University of Tunis
Tunis Business School

Satellite Tracking Web Application

Web Services Final Project Report

Prepared by
Imen Cherif
BA/IT

Supervised by
Pr. Montassar Ben Messaoud

2023-2024

Declaration

I certify that I am the author of this report and that any assistance I have received in its preparation is fully acknowledged and disclosed in this report. I have also cited any source from which I used data, ideas, or words, either quoted or paraphrased. Further, this report meets all of the rules of quotation and referencing in use at TBS, as well as adheres to the fraud policies listed in the TBS honor code.

No portion of the work referred to in this report has been submitted in support of an application for another degree or qualification to this or any other university or academic institution.

Abstract

The project emerged from the intersection of telecommunication and aerospace, aiming to create a user-friendly Satellite Tracking Web Application. Using Python, Django, and external APIs, the application enables user registration, real-time satellite tracking, and access to a comprehensive satellite database. Key security measures, code reuse practices, and Docker deployment enhance the platform's robustness.

In the Tunisian context, the project acquires additional significance, aligning with the country's recent rise into space exploration with the launch of its first satellite in 2021. The initiative seeks to bridge the gap in space culture and technology, offering a centralized platform for satellite enthusiasts.

Keywords: Satellite Tracking, Web Application, Telecommunication, Aerospace, Security.

Acknowledgements

I extend my sincere gratitude to Professor Montassar Ben Messaoud, the instructor of the Web Services course, for assigning me this challenging yet rewarding project. Tackling the development of the Satellite Tracking Web Application has been a journey of discovery and growth. The challenges of this project have served as invaluable learning experiences, enhancing my proficiency in web development, database management, security implementation, and the integration of external APIs. The skills acquired during this endeavor will undoubtedly contribute to my academic and professional journey.

Thank you, Professor, for providing this opportunity and for your dedication to fostering a culture of exploration and innovation.

Contents

EXECUTIVE SUMMARY

The primary objective of this project initiative is to provide a user-friendly platform for satellite enthusiasts, emphasizing real-time tracking of satellites, particularly the International Space Station (ISS). Developed as part of the Web Services class, the project merges telecommunication and aerospace technology.

1. **User Registration and Authentication:** The application allows users to create accounts securely, with Django authentication ensuring reliable access.
2. **Satellite Information Retrieval:** Users can access a comprehensive database, stored in PostgreSQL, containing detailed satellite information.
3. **Real-time Tracking:** The ISS tracking feature uses code reuse and external APIs, offering dynamic updates on the ISS current position.
4. **Security Measures:** The project prioritizes user security with HTTPS communication, password hashing, and data validation to prevent common vulnerabilities.
5. **Database Management:** PostgreSQL, MongoDB, and Geospatial Databases handle efficient data storage. Database migration ensures adaptability.
6. **Web Technologies:** Python, Django, HTML, CSS, and JavaScript create an interactive and visually appealing user interface.
7. **Friendly User Interface:** The user interface is designed to be user-friendly, providing an intuitive and engaging experience. HTML, CSS, and JavaScript are used to craft visually pleasing and responsive elements, ensuring ease of navigation and understanding.
8. **Code Reuse:** The ISS tracking code showcases modular and efficient code practices.
9. **External API Integration:** Real-time ISS and astronauts tracking APIs enhance project functionality. Google Fonts API is employed for dynamic font for the user interface.
10. **Docker and Swagger Implementation:** Docker ensures easy deployment, while Swagger documentation provides clear API endpoints.

GENERAL INTRODUCTION

In the center of modern technology, where telecommunication and aerospace converge, the Satellite Tracking Web Application counts as a pioneering initiative to offer users a holistic and accessible view of satellite information. The project's background lies in the recognition of the pivotal role played by satellites in telecommunication systems and the broader field of space exploration.

The convergence of these two fields was not only a strategic choice but also a personal passion, fostering a unique blend of technological innovation and space curiosity. The project embarks on a mission to fill a void in the digital landscape by providing a centralized platform for users to access real-time satellite data, track the ISS, and engage with a wealth of information related to space.

This endeavor is not without its challenges, but each obstacle encountered serves as an opportunity for growth and learning. The development process involved the use of Python Django for web applications, PostgreSQL for robust data storage, Docker for efficient deployment, and the integration of external APIs for real-time updates. Security considerations are important, with measures such as password hashing implemented to ensure the integrity and confidentiality of user data.

Chapter 1

Motivation and Topic Choice

1.1 Background and Significance

1.1.1 Telecommunication and Satellite Importance

In the world of communication, satellites are like global messengers. These orbiting entities facilitate communication across vast distances, connecting remote regions and supporting various applications, including television broadcasting, internet services, and secure communication channels. The unique ability of satellites to cover expansive geographic areas makes them indispensable in bridging communication gaps and fostering a globally interconnected society.

1.1.2 Tunisia's Leap into Space

In 2021, Tunisia launched its first satellite, Challenge ONE. This was a great achievement since it marked Tunisia's entry into the space age, emphasizing the nation's commitment to advancing technology and contributing to global space efforts.

1.2 Motivation Behind the Project

1.2.1 Combining Passion for Aerospace and Telecommunication

As part of my web services project, the opportunity to explore the convergence between aerospace and telecommunication inspired the choice of satellites as the focal point of my work. The relationship between these two fields became apparent, especially considering the crucial role satellites play in enhancing telecommunication capabilities.

1.2.2 Space Tech for Tunisia's Growth

This project aims to contribute to Tunisia's space endeavors. Acknowledging the potential benefits of satellite technology in fostering communication, data transmission, and scientific research, the project seeks to bridge the gap between passion and practical application, since for a country to grow, space tech is crucial.

1.3 Telecommunication Satellites and Tunisia's Space Culture

1.3.1 Space Interest in Tunisia

In Tunisia, we're curious about space, but we can do more to get people interested. Even though we know about the ISS, there's no easy way to find real time information about other satellites. This project wants to change that, trying to cover the lack of a comprehensive platform offering detailed information about various satellites in a user-friendly manner.

