**Critical Design Review Document**

**Web Scraper For Git Projects**

**Development team:**

Marouthu, Sai Nikith

Rapolu, Ravi Teja

Gudiseva, Bhavya Sree

Cheedella, Prudhvi Lakshmi Sesha Sai

Gummadi, Karthik

Distribution clause:

This document was created as a deliverable for the spring 2023 CSCI 6235 course at the George Washington University, 1918 F Street, NW, Washington, DC 20052. Distribution is unrestricted provided the source is referenced.

Table of Contents

[Introduction 1](#_Toc182494592)

[Software Functionality 2](#_Toc182494593)

[Accessing the Landing Page 2](#_Toc182494594)

[Entering Keywords into the Search Box 3](#_Toc182494595)

[Keyword Limitation Validator 5](#_Toc182494596)

[Character Limitation Validator 6](#_Toc182494597)

[Searching After Selected Keywords - Loading Screen 6](#_Toc182494598)

[Getting the results (loading screen and listing screen) 7](#_Toc182494599)

[Pagination of Project List 9](#_Toc182494600)

[Navigating to the Git page via the project name 11](#_Toc182494601)

[Accessing Additional details for a project 12](#_Toc182494602)

[Displaying project dependencies visualization 13](#_Toc182494603)

[Feedback received during Critical Design Review 15](#_Toc182494604)

[Feedback Resolution 16](#_Toc182494605)

# **Introduction**

The Web Scrapper for GIT Projects is designed to automate the process of retrieving data from GIT repositories. This tool will gather information about GIT projects, categorize them based on predefined keywords, and make the projects easily searchable. Users can click directly on the project names to navigate to the respective GIT pages. Additionally, the project will provide a visual representation of relationships between projects using a graphical interface, helping users understand the connections and similarities between different repositories.  
  
  
**List of Stakeholders:**

**Software Developers:** Developers can use the tool to quickly find relevant GIT repositories based on keywords and explore similar projects.

**Project Managers:** They will be able to oversee project relationships and dependencies, aiding in task allocation and team coordination.

**Researchers:** They can easily discover relevant projects and understand project relationships for academic or industrial research purposes.

**Open-Source Contributors:** Contributors looking for projects to engage with can search by interest area and directly access GIT pages for contribution opportunities.

**Technology Enthusiasts:** Casual users interested in exploring open-source projects will benefit from the easy navigation and visual exploration features of the tool.

This document outlines the functionality of the Git Web Scrapper project, explaining how its stakeholders will utilize the system and interact with its features. It details the core functionality, interface expectations, and intended user experience. All stakeholders are expected to review and acknowledge that the functionalities and specifications described herein will meet their project requirements.

# **Software Functionality**

## **Accessing the Landing Page**

The **Landing Page** for the “GIT WEB SCRAPPER” serves as the entry point for the users to begin their search for relevant GIT repositories. It offers a simple and intuitive interface, making it easy to navigate.

