**Bug Tracking Script Documentation**

**Introduction**

The "Bug Tracking Script" (sendable\_bug\_tracker.py) is designed to fetch bug data from the Bugzilla API, analyze the data, and generate a chart in an Excel file. This script offers a user-friendly way to visualize bug statistics over a two-week period. The script also provides options for customizing the bug search criteria, such as project, component, and severity.

**Functionality**

1. **Function Signature**
   * **run\_script(product: str, component: str, severity: str) -> None**
2. **Inputs**
   * **product (str):** The name of the project or product for bug tracking.
   * **component (str):** The specific component within the product.
   * **severity (str):** The severity level of the bugs to track.
3. **Outputs**
   * A chart displaying bug statistics over the last two weeks is generated in an Excel file.
   * The chart is also saved as a PNG image. It is pulled up on the device for viewing.

**Steps**

1. **Calculate Dates**
   * The script calculates the last 14 dates in the format MM/DD/YYYY for the horizontal axis of the chart.
   * The same dates in the format YYYY-MM-DD are calculated as well for querying the Bugzilla API for bug data.
2. **API Data Retrieval**
   * The script uses the provided product, component, and severity to construct a URL for querying the Bugzilla API.
   * The API key is used for authentication.
   * For each date in the last two weeks, the script fetches bug data related to the specified criteria.
3. **Data Processing**
   * The script counts the number of open and closed bugs for each date.
   * The counters are stored in arrays.
4. **Bugs Created Calculation**
   * The script calculates the number of bugs created on a specific day by subtracting the count of bugs created on (and after) that day from the count of bugs created on (and after) the next day.
5. **DataFrame Creation**
   * The collected data is structured into a DataFrame.
   * The DataFrame contains columns: "Day," "Open Bugs," and "Closed Bugs."
6. **Excel Interaction**
   * The script opens an existing Excel file (xlsx format).
     1. Ensure that the Excel file path is -

"D:/OneDrive - TMEIC/Desktop/Bugzilla Tracker/SearchedBugsPlot.xlsx"

* + The DataFrame data is written to a specified range on a sheet (“Sheet1”).
  + The Excel chart named "Chart 1" on the sheet is updated with the new data.

1. **Chart Image Generation**
   * The script captures the updated chart as a PNG image.
   * The image is saved to the current directory as "chart\_screenshot.png."
2. **Script Conclusion**
   * The Excel workbook is saved and closed.
   * The Excel application is quit.
   * The generated chart image is displayed using the PIL library.

**Usage**

1. Import the necessary libraries: **requests**, **json**, **pandas**, **datetime**, **timedelta**, **xlwings**, **PIL**, and **os**.
2. Call the **run\_script** function with the following parameters:
   * **product:** The project or product name for bug tracking.
   * **component:** The component related to the bugs.
   * **severity:** The severity level of the bugs.
3. Observe the generated Excel chart displaying bug statistics over the last two weeks.

**Conclusion**

The "Bug Tracking Script" provides an automated way to fetch and visualize bug statistics from the Bugzilla API. By specifying the project, component, and severity, users can quickly analyze bug trends over a two-week period. The script's integration with Excel and chart visualization enhances bug tracking and analysis capabilities.