1.3.2 Project Boost for Space Interest

This project emerges as a pioneering effort to address this gap. By creating a user-friendly platform that consolidates information about diverse satellites, the goal is to elevate the space culture in Tunisia. The aim is not only to make space-related information accessible but also to instill a sense of pride and curiosity among Tunisians about their country's contributions to the space domain.

1.4 ISS Tracking and Beyond

1.4.1 Starting with ISS, Dreaming Bigger

While ISS tracking has garnered attention, the project's broader vision is to extend beyond the International Space Station. The intent is to provide a comprehensive platform that encompasses information about all satellites, fostering a deeper understanding of their roles, launches, and trajectories.

1.4.2 User-Friendly Space Exploration

By going beyond the current ISS tracking initiatives, this project seeks to democratize access to space-related knowledge. Through a user-friendly interface, Tunisians can explore and appreciate the multitude of satellites orbiting the Earth, laying the foundation for an informed and space-aware society.

Chapter 2

Software and Tools Choice

2.1 Python Django: A Foundation for Functionality

2.1.1 Python Django

Choosing Python Django as the backbone for this project was a deliberate decision. Django, a high-level web framework, provides a robust and efficient way to build web applications. Its simplicity and flexibility are key factors that make it an ideal choice, enabling the implementation of various functionalities with ease.

2.1.2 Enabling Functionality through Django

Django's "batteries-included" philosophy means it comes with a set of tools and features ready to use. This includes an authentication system, database management, and a powerful ORM (Object-Relational Mapping) system. This allows us to focus more on the project's unique features, like satellite tracking, rather than reinventing the wheel.

2.2 PostgreSQL and PGAdmin: Managing Satellite Information

2.2.1 Storing Satellite Data

Satellite information is crucial for our project, and PostgreSQL serves as the reliable database to store this data. Its support for geospatial data and robust features make it an excellent choice for managing satellite information.

2.2.2 PGAdmin for Database Administration

PGAdmin, a popular PostgreSQL administration and management tool, provides a user-friendly interface for handling the database. This makes it easier to manage and visualize the satellite data, ensuring smooth interaction with the stored information.

2.3 Swagger: API Documentation

Swagger plays a key role in documenting our API. It provides a clear and interactive documentation platform that helps users understand how to interact with our API endpoints. This ensures transparency and facilitates integration for developers.

2.4 Docker

2.4.1 Containerization with Docker

Docker simplifies the deployment process by containerizing our application. This means that all dependencies, configurations, and the application itself are packaged into a single container. This approach streamlines deployment, making it consistent across different environments and reducing compatibility issues.

2.4.2 Ensuring Consistency with Docker

Using Docker ensures that the application runs consistently, regardless of where it's deployed. It enhances scalability and flexibility, allowing for easy scaling and adapting to changing requirements.

2.5 Frontend Technologies: HTML, CSS, and JavaScript

The frontend of the application is built using HTML, CSS, and JavaScript to create an intuitive and user-friendly interface. These technologies work together to ensure a responsive and visually appealing design, enhancing the overall user experience, and easily interacting with the Django app.

2.6 Postman: Testing and Development Aid Tool

Postman, a widely used API development and testing tool, plays a vital role in ensuring the functionality of our API. It allows for thorough testing of endpoints, ensuring that our API behaves as expected and meets the project requirements.

2.7 Requirements.txt: Dependency Management

The requirements.txt file serves as a guide for managing project dependencies. It lists the necessary Python packages, including Django, dnspython, psycopg2-binary, pymongo, sqlparse, tzdata, and asgiref, ensuring a standardized and reproducible development environment.

Chapter 3

Project Features

3.1 User Registration

3.1.1 Feature Overview

The user registration feature serves as the welcoming door to the project, allowing space enthusiasts to create personalized accounts. Beyond its technical implementation, this feature symbolizes the inclusivity and engagement the project aims to foster. By enabling users to sign up with a unique username and email, it provides them with a sense of belonging to the platform and makes them more interested in discovering the space industry.

3.1.2 User-Centered Impact

User registration is not only a technical process; it represents the initiation of a user into a community interested in satellite exploration. The welcome email adds a personal touch, expressing gratitude for their involvement. This feature aims to build a sense of community and acknowledges the user's contribution to the project. The user will also need to sign a project charter in order to learn more about how the website works. The user's information is stored in a database only accessible by the admin user.

3.2 User Login

3.2.1 Feature Overview

The user login feature ensures secure access for registered users, emphasizing the importance of privacy and user accounts. The login process signifies a user's return to the platform and their continued interaction with satellite-related content.

3.2.2 User-Centered Impact

User login is a user-friendly aspect that recognizes returning users. The authentication failure message communicates transparently with users, maintaining a user-centred approach by assisting them in resolving login issues. This feature contributes to a sense of security and trust within the platform. The login feature only works when the user is registered in the database.

3.3 World Satellites Tracking

3.3.1 Feature Overview

World satellites tracking represents the core of the project's mission—to provide users with a comprehensive view of satellites orbiting the Earth. While the technical implementation involves rendering a webpage, the feature's essence lies in granting users access to a wealth of information about satellites from different parts of the world. In this project, 12 different APIs were implemented, verying from GET and POST to PUT and DELETE. Their objective is to allow the user to easily integrate with the large database of satellites information:

- Satellites by Country
- Satellite by Norad Number
- Satellites by Operator
- Countries by Purpose
- Country Satellite Ranking
- Delete Satellites by Country
- Modify Purpose by Norad Number
- Add Satellite
- Satellites by Launch Site
- Delete Satellite by Norad Number
- Update Satellite Country by Norad Number
- Add Simple Satellite

3.3.2 User-Centered Impact

This feature opens a window to the universe for users, encouraging exploration and learning. By creating a space where users can delve into satellite data, the project aspires to kindle curiosity and enhance the user's understanding of the vast array of satellites orbiting our planet. The Swagger link with the project also allows the user to see the documentation of each of the APIs all while testing them.

