Web Scraping

By

Sania Sikandar	**2021-GCUF-07088**
Mahnoor masood	**2019-GCUF-080847**

BACHELOR OF SCIENCE

IN

COMPUTERSCIENCE

DEPARTMENT OF COMPUTER SCIENCE

Government College University Faisalabad

AKNOWLEDGEMENT

All Praises for "**Almighty Allah**" Lord of all worlds the most Affectionate, the most Merciful, who presented me in a Muslim community and who is the entire source of all knowledge and wisdom. After that we thanks for the Holy Prophet Muhammad (PBUH) who is internal torch of guidance for humanity and enabled us to recognize our creator, forever model of guidance and knowledge for humanity.

We are also grateful to our **Supervisor ------ Uzma Jameel ----------**for this invaluable guidance and blessings. We are very grateful to our **Coordinator-----(Name)** **----------** fo

r

providing us with an environment to complete our project successfully.

We are deeply indebted to our **Teachers ----------**who modeled us both technically and morally

for achieving greater success in life. We also thank all the staff members of our college and technicians for their help in making this project a successful on finally, we take this opportunity to extend our deep appreciation to our **family and friends**, for all that they meant to us during the crucial times off the completion of our project.

Name: Sania Sikandar

Registration No. 2021-GCUF-07088

Name: Mahnoor masood

Registration No. 2019-GCUF-080847

Dedication

I dedicate this project to all those whose unwavering support and guidance have made this journey possible.

To our beloved supervisor [Supervisor's Name], thank you for being the guiding light throughout this project. Your expertise, encouragement, and patience have been instrumental in shaping our ideas and honing our skills. Your constant motivation and insightful feedback have pushed us to achieve greater heights. We are grateful for your mentorship and the invaluable knowledge we have gained under your tutelage.

To our ever-supportive parents, who have been the pillars of strength in our lives, this project is a tribute to your boundless love and sacrifices. Your unwavering belief in our abilities has been a constant source of inspiration. Your encouragement and understanding during challenging times have helped us stay focused and determined. We are forever indebted to you for instilling in us the values of perseverance and hard work.

This project is a testament to the power of collaboration and the impact of unwavering support. To all our friends and loved ones who stood by us throughout this journey, your presence has made this experience more fulfilling.

Lastly, we dedicate this work to the pursuit of knowledge and the spirit of innovation. May this project contribute to the broader academic community and inspire others to explore the limitless possibilities in the field of web scraping.

Thank you all for being an integral part of this journey.

With heartfelt gratitude,

Name: Sania Sikandar

Registration No. 2021-GCUF-07088

Name: Mahnoor masood

Registration No. 2019-GCUF-080847

Table of Contents

AKNOWLEDGEMENT....3
Dedication....4
List of Tables....7
Chapter No 1: Introduction to the Problem....9
1.1 Introduction....9
1.2 Background....10
1.3 Purpose....11
1.4 Scope....13
1.5 Objective....15
1.6 Intended Audience and Reading Suggestions....17
1.7 Document Conventions....19
Chapter No 2: Software Requirement Specification....21
2.1 Overall Description....21
2.1.1 Product Perspectives....21
2.1.2 Product Features....23
2.1.3 Design and Implementation Constraints....25
2.1.4 Assumption and Dependencies....26
2.2 System Features....29
2.3 External Interface Requirements....32
2.3.1 User Interfaces....32
2.3.2 Hardware Interfaces....33
2.3.3 Software Interfaces....34
2.3.4 Communications Interfaces....35
2.4 Other Nonfunctional Requirements....36
2.4.1 Performance Requirements....36
2.4.2 Safety Requirements....38
2.2.3 Security Requirements....40
Chapter No 3: Analysis (use case model)....43
3.1 Identifying Actors and Use Cases using Textual Analysis....43
3.2 Forming Use Case Diagram with Candidate and Use Cases....45
3.3 Describe the Events Flow for Use Case....45
Chapter No 4: Design....47
4.1 Architecture Diagram....47

4.2 ERD with Data Dictionary 48
4.3 Data Flow Diagram (Level 0 and Level 1) 49
4.4 Class Diagram 51
4.5 Object Diagram 51
4.6 Sequence Diagram 52
4.7 Activity Diagram 53
4.8 Collaboration Diagram 54
4.9 State Transition Diagram 55
Chapter No 5: Implementation 57
5.1 Component Diagram 57
5.2 Deployment Diagram 58
5.3 Database Architecture (1- Tier, 2-Tier, 3- Tier Architecture) 59
1. Presentation Layer (Tier 1): 59
2. Application Layer (Tier 2): 59
1. Data Layer (Tier 3): 60
Chapter No 6: Testing (Software Quality Attributes) 61
6.1 Test Case Specification 61
6.2 Black Box Test Cases: 61
6.2.1 BVA or Boundary Value Analysis 61
6.2.2 Equivalence Class Partitioning 61
6.2.3 State Transition Testing 61
6.2.4 Decision Table Testing 61
6.2.5 Graph Base Testing 62
6.3 White Box Testing: 62
6.3.1 Statement Coverage 62
6.3.2 Branch Coverage 62
6.3.3 Path Coverage 62
Test Cases: 63
Chapter No 7: Tools and Technologies 71
7.1 Programming Languages 71
7.2 Operating Environment 71
Appendix A: User documentation 72
Appendix B: Source Code 73

List of Tables

Test Case – User Registration 63

Test Case – User Login 65

Test Case – Downloads 67

Test Case – Viral Finder 69

List of Figures

Loading page 29

User registration 29

User Dashboard 30

Viral finder 30

My subscription 31

Download 31

Use Case Diagram 45

Architecture Diagram 47

ERD with Data Dictionary 49

Data Flow Diagram (Level 0 and Level 1) 50

Class Diagram 51

Object Diagram 52

Sequence Diagram 53

Activity Diagram 54

Collaboration Diagram 55

State Transition Diagram 56

Component Diagram **57**

Deployment Diagram 58

Database Architecture (1- Tier, 2-Tier, 3- Tier Architecture) 59

Chapter No 1: Introduction to the Problem

1.1 Introduction

In today's digital era, the internet has become an unprecedented repository of information, encompassing vast amounts of data on a wide array of topics. This abundance of online data has fueled the need for extracting, analyzing, and utilizing relevant information efficiently. Web scraping, a technique of extracting data from websites, has emerged as a powerful tool to harness this wealth of information and transform it into valuable insights.

The objective of this project is to explore the realm of web scraping and its applications in various domains. Web scraping involves automating the process of extracting data from websites, allowing us to gather large volumes of information rapidly and systematically. This project delves into the principles, methodologies, and tools used for web scraping, as well as the ethical considerations and challenges associated with this practice.

Significance:

The significance of web scraping lies in its ability to democratize access to data and empower businesses, researchers, and individuals with valuable information. By automating the data extraction process, web scraping streamlines data collection, saving time and effort. The knowledge gained from this project will be invaluable for anyone seeking to harness the power of web scraping for data-driven decision-making, research, and analysis.

With a thorough understanding of web scraping techniques, ethical awareness, and practical implementation examples, this project seeks to equip readers with the tools they need to embark on their own web scraping endeavors and contribute to the ever-evolving landscape of data analytics and knowledge discovery.

Let the journey into the world of web scraping begin!

1.2 Background

In the age of information, the internet has evolved into an enormous repository of data and knowledge, encompassing virtually every aspect of human life. Websites, social media platforms, e-commerce portals, news outlets, and more generate an overwhelming amount of information on a daily basis. Extracting, organizing, and analyzing this vast trove of data can provide valuable insights and drive decision-making across various domains, including business, academia, research, and social sciences.

Web scraping, also known as web harvesting or web data extraction, has emerged as a crucial technique to gather and process this wealth of data from websites. It involves automating the retrieval of data from web pages, transforming unstructured web content into structured data that can be stored, analyzed, and utilized effectively.

The roots of web scraping can be traced back to the early days of the internet, when simple programs were developed to extract data from HTML pages. Initially used for basic data retrieval, web scraping has evolved significantly over the years, keeping pace with advancements in web technologies and programming languages.

The proliferation of web scraping tools and frameworks has opened up a world of possibilities for businesses and researchers alike. From price monitoring for e-commerce websites to sentiment analysis of social media posts, from gathering financial data for market research to aggregating news articles for analysis, web scraping has found applications in almost every sector.

The Advantages of Web Scraping:

1. Data Access and Aggregation: Web scraping allows users to access data from multiple sources and consolidate it into a single database. This aggregated data can be further analyzed to identify patterns, trends, and correlations.

2. Real-time Updates: With web scraping, users can fetch real-time data, ensuring that the information they have is up-to-date and accurate.

3. Automation and Efficiency: Web scraping automates the data extraction process, saving significant time and effort compared to manual data collection methods.

4. Business Intelligence: Web scraping enables businesses to monitor competitors, track market trends, and gather valuable business intelligence to make informed decisions.

5. Academic and Research Insights: Researchers can leverage web scraping to collect data for academic studies, social science research, sentiment analysis, and more.

Challenges and Ethical Considerations:

While web scraping offers immense benefits, it also presents certain challenges and ethical considerations. Websites may implement measures to prevent web scraping, leading to technical hurdles for data extraction. Moreover, web scraping can raise ethical concerns related to data privacy, copyright, and the potential misuse of extracted data.

Navigating these challenges and ensuring responsible and legal web scraping practices are essential aspects that any web scraping project must address.

Project Aim:

The aim of this project is to explore the world of web scraping comprehensively, providing insights into the techniques, tools, and methodologies involved. By delving into practical implementations, ethical considerations, and real-world applications, this project aims to equip readers with the knowledge and skills to harness web scraping for data-driven decision-making and research endeavors. Ultimately, the project seeks to empower users to unlock the vast potential of web scraping and leverage it responsibly to uncover valuable information in the digital age.

1.3 Purpose

The purpose of this project on web scraping is multi-faceted and encompasses several key objectives:

1. Exploring Web Scraping Techniques: The project aims to provide a comprehensive understanding of various web scraping techniques, such as parsing HTML, using XPath, navigating websites, and

handling dynamic content. By exploring these techniques, readers will gain insights into how data can be extracted efficiently from different types of web pages.

2. Hands-on Learning: Through practical implementation examples and code samples, the project aims to offer a hands-on learning experience. Readers will be able to follow step-by-step instructions to implement web scraping in real-world scenarios, thereby gaining valuable skills that they can apply to their own projects.

3. Understanding Data Preprocessing: Web pages often contain unstructured or messy data. The project focuses on data preprocessing techniques that help clean and organize the extracted data, making it suitable for analysis and further processing.

4. Ethical Considerations: Web scraping can raise ethical concerns related to data privacy, copyright, and terms of service violations. The project addresses these ethical considerations, guiding readers on how to conduct web scraping responsibly and in compliance with legal and ethical boundaries.

5. Showcasing Applications: The project aims to showcase the practical applications of web scraping across various domains. By demonstrating real-world use cases, such as market research, sentiment analysis, content aggregation, and competitive intelligence, readers will understand the broad range of possibilities web scraping offers.

6. Empowering Data-driven Decision-making: One of the primary purposes of web scraping is to provide valuable data for informed decision-making. By equipping readers with web scraping knowledge, the project enables them to leverage data-driven insights to support business strategies, research endeavors, and other data-intensive projects.

7. Encouraging Innovation: Web scraping opens up opportunities for innovation and creativity. By providing a foundation in web scraping techniques, the project encourages readers to explore novel applications and contribute to the advancement of data analytics and knowledge discovery.

8. Democratizing Data Access: Web scraping democratizes access to information that may otherwise be locked behind proprietary systems or paywalls. The project seeks to empower readers with the ability to access and utilize publicly available data effectively, regardless of their background or resources.

9. Contributing to Academic Community: As web scraping plays a significant role in academic research, this project aims to contribute to the academic community by providing valuable insights into the methodologies and best practices of web scraping.

10. Fostering Technical Skills: By engaging in web scraping projects and understanding its underlying technical aspects, readers can enhance their programming skills, data manipulation capabilities, and data analysis proficiency.

Overall, the purpose of this project is to offer a comprehensive and practical resource that equips readers with the knowledge, skills, and ethical awareness needed to harness the potential of web scraping for various purposes while adhering to responsible data practices.

