanel: number,
    numPanels: number,
    powerConversionEfficiency: number
): number {
    const adjustedPowerPeak = powerPeak * powerConversionEfficiency;
    const adjustedSolarIrradiance = adjustedPowerPeak * (solarIrradiance / this.referenceIrradiance);
    const orientationFactor = this.calculateOrientationFactor(orientation);
    const tiltAngleFactor = this.calculateTiltAngleFactor(tiltAngle);
    const totalPanelsArea = numPanels * areaPerPanel;

    const electricityProduced =
        adjustedSolarIrradiance *
        orientationFactor *
        tiltAngleFactor *
        totalPanelsArea;

    return electricityProduced / 1000;
}
```

Figure 26 - Electricity Generation Formula

The report can be visualized as a chart in both the hourly and daily formats. The report can be zoomed in, out and panned. It can also be downloaded in JSON and PNG formats using the application.

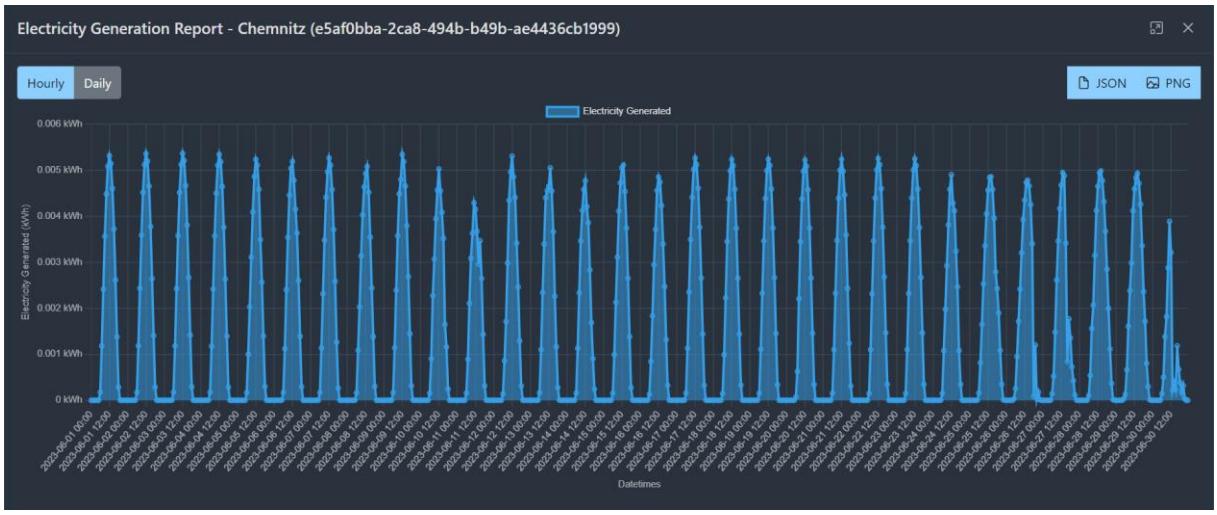


Figure 27 - Electricity Generation Report (Hourly)

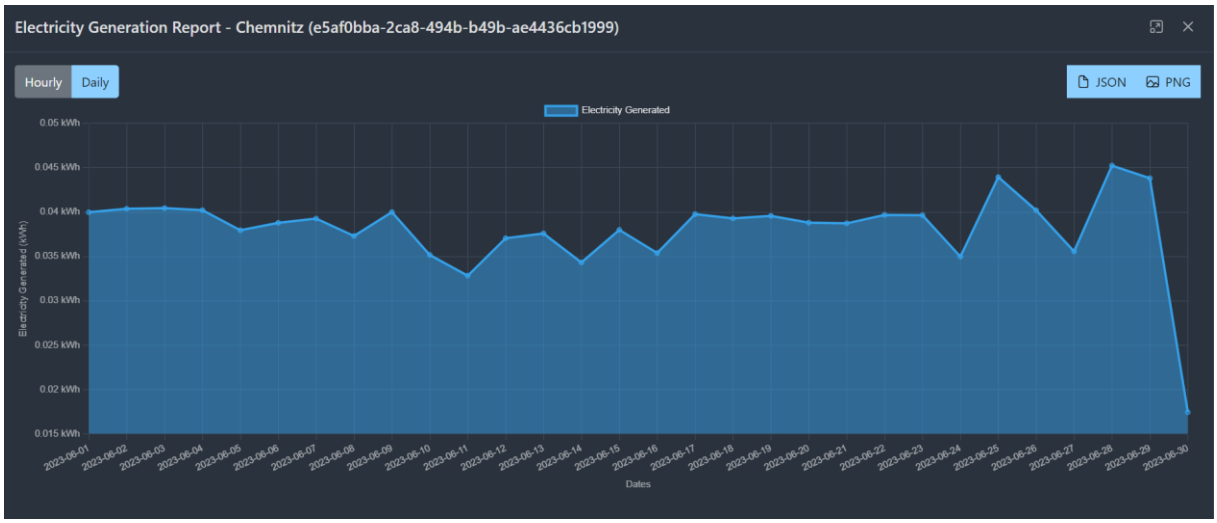
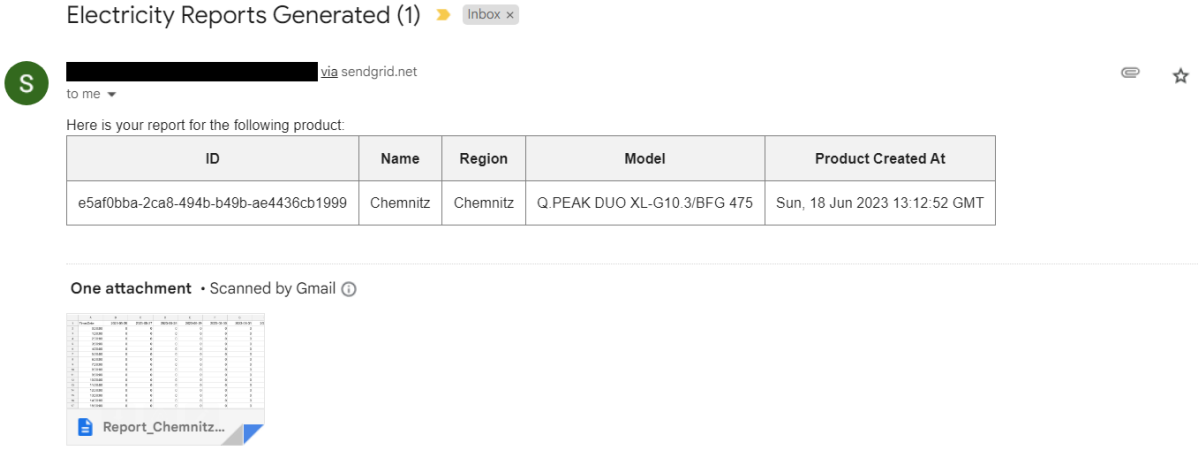


Figure 28 - Electricity Generation Report (Daily)

The report is also sent as an email to the user's email address.