3.4 Telecommunication Satellites

3.4.1 Feature Overview

Dedicated to telecommunication satellites, this feature is a nod to the project's roots in the telecommunication field. It provides a focused space for users interested in the specific category of satellites used for communication purposes. The webpage embeds a youtube video enabling them to build their own telecommunication satellite from the comfort of their homes.

3.4.2 User-Centered Impact

By spotlighting telecommunication satellites, the project acknowledges the diverse interests within the user community. Users passionate about communication technology can easily find the opportunity to construct their DIY telecommunication satellite, creating a more personalized and engaging experience.

3.5 International Space Station Tracking

3.5.1 Feature Overview

The last feature is the International Space Station (ISS) tracking. It injects a dynamic and real-time element into the project. This feature allows users to stay updated on the ISS's position and movements, but also get to learn more about the astronauts currently in the station.

3.5.2 User-Centered Impact

ISS tracking adds an element of excitement and relevance, especially for users intrigued by human space exploration. This feature connects users with ongoing space activities, fostering a sense of participation and interest in the latest developments beyond Earth. The user-friendly aspect of the Python turtle library and the generated txt file makes it easier for the user to be up-to-date on the ISS position.

Chapter 4

Project Functionalities

4.1 Database Management

4.1.1 Database Structure

The project adopts a robust database structure, leveraging the capabilities of PostgreSQL for satellite information and Django's user creation database for user authentication. The PostgreSQL database serves as a repository for extensive satellite details, encompassing critical attributes such as NORAD number, country of operator, purpose, launch information, and more. This structure ensures a comprehensive and organized storage of data, facilitating efficient retrieval and manipulation.

The Django user creation database plays a pivotal role in user management, offering a secure and structured framework for handling user authentication. It stores essential user details like username, email, and hashed passwords. This separation of databases aligns with best practices, ensuring clarity in data organization and enhancing the scalability of the system.

4.1.2 Migration Process

The migration process, orchestrated through Django's migration framework, exemplifies the project's commitment to maintainability and adaptability. The provided migration script captures the evolution of the database schema over time, ensuring fluid updates and additions to the data model.

The creation of two essential models: CustomUser and Satellite demonstrates the versatility of Django's migration system. The CustomUser model manages user accounts, incorporating fields for essential user information, while the Satellite model encapsulates attributes specific to satellite data. These models serve as the foundation for structured data storage and retrieval.

The migration operations extend beyond simple table creation. Relationships between tables are established, allowing for the efficient retrieval of related data. Necessary indexes are added, optimizing query performance and ensuring quick access to information. This strategic use of migration operations guarantees a well-organized and responsive database structure.

Additionally, the migration process caters to the integration of an admin account, providing the capability to manage satellite information dynamically. The admin can add new satellites or remove outdated entries, keeping the database up to date. This functionality is very important in ensuring that the project stays up-to-date with the latest satellite information, aligning with the dynamic nature of space exploration.

In essence, the database management component of the project not only establishes a solid foundation for data storage but also incorporates dynamic elements that allow for continuous updates and enhancements, driven by the migration process and administrative capabilities.

4.2 Security

4.2.1 Password Hashing

Ensuring the security and confidentiality of user credentials was one of my greatest concerns in the project. To address this, a robust password hashing mechanism was implemented. The Django user creation database uses industry-standard password storage techniques, hashing passwords before storage. Hashing

transforms user passwords into irreversible strings of characters, significantly enhancing protection against unauthorized access. In the event of a data breach, hashed passwords provide an additional layer of defense, as they cannot be easily decrypted. This security measure underscores the project's commitment to safeguarding user information and privacy.

4.2.2 Email Confirmation

A critical aspect of user security and verification is the implementation of an email confirmation system. Upon successful registration, users receive a welcome email, introducing a personalized touch to the onboarding process. This email confirmation serves a dual purpose—it provides users with a tangible confirmation of their account creation and contributes to a more secure user environment. This additional layer of verification adds an essential element to the overall security of the application.

4.2.3 Admin Account and Permissions

Django's built-in admin interface is use to facilitate the management of user accounts and gain valuable insights into user activities. An admin account, distinct from regular user accounts, is created to oversee administrative tasks efficiently. The admin interface provides an intuitive dashboard, allowing administrators to view and manage user registrations, permissions, and other relevant information. This capability empowers administrators to monitor and control the platform securely, ensuring that user accounts adhere to the established security standards. The separation of administrative functions through Django's built-in permissions system adds granularity to access control, minimizing the risk of unauthorized modifications and reinforcing the overall security architecture.

4.3 Code Reuse

The implementation of the International Space Station (ISS) tracking feature was an example of code reuse and modularity. The code responsible for fetching the ISS position is designed with a focus on reusability. This modular approach ensures that the code for tracking the ISS can be efficiently used in various parts of the application. By avoiding redundancy and leveraging existing solutions, the project not only enhances development efficiency but also establishes a foundation for future expansions and feature integrations. This practice aligns with industry best practices and showcases the project's dedication to efficient software development.

4.4 External APIs Integration

4.4.1 ISS and Astronauts Tracking API

The project embraces the integration of external APIs to augment its functionality and provide users with a richer experience. The ISS tracking feature leverages an external API to fetch real-time position data, offering users dynamic and accurate information about the ISS's current position. This integration expands the project's scope beyond satellite tracking, introducing real-time data from space missions. Additionally, the incorporation of an API for tracking astronauts in space adds a layer of complexity and depth to the platform, offering users insights into human space exploration activities. This strategic use of external APIs enhances the project's value proposition and aligns with the dynamic nature of space-related data.