1.4 Scope

The scope of this project on web scraping is ambitious and covers a wide range of topics related to web data extraction and analysis. It aims to provide a holistic understanding of web scraping techniques, ethical considerations, and practical applications. Below are the key aspects that fall within the scope of this project:

1. Web Scraping Techniques: The project covers various web scraping techniques, including parsing HTML, using XPath expressions, navigating websites, handling AJAX-based content, and interacting with forms. It delves into the implementation details of these techniques, enabling readers to grasp the intricacies of data extraction from different types of web pages.

2. Data Preprocessing and Cleaning: Efficient data preprocessing is crucial for data quality and analysis. The project addresses data cleaning challenges specific to web scraping, such as dealing with missing values, removing duplicate entries, and handling inconsistent data formats.

3. Web Scraping Libraries and Frameworks: It explores popular web scraping libraries and frameworks like BeautifulSoup, Scrapy, and Selenium. Readers will gain practical knowledge of using these tools to streamline the web scraping process.

4. Legal and Ethical Considerations: Web scraping can raise ethical and legal issues, including data privacy, copyright violations, and terms of service compliance. The project emphasizes the importance of responsible web scraping and educates readers on best practices to avoid potential pitfalls.

5. Real-world Application Scenarios: The project showcases diverse real-world application scenarios for web scraping. Examples include gathering e-commerce product data for price monitoring, extracting social media content for sentiment analysis, aggregating news articles for content curation, and collecting financial data for market research.

6. Data Analysis and Visualization: While the primary focus is on web scraping, the project briefly touches upon data analysis and visualization techniques to demonstrate how the extracted data can be further analyzed and presented in a meaningful way.

7. Hands-on Implementation: Practical implementation examples are included throughout the project, allowing readers to follow along and gain hands-on experience. These examples cover a variety of web scraping tasks, making the project accessible to both beginners and those with some programming experience.

8. Challenges and Workarounds: The project addresses common challenges faced during web scraping, such as anti-scraping measures implemented by websites. It offers potential workarounds and strategies to overcome these challenges.

9. Educational Resource: The project aims to serve as an educational resource for individuals interested in learning about web scraping. It provides detailed explanations, code samples, and tutorials to support the learning process.

10. Limitations and Future Scope: The project acknowledges the limitations of web scraping, such as the reliance on website structures and potential changes to website layouts. It also discusses potential areas for future exploration and expansion within the realm of web scraping.

While the project covers a comprehensive range of topics related to web scraping, it is essential to note that the field of web scraping is continuously evolving. Therefore, the scope of this project should be

seen as a foundation upon which readers can build and explore further as they delve into the dynamic world of web data extraction and analysis.

1.5 Objective

The objectives of the project on web scraping are as follows:

1. Explore Web Scraping Techniques: The primary objective of this project is to provide a comprehensive exploration of various web scraping techniques, including parsing HTML, using XPath expressions, interacting with forms, and handling AJAX-based content. By understanding these techniques, readers will be equipped with the knowledge to extract data from diverse web page structures.

2. Hands-on Implementation: The project aims to offer practical, hands-on experience in web scraping. Through step-by-step implementation examples and code samples, readers will gain the skills to perform web scraping tasks effectively and independently.

3. Understand Data Preprocessing: Effective data preprocessing is essential for clean and accurate analysis. The project focuses on teaching readers how to preprocess and clean the extracted data, ensuring its reliability and suitability for various analytical tasks.

4. Examine Web Scraping Libraries and Frameworks: The project will explore popular web scraping libraries and frameworks, such as BeautifulSoup, Scrapy, and Selenium. Readers will understand the functionalities and capabilities of these tools, empowering them to choose the most suitable one for their specific projects.

5. Address Ethical Considerations: Web scraping can raise ethical concerns related to data privacy and copyright. The project aims to address these ethical considerations and educate readers on responsible web scraping practices, ensuring compliance with legal and ethical guidelines.

6. Showcase Real-world Applications: The project will showcase practical application scenarios of web scraping across different domains, including e-commerce, social media analysis, news aggregation,

and market research. By presenting real-world use cases, readers will see the diverse applications of web scraping.

7. Empower Data-driven Decision-making: Web scraping enables data-driven decision-making by providing valuable insights and information. The project's objective is to equip readers with the skills to use web scraping for data acquisition, analysis, and informed decision-making.

8. Explore Data Analysis and Visualization: While the primary focus is on web scraping, the project will briefly introduce data analysis and visualization techniques. Readers will gain an understanding of how to analyze and present the extracted data effectively.

9. Address Challenges and Workarounds: Web scraping may encounter challenges such as anti-scraping measures employed by websites. The project will address common obstacles and provide strategies to overcome them, ensuring a smooth web scraping process.

10. Serve as an Educational Resource: This project aims to serve as a comprehensive educational resource for individuals interested in web scraping. By providing detailed explanations, examples, and tutorials, it strives to make web scraping accessible to learners of various skill levels.

11. Encourage Further Exploration: The project acknowledges that web scraping is an evolving field. It seeks to encourage readers to explore advanced topics and continue their learning journey beyond the project's scope.

By achieving these objectives, the project aspires to equip readers with the knowledge, skills, and ethical awareness required to perform web scraping proficiently and responsibly, thus contributing to the broader realm of data analytics and knowledge discovery.

1.6 Intended Audience and Reading Suggestions

Intended Audience:

The project on web scraping is designed to cater to a diverse audience with varying levels of technical expertise and interests. The following groups of individuals will find this project particularly valuable:

1. Students and Researchers: University students and researchers in fields such as computer science, data science, business, social sciences, and academia will benefit from this project. It serves as a comprehensive resource for understanding web scraping techniques and applying them in research endeavors.

2. Data Enthusiasts and Analysts: Aspiring data analysts and enthusiasts seeking to enhance their data manipulation and analysis skills will find this project useful. Web scraping provides a means to access valuable data for analysis and decision-making.

3. Developers and Programmers: Software developers and programmers interested in data extraction and automation will gain insights into web scraping techniques, libraries, and frameworks. The project caters to both beginners and those with programming experience.

4. Business Professionals: Business professionals involved in market research, competitive intelligence, and data-driven decision-making will find this project relevant. Web scraping offers valuable insights into competitor analysis, market trends, and customer behavior.

5. Entrepreneurs and Startups: Entrepreneurs and startup founders seeking data to validate business ideas, understand market demands, and identify opportunities can leverage web scraping techniques presented in this project.

6. Web Content Aggregators: Individuals or organizations involved in content curation and aggregation, such as news portals or bloggers, will benefit from learning how to scrape and consolidate information from different sources.

7. Data Privacy and Compliance Officers: Professionals concerned with data privacy and ethical considerations will find the project's focus on responsible web scraping practices and compliance valuable.

Reading Suggestions:

1. Beginners: For those new to web scraping or programming, it is recommended to start from the beginning of the project and progress sequentially. Focus on understanding the fundamental concepts, web scraping techniques, and hands-on implementation examples.

2. Intermediate Learners: Individuals with some programming knowledge can explore specific chapters or sections of interest. Focus on practical implementation examples and explore the web scraping libraries and frameworks showcased in the project.

3. Advanced Users: Experienced web scrapers may choose to skim through initial chapters to review fundamentals and then delve into more complex topics like handling dynamic content, overcoming anti-scraping measures, and advanced data preprocessing techniques.

4. Legal and Ethical Considerations: Readers concerned with legal and ethical aspects of web scraping should pay close attention to the chapters dedicated to ethical considerations and best practices to ensure responsible web scraping.

5. Application Scenarios: For readers interested in practical use cases, the chapters on real-world applications demonstrate how web scraping can be applied in diverse domains, providing inspiration for their own projects.

6. Further Exploration: Those looking to expand their knowledge beyond this project can explore additional web scraping resources, advanced web scraping techniques, or delve into data analysis and visualization in more depth.

Regardless of the reader's background or level of expertise, the project encourages a hands-on approach. Readers are encouraged to follow along with the implementation examples and experiment with web scraping on their own to gain a deeper understanding and proficiency in this valuable skill.

1.7 Document Conventions

Document conventions are guidelines and standards used in the project to ensure consistency and clarity in presenting information. These conventions make it easier for readers to navigate and understand the content. Here are the updated document conventions used in this project on web scraping:

1. Headings and Subheadings: Headings are formatted in Times New Roman font, size 16, and are presented in bold. Subheadings are also formatted in Times New Roman font, size 14, and may be further distinguished by different font styles.

2. Text: The main text content is presented in Times New Roman font, size 12, which ensures readability and clarity throughout the project.

3. Code Formatting: Code snippets and examples are presented in a monospaced font, such as Courier New, to differentiate them from regular text. Code sections may be highlighted or enclosed in code blocks to enhance readability.

4. Variables and User Input: When referring to variables, user input, or placeholders in code examples, they are presented in a monospaced font and may be highlighted for better visibility.

5. Bullet Points and Numbering: Lists are used to present items or concepts that need to be emphasized or presented in a sequential manner. Bullet points are used for unordered lists, while numbering is used for ordered lists.

6. Emphasis and Italics: Text that requires emphasis is presented in italics. This may include important concepts, key terms, or warnings.

8. Callout Boxes: Callout boxes or side notes may be used to draw attention to important tips, reminders, or additional information that complements the main text.

9. Figures and Illustrations: Figures, charts, diagrams, and other visual elements are used to enhance understanding. They are typically labeled with captions for clarity.

10. Annotations and Comments: Explanatory annotations and comments may be used within code examples to provide additional context or clarify specific steps.

11. Terminology and Glossary: Key terminology and technical jargon may be presented in bold or italics when first introduced. A glossary section at the end of the project may also be provided to define essential terms.

13. Navigation and Table of Contents: A clear table of contents is provided at the beginning of the project to help readers quickly navigate to specific sections of interest.

Chapter No 2: Software Requirement Specification

2.1 Overall Description

2.1.1 Product Perspectives

The web scraping project operates within the context of the broader field of data extraction and analysis. It serves as a valuable tool and resource for individuals and organizations seeking to harness the power of web scraping to acquire, process, and analyze data from various websites. The project is designed to be a standalone educational resource that provides comprehensive insights into web scraping techniques, ethical considerations, and practical implementations.

Integration with Existing Tools and Projects:

The project can be integrated into existing data analysis pipelines and projects that require web data extraction. As web scraping is a foundational skill in data-driven decision-making, the knowledge gained from this project can complement other data analysis and machine learning initiatives.

Supporting Data-driven Decision-making:

The project aligns with the goal of empowering data-driven decision-making. By enabling users to extract and analyze relevant data from the web, it supports users in making informed choices, whether in business strategy, academic research, market analysis, or content curation.

Learning and Skill Building:

From a learning perspective, the project aims to equip its audience with a strong understanding of web scraping principles, methodologies, and best practices. It caters to readers with varying levels of technical expertise, offering a progressive learning experience, from foundational concepts to practical implementation.

Stand-alone Educational Resource:

As a stand-alone project, it does not require dependencies on external systems or tools to function. The project provides all the necessary code examples, explanations, and resources for users to learn and apply web scraping techniques independently.

Focus on Ethical Web Scraping:

The project underscores the significance of ethical web scraping practices, emphasizing compliance with legal and ethical considerations. By promoting responsible data extraction, it aims to instill in its users the importance of respecting data privacy and adhering to website terms of service.

Potential Extensions and Applications:

While the project covers a broad range of web scraping topics, it also opens avenues for further exploration and extension. Users may build upon the knowledge gained to explore more advanced web scraping techniques, automate data collection for specific domains, or integrate web scraping into larger data analytics projects.

Continuous Relevance and Adaptability:

The project's content is designed to be relevant and adaptable to the ever-changing landscape of web technologies. While the underlying principles of web scraping remain constant, the project acknowledges that websites and technologies continuously evolve. As such, readers are encouraged to continuously update their knowledge and adapt their approaches as needed.

In summary, the web scraping project operates as an educational resource with a strong focus on enabling responsible data extraction and analysis. It empowers users with valuable skills to gather data from the web and supports data-driven decision-making across diverse domains. As a stand-alone project, it caters to users with various levels of expertise, fostering a culture of continuous learning and exploration within the realm of web scraping and data analytics.