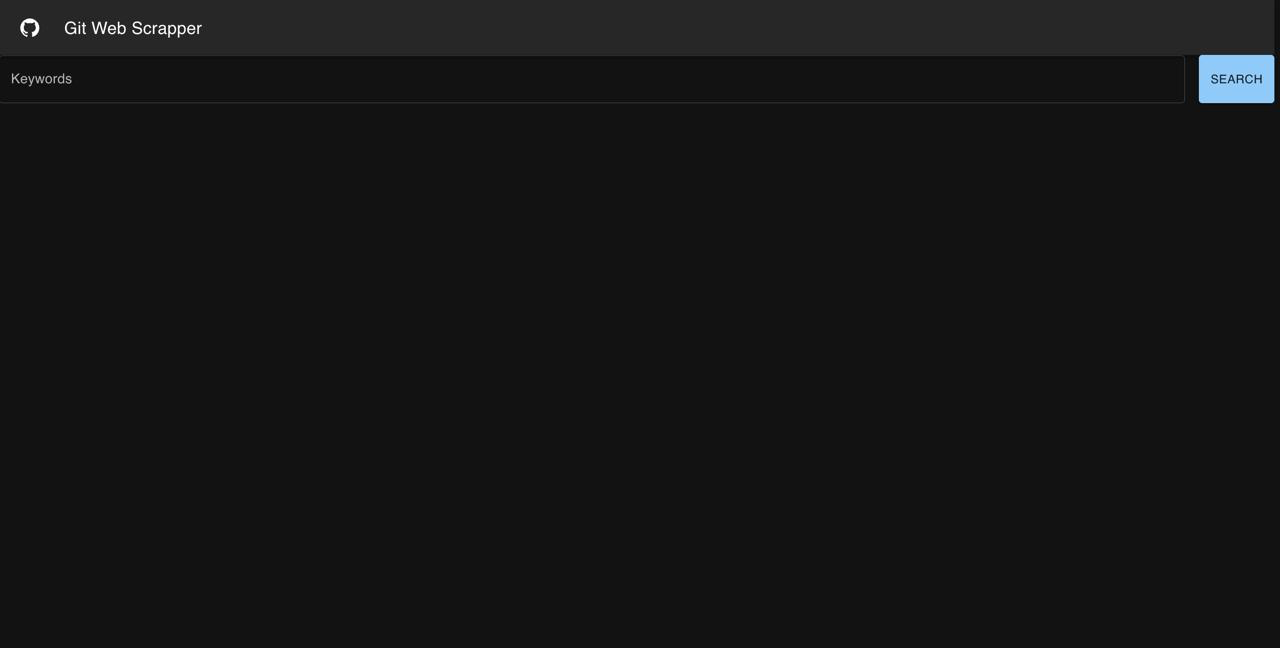


Figure 1. Landing page for Git Web Scraper

***Key Features:***

**Header:** At the top, the header contains the project name "GIT Web Scrapper", clearly indicating the purpose and title of the tool. The GitHub icon on the left emphasizes the platform from which data will be retrieved.

**Keyword Search:** The main functionality on the landing page is a search bar, where users can input keywords related to their desired projects. After entering the keywords, users can click on the "Search" button to retrieve and filter GIT repositories based on the provided keywords.

**Minimalistic Design:** The page adopts a clean, black theme for easy readability, allowing users to focus on the core task of searching GIT projects. This design provides a distraction-free environment, optimized for efficiency.

## **Entering Keywords into the Search Box**

This page allows users to input specific keywords to search for relevant GIT repositories. The keyword search is the primary function that drives the data retrieval process from the GIT platform.

A black rectangular object with blue lines

Description automatically generated

Figure 2: Entering the keywords into Search Box

#### **Key Features:**

**Keywords Input**: Users can enter one or more keywords into the **"Keywords"** search box, as shown in the screenshot. This input field is the focal point where users define their search parameters, specifying project names, technologies, or any relevant tags they are interested in.

**Keyword Entry - Pill Conversion**

Once a keyword is entered into the search box and the user presses Enter, the entered keyword is transformed into a pill-shaped tag. This feature allows users to visually manage their search terms more effectively.

A black screen with blue lines

Description automatically generated

Figure 3 Pressing Enter makes keyword a Pill

**Pill Representation:** After pressing Enter, the keyword "undefined" appears as a pill-shaped label. This label represents the keyword that will be used in the search. The pill format is a convenient way to display keywords, making it easy for users to distinguish between multiple search terms.

**Keyword Management:** Users can add multiple keywords, and each new keyword will also be transformed into a pill. If necessary, the user can remove a keyword by clicking the "X" button on the right side of each pill.

**Search Process:** The pill-based keyword entry system helps organize search terms, ensuring clarity for users as they refine their searches. The Search button, which remains on the right side, will execute the query based on the pills currently displayed.

## **Keyword Limitation Validator**

To ensure users input a manageable number of search terms, the system enforces keyword limitations. In this case, a maximum of **five keywords** can be entered into the search box.

A black and red rectangle with a black border

Description automatically generated

Figure 4 Pressing Enter after typing 6th keyword

***Key Features:***

**Keyword Entry Limit:** The Figure 4 shows five keywords ("matrix", "MIT", "addition", "sub", and "abc") entered as pill-shaped tags. Upon attempting to add a sixth keyword, the system activates the keyword limit validator.