4.4.2 Google Fonts API

In the pursuit of an aesthetically pleasing and visually engaging user interface, the project integrates the Google Fonts API for dynamic font rendering in the HTML code. This external API merges with the project, providing access to a diverse collection of fonts without the need for local font files. By using the Google Fonts API, the project ensures a consistent and appealing typography style across different browsers and devices. This integration contributes to the overall user experience, emphasizing the project's attention to detail and commitment to creating a visually compelling interface. The use of external APIs for both functional and design-related features showcases the project's versatility and adaptability in incorporating third-party services to enhance the user journey.

Chapter 5

Areas of Improvement

5.1 Enhanced Security Measures

While the current security measures provide a strong foundation, there is an opportunity to enhance security by introducing additional layers of protection. Implementing measures such as intrusion detection systems and continuous monitoring can further fortify the platform against potential threats. The integration of security best practices and staying updated on emerging threats will be pivotal in ensuring the resilience of the system.

5.2 Role-Based Access Control

Introducing role-based access control (RBAC) can bring a deeper level of control over the functionalities accessible to different user roles. By assigning specific roles to users, such as regular users and administrators, the platform can restrict or grant access to particular APIs and features. This not only enhances security but also allows for more tailored user experiences, aligning with the principle of least privilege.

Conclusion

In conclusion, the development and implementation of the Satellite Tracking Web Application have yielded a robust platform that seamlessly integrates technology, data, and user-centred features. The project's inception was driven by a convergence of passions—telecommunication and aerospace—and aimed at addressing the lack of comprehensive satellite information platforms, particularly tailored for the Tunisian context.

The significance of satellites in telecommunication, as well as their broader role in advancing technological capabilities, has been a driving force. This project not only provides real-time tracking of satellites, including the International Space Station, but also acts as a gateway to fostering a space-aware culture among Tunisians.

The chosen technological stack, featuring Python Django, PostgreSQL, Docker, and external API integrations, underscores the project's commitment to leveraging powerful and scalable tools. The use of Docker and Swagger enhances deployment efficiency and documentation accessibility, contributing to a seamless user experience.

Security has been a core concern, with the implementation of measures such as password hashing, email confirmation, and user role-based access control. However, the project acknowledges opportunities for improvement, particularly in enhancing security layers and adopting role-based access control for other permissions.

Code reuse principles are used in the modular implementation of the ISS tracking feature, showcasing the project's dedication to maintainable and efficient coding practices. External API integrations, including ISS and astronaut tracking, as well as the Google Fonts API, contribute to a feature-rich and visually appealing user interface.

As the project moves forward, identified areas of improvement, including enhanced security measures, role-based access control, and the creation of a dedicated API for tracking Tunisian satellites, will guide its evolution. Collaboration with Tunisian space organizations and international agencies will be instrumental in overcoming challenges and enriching the platform with accurate and comprehensive satellite information.

In essence, the Satellite Tracking Web Application not only represents a technological achievement but also signifies a step towards fostering space awareness and technological curiosity, particularly in the Tunisian context. The journey continues, with a commitment to refinement, innovation, and the exploration of new frontiers in satellite tracking and beyond.

References

5.3 Celestrak NORAD Elements

- Celestrak NORAD Elements. Available at https://celestrak.org/NORAD/elements/.

5.4 Django Documentation: Forms

- Django Documentation: Forms. Available at https://docs.djangoproject.com/en/5.0/topics/forms/.

5.5 Django Extensions - RunScript

- Django Extensions: RunScript Documentation. Available at https://django-extensions.readthedocs.io/en/latest/runscript.html#:~:text=To%20run%20any%20script%20you,that%20you%20want%20to%20run.&text=Note%3A%20The%20command%20first%20checks,in%20the%20project_root%2Fscripts%20folder.

5.6 GitHub Repository - NASA Project with Python

- NASA Project with Python on GitHub. Available at https://github.com/BekBrace/NASA-Project-with-Python

5.7 N2YO Satellite Tracking

- N2YO Satellite Tracking. Available at https://www.n2yo.com/.

5.8 Open Notify API - Astronauts

- Open Notify API: Astronauts. Available at http://api.open-notify.org/astros.json.

5.9 Open Notify API - ISS Now

- Open Notify API: ISS Now. Available at http://api.open-notify.org/iss-now.json.

5.10 Swagger: Documenting APIs with Swagger

- Swagger: Documenting APIs with Swagger. Available at https://swagger.io/resources/articles/documenting-apis-with-swagger/.