2.1.2 Product Features

The web scraping project offers a comprehensive set of features that cater to learners with varying levels of technical expertise and interests. These features enable users to gain a solid understanding of web scraping techniques, apply them in real-world scenarios, and make informed decisions based on the extracted data. The key product features of this project are as follows:

1. Web Scraping Techniques: The project provides an in-depth exploration of various web scraping techniques, such as parsing HTML, using XPath expressions, interacting with forms, and handling AJAX-based content. Users will learn how to extract data from diverse web page structures.

2. Practical Implementation Examples: The project includes numerous hands-on implementation examples and code samples. This feature allows users to follow step-by-step instructions to perform web scraping tasks in a real-world context, enhancing their practical skills.

3. Data Preprocessing and Cleaning: Effective data preprocessing is crucial for data quality and analysis. The project focuses on teaching users how to preprocess and clean the extracted data, ensuring its reliability and suitability for various analytical tasks.

4. Web Scraping Libraries and Frameworks: The project explores popular web scraping libraries and frameworks, such as BeautifulSoup, Scrapy, and Selenium. Users will understand the functionalities and capabilities of these tools, empowering them to choose the most suitable one for their specific projects.

5. Ethical Considerations and Best Practices: The project addresses ethical concerns related to web scraping, emphasizing responsible data extraction and compliance with legal boundaries. It provides best practices to ensure users conduct web scraping activities ethically.

6. Real-world Application Scenarios: The project showcases practical application scenarios of web scraping across different domains. Examples include gathering e-commerce product data for price monitoring, sentiment analysis of social media content, aggregating news articles for content curation, and collecting financial data for market research.

7. Data Analysis and Visualization: While the primary focus is on web scraping, the project briefly introduces data analysis and visualization techniques. Users will gain an understanding of how to analyze and present the extracted data effectively.

8. Challenges and Workarounds: The project addresses common challenges faced during web scraping, such as anti-scraping measures implemented by websites. It provides strategies and workarounds to overcome these challenges and ensure a successful web scraping process.

9. Educational Resource and Learning Progression: The project is designed to be an educational resource that caters to both beginners and intermediate learners. It follows a logical learning progression, allowing users to start from foundational concepts and gradually advance to more complex topics.

10. Navigation and Table of Contents: A clear table of contents is provided at the beginning of the project, enabling users to quickly navigate to specific sections of interest. This feature enhances the project's user-friendliness and accessibility.

11. Supporting Documentation and Glossary: The project may include supporting documentation, such as reference guides and glossaries, to explain key terminology and technical concepts in more detail. This feature enhances the project's comprehensibility for users with varying levels of technical knowledge.

12. Continuous Relevance and Updates: The project acknowledges the dynamic nature of web scraping and the web itself. It encourages users to continuously update their knowledge and adapt their approaches as web technologies evolve.

These product features collectively contribute to a comprehensive and user-friendly learning experience, empowering users to harness the potential of web scraping for data acquisition, analysis, and informed decision-making.

2.1.3 Design and Implementation Constraints

During the development of the web scraping project, several design and implementation constraints need to be considered to ensure the project's effectiveness, efficiency, and ethical compliance. The following are some of the key constraints that should be addressed:

1. Website Structure and Changes: Web scraping heavily relies on the structure of the target websites. Changes in the website's HTML layout or underlying technologies can impact the effectiveness of the scraping process. To mitigate this constraint, the project should provide guidance on handling dynamic content and adapting scraping strategies to evolving website structures.

2. Anti-Scraping Measures: Websites may implement anti-scraping measures, such as rate limiting, CAPTCHAs, or IP blocking, to deter automated data extraction. The project should address ways to circumvent such measures ethically or provide alternative approaches for data collection.

3. Legal and Ethical Considerations: Web scraping can raise legal and ethical concerns, especially concerning data privacy, copyright, and terms of service violations. The project must emphasize responsible web scraping practices and educate users on adhering to legal and ethical boundaries.

4. Resource Consumption: Web scraping can consume substantial server resources and bandwidth, affecting website performance and potentially leading to service disruptions. The project should highlight the importance of being considerate and respectful of the target websites' resources.

5. Data Volume and Storage: Extracting data from numerous web pages may result in large data volumes. Users must be aware of the storage requirements and potential costs associated with managing and storing scraped data.

6. Target Website Policies: Some websites explicitly prohibit or limit web scraping in their terms of service. Users should be aware of these policies and exercise caution when scraping data from such websites.

7. Authentication and Access Restrictions: Some websites require user authentication or have access restrictions for certain pages. The project should cover techniques to handle login forms or deal with password-protected content.

8. Data Quality and Accuracy: Web scraping is susceptible to errors, leading to incomplete or inaccurate data. The project should emphasize data validation, error handling, and the importance of verifying scraped data for accuracy.

9. Performance and Scalability: Large-scale web scraping projects may encounter performance and scalability challenges. Users should consider the efficiency of their scraping scripts and potentially implement distributed scraping strategies for scalability.

10. Dependency and Library Compatibility: The project's implementation should consider compatibility issues with different web scraping libraries and dependencies, ensuring the code examples work across various environments.

11. Learning Curve for Beginners: For users with limited programming experience, web scraping may present a learning curve. The project should provide clear explanations and step-by-step tutorials to accommodate learners at different skill levels.

Addressing these design and implementation constraints will contribute to a well-rounded and responsible web scraping project, enabling users to navigate potential challenges while adhering to ethical standards and maximizing the effectiveness of their data extraction efforts.

2.1.4 Assumption and Dependencies

Assumption:

1. Internet Connectivity: The project assumes that users have access to the internet to retrieve data from websites. Web scraping relies on fetching data online, and without internet connectivity, the process cannot be performed.

2. Web Page Structure Consistency: The project assumes that the target websites' structures remain relatively consistent during the scraping process. Significant changes in the website's layout or structure may require adjustments to the scraping code.

3. Publicly Accessible Data: Web scraping focuses on extracting publicly accessible data from websites. It assumes that the data to be scraped is not protected by login credentials or access restrictions.

4. Respectful Use of Resources: The project assumes that users will use web scraping responsibly and respect the resources of target websites. Users should avoid excessive scraping that may lead to server overload or service disruptions.

5. Legal Compliance: It is assumed that users will comply with all relevant laws, regulations, and website terms of service while performing web scraping. Users should not engage in activities that violate copyright or data privacy laws.

6. No Malicious Intent: The project assumes that web scraping will be conducted for legitimate purposes and not for any malicious or harmful intent, such as data theft, hacking, or spamming.

Dependencies:

1. Programming Language and Libraries: The project depends on the choice of programming language and web scraping libraries, such as Python with BeautifulSoup, Scrapy, or Selenium. Users need to have the necessary programming environment and dependencies installed.

2. Website Accessibility: Web scraping is dependent on the availability and accessibility of the target websites. If a website is down or inaccessible, the scraping process cannot be performed.

3. Website Structure and Content: The effectiveness of web scraping relies on the structure and organization of the target websites. Changes to the website's structure or content may affect the scraping process.

4. Internet Connection: As web scraping involves fetching data from websites, a stable internet connection is essential for successful data retrieval.

5. Server Response Time: The speed and efficiency of web scraping can be influenced by the target website's server response time. Slow server responses may affect the scraping process.

6. Data Storage and Processing Capacity: The size and complexity of the scraped data may require adequate storage and processing capacity to handle and analyze the collected data effectively.

7. Website Terms of Service: Users must be aware of and adhere to the terms of service of the target websites they are scraping to avoid any legal or ethical issues.

8. Ethical Considerations: Users should consider ethical factors when conducting web scraping, such as avoiding excessive requests that may burden the target website's resources or collecting sensitive or private information.

9. Continuous Monitoring and Maintenance: Depending on the scope and frequency of web scraping, users may need to continuously monitor the target websites for changes and perform maintenance on the scraping scripts when necessary.

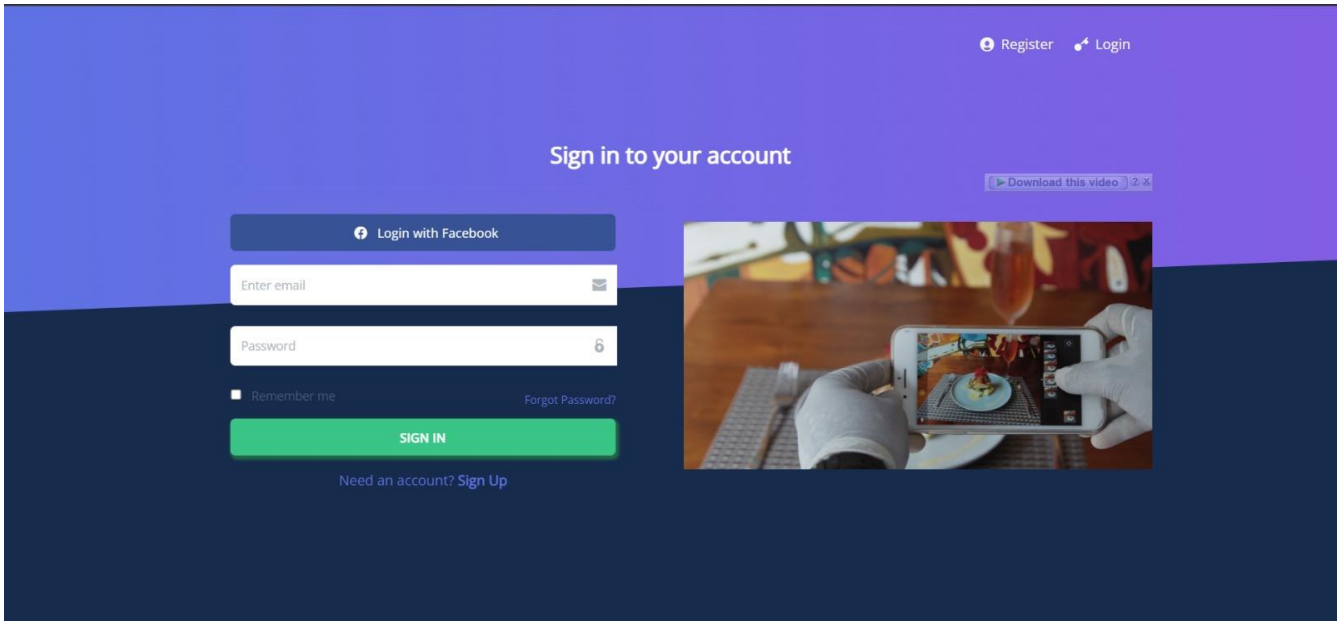
Addressing these assumptions and dependencies will ensure a smooth and responsible web scraping process, enabling users to effectively extract data from websites while being mindful of legal and ethical considerations.