**Validation Message:** Below the keyword entry box, a message is displayed: "You can only add up to 5 keywords." This informs the user that no further keywords can be added beyond this limit, maintaining simplicity and preventing overwhelming searches.

**Visual Feedback:** The search box and entered keyword pills are highlighted with a red border, signaling to the user that they've reached the maximum allowable keywords and prompting them to remove a keyword if they wish to add a new one.

## **Character Limitation Validator**

To ensure users input concise search queries, the system enforces a character limit. In this case, a maximum of 256 characters can be entered into the search box.

**A screenshot of a computer

Description automatically generated**Figure 5 Pressing Enter after typing more than 256 characters.

**Character Entry Limit:** Users can type up to 256 characters, encompassing all keywords, operators, and qualifiers within a single search query. As shown in Figure 5, once the 256-character limit is reached, no additional characters can be typed.

**Validation Message:** When users attempt to enter more than 256 characters, a message appears below the search box, stating: "Search queries cannot exceed 256 characters." This informs users that they have reached the maximum character limit and need to shorten their query if they wish to proceed.

**Visual Feedback:** The search box and any text beyond the character limit are highlighted with a red border, signaling to the user that they have exceeded the allowable query length. This prompts them to adjust their search content for clarity and relevance.

## **Searching After Selected Keywords - Loading Screen**

Once the user has entered the desired keywords and pressed the Search button, the system initiates the search process, displaying a loading screen as it queries the GIT repositories.

A black screen with a white circle

Description automatically generated

Figure 6 Searching after selected keywords (Loading Screen)

***Key Features:***

**Keyword Pills Displayed:** At the top of the screen, the keywords "undefined" and "not" are represented as pill-shaped tags. This confirms that the system has recognized the entered keywords and is processing the search based on those inputs.

**Loading Indicator:** A loading icon appears at the center of the screen, signifying that the search process is underway. This loading indicator provides visual feedback to the user, ensuring them that the system is actively fetching data from GIT.

**Search Button Disabled:** During the loading phase, the search button is typically disabled, preventing additional inputs or actions until the search results are retrieved.

## **Getting the results (loading screen and listing screen)**

After the user submits their keyword search, the results are displayed in a tabular format. This organized presentation of GIT repositories provides a detailed overview of the relevant projects based on the keywords entered.

A screenshot of a computer

Description automatically generated

Figure 7 Search Results in a table

***Key Features:***

**Table Structure:** The results are presented in a neatly structured table format, with each row representing a GIT repository that matches the user's search criteria. This format allows for easy comparison between different projects.

**Column Breakdown:**

**ID:** A unique identifier for each repository.

**Repo Name:** The name of the repository, which includes the owner and the project name. The names are clickable links that redirect users to the specific GIT page for the project.

**Description:** This column provides a short description of the repository’s description.

**Origin and Pedigree:** This column provides additional context about the repository's origin or the primary contributors.

**Support:** This displays the last time the repository was updated, giving users an idea of how active the project is.

**License:** The type of license the repository uses, such as MIT License or Apache License 2.0, providing critical information on how the project can be reused.

**Details Page:** A button labeled "View Details" that, when clicked, opens a detailed view of the repository, providing more in-depth information.

**Pagination Controls:** At the bottom right, pagination controls allow users to navigate through the search results if more than 10 repositories are returned. This keeps the page from becoming overloaded with too many entries at once.

## **Pagination of Project List**

The **pagination feature** allows users to navigate through large sets of search results by dividing them into pages. This ensures that the user interface remains manageable, even when dealing with numerous project results.

A screenshot of a computer

Description automatically generated

Figure 8 Pagination of projects

***Key Features:***

**Pagination Controls:** The pagination controls appear at the bottom right of the table. Users can navigate through the result pages using the forward and backward arrows, which allow them to move between pages.

The label indicates that the user is currently viewing results 1–20 of 25. This provides a clear understanding of how many results are available and which ones are currently visible.

**Efficient Navigation:** Pagination ensures that users are not overwhelmed by too many entries at once, making the interface cleaner and more user-friendly. It also reduces load times, as fewer results are displayed at once.