- **Subject**: Number of reports generated
- **Body**: Overview of the products whose reports are generated
- **Attachment**: Electricity generation reports in CSV format



ID	Name	Region	Model	Product Created At
e5af0bba-2ca8-494b-b49b-ae4436cb1999	Chemnitz	Chemnitz	Q.PEAK DUO XL-G10.3/BFG 475	Sun, 18 Jun 2023 13:12:52 GMT

Figure 29 - Electricity Generation Report (Email)

6. Future Work

- Show current weather icons in place of the map marker for every product.
- Implement OAuth 2.0
- Show live weather data as an overlay on the map.
- Add more fields in the user profile page, e.g., image.
- Add more parameters in the electricity generation formula.

7. Conclusion

In summary, the prototype web-based calculation system outlined in this project offers a practical and user-friendly solution for efficient planning of photovoltaic systems. By harnessing weather data and focusing on a specific manufacturer and product, the system enables precise assessment and effective planning of solar installations. The challenge of accurately determining the usability and effectiveness of solar systems for specific applications, locations, and manufacturers is effectively addressed through the development of this comprehensive and intuitive application. With its potential to streamline energy planning processes, the web application serves as a valuable resource for individuals and businesses, empowering them to optimize their solar installations and make well-informed decisions in an increasingly digital landscape.

8. References

[1] D. Flanagan, JavaScript - The Definitive Guide, 5th ed., Sebastopol, CA: O'Reilly, 2006, p. 497.

[2] Angular, “Angular,” 2019. [Online]. Available: https://angular.io/. [Accessed 01 July 2023].

[3] Microsoft, “TypeScript: JavaScript With Syntax For Types.,” 2015. [Online]. Available: https://www.typescriptlang.org/. [Accessed 01 July 2023].

[4] “Sass: Syntactically Awesome Style Sheets,” 2019. [Online]. Available: https://sass-lang.com/. [Accessed 01 July 2023].

[5] “TypeScript: Documentation - Angular,” [Online]. Available: https://www.typescriptlang.org/docs/handbook/angular.html. [Accessed 01 July 2023].

[6] “RxJS,” [Online]. Available: https://rxjs.dev/. [Accessed 01 July 2023].

[7] “Angular - The RxJS library,” [Online]. Available: https://angular.io/guide/rx-library. [Accessed 01 July 2023].

[8] “PrimeNG - Angular UI Component Library,” [Online]. Available: https://primeng.org/. [Accessed 01 July 2023].

[9] “Angular Icon Library - PrimeNG,” [Online]. Available: https://primeng.org/icons. [Accessed 01 July 2023].

[10] “Build Dynamic Maps with Mapbox,” [Online]. Available: https://www.mapbox.com/about/maps/. [Accessed 01 July 2023].

[11] “OpenStreetMap,” [Online]. Available: https://www.openstreetmap.org/about/. [Accessed 01 July 2023].

[12] OpenJS Foundation, “Node.js,” [Online]. Available: https://nodejs.org/. [Accessed 01 July 2023].

[13] OpenJS Foundation, “Express - Node.js web application framework,” 2017. [Online]. Available: https://expressjs.com/. [Accessed 01 July 2023].

[14] R. T. Fielding, “Architectural Styles and the Design of Network-based Software Architectures,” University of California, Irvine, California, 2000.

[15] “Firebase Authentication,” [Online]. Available: https://firebase.google.com/docs/auth/. [Accessed 01 July 2023].

[16] “Privacy and Security in Firebase,” [Online]. Available: https://firebase.google.com/support/privacy/. [Accessed 01 July 2023].

[17] M. B. Jones, B. Bradley and S. Sakimura, “JSON Web Token (JWT),” *RFC 7519, IETF,* May 2015.

[18] “Firestore | Firebase,” [Online]. Available: https://firebase.google.com/docs/firestore/. [Accessed 01 July 2023].

[19] “Firestore: NoSQL document database | Google Cloud,” [Online]. Available: https://cloud.google.com/firestore/. [Accessed 01 July 2023].

[20] Comtronic, “Automation with Cron job on Centos 8 – Comtronic Blog,” 06 April 2020. [Online]. Available: https://comtronic.com.au/automation-with-cron-job-on-centos-8/. [Accessed 01 July 2023].

[21] Visual Crossing Corporation, “Visual Crossing Weather,” 2023. [Online]. Available: https://www.visualcrossing.com/. [Accessed 01 July 2023].

[22] Photovoltaic-software.com, “How to calculate output energy of PV solar systems?,” [Online]. Available: https://photovoltaic-software.com/principle-ressources/how-calculate-solar-energy-power-pv-systems. [Accessed 01 July 2023].

[23] “Here is how you can calculate the annual solar energy output of a photovoltaic system,” [Online]. Available: https://www.saurenergy.com/solar-energy-blog/here-is-how-you-can-calculate-the-annual-solar-energy-output-of-a-photovoltaic-system. [Accessed 01 July 2023].

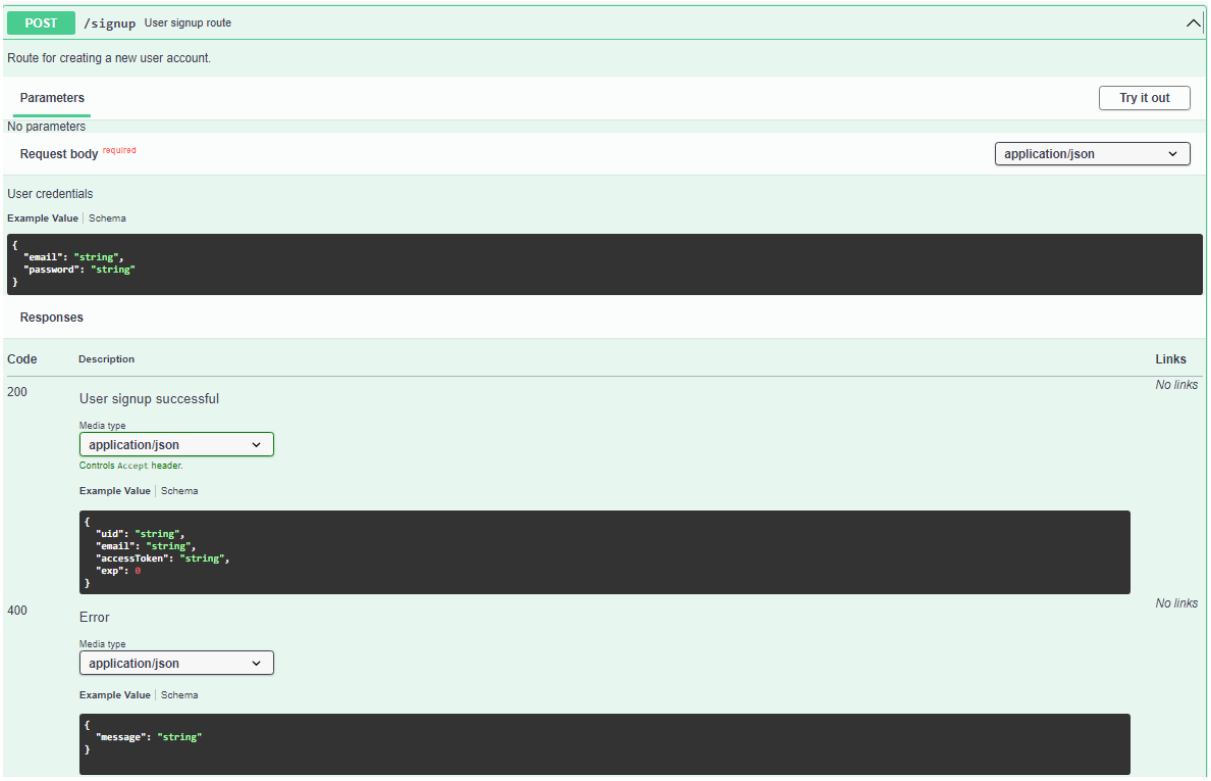
[24] A. P. Dobos, “PVWatts Version 5 Manual,” 04 September 2014. [Online]. Available: https://www.nrel.gov/docs/fy14osti/62641.pdf. [Accessed 01 July 2023].