2.2 System Features

Some of the System features are:

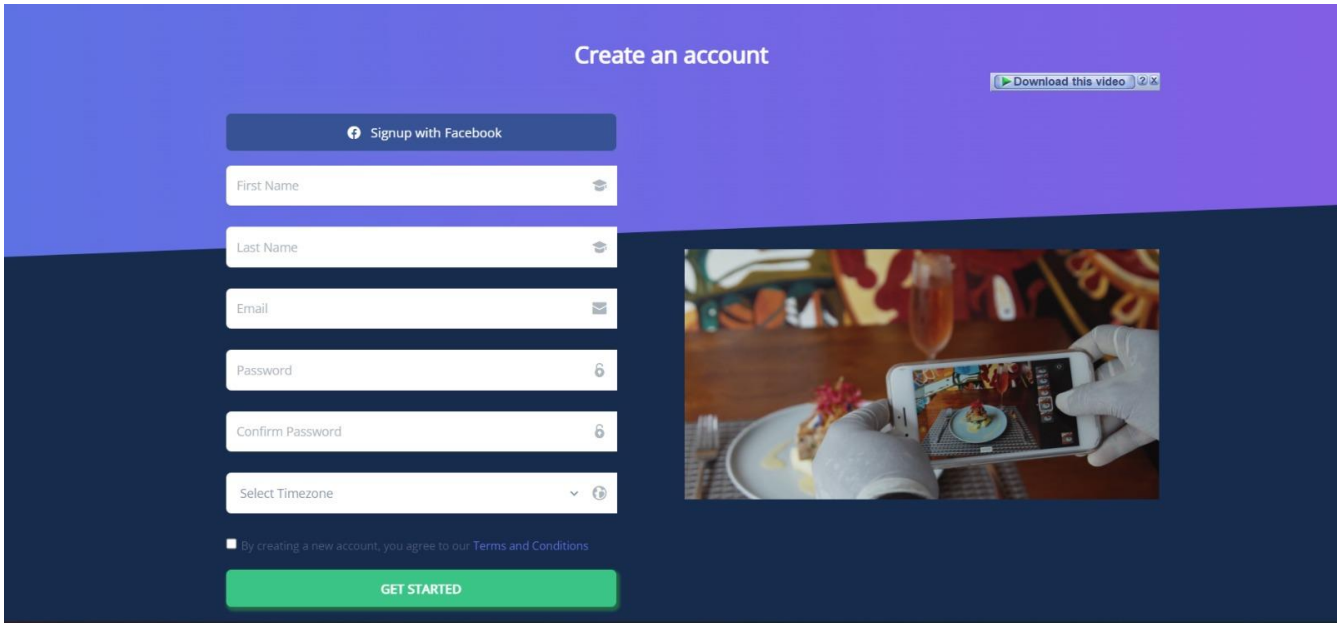
Landing page

This is the landing page where user can register and login himself



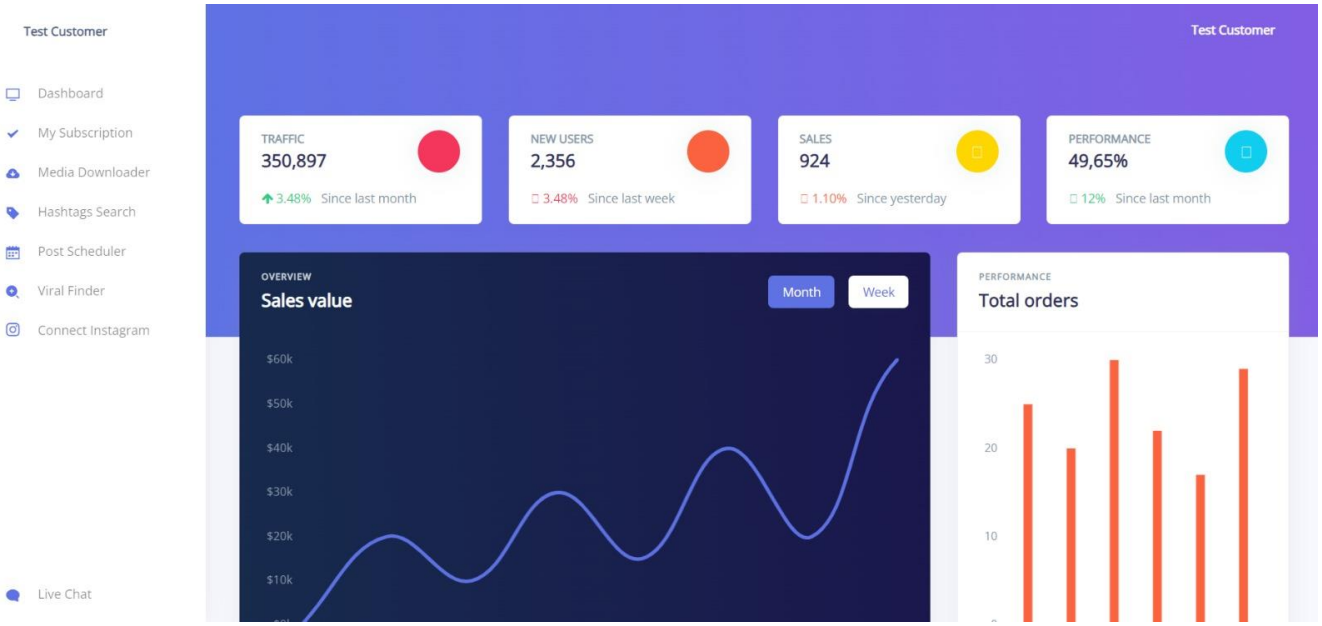
Register page:

Here user can register himself by giving his information like his name email and create password



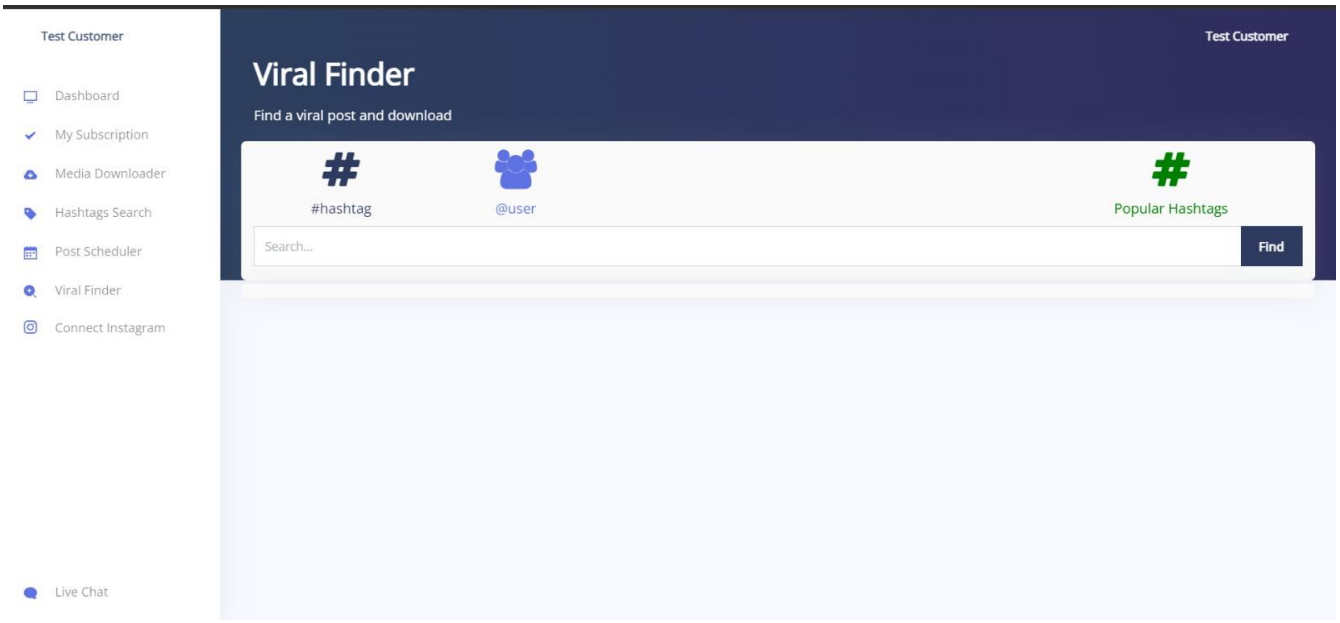
Dashboard;

In dashboard user have all the available option like my subscripton, viral finder etc and can see all the information from here.



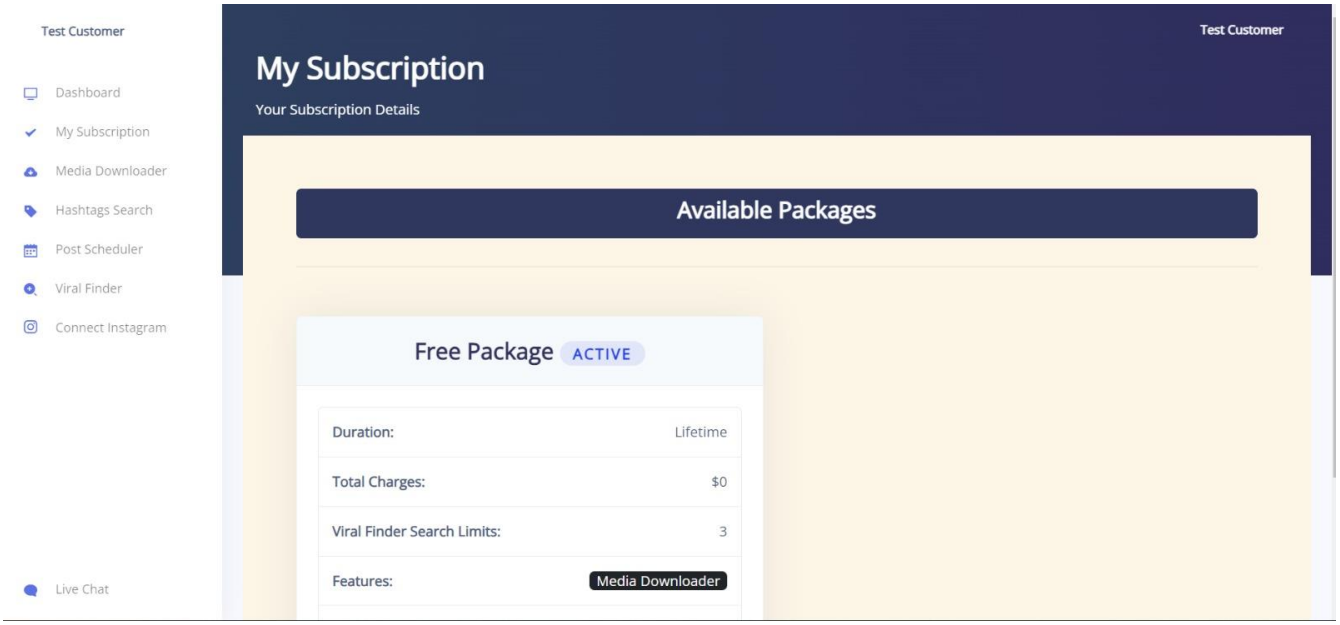
Viral finder:

In viral finder user can find all the viral stuff of the internet



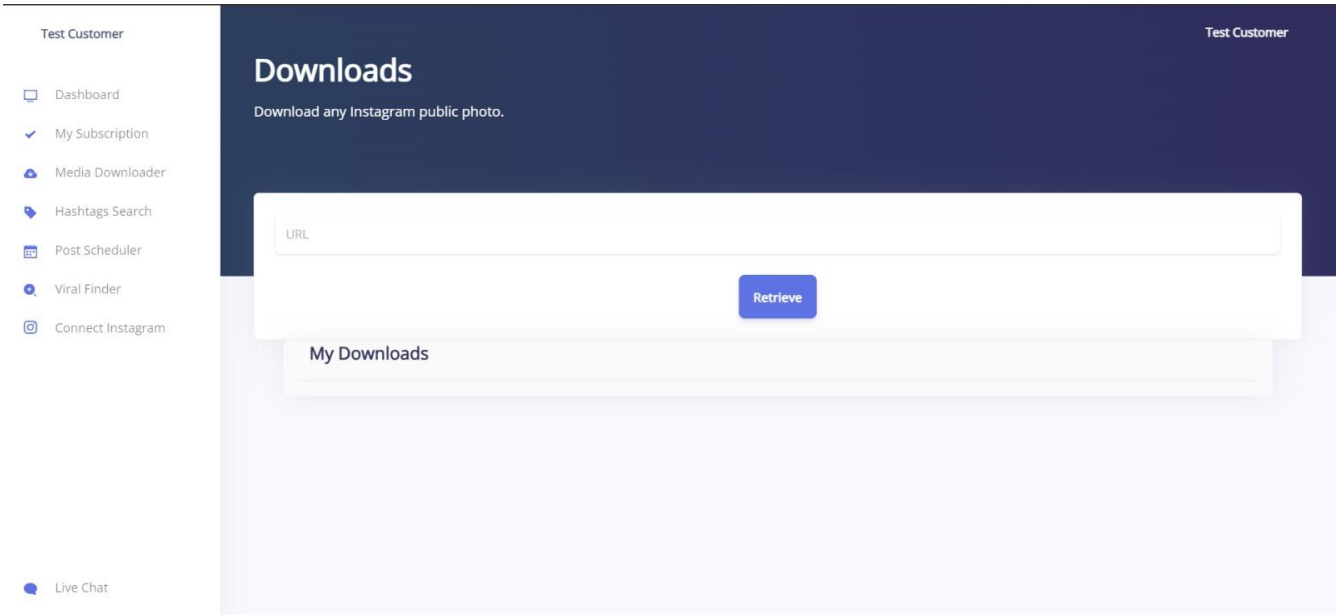
My subscription:

In this tab user can see all the subscription that he did and acces them.



Downloads:

Here user can watch all the downloads he did.