## **Navigating to the Git page via the project name**

The project list includes clickable hyperlinks for each project name, allowing users to directly navigate to the corresponding GIT repository page.

A screenshot of a computer

Description automatically generated

Figure 9 Clicking on the Hyperlink as title of the project

***Key Features:***

**Hyperlink Project Names:** In the Repo Name column, each project title is displayed as a clickable hyperlink. Clicking on the project name redirects users to the project's GIT repository page, providing easy access to the project's code, documentation, and other related information.

**Direct Navigation:** This feature simplifies the process of exploring specific GIT repositories, as users can immediately visit the GIT page without manually searching for the project elsewhere.

**Additional Project Details:** The table also retains other relevant information stars, Origin and Pedigree, Last Updated, and License. These details give users context about the project before they choose to visit the GIT repository.

**View Details Button:** In addition to clicking the project name, users can also click the View Details button to gather more information directly within the web scrapper tool before deciding to visit the GIT repository.

## **Accessing Additional details for a project**

This page provides in-depth information about a specific project, focusing on key attributes such as Pluggability, Extensibility, and Dependencies, which are critical for understanding how easily the project can be integrated or expanded.

A screenshot of a computer

Description automatically generated

Figure 10 Project Details Page containing Pluggability and Extensiility

***Key Features:***

**Pluggability and Extensibility Scores:**

The page displays Pluggability and Extensibility scores, each assigned a value (in this case, both are 100). These scores reflect the project's flexibility in terms of integrating with other systems (pluggability) and adapting to new features or changes (extensibility). Higher scores suggest that the project is more versatile and easier to integrate into larger workflows.

**Dependency Tree:** A detailed Dependency Tree is shown on the right, listing key dependencies of the project. These are other libraries or modules the project relies on. Users can expand or collapse each dependency to view additional information, making it easier to assess the project's ecosystem and compatibility.

**Support and Updates:** The Support section provides the last update time for the project, helping users gauge how actively the project is maintained. In this example, the last update occurred on September 2, 2024.

**Origin and Pedigree:** The Origin and Pedigree section shows the project's primary contributors or maintainers, allowing users to trace the project back to its developers or source.

**License:** The project’s License is displayed, which is Apache License 2.0 in this case. This helps users understand the legal framework under which the project can be used or modified.

## **Displaying project dependencies visualization**

This section of the project details page provides a visualization of the project's dependency tree, allowing users to explore the hierarchy and relationships of the project's dependencies.