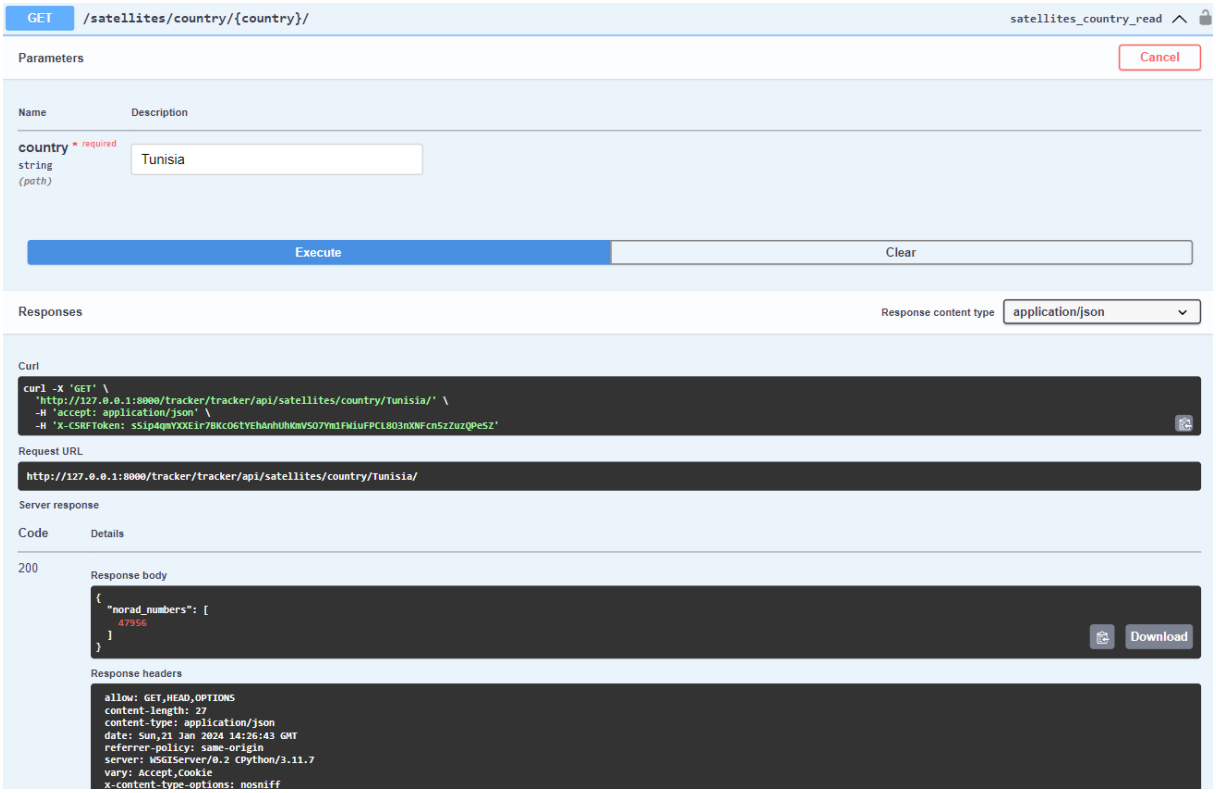
5.11 YouTube Video - Python Satellite Tracking

- YouTube Video: Python Satellite Tracking. Available at https://www.youtube.com/watch?v=5UWeOfdESZE&t=698s.