9. Appendix

9.1. Web Server API Routes Documentation

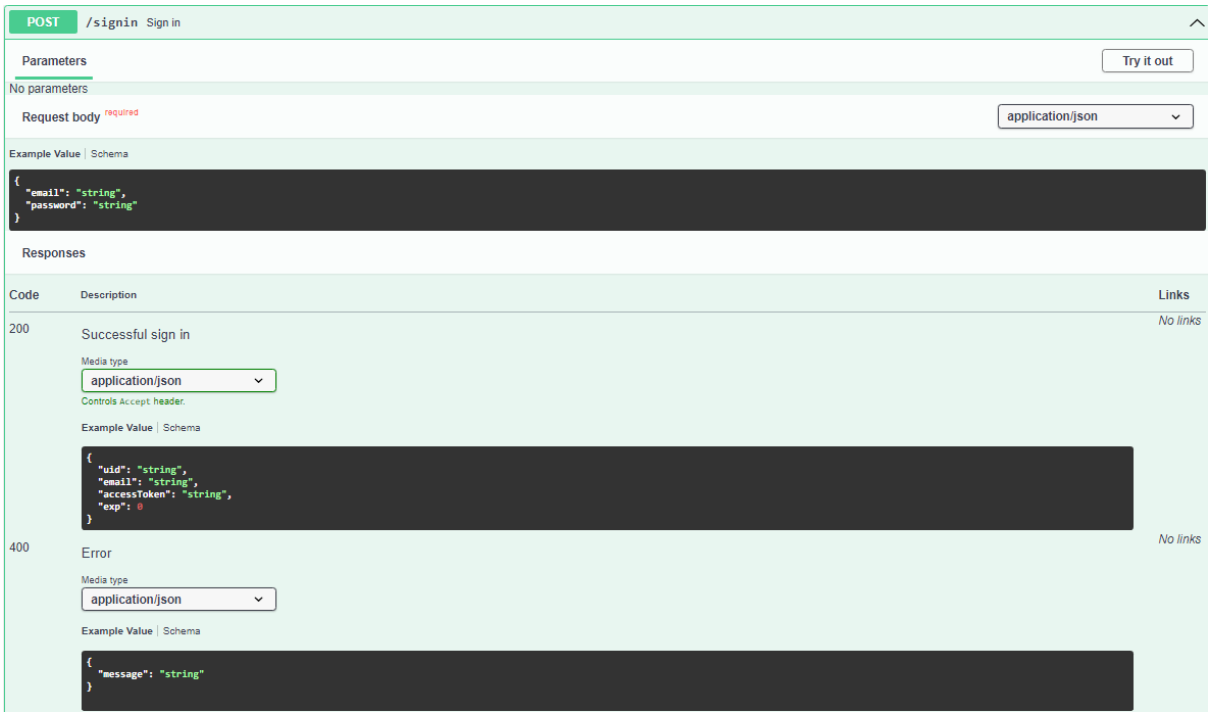
9.1.1. /signup

9.1.1.1. POST



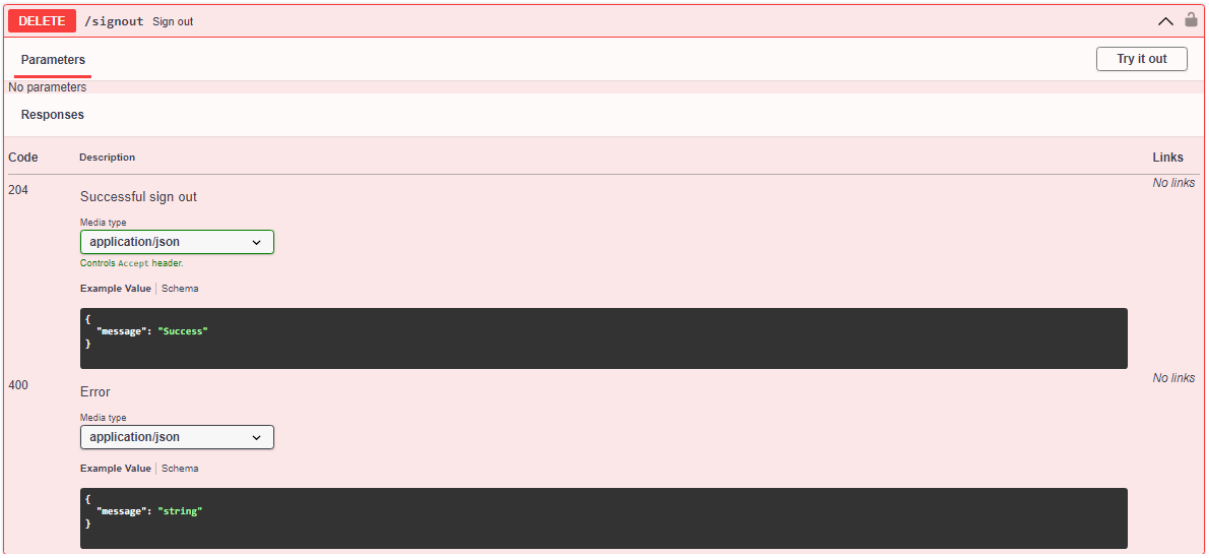
9.1.2. /signin

9.1.2.1. POST



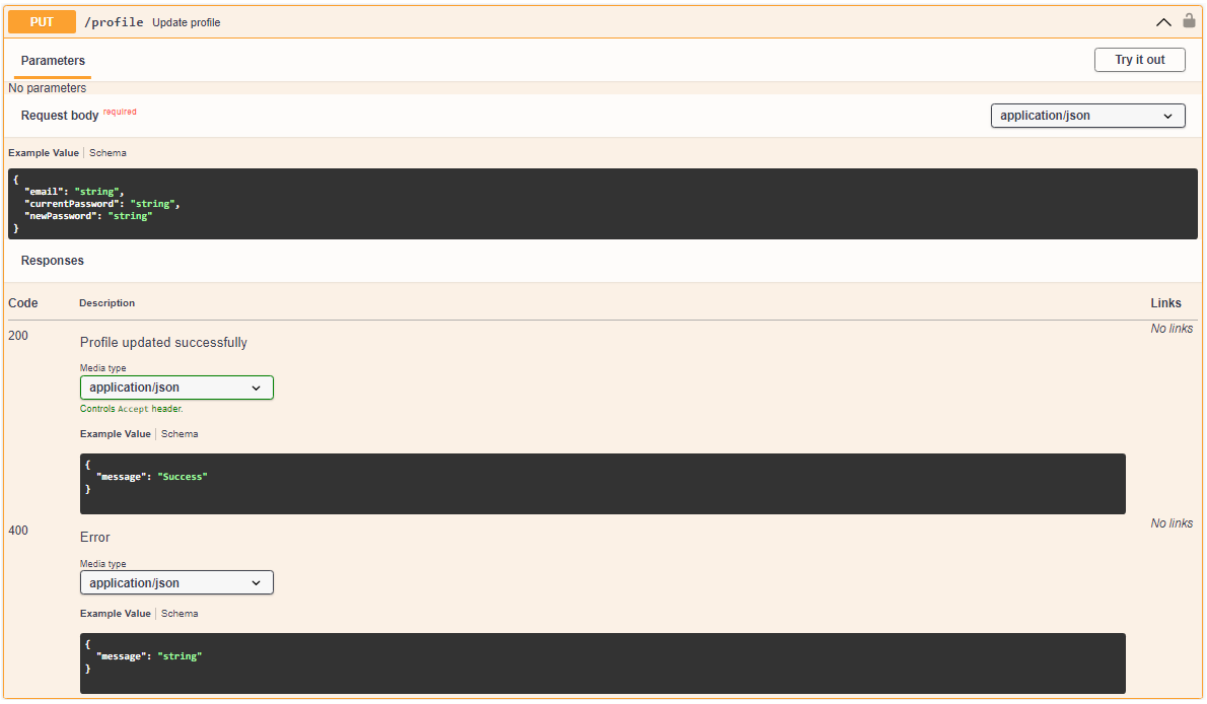
9.1.3. /signout

9.1.3.1. DELETE

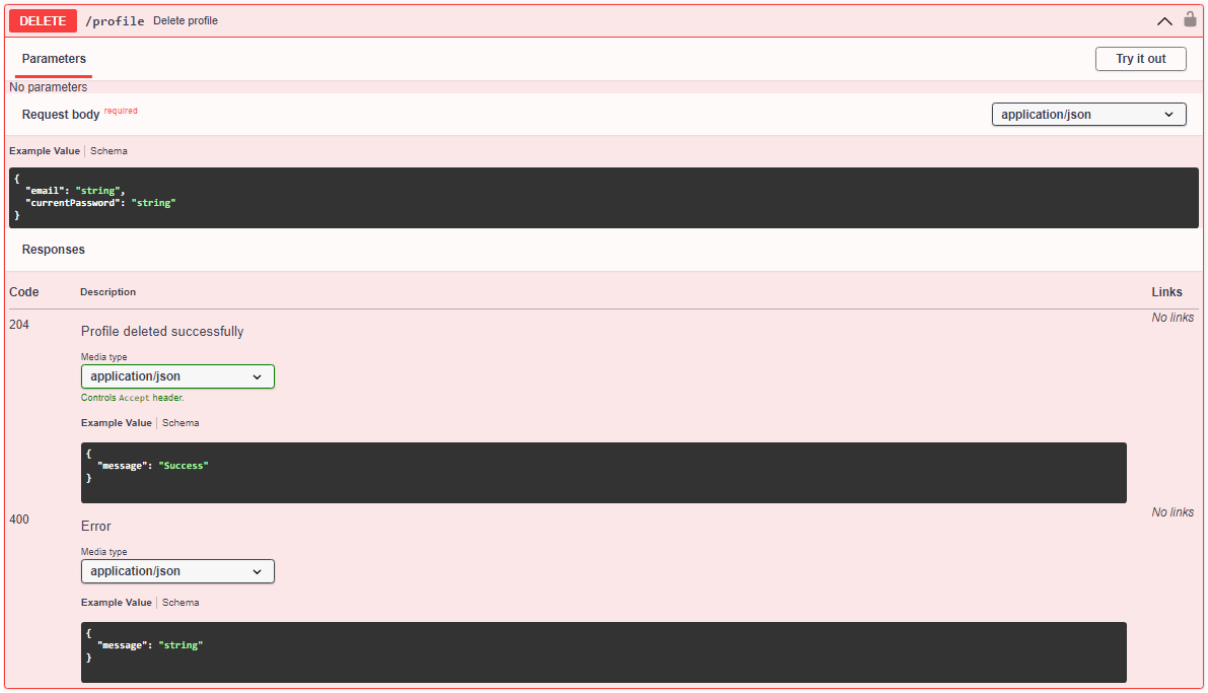


9.1.4. /profile

9.1.4.1. PUT

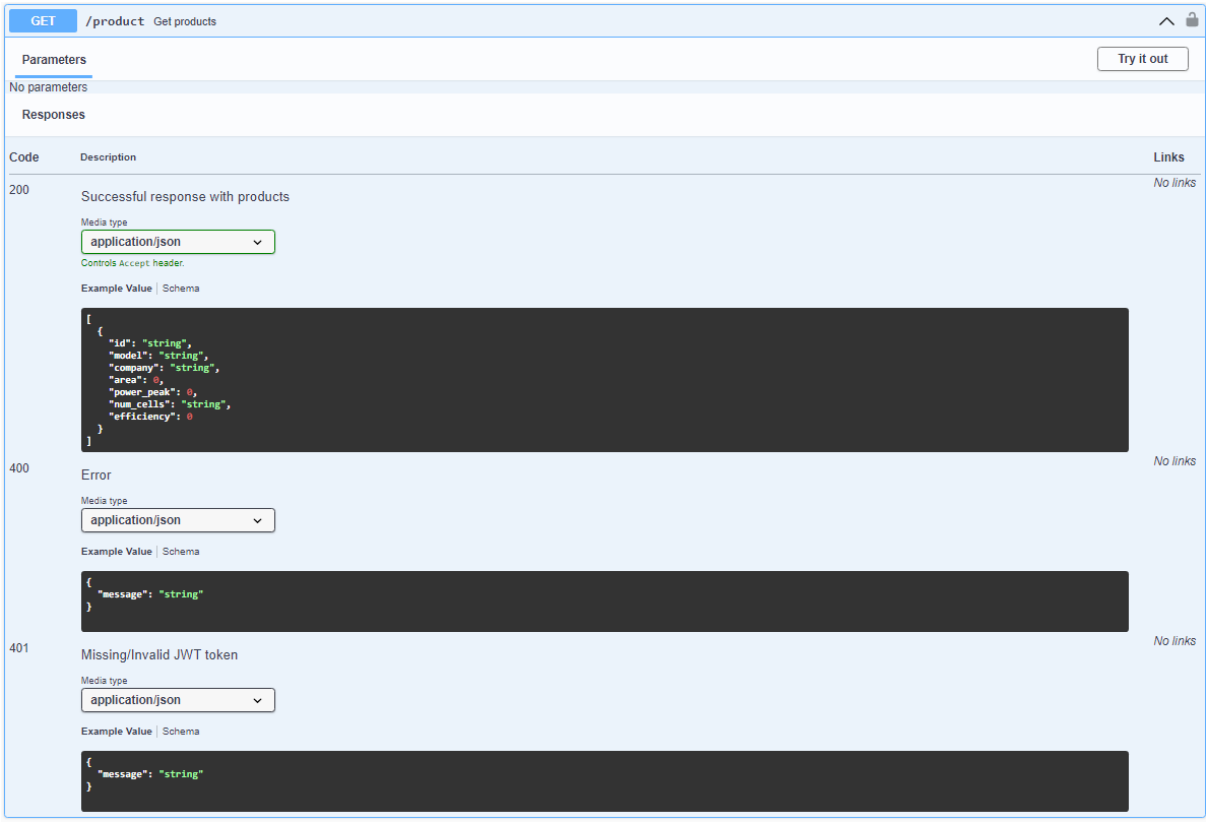