2.3 External Interface Requirements

2.3.1 User Interfaces

User Interface (UI): An Overview

A User Interface (UI) refers to the point of interaction between a user and a digital system or software application. It encompasses the visual elements, design, and interactive components that facilitate user interaction and communication with the system. The primary goal of a well-designed user interface is to create a seamless and intuitive user experience (UX) by presenting information and functionality in a clear, efficient, and aesthetically pleasing manner.

Key Components of a User Interface:

1. Visual Design: The visual design of a UI involves the layout, color scheme, typography, icons, and other graphical elements. An appealing and consistent visual design enhances the user's perception of the software and helps in establishing a brand identity.

2. Layout: The arrangement of elements on the screen is crucial for easy navigation and comprehension. A well-organized layout ensures that important features and information are readily accessible, reducing the user's cognitive load.

3. Navigation: Effective navigation is vital for guiding users through the application or website. Intuitive navigation menus, breadcrumbs, and clear calls-to-action (CTAs) help users find what they need quickly and effortlessly.

4. Input Controls: Input controls allow users to interact with the system, such as buttons, text fields, checkboxes, radio buttons, sliders, and drop-down lists. They should be self-explanatory and responsive to user actions.

5. Feedback and Alerts: The UI should provide real-time feedback to users about the success of their actions or any errors that may occur. Informative alerts and notifications keep users informed and help prevent confusion.

6. Consistency: Consistency in design elements and interactions across the entire application ensures a coherent user experience. Users should be able to predict how different parts of the system will behave based on their prior interactions.

7. Accessibility: A good UI is inclusive and considers the needs of all users, including those with disabilities. Providing accessible features, such as screen readers compatibility and keyboard navigation, makes the application more usable to a broader audience.

8. Responsive Design: With the prevalence of various devices like smartphones, tablets, and desktops, a responsive UI adapts to different screen sizes and orientations, providing an optimal experience on any device.

9. Interactivity: A UI should be interactive, engaging users through animations, transitions, and other dynamic elements. However, excessive animations can be distracting and slow down the user experience.

10. Error Handling: The UI should handle errors gracefully, providing clear error messages and suggestions to rectify them. Avoid cryptic error messages that confuse users further.

User Interface Design Process:

1. User Research: Understand the target audience and their needs through user interviews, surveys, and usability testing.

2. Wireframing: Create low-fidelity wireframes to outline the layout and basic elements of the UI.

3. Prototyping: Develop interactive prototypes to test and refine the user flow and interactions.

4. Visual Design: Apply the visual elements, branding, and aesthetics to the wireframes to create high-fidelity mockups.

5. Usability Testing: Conduct usability testing with real users to gather feedback and identify areas for improvement.

6. Iterative Design: Continuously refine the UI based on user feedback and data analysis.

In conclusion, a well-designed user interface is crucial for creating a positive user experience, increasing user engagement, and achieving the goals of a software application or website. It involves careful consideration of visual design, layout, navigation, interactivity, and accessibility to ensure that users can interact with the system effortlessly and enjoy a seamless digital experience.

2.3.2 Hardware Interfaces

All of the evaluation processes were run on the following hardware:
• Intel Core i7 2,2 GHz processor
• 16 GB2400MHzDDR4RAM
•Intel UHD Graphics 630 1536 MB Graphics card

2.3.3 Software Interfaces

Web scraping project involves developing a user-friendly and efficient program to interact with the web scraping functionality.

1. User Authentication: Web scraping project requires access to authenticated or restricted content, implement a user authentication mechanism, such as username/password or API keys.

2. Input Parameters: Design the interface to accept user input, such as the target website's URL, search queries, data filters, or any other parameters required for web scraping.

3. Data Output Options: Provide users with options to specify the desired output format (e.g., CSV, JSON, Excel) or data storage location (e.g., local file, database) for the scraped data.

4. Data Preview (Optional): Include a preview feature that allows users to see a sample of the data before initiating the web scraping process.

5. Start/Stop Control: Implement buttons or commands to start and stop the web scraping process. **We e**nsure that users can easily interrupt the process if needed.

6. Progress Indicators: Include progress indicators (e.g., progress bars or status updates) to keep users informed about the current status of the web scraping operation.

7. Error Handling and Reporting: Design the interface to handle errors gracefully. Display clear error messages and provide suggestions for troubleshooting when issues occur during the scraping process.

8. User-Friendly Messages: Use descriptive labels, tooltips, and hints to guide users on how to use the interface effectively.

9. Save and Load Configurations (Optional): Allow users to save their web scraping configurations as presets for future use and provide the option to load saved configurations when needed.

10. Logging and Reporting: Implement a logging feature to record the details of each web scraping session, such as the start time, end time, number of scraped pages, and any errors encountered.

2.3.4 Communications Interfaces

The communication interface would involve the interaction between the web scraping script running on the client-side and the target website's server. The goal is to enable the script to request web pages, retrieve data, and handle any necessary authentication while adhering to best practices and respecting the website's policies.

1. HTTP(S) Requests:

- Use HTTPS for secure communication to encrypt data transmitted between the client (web scraping script) and the server (target website).
- Implement HTTP GET or POST requests to request web pages or data from the target website's server.

2. User-Agent Spoofing (Optional):

- To mimic a regular user's browser behavior, set an appropriate User-Agent header in the HTTP request to avoid detection as a bot.

3. Handling Cookies (Optional):

- If the website relies on cookies for user sessions, ensure the web scraping script can handle and store cookies appropriately to maintain session continuity.

4. User Authentication (Optional):

- If the target website requires user authentication, use appropriate authentication mechanisms such as OAuth, API keys, or username/password.
- Securely transmit the authentication credentials with HTTPS.

5. Rate Limiting:

- Implement rate limiting on the client-side to control the number of requests sent to the server within a specific time frame.
- Prevent overwhelming the server and adhere to the website's usage policies to avoid potential IP blocking.

6. Data Parsing and Extraction:

- On the client-side, parse the received HTML or JSON data from the server using libraries like BeautifulSoup (Python) or other parsing tools.
- Extract the desired information according to the web scraping requirements.

7. Error Handling and Logging:

- Implement error handling mechanisms to gracefully manage connection errors, timeouts, and other exceptions that may occur during web scraping.
- Log errors securely to facilitate troubleshooting and debugging if issues arise.

8. Data Storage and Output:

- Provide options for the user to specify the desired output format (e.g., CSV, JSON, Excel) and the location to save the scraped data.

9. HTTPS Certificate Verification:

- Validate the server's SSL/TLS certificate to ensure the communication is secure and prevent potential man-in-the-middle attacks.

10. Respecting Robots.txt:

- Check the website's robots.txt file to respect its rules and avoid scraping disallowed content.

By incorporating these communication interface features into web scraping project, **we** can ensure secure, efficient, and respectful interaction with the target website while extracting the desired data. Always review and comply with the terms of service and privacy policies of the target website to avoid any legal issues during the web scraping process.

2.4 Other Nonfunctional Requirements

2.4.1 Performance Requirements

Performance requirements for a web scraping project focus on optimizing the efficiency, speed, and resource usage of the data extraction process. The following are key performance requirements to ensure the project operates effectively:

1. Scraping Speed: The project should achieve reasonable scraping speeds to minimize the time required to extract data from web pages. Faster scraping speeds improve productivity and reduce the overall scraping time, especially for large-scale data extraction tasks.

2. Resource Utilization: The project should be designed to use system resources efficiently. It should avoid excessive memory consumption or CPU usage to prevent system slowdowns and ensure smooth execution on various hardware configurations.

3. Concurrency and Parallelism (Optional): For large-scale scraping tasks, the project may implement concurrency and parallelism techniques to process multiple web pages simultaneously. This approach improves data retrieval efficiency and allows the system to take advantage of multi-core processors.

4. Robust Error Handling: Effective error handling mechanisms should be in place to address potential issues that may arise during the scraping process. Graceful error handling ensures the project recovers from errors without significant disruptions.

5. Data Processing Efficiency: The data processing phase, including cleaning, transformation, and validation, should be optimized for efficiency. Reducing processing time enhances the overall project performance and allows for faster data analysis.

6. Optimized Network Requests: The project should minimize the number of unnecessary network requests to avoid overloading the target website and reduce potential IP blocking risks. Caching and smart request management can help achieve this goal.

7. Asynchronous Requests (Optional): Asynchronous programming techniques, if implemented, can enhance scraping efficiency by allowing the system to process multiple requests simultaneously without waiting for each response individually.

8. Response Time Tolerance: The system should be able to handle variations in response times from target websites gracefully. Implementing timeout and retry mechanisms helps manage slow or unresponsive websites.

9. Scalability (Optional): Depending on the scale of the scraping task, the project may need to be scalable to handle a large volume of data. Efficient data storage and retrieval strategies contribute to the project's scalability.

10. Adaptability to Website Changes: The project should be adaptable to website changes, such as alterations in HTML structure or URL patterns. Regular maintenance and updates ensure continued functionality as websites evolve.

11. Avoiding IP Blocking: The project should incorporate mechanisms to prevent IP blocking by target websites. Implementing user-agent rotation and respecting website access frequency guidelines can help avoid detection as a bot.

12. Responsiveness of User Interface: If the project includes a user interface, it should be responsive and provide timely updates, progress reports, and error notifications to keep users informed of the scraping process.

Meeting these performance requirements ensures that the web scraping project operates efficiently, delivers timely results, and utilizes system resources optimally to achieve the desired data extraction objectives.

2.4.2 Safety Requirements

Safety requirements for a web scraping project are essential to ensure the security, privacy, and ethical compliance of the data extraction process. These requirements address potential risks and mitigate adverse impacts on both the project and the target websites. The following are key safety requirements for a responsible web scraping project:

1. Ethical Web Scraping Practices: The project must adhere to ethical web scraping practices and comply with all legal regulations and website terms of service. Users should be educated on the importance of respecting website policies and avoiding activities that may harm the target website or its users.

2. Privacy Protection: The project should not extract or store any sensitive or personally identifiable information (PII) without explicit user consent. Users should be aware of data privacy considerations and avoid scraping content that violates privacy regulations.

3. Crawling Politeness and Frequency Limits: The project should implement crawling politeness mechanisms to avoid overloading target websites with excessive requests. Users should adhere to website-specific frequency limits and respect robots.txt guidelines.

4. IP Blocking Avoidance: Users should employ strategies to prevent IP blocking by target websites, such as rotating user-agents and implementing delay timers between requests. IP blocking can disrupt the scraping process and lead to potential legal issues.

5. Error Handling for Security and Anomaly Detection: The project should include robust error handling mechanisms to detect and respond to potential security-related errors, such as unexpected responses or security measures triggered during scraping.

6. Regular Maintenance and Monitoring: Users should conduct regular maintenance to ensure the project remains up-to-date with website changes and avoids scraping sensitive or restricted content inadvertently.

7. Access Control and Authentication Handling: If the project requires handling login forms or password-protected content, users should ensure that any credentials used are securely managed and not shared with others.

8. User-Agent Rotations: Implementing user-agent rotations helps to mask scraping activities and avoid detection as a bot. Users should use a variety of user-agent headers to emulate human-like behavior.

9. Responsible Data Storage and Handling: The project should store scraped data securely and in compliance with relevant data protection laws. Users should avoid storing sensitive or private information and be cautious about sharing data with third parties.

10. Error and Exception Logging: Logging errors and exceptions during the scraping process helps identify potential issues and take appropriate corrective actions. Users should be aware of any errors or anomalies in the scraping process.

11. Clear User Documentation: The project's documentation should include safety guidelines and best practices, making users aware of potential risks and providing guidance on how to mitigate them.

12. Responsible Use of Scraped Data: Users should employ the scraped data responsibly and ensure it is used for legal and ethical purposes. They should avoid engaging in activities that could harm individuals or organizations based on the extracted information.

Meeting these safety requirements ensures that the web scraping project operates responsibly, respecting website policies, data privacy, and ethical standards. By following these guidelines, users can conduct web scraping activities in a safe and compliant manner, mitigating potential risks and fostering a positive impact on the web scraping community.

