Overview:

This Python script is designed as a bot for web scraping foreclosure auction data from various county websites. It extracts foreclosure dates from calendars, navigates through auction listings, retrieves auction items, and performs various actions based on the extracted data. The bot utilizes Selenium for web automation, ParSel for HTML parsing, and makes use of external APIs and services for data storage and notifications.

Dependencies:

- **time.sleep**: Used for adding delays in the script execution.
- **Selector** from **parsel**: A library for extracting data from HTML and XML using XPath and CSS selectors.
- **pandas**: A powerful data manipulation library.
- **requests**: A library for making HTTP requests.
- **urlparse** from **urllib.parse**: Used for parsing URLs.
- **datetime**: Provides classes for manipulating dates and times.
- **pprint**: Used for pretty-printing data structures.

Custom Modules:

- **helper_functions**: Contains various helper functions used by the bot.

Configuration:

- **API_ENDPOINT_PRODUCTION**: Endpoint for the production CRM API.
- **API_ENDPOINT_TESTING**: Endpoint for the testing CRM API.
- **today_date_time**: Current date and time formatted as "YYYY-MM-DD_HH-MM".
- Boolean flags (**send_to_api**, **save_to_csv**, **save_to_google_sheets**, **send_alerts_to_telegram**) control the behavior of the bot.

Main Functionality:

1. **Scraping Workflow**:
    - Fetches URLs to scrape from a file.
    - Iterates through each URL.
    - Navigates to the URL, retrieves foreclosure dates, and proceeds to scrape auction data.
    - Handles errors and retries for robustness.
2. **Data Extraction and Processing**:
    - Retrieves foreclosure dates from calendar elements.

- Scrapes auction data from auction listing pages.
- Processes extracted data and performs various actions based on configuration settings.

3. **Data Storage and Reporting**:
    - Saves scraped data to CSV files if enabled.
    - Sends data to Google Sheets if enabled.
    - Sends alerts to Telegram at the start, finish, and upon errors if enabled.

Error Handling:

- Implements error handling mechanisms to deal with exceptions during the scraping process.
- Retries certain operations in case of failures to enhance robustness.

Main Function:

- The **main()** function orchestrates the entire scraping process.
- It iterates through URLs, extracts data, performs actions, and manages reporting and error handling.

Entry Point:

- The script's main block ensures continuous execution and handles exceptions gracefully.
- It sends alerts to Telegram at various stages of the execution.