9.1.4.2. DELETE



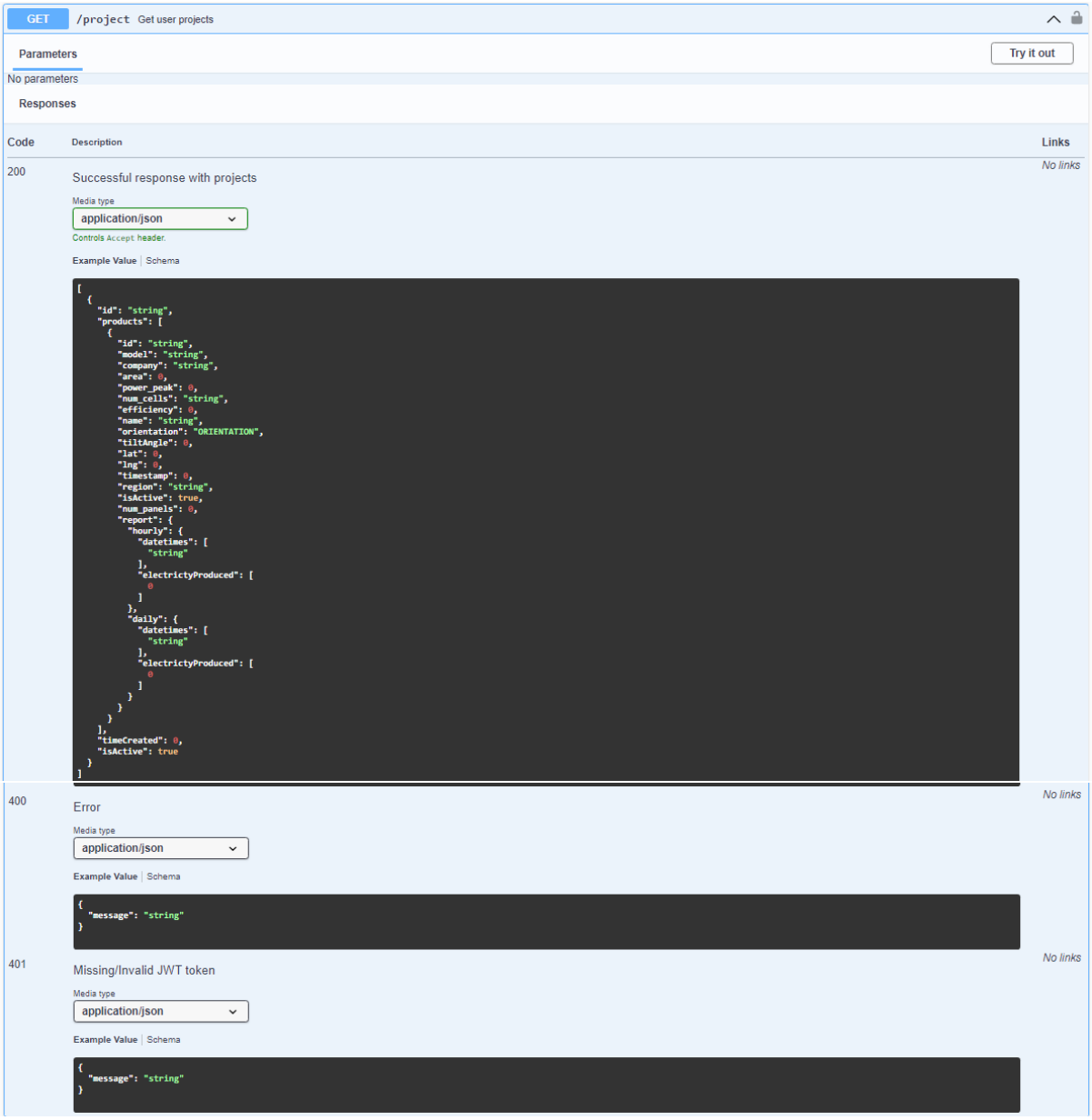
9.1.5. /product

9.1.5.1. GET

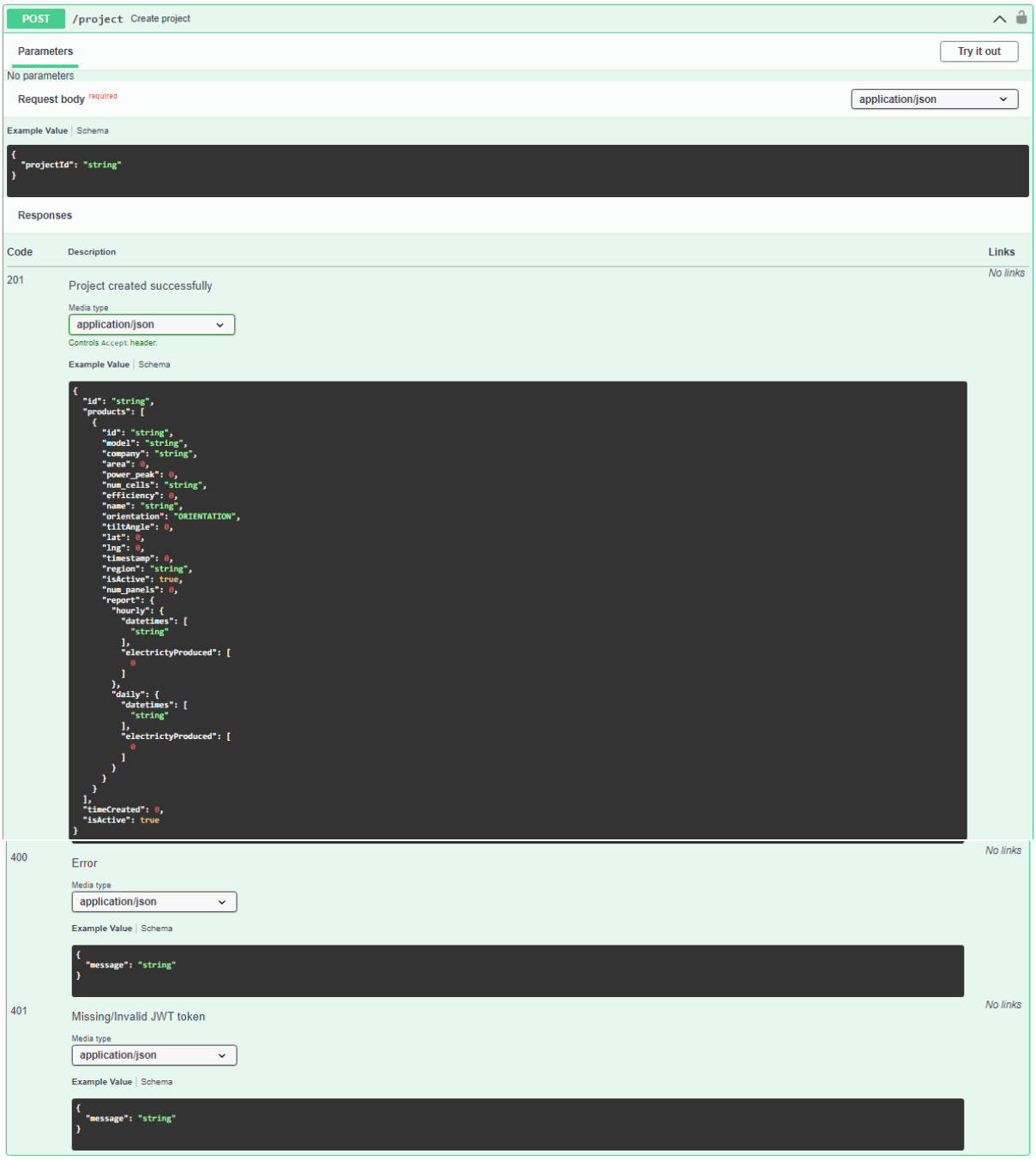


9.1.6. /project

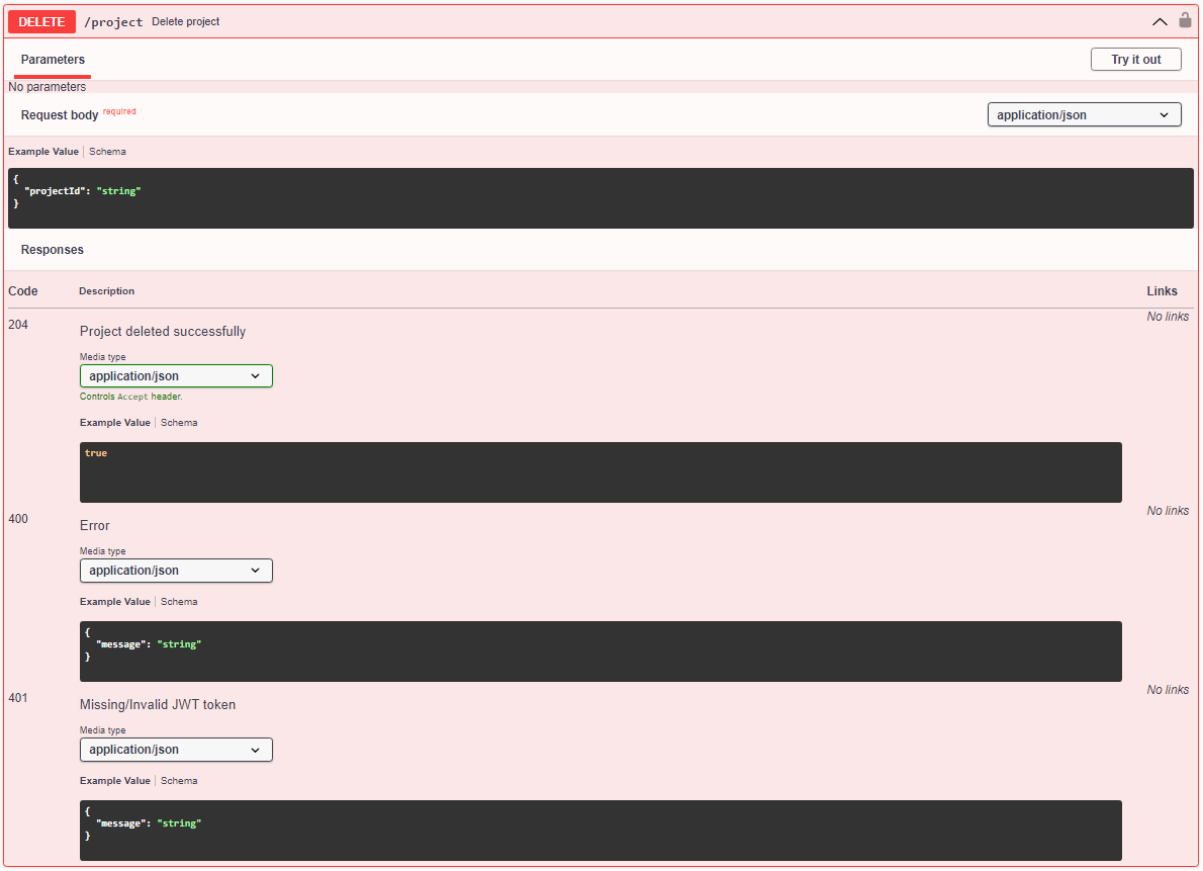
9.1.6.1. GET



9.1.6.2. POST

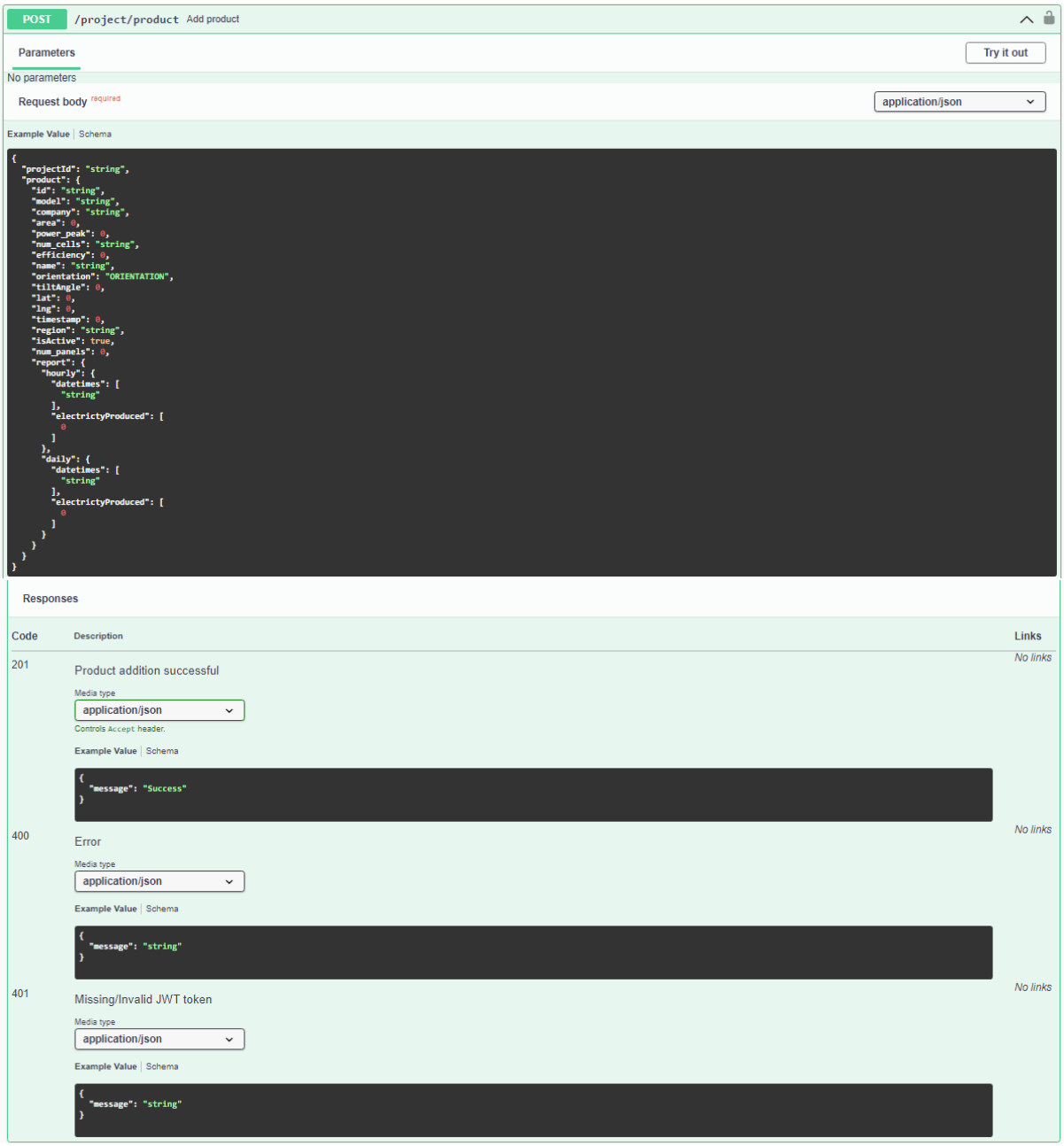

POST /project Create project
Parameters
Try it out
No parameters
Request body required
application/json
Example Value | Schema
{
"projectId": "string"