2.2.3 Security Requirements

Security requirements for a web scraping project are crucial to protect the project, the data being extracted, and the target websites from potential threats and vulnerabilities. These requirements aim to ensure the confidentiality, integrity, and availability of data and prevent unauthorized access or malicious use. The following are key security requirements for a secure web scraping project:

1. Data Encryption in Transit: The project should use secure communication protocols, such as HTTPS, to encrypt data transmitted between the web scraper and the target websites. This prevents eavesdropping and unauthorized access to sensitive information during data retrieval.

2. Secure Credential Management: If the project requires authentication for accessing certain websites, user credentials should be securely stored and transmitted. Passwords and sensitive information should be properly encrypted and protected from unauthorized access.

3. Input Validation and Sanitization: The project should validate and sanitize all user input and URL parameters to prevent potential injection attacks, such as SQL injection or cross-site scripting (XSS).

4. Authentication and Access Control: If the project includes user authentication or access control mechanisms, strong authentication methods should be used to prevent unauthorized access to sensitive functionalities or data.

5. Data Privacy Compliance: The project should adhere to relevant data privacy regulations and guidelines, ensuring that the data being scraped does not include personally identifiable information (PII) without proper user consent.

6. Protection Against Bot Detection: The project should employ techniques, such as rotating user-agents and implementing delays, to avoid detection as a bot by target websites' anti-scraping measures.

7. Secure Data Storage: Scraped data should be stored securely, with proper access controls, encryption, and backups to protect against data breaches and unauthorized access.

8. Handling of Cookies and Sessions: If the project requires handling cookies and sessions, users should ensure that sensitive data, such as session tokens, are appropriately managed and not exposed.

9. Avoiding Denial-of-Service (DoS) Attacks: The project should implement rate limiting and other measures to prevent unintentional DoS attacks on the target websites due to excessive scraping requests.

10. Security Auditing and Monitoring: Regular security audits and monitoring should be conducted to identify potential vulnerabilities or suspicious activities in the web scraping process.

11. Protection Against Captchas and Bot Detection: Users should be aware of and comply with websites' anti-bot mechanisms, such as CAPTCHAs, to avoid violations and ensure smooth data extraction.

12. Limiting Scope of Scraping: The project should be scoped to extract only the necessary data, reducing the risk of inadvertently scraping sensitive or restricted content.

13. User Notification of Security Issues: Users should be informed of any potential security concerns or vulnerabilities related to the project, enabling them to take appropriate actions promptly.

By adhering to these security requirements, users can ensure that their web scraping project operates securely, safeguarding data, respecting privacy, and minimizing potential risks and impacts on both the project and the target websites.

Chapter No 3: Analysis (use case model)

3.1 Identifying Actors and Use Cases using Textual Analysis

Identifying Actors and Use Cases for web scraping using textual analysis involves analyzing the text to extract potential actors (users or external systems) and the corresponding use cases (functionalities) that the web scraping system should support. Below are some examples of potential actors and use cases:

Example 1: Web Scraping Tool for News Aggregation

Actors:

1. **User**: The end-user who interacts with the web scraping tool to access aggregated news from various websites.

Use Cases:

1. **Perform News Aggregation**: The user can initiate the web scraping tool to perform news aggregation from multiple news websites.
2. **Configure News Sources**: The user can specify the news websites or sources to include in the aggregation process.
3. **Schedule News Updates**: The user can schedule periodic updates to fetch the latest news from the specified sources.
4. **View Aggregated News**: The user can view the aggregated news from different sources in a user-friendly format.

Example 2: E-commerce Price Comparison Application

Actors:

1. **User**: The end-user who uses the e-commerce price comparison application to compare prices of products across various online retailers.

Use Cases:

1. **Search Product**: The user can input a product name or keyword to search for various online retailers selling that product.
2. **Retrieve Product Prices**: The application can scrape the prices of the searched product from different online retailers.
3. **Filter Results**: The user can filter the results based on price range, brand, availability, etc.

4. **Sort Results**: The user can sort the results based on price, popularity, rating, etc.
5. **View Product Details**: The user can view detailed information about a particular product, including its specifications and reviews from various retailers.

Example 3: Data Analytics Platform with Web Scraping Capabilities

Actors:

1. **Data Analyst**: The user who utilizes the data analytics platform to gather and analyze data from different online sources.

Use Cases:

1. **Configure Data Sources**: The data analyst can configure the online sources (e.g., websites, APIs) from which data needs to be scraped.
2. **Perform Data Scraping**: The data analytics platform can perform web scraping to extract data from the specified sources.
3. **Store Scraped Data**: The platform can store the scraped data in a structured format (e.g., databases, data warehouse) for further analysis.
4. **Clean and Transform Data**: The data analyst can clean and transform the scraped data to make it suitable for analysis.
5. **Analyze Data**: The data analyst can perform various analytical tasks on the scraped data, such as creating reports, visualizations, or running machine learning algorithms.

3.2 Forming Use Case Diagram with Candidate and Use Cases

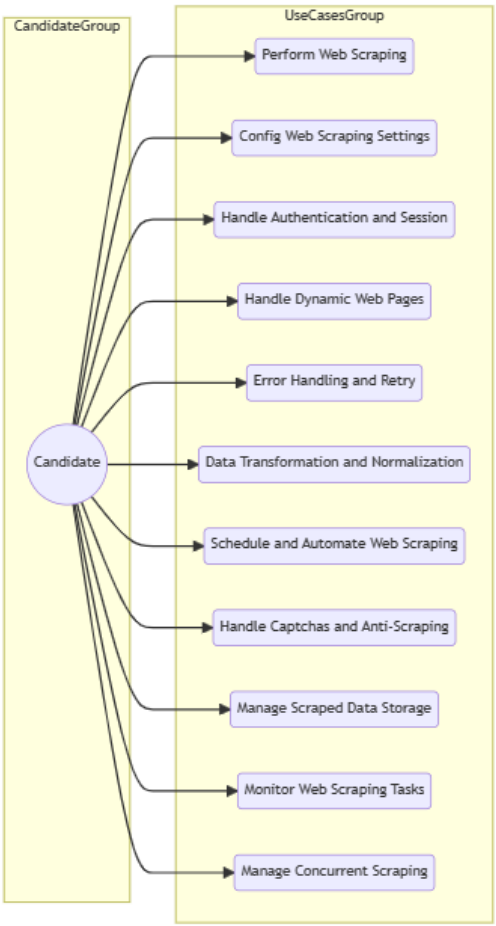


Fig 3.2 Use case diagram

3.3 Describe the Events Flow for Use Case

The events flow for the use case of web scraping involves describing the sequence of events or steps that occur when a user interacts with the web scraping system to perform a specific task. Let's consider the "Perform Web Scraping" use case as an example:

Use Case: Perform Web Scraping

Actor: User (Application or Developer)

Events Flow:

1. The user launches the web scraping tool or initiates the web scraping process through their application or script.
2. The user configures the web scraping settings, including specifying the target URLs or websites from which data needs to be scraped. The user may also define data extraction rules or patterns to identify the relevant data on the web pages.
3. The web scraping tool sends HTTP requests to the target websites' servers to access the web pages and retrieve the content.
4. The target websites' servers respond to the requests, serving the web pages' content back to the web scraping tool.
5. The web scraping tool parses the HTML or other structured content received from the websites to extract the desired data based on the configured extraction rules.
6. The tool may perform additional processing steps, such as cleaning and transforming the scraped data, to ensure it is in the desired format and quality.
7. If the web scraping task involves scraping multiple web pages or websites, the tool may repeat the process for each specified URL or website.
8. The scraped data is stored in the designated storage system, such as a database, a data warehouse, or files, for further processing or analysis.
9. The user can access and utilize the scraped data for various purposes, such as data analysis, reporting, or integration with other applications.
10. If the user has scheduled recurring web scraping tasks, the system may automatically repeat the entire events flow at the specified intervals.

Chapter No 4: Design

4.1 Architecture Diagram

1. 1. **Web Scraping Tool (A)**: This component represents the software tool or library that performs the web scraping. It sends requests to the Web Scraping Server to scrape data from the target websites.
2. **Target Websites (B, C, D)**: These components represent the websites that need to be scraped for data. The Web Scraping Server sends requests to these websites to extract the desired information.
3. **Web Scraping Server (E)**: This component acts as an intermediary between the Web Scraping Tool and the Target Websites. It receives requests from the Web Scraping Tool and forwards them to the respective Target Websites. It also receives the scraped data from the Target Websites and passes it back to the Web Scraping Tool. The Web Scraping Server may handle tasks like IP rotation, managing multiple requests in parallel, handling CAPTCHAs, etc.
4. **Data Storage/Processing System (F)**: This component represents the system where the scraped data is stored and further processed or analyzed. It could be a database, a data warehouse, or any other storage system depending on the requirements of the web scraping project.

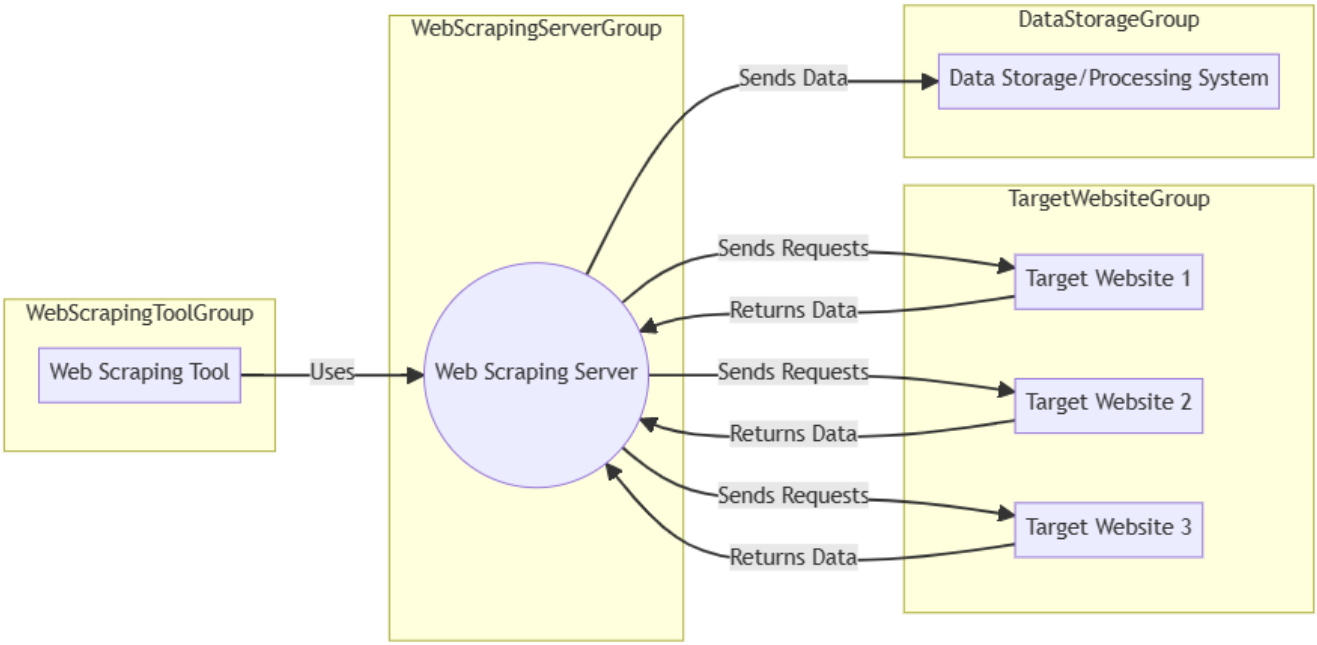


Figure 4.1 Architecture Diagram

4.2 ERD with Data Dictionary

User (Entity):

- user_id: Primary key, unique identifier for users.
- first_name: User's first name.
- last_name: User's last name.
- email: User's email address (Unique constraint to ensure uniqueness).
- password: User's password (hashed and salted).
- timezone: User's selected timezone.
- registration_date: Date and time when the user registered.

MediaDownload (Entity):

- media_id: Primary key, unique identifier for media downloads.
- media_link: Link to the Instagram picture for download.
- download_date: Date and time when the download was initiated.

Subscription (Entity):

- subscription_id: Primary key, unique identifier for subscriptions.
- instagram_username: Instagram account username subscribed to.

HashtagSearch (Entity):

- search_id: Primary key, unique identifier for hashtag searches.
- hashtag: Hashtag entered by the user for search.
- search_date: Date and time when the hashtag search was performed.