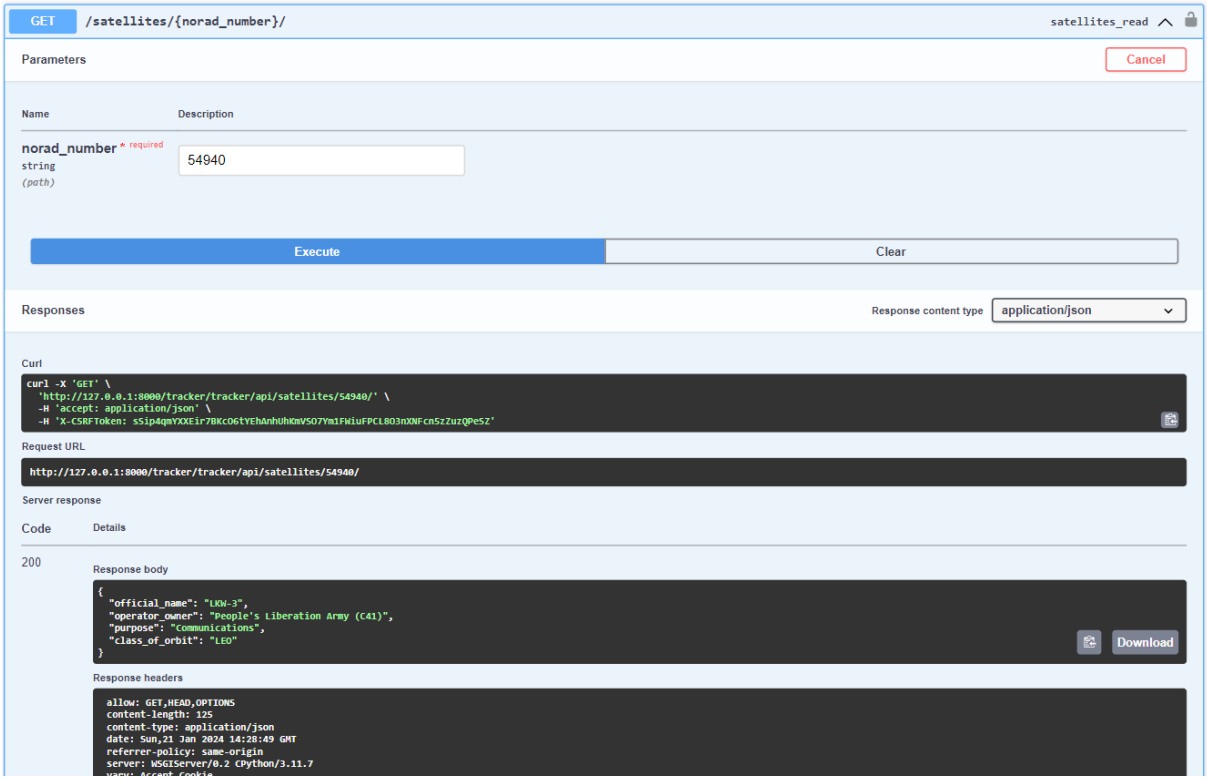
Appendix

Endpoint Testing Screenshots

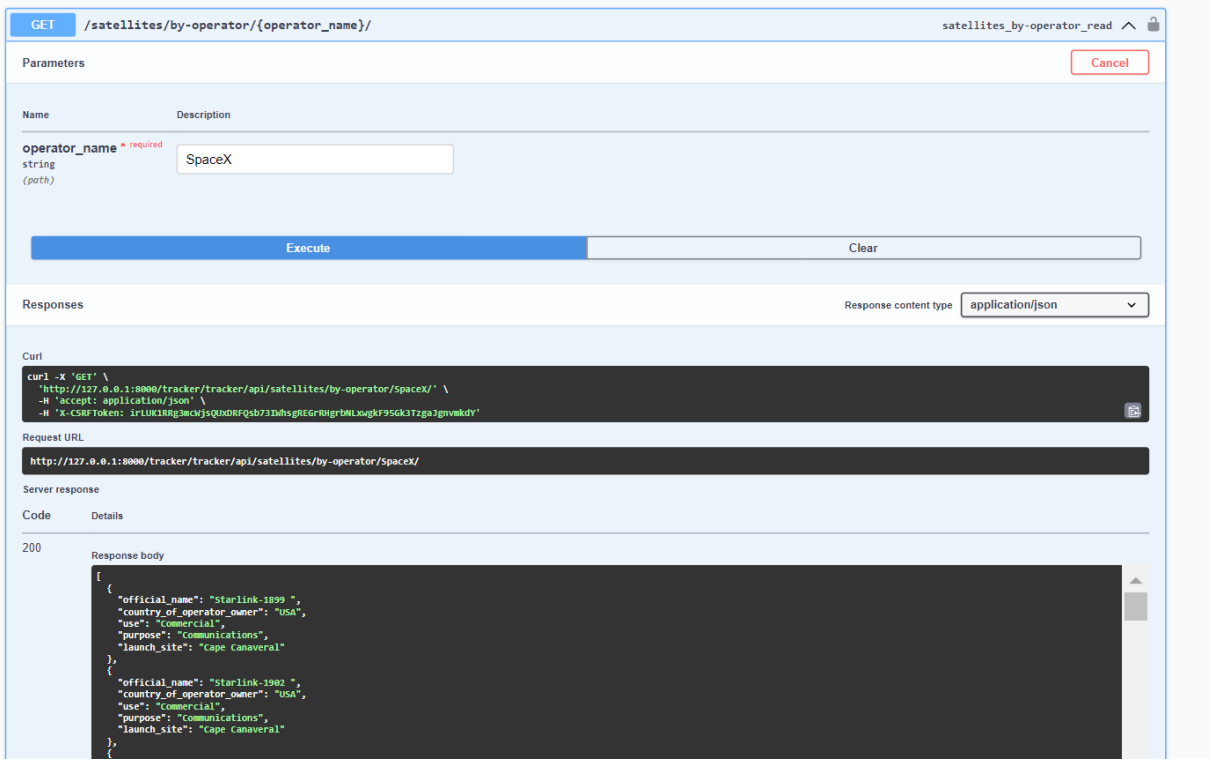
GET Satellites by Country



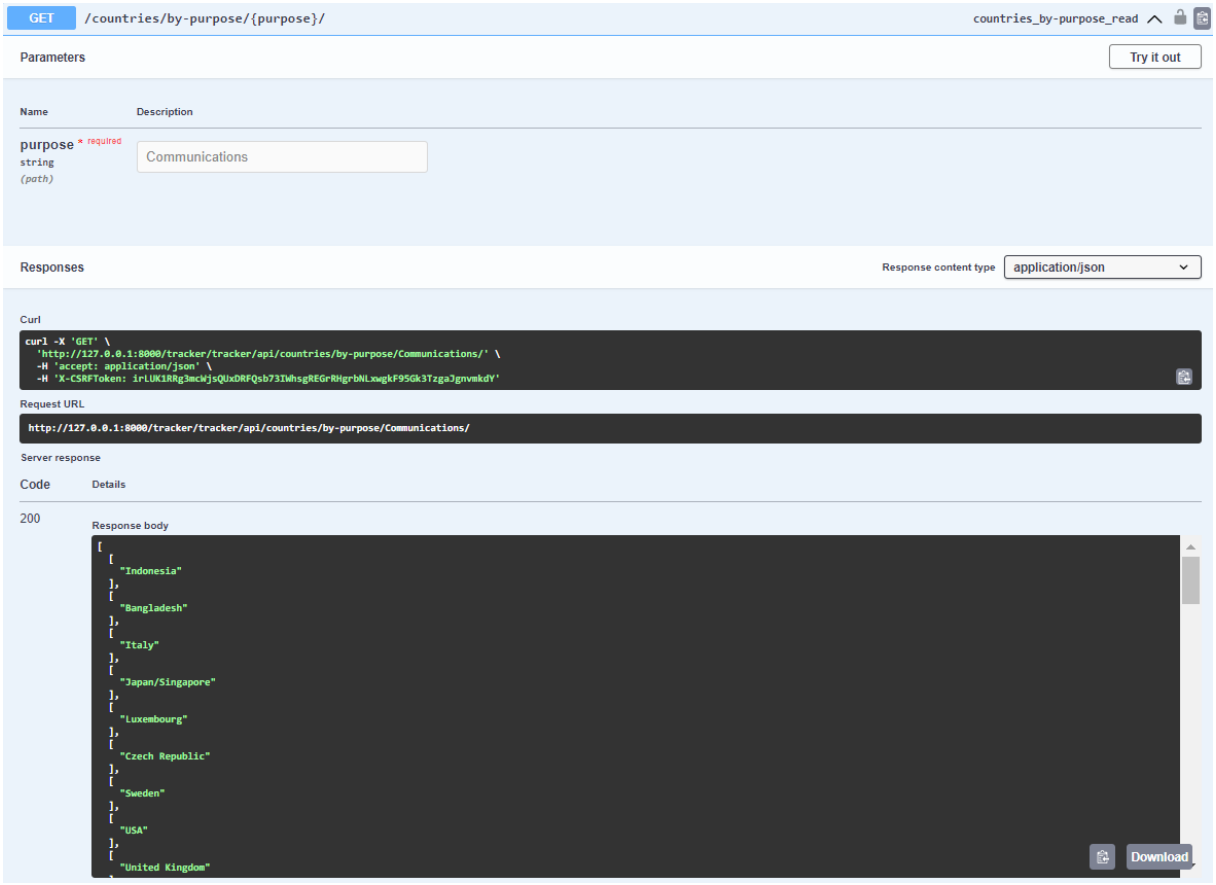
