**Technical Documentation: AIS Google Map Viewer Application**

This documentation provides a detailed description of the **AIS Google Map Viewer** application, which displays real-time ship positions using AIS (Automatic Identification System) data and Google Maps. The application consists of a backend in Python (using Flask) and a frontend in HTML/JavaScript.

**Table of Contents**

1. **Application Overview**
2. **Code Structure**
   * Backend (Python)
   * Frontend (HTML/JavaScript)
3. **Method Details**
   * Backend
   * Frontend
4. **Configuration and Execution**
5. **Dependencies**
6. **Extensions and Customizations**

**1. Application Overview**

The application connects to an AIS server to receive real-time data on ship positions. This data is processed and displayed on an interactive Google Map. The user can configure the AIS server IP, port, Google API key, and the msim value of the ship to track.

**2. Code Structure**

**Backend (Python)**

* **Flask**: Web framework for handling HTTP requests and serving the HTML page.
* **PyAIS**: Library for connecting to an AIS server and decoding messages.
* **Threading**: Used to manage the AIS connection in a separate thread.

**Frontend (HTML/JavaScript)**

* **Google Maps API**: Displays the map and ship markers.
* **Fetch API**: Updates ship data in real time.
* **HTML/CSS**: User interface and page styling.

**3. Method Details**

**Backend (Python)**

**connect\_to\_ais\_stream()**

* **Description**: Connects to the AIS server using the specified IP and port.
* **Parameters**: None.
* **Functionality**:
  + Establishes a TCP connection to the AIS server.
  + Decodes AIS messages and calls process\_ais\_data to process them.

**process\_ais\_data(decoded\_message)**

* **Description**: Processes decoded AIS messages and updates the SHIPS dictionary.
* **Parameters**:
  + decoded\_message: Decoded AIS message.
* **Functionality**:
  + Extracts the fields mmsi, lat, lon, status, turn, speed, course, and heading.
  + Updates the SHIPS dictionary with the ship's data.

**index()**

* **Description**: Returns the main HTML page with the map.
* **Parameters**: None.
* **Functionality**:
  + Renders the map.html template, passing the Google API key and target\_msim value.

**get\_ships()**

* **Description**: Returns ship data in JSON format.
* **Parameters**: None.
* **Functionality**:
  + Converts the SHIPS dictionary to JSON and returns it.

**configure()**

* **Description**: Handles the configuration page to set application parameters.
* **Parameters**: None.
* **Functionality**:
  + If the request is a POST, updates global variables with new values.
  + Returns an HTML form for configuration.

**start\_ais\_thread()**

* **Description**: Starts a separate thread for the AIS connection.
* **Parameters**: None.
* **Functionality**:
  + Creates and starts a thread that runs connect\_to\_ais\_stream.

**Frontend (HTML/JavaScript)**

**initMap()**

* **Description**: Initializes the Google Map.
* **Parameters**: None.
* **Functionality**:
  + Creates a new instance of google.maps.Map centered at (0, 0).
  + Sets an interval to update the map every 2 seconds.

**updateMap()**

* **Description**: Updates the markers on the map with ship data.
* **Parameters**: None.
* **Functionality**:
  + Makes a fetch request to /get\_ships to retrieve ship data.
  + Updates or creates markers on the map based on the received data.

**4. Configuration and Execution**

**Steps to Run the Application**

1. **Install Dependencies**:

bash

Copy

pip install flask pyais

1. **Run the Application**:

bash

Copy

python app.py

1. **Open the Browser**:
   * Go to http://127.0.0.1:5000 to view the map.
   * Go to http://127.0.0.1:5000/configure to modify parameters.

**5. Dependencies**

**Backend**

* **Flask**: Web framework.
* **PyAIS**: Library for decoding AIS messages.
* **Threading**: For managing the AIS connection in a separate thread.

**Frontend**

* **Google Maps API**: For displaying the map.
* **Fetch API**: For updating ship data.

**6. Extensions and Customizations**

**Adding New AIS Fields**

* Modify the process\_ais\_data method to extract and handle new fields from AIS messages.

**Customizing Map Style**

* Update the map.html file to customize the appearance of the map and markers.

**Creating an Executable**

* Use PyInstaller to create an executable that includes the templates folder and other necessary files.

**Conclusion**

The **AIS Google Map Viewer** application is a flexible and configurable tool for visualizing real-time ship positions. Thanks to its modular architecture, it can be extended and customized to meet various needs.