ScheduledPost (Entity):

- post_id: Primary key, unique identifier for scheduled posts.
- post_content: Content of the Instagram post to be scheduled.
- post_datetime: Date and time when the post is scheduled to be published.

ViralFinder (Entity):

- viral_id: Primary key, unique identifier for viral posts.
- search_criteria: Criteria used for searching viral posts.
- search_date: Date and time when the viral search was performed.

ConnectedInstagram (Entity):

- connection_id: Primary key, unique identifier for connected Instagram accounts.
- instagram_access_token: Access token for the connected Instagram account.

- connection_date: Date and time when the Instagram account was connected.

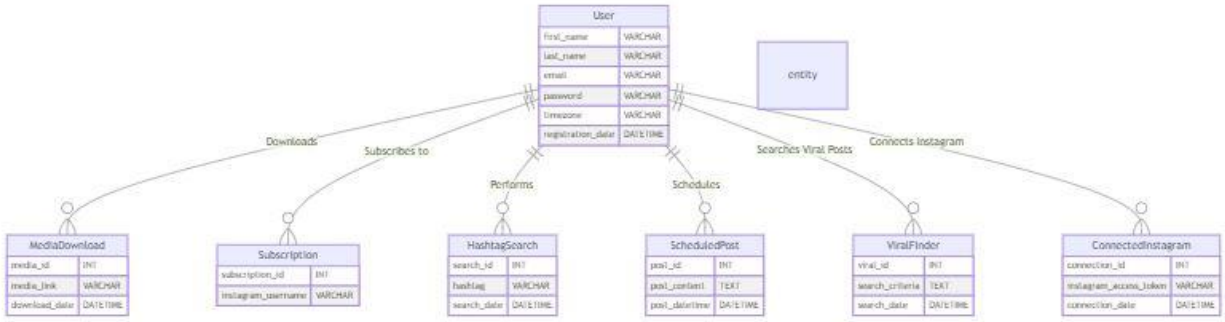


Figure 4.2 ERD Diagram with Data Dictionary

4.3 Data Flow Diagram (Level 0 and Level 1)

The User enters a hashtag to search, which is the input to the Application (B).
The Application (B) checks if the hashtag entered is valid at the decision point (C).
If the hashtag is valid (Yes path from C), the Application performs the hashtag search (D).
The search results are then displayed to the User (E)

Fig 4.3.1 Level 0

The User enters a hashtag to search, which is the input to the Application (B).
The Application (B) validates the hashtag and sends it to the Hashtag Search process (C).
The Hashtag Search process (C) performs the search and returns the search results to the Application (B).
The Application (B) then displays the search results to the User (D).

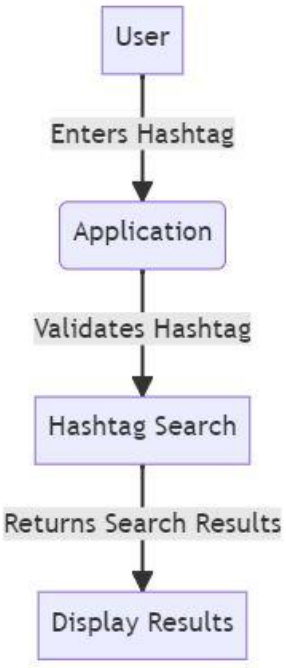


Fig 4.3.2 Level 1

4.4 Class Diagram

In this Class Diagram, each class represents a major entity in your project with its attributes and methods. The relationships between classes are represented using associations, where "1" denotes a one-to-many relationship. For example, a User can have multiple Media Downloads, Subscriptions, Hashtag Searches, Scheduled Posts, Viral Finders, and Connected Instagram.

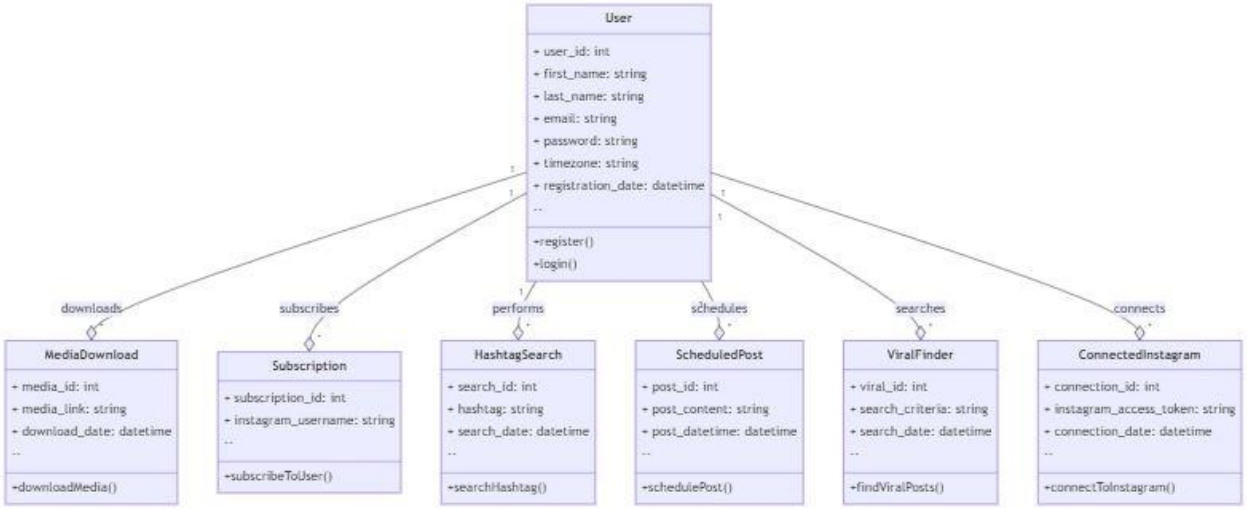


Fig 4.4 Class diagram

4.5 Object Diagram

In this Object Diagram, each class represents a major entity in your project with its attributes and methods, similar to the Class Diagram representation. However, in an Object Diagram, we can specify specific instances (objects) of each class with their unique names. For example, we have objects like "user1," "mediaDownload1," "subscription1," etc., each representing a specific instance of the corresponding class.

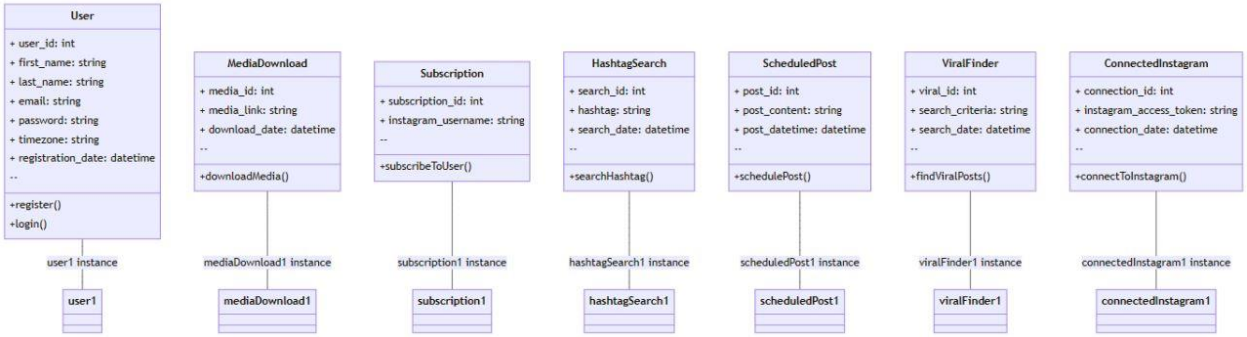


Fig 4.5 Object diagram

4.6 Sequence Diagram

- The User enters a hashtag to search in the Application.
- The Application sends a request to the Instagram API to search for the specified hashtag.
- The Instagram API processes the request and sends back the search results to the Application.
- The Application receives the search results from the Instagram API and displays them to the User.

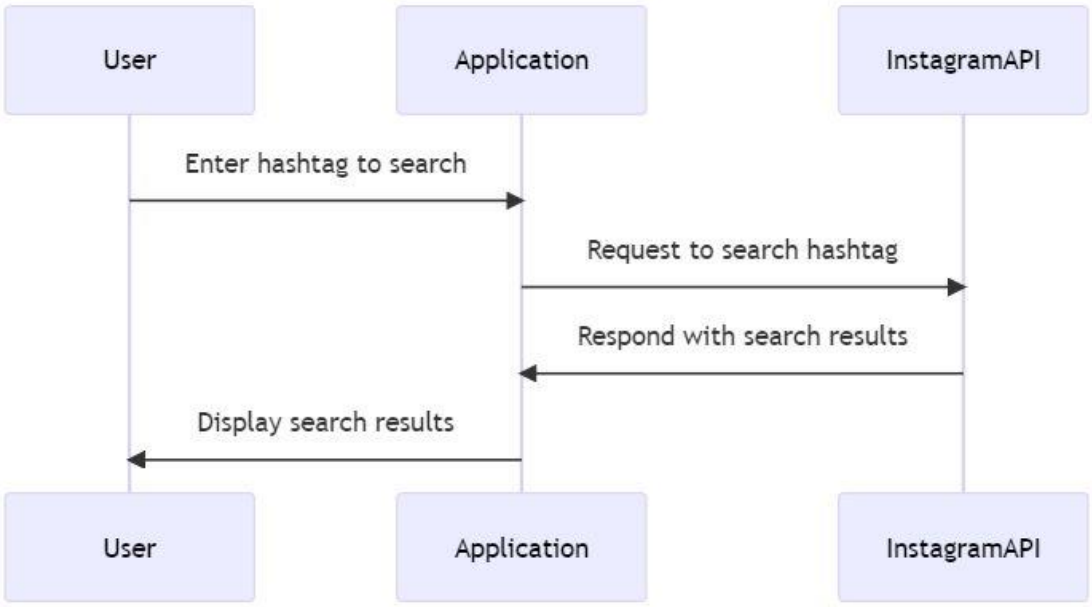


Fig 4.6 Sequence diagram

4.7 Activity Diagram

- The process starts at activity A (Start).
- The user enters a hashtag at activity B (Enter Hashtag).
- At activity C (Valid Hashtag?), a decision is made to check if the hashtag entered is valid.
- If the hashtag is valid (Yes path from C), the system performs the hashtag search at activity D (Perform Hashtag Search).
- The search results are then displayed to the user at activity E (Display Search Results).
- The process ends at activity F (End)

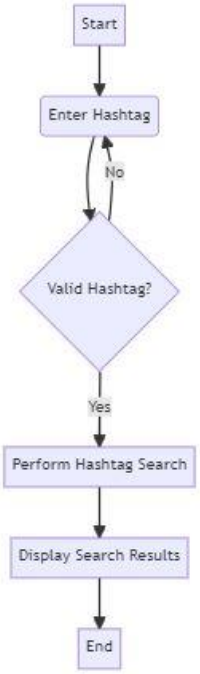


Fig 4.7 Activity diagram

4.8 Collaboration Diagram

- The User enters a hashtag to search in the Application.
- The Application validates the hashtag entered.
- If the hashtag is valid, the Application performs the hashtag search and displays the search results to the User.
- If the hashtag is invalid, the Application shows an error message to the User.

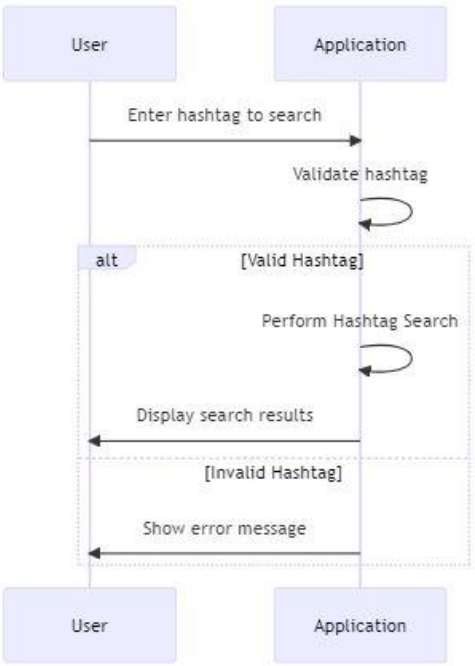


Fig 4.8 Collaboration diagram

4.9 State Transition Diagram

The initial state [*] represents the application's starting point.
Users start in the "NotLoggedIn" state.
When a user logs in, the state transitions to "LoggedIn."
When the user logs out, the state transitions back to "NotLoggedIn."
From the "LoggedIn" state, the user can transition to various other states:
"HashtagSearch" when the user performs a hashtag search.
"MediaDownloader" when the user requests media download.
"MySubscriptions" when the user accesses the subscription page.
"PostScheduler" when the user schedules a post.
"ViralFinder" when the user searches for viral posts.
From each specific state, the user can return to the "LoggedIn" state, representing a return to the dashboard.