GET Satellite by Norad Number



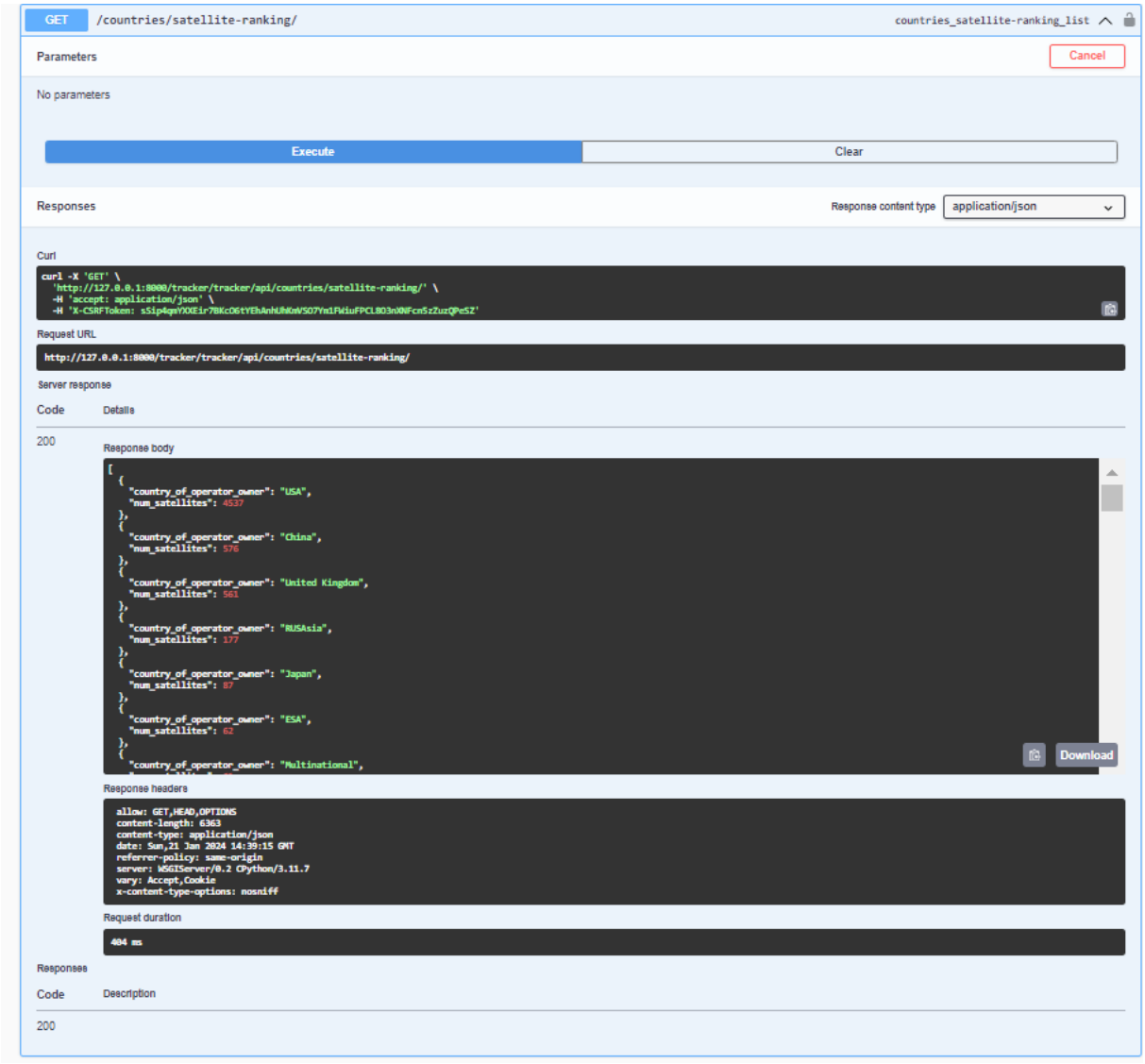
GET Satellites by Operator



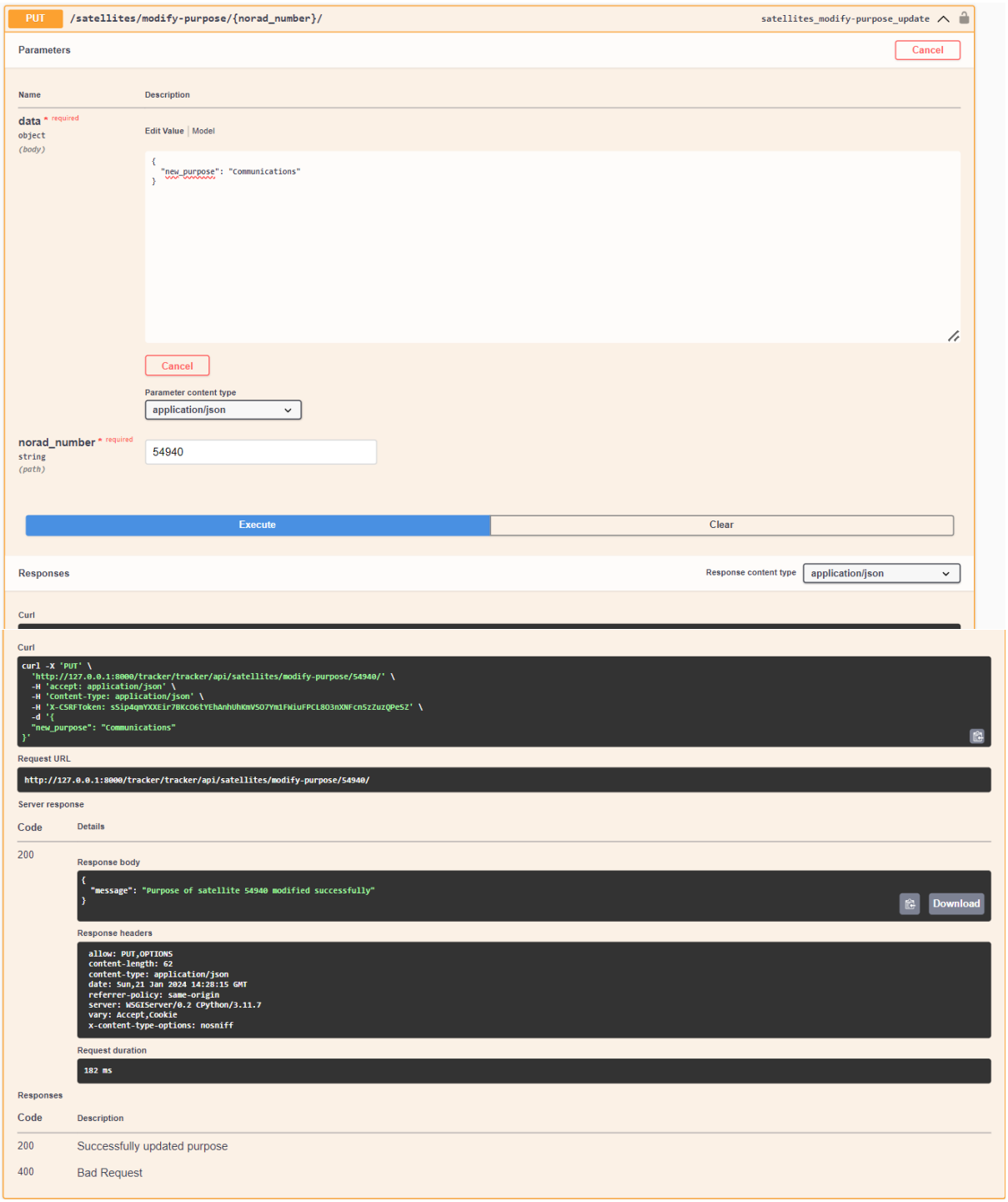