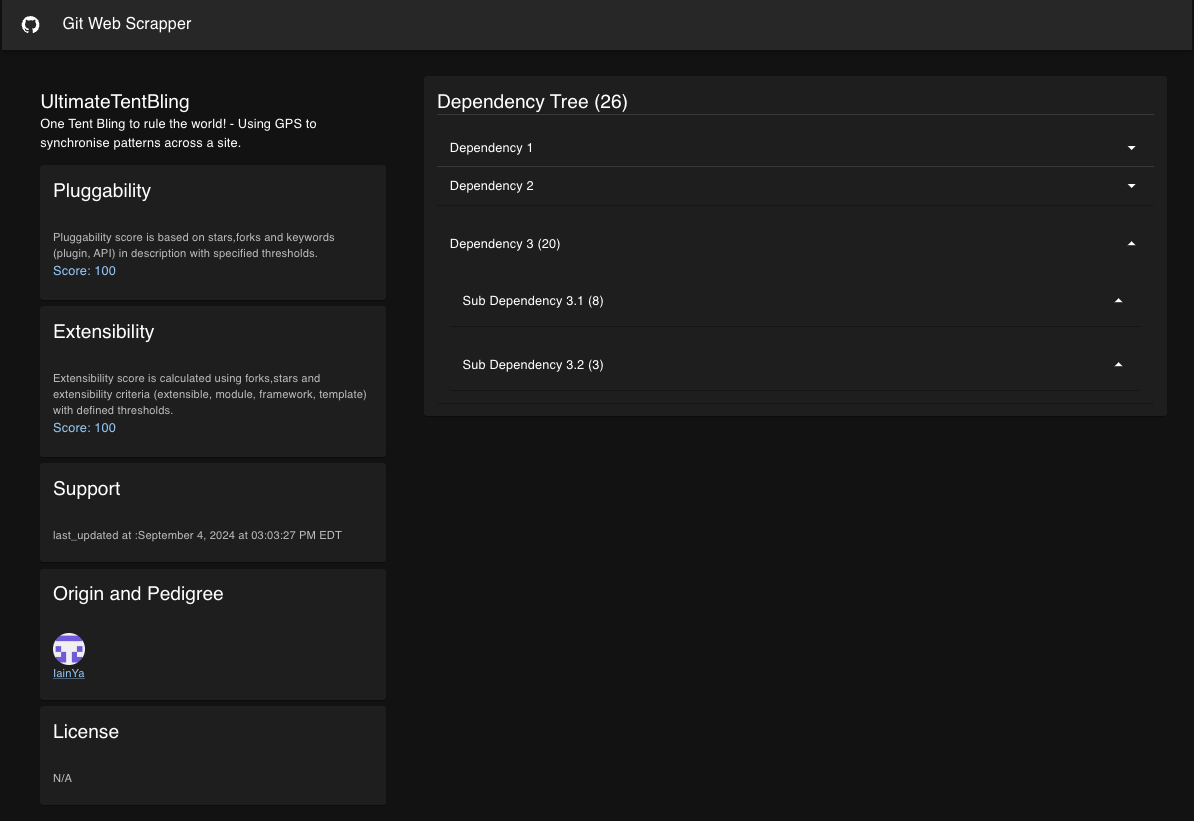


Figure 11 Dependency tree of a project expandable

***Key Features:***

**Dependency Tree Visualization:** The Dependency Tree, located on the right side of the page, displays a list of dependencies that the project relies on, with the total count of dependencies shown in parentheses beside the main "Dependencies" label. This expandable structure allows users to click on each dependency to reveal more details, including any sub-dependencies. When a dependency is expanded, a count is displayed next to each item, indicating the number of dependencies at each level. The tree can display up to 3 levels of dependencies, if available on GitHub. If fewer levels are available, only those levels will be shown. For example, expanding Dependency 3 reveals Sub Dependency 3.1 and Sub Dependency 3.2, along with their respective counts, providing a clear view of how dependencies are interconnected and highlighting the overall structure and hierarchy of project dependencies.

**Interactive Exploration:** Users can interact with the dependency tree, expanding or collapsing dependencies to view their structure in greater detail. This helps in understanding the full scope of external libraries or modules the project requires.

**Pluggability and Extensibility Scores:** On the left, the Pluggability and Extensibility scores provide insight into the project's flexibility for integration and adaptation. Both scores are displayed as 100, indicating a highly adaptable and pluggable project. Additionally, there is a static field that explains how these scores have been calculated, offering users transparency into the criteria and methodology behind the scoring system.

**Support and Maintenance:** The Support section provides the last updated time for the project, ensuring that users can assess how actively the project is maintained. In this example, the last update occurred on September 4, 2024.

**Origin and Pedigree:** The Origin and Pedigree section displays the key contributors or maintainers of the project, helping users understand the project's source and potential support structure.

# **Feedback received during Critical Design Review**

1. Stakeholder: Renato, Levy
   1. Replace the stars and forks on the Project Listing Screen with the short description of the project
   2. Display the count of each dependency's sub-dependencies in parentheses within the dependency tree.

# **Feedback Resolution**

A. The "forks" column and “stars” on the Project Listing Screen has been removed and replaced with a "description" column. This update pulls the short description directly from the project repository, providing users with a clearer overview of each project briefly.

B. The Dependency Tree now displays the total count of dependencies at the top of the tree, providing a quick overview of the project's overall complexity. Each dependency has the count of its sub-dependencies shown in parentheses beside it, offering an at-a-glance view of its hierarchical structure. When a dependency is expanded, the sub-dependency count updates dynamically, revealing the number of dependencies nested within each level. This interactive feature allows users to efficiently explore and assess the full dependency scope by clicking to expand each level.