}
Responses
Code
Description
Links
201
Project created successfully
No links
Media type
application/json
Controls Accept header.
Example Value | Schema
{
"id": "string",
"products": [
{
"id": "string",
"model": "string",
"company": "string",
"area": 0,
"power_peak": 0,
"num_cells": "string",
"efficiency": 0,
"name": "string",
"orientation": "ORIENTATION",
"tiltAngle": 0,
"lat": 0,
"lng": 0,
"timestamp": 0,
"region": "string",
"isActive": true,
"num_panels": 0,
"report": {
"hourly": {
"datetimes": [
"string"
],
"electricityProduced": [
0
]
},
"daily": {
"datetimes": [
"string"
],
"electricityProduced": [
0
]
}
}
}
],
"timeCreated": 0,
"isActive": true
}
400
Error
No links
Media type
application/json
Example Value | Schema
{
"message": "string"
}
401
Missing/Invalid JWT token
No links
Media type
application/json
Example Value | Schema
{
"message": "string"
}

9.1.6.3. DELETE

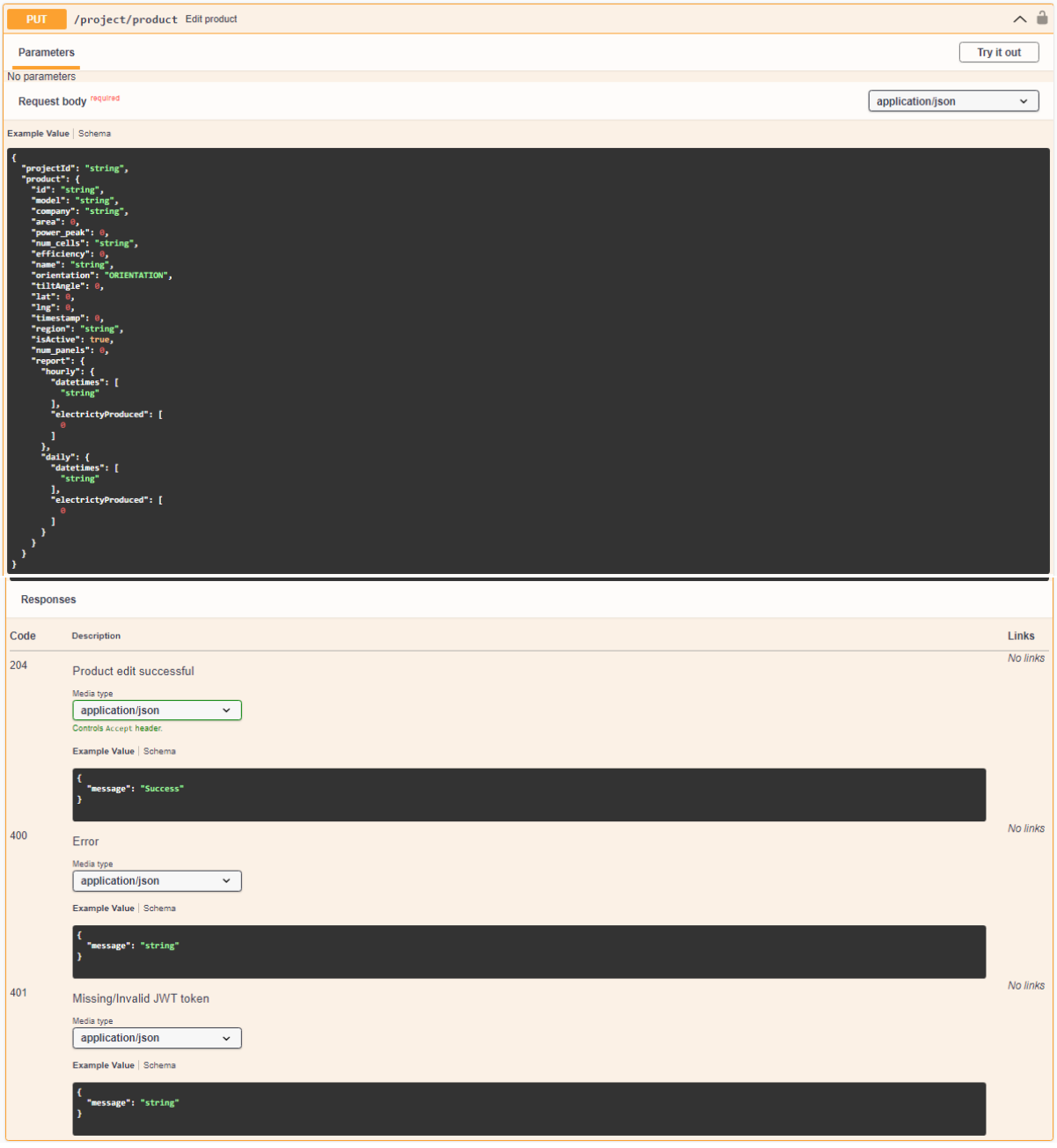

DELETE
/project
Delete project
Parameters
Try it out
No parameters
Request body
required
application/json
Example Value | Schema
{
"projectId": "string"
}
Responses
Code
Description
Links
204
Project deleted successfully
Media type
application/json
Controls Accept header.