GET Countries by Purpose



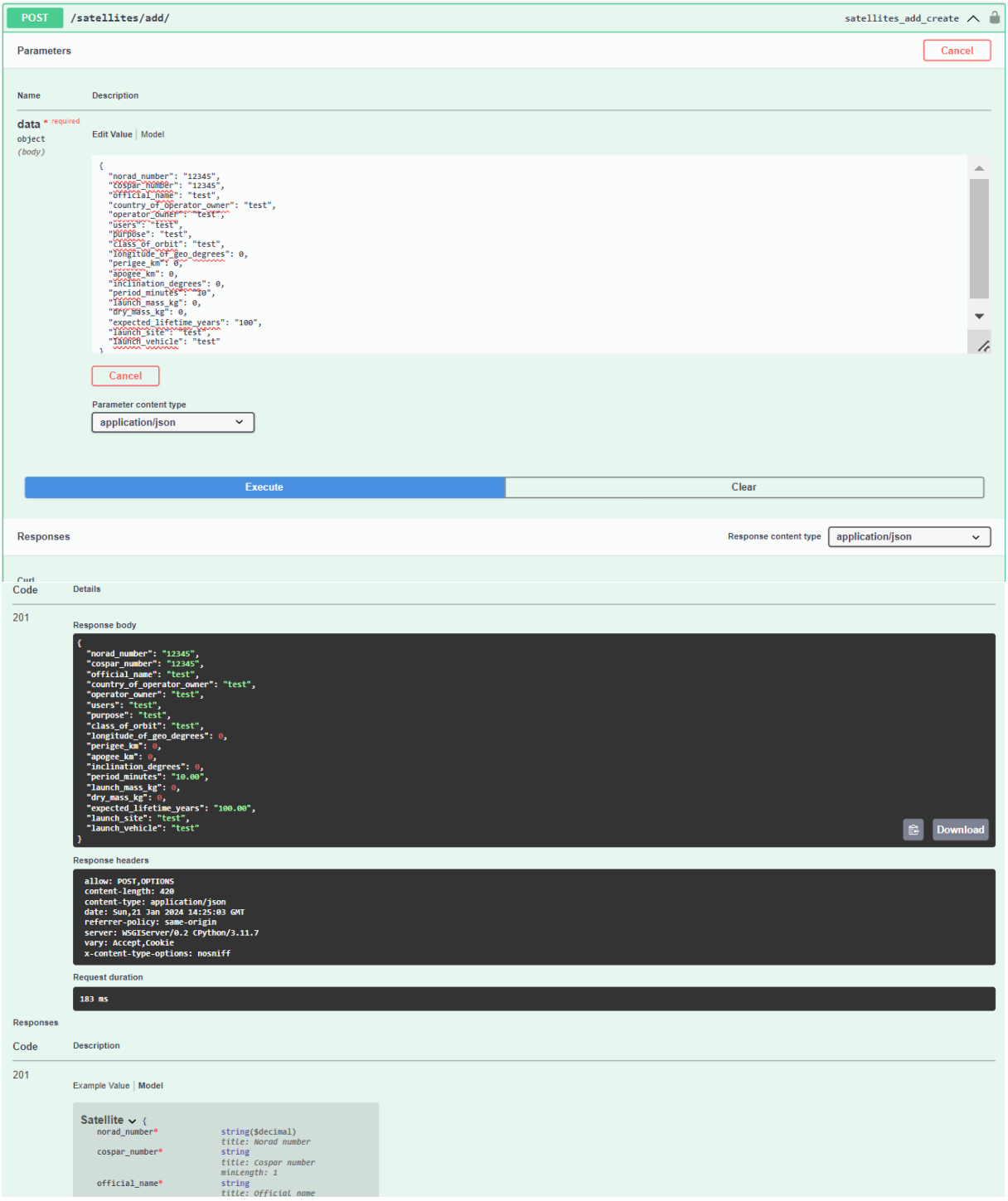
GET Country Satellite Ranking



Modify Purpose by Norad Number



Add Satellite



Delete Satellite by Norad Number