|  |
| --- |
| Google Reviews Scraper  Deliverable 1 |

|  |  |  |
| --- | --- | --- |
| Group 1 |  |  |
| Monique Dai/ Chester Rae de Vera/ Heping Song  Snehalata / Hua Zhang / Yateng Geng / Chester Rae de Vera/ Heping Song |  |  |



Contents

1. Requirements \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3

1.1 Functional Requirements \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3

1.2 Non-functional Requirements \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3 & 4

1.3 UI/UX Considerations \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 4

2. Project Plan \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 5

2.1 Milestones \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 5

2.2 Tasks \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 5 & 6

2.3 Deadlines \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 6

3. Roles and Responsibilities \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7

3.1 Monique Dai \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7

3.2 Chester Rae de Vera \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 7

3.3 Heping Song \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 8

# 1. requirements

**1.1 Functional Requirements**

**a. User Inputs:**

1. Allow the user to enter the business or location name for which they want to scrape reviews.

**b. Scraping:**

1. Use Python to automate the process of querying Google and extracting review data.
2. Retrieve review details, including reviewer name, rating, date, and comments.
3. Handle pagination to ensure all reviews are scraped.

**c. Data Storage:**

1. Store the scraped review data in a structured format, such as CSV, JSON, or a database.

**d. User Interface:**

1. Develop a simple and intuitive command-line or graphical user interface (GUI) to interact with the scraper.
2. Display progress indicators during the scraping process.
3. Provide feedback or error messages for invalid inputs or connection issues.

**e. Exception Handling:**

1. Implement error handling mechanisms to address common exceptions, such as connection errors or timeouts.

**f. Logging:**

1. Implement logging functionality to record important events, errors, and debugging information.

**1.2 Non-functional Requirements**

**a. Performance:**

1. Optimize the scraper for efficient and timely retrieval of reviews.

**b. Reliability:**

1. Ensure the scraper handles unexpected scenarios gracefully and recovers from failures.

**c. Security:**

1. Handle user inputs securely to prevent injection attacks.

**d. Maintainability:**

1. Write clean, modular, and well-documented code for easy maintenance and future enhancements.

**1.3 UI/UX Considerations**

**a. Framework Selection:**

1. Choose an appropriate framework for developing the user interface based on requirements, such as PyQt, Tkinter, or Flask.

**b. User-Friendly Design:**

1. Design an intuitive and visually appealing user interface.
2. Provide clear instructions and labels for input fields and buttons.

**c. Progress Indicators:**

1. Display loading bars, spinners, or progress percentages to indicate the progress of the scraping process.

**d. Error Handling:**

1. Display user-friendly error messages when inputs are invalid or when the scraper encounters issues.

**e. Responsiveness:**

1. Ensure the user interface is responsive and provides timely feedback during the scraping process.

# 2. project plan

**2.1 Milestones**

**a.** **Project Setup and Requirements Gathering** - Familiarization with Python and web scraping techniques, requirements elicitation and documentation, project plan creation, and role assignment. (Due: June 14, 2023)

**b.** **Core Functionality and Scraper Implementation** - Design of scraper architecture, implementation of the core functionality and scraping logic, and setup of error handling mechanisms. (Due: June 21, 2023)

**c.** **User Interface Development and Testing** - Selection of UI framework, UI design and integration with scraper functionality, comprehensive system testing. (Due: June 28, 2023)

**d.** **Refinement, Documentation, and Finalization** - Codebase refinement, comprehensive testing, user documentation creation, final report, final code review, and project deployment. (Due: July 5, 2023)

**2.2 Tasks**

**a. Deliverable 1: Project Setup and Requirements Gathering**

1. Familiarize team with Python and web scraping techniques.
2. Conduct requirements elicitation sessions and document the requirements.
3. Create a detailed project plan including tasks and deadlines.
4. Assign roles and responsibilities to team members.

**b. Deliverable 2: Core Functionality and Scraper Implementation**

1. Design the scraper's architecture.
2. Implement the core functionality of the scraper.
3. Implement the scraping logic to retrieve review details and handle pagination.
4. Set up error handling mechanisms.

**c. Deliverable 3: User Interface Development and Testing**

1. Select a suitable framework for UI development.
2. Design the user interface and integrate with the scraper's functionality.
3. Conduct thorough testing of the system, including the user interface.
4. Handle user inputs securely to prevent injection attacks.

**d. Deliverable 4: Refinement, Documentation, and Finalization**

1. Refine the codebase and ensure adherence to coding standards.
2. Conduct comprehensive testing and debugging.
3. Prepare user documentation.
4. Prepare a final report summarizing the project.
5. Conduct a final code review.

**2.3 Deadlines**

1. Deliverable 1: Project Setup and Requirements Gathering: June 14, 2023
2. Deliverable 2: Core Functionality and Scraper Implementation: June 21, 2023
3. Deliverable 3: User Interface Development and Testing: June 28, 2023
4. Deliverable 4: Refinement, Documentation, and Finalization: July 5, 2023

# 3. roles and responsibilities

**3.1 Monique Dai**

1. **Role: Project Manager, QA engineer**
2. **Responsibilities:**
3. Familiarize the team with Python and web scraping techniques (Deliverable 1).
4. Conduct thorough testing of the system, including the user interface (Deliverable 3).
5. Refine the codebase and ensure adherence to coding standards (Deliverable 4).
6. Conduct comprehensive testing and debugging (Deliverable 4).
7. Prepare user documentation (Deliverable 4).
8. Prepare a final report summarizing the project (Deliverable 4).
9. Upload solutions (Deliverable 1,2,3 and 4).
10. Monitor project progress, manage risks, and handle any issues that may arise during the development process (Deliverable 1,2,3 and 4).
11. Oversee Github repository management and track issues, ensuring tasks are clearly defined and distributed.

**3.2 Chester Rae de Vera**

1. **Role:** **Full stack Developer**
2. **Responsibilities:**
3. Conduct requirements elicitation sessions and document the requirements (Deliverable 1).
4. Create a detailed project plan including tasks and deadlines (Deliverable 1).
5. Asist in designing the scraper's architecture (Deliverable 2).
6. Set up error handling mechanisms (Deliverable 2).
7. Implement login system and the database related (Deliverable 3).
8. Handle user inputs securely to prevent injection attacks (Deliverable 3).
9. Design the user interface and integrate with the scraper's functionality (Deliverable 3).
10. Assist in comprehensive testing and debugging (Deliverable 4).
11. Assist in a final code review (Deliverable 4).
12. Create Github repository, developing features, making regular code commits to his branch, and reviewing code from other branches before merging.

**3.3 Heping Song**

1. **Role: Full stack Developer**
2. **Responsibilities:**

Responsibilities:

1. List and assign roles and responsibilities to team members (Deliverable 1).
2. Implement the core functionality of the scraper (Deliverable 2).
3. Implement the scraping logic to retrieve review details and handle pagination (Deliverable 2).
4. Design the scraper's architecture (Deliverable 2).
5. Select a suitable framework for UI development (Deliverable 3).
6. Design the user interface and integrate with the scraper's functionality (Deliverable 3).
7. Conduct a final code review (Deliverable 4).
8. Assist in comprehensive testing and debugging (Deliverable 4).
9. Handle Github pull requests, merge approved changes into the main branch, and contribute to maintaining and updating the project documentation.