Example Value | Schema
true
No links
400
Error
Media type
application/json
Example Value | Schema
{
"message": "string"
}
No links
401
Missing/Invalid JWT token
Media type
application/json
Example Value | Schema
{
"message": "string"
}
No links

9.1.7. /project/product

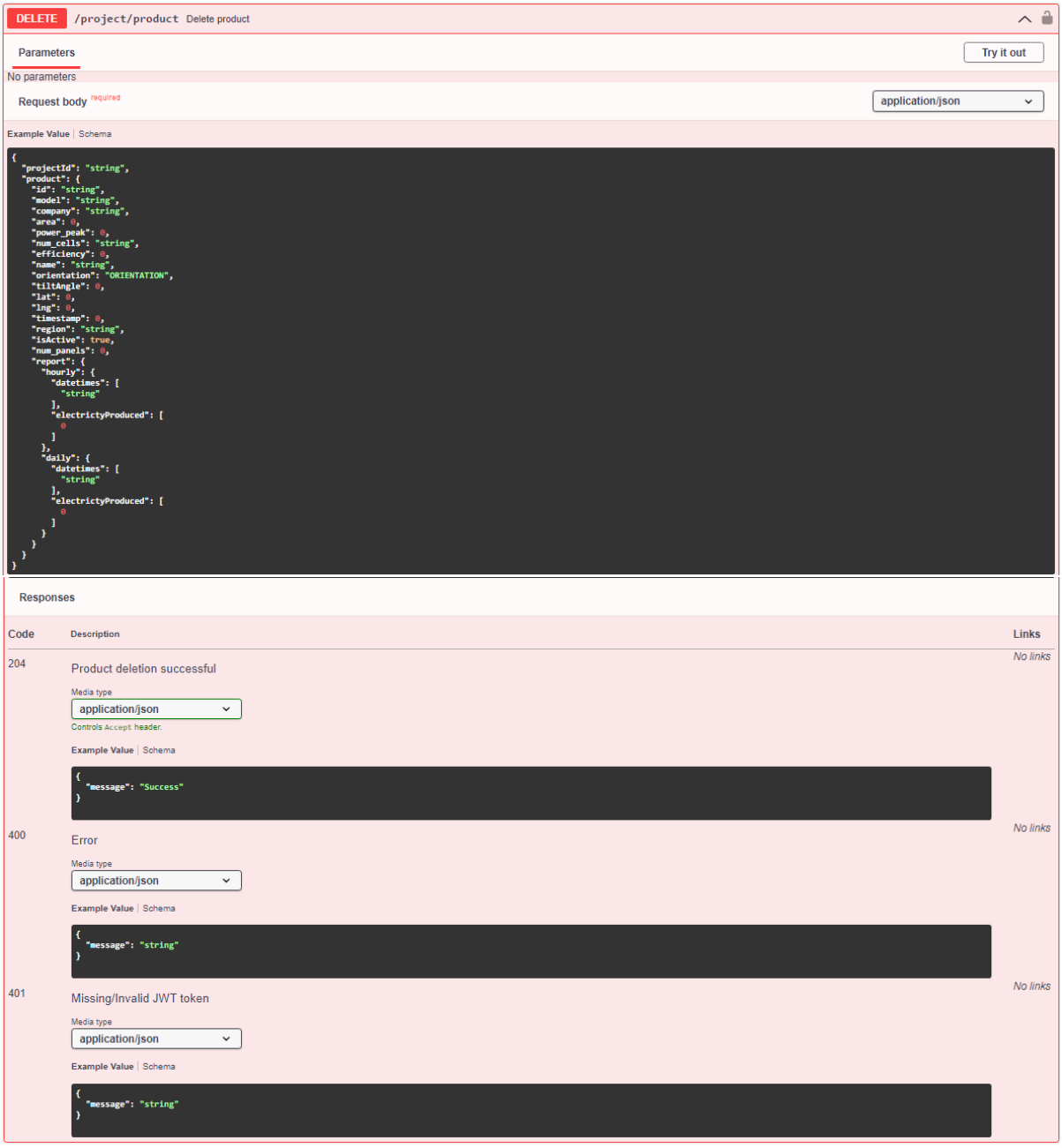
9.1.7.1. POST



9.1.7.2. PUT


PUT
/project/product
Edit product
Parameters
Try it out
No parameters
Request body
required
application/json
Example Value | Schema
{
"projectId": "string",
"product": {
"id": "string",
"model": "string",
"company": "string",
"area": 0,
"power_peak": 0,
"num_cells": "string",
"efficiency": 0,
"name": "string",
"orientation": "ORIENTATION",
"tiltAngle": 0,
"lat": 0,
"lng": 0,
"timestamp": 0,
"region": "string",
"isActive": true,
"num_panels": 0,
"report": {
"hourly": {
"datetimes": [
"string"
],
"electricityProduced": [
0
]
},
"daily": {
"datetimes": [
"string"
],
"electricityProduced": [
0
]
}
}
}
}
Responses
Code
Description
Links
204
Product edit successful
Media type
application/json
Controls Accept header.
Example Value | Schema
{
"message": "Success"
}
No links
400
Error
Media type
application/json
Example Value | Schema
{
"message": "string"
}
No links
401
Missing/Invalid JWT token
Media type
application/json
Example Value | Schema
{
"message": "string"
}

9.1.7.3. DELETE


DELETE
/project/product
Delete product
Parameters
Try it out
No parameters
Request body
required
application/json
Example Value | Schema
{
"projectId": "string",
"product": {
"id": "string",
"model": "string",
"company": "string",
"area": 0,
"power_peak": 0,
"num_cells": "string",
"efficiency": 0,
"name": "string",
"orientation": "ORIENTATION",
"tiltAngle": 0,
"lat": 0,
"lng": 0,
"timestamp": 0,
"region": "string",
"isActive": true,
"num_panels": 0,
"report": {
"hourly": {
"datetimes": [
"string"
],
"electricityProduced": [
0
]
},
"daily": {
"datetimes": [
"string"
],
"electricityProduced": [
0
]
}
}
}
}
Responses
Code
Description
Links
204
Product deletion successful
No links
Media type
application/json
Controls Accept header.
Example Value | Schema
{
"message": "Success"
}
400
Error
No links
Media type
application/json
Example Value | Schema
{
"message": "string"
}
401
Missing/Invalid JWT token
No links
Media type
application/json
Example Value | Schema
{
"message": "string"
}

9.1.8. /project/product/report

9.1.8.1. POST