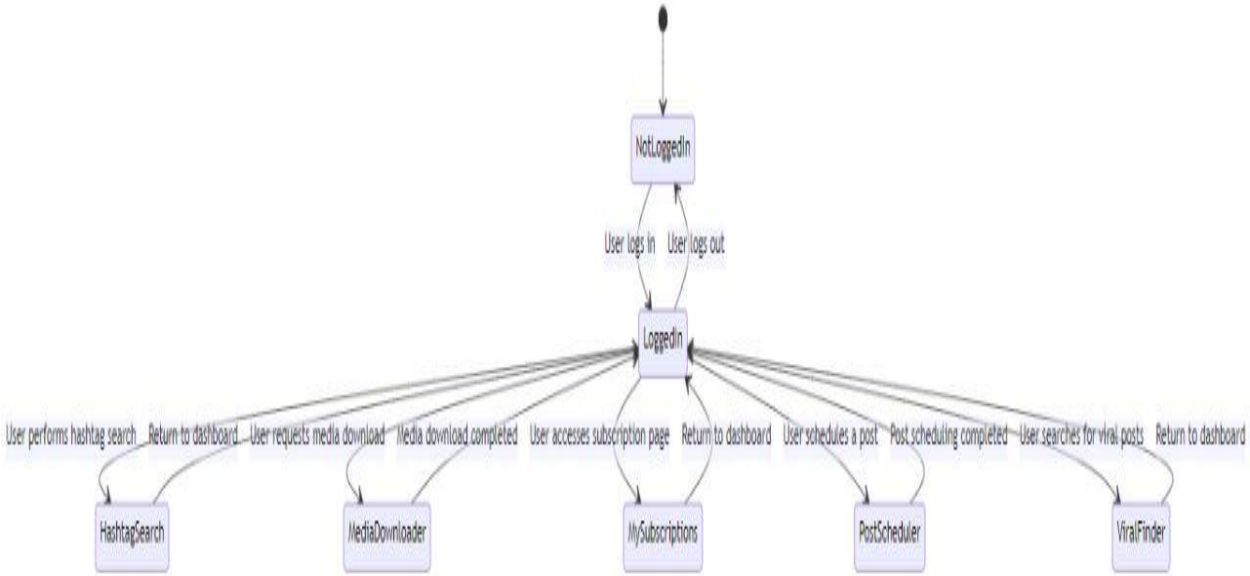


Fig 4.9 State Transition diagram

Chapter No 5: Implementation

5.1 Component Diagram

The Components of the Web scraping are so clear that here the diagram of the admin component that admin have the below components and that all stored to the database.
The component diagram for the online Web scraping illustrates the high-level architecture and interactions between key components. These components include the Admin Interface for admin interactions, an Authentication Module for admin login and access control, saw all the documents that gets from the user and then respond to it. And saved it to database. Components are.

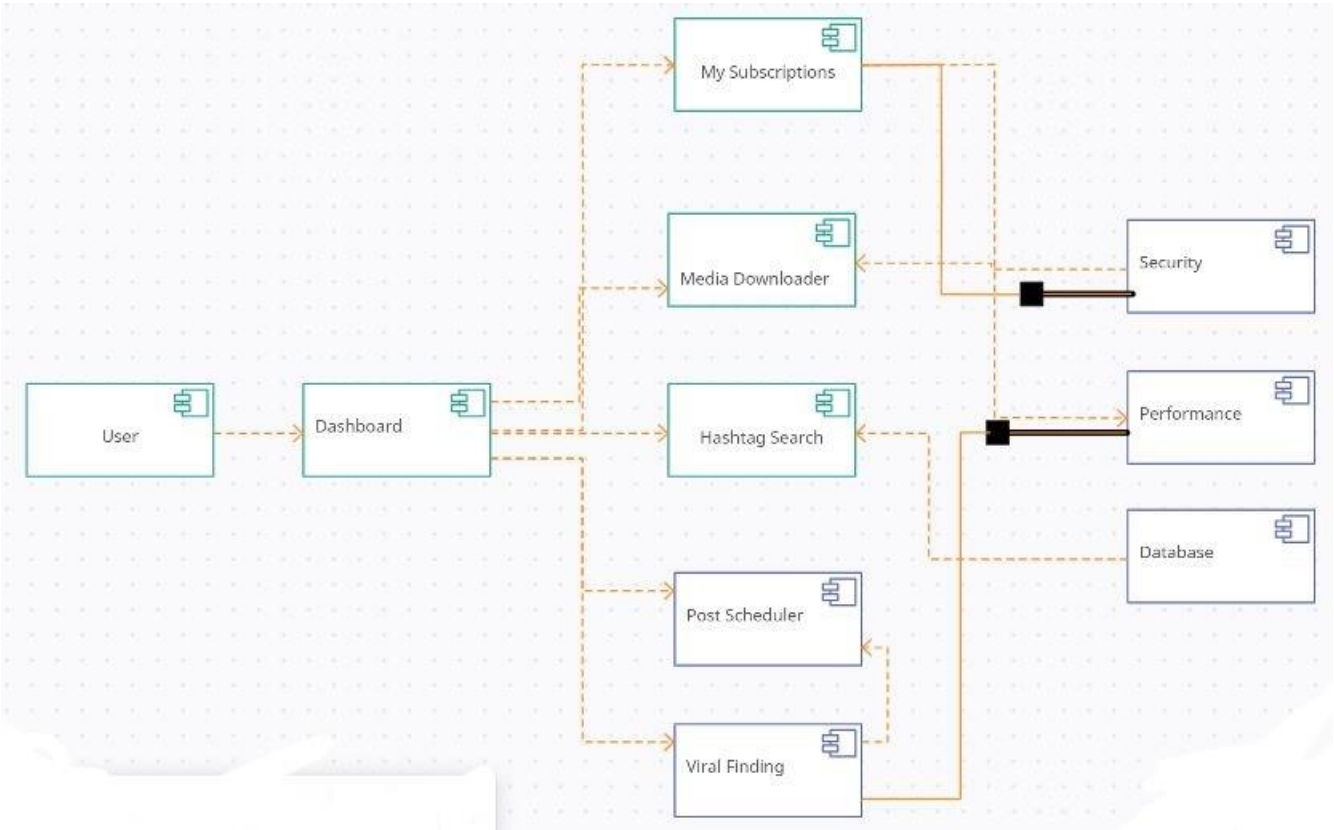


Fig 5.1 component diagram

5.2 Deployment Diagram

The deployment diagram for the online web scraping showcases the distribution of hardware and software components across various nodes. The "Application Server" forms the core, hosting modules like User Interface, Authentication, doc Manager, Results Handler, and Analytics within a web server container or application server. The "Database Server" stores the documents and User Profile databases. "Client Devices" represent endusers' devices accessing the system through web browsers. A "Load Balancer" ensures efficient resource utilization, while "External Services" integrate external systems or APIs. The "Notification Service" handles user notifications. This diagram highlights how these components collaborate to deliver a seamless and scalable web scraping experience.

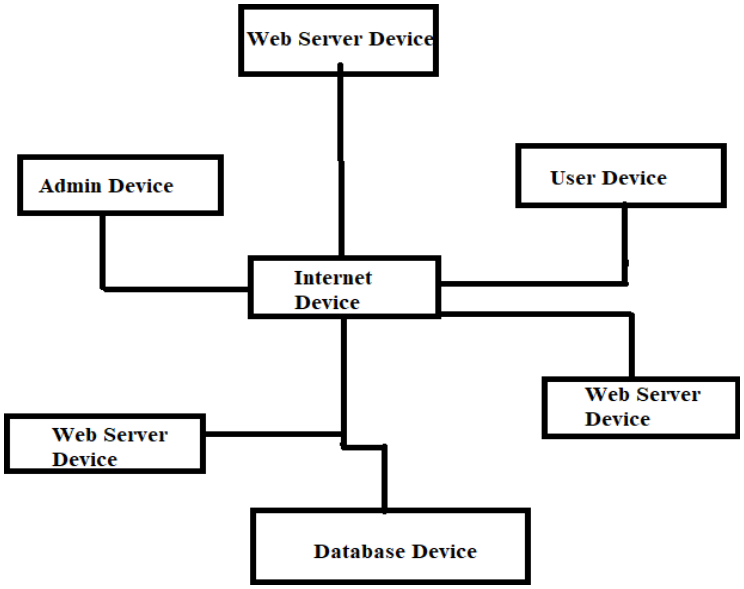


Fig 5.2 deployment diagram

5.3 Database Architecture (1- Tier, 2-Tier, 3- Tier Architecture)

For an Web scraping, the most suitable database architecture is the3-tier architecture. The 3-tier architecture provides better separation of concerns, scalability, and maintainability, making it well-suited for modern web-based applications like online web scrapings. Let's see how the components are organized in the 3-tier architecture for this project:

1. Presentation Layer (Tier 1):

The Presentation Layer is the user interface where users interact with the system. It consists of the frontend components such as web pages or mobile app interfaces. In the context of the online web scraping, this layer would include the user interface that allows usersto log in, register, view downloads, viral finder, and view their subscriptions.

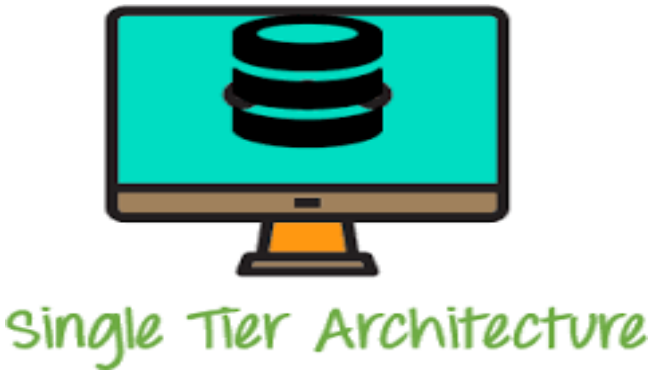


Fig 5.3.1 database architecture Tier 1

2. Application Layer (Tier 2):

The Application Layer, also known as the Business Logic Layer, is responsible for processing user requests and handling the business logic of the online web scraping. This layer contains the application code that manages web scraping.

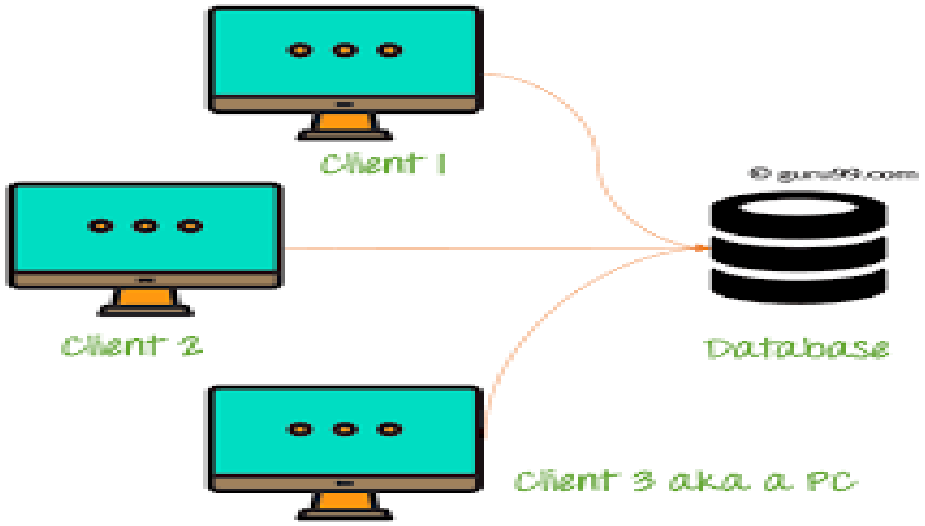


Fig 5.3.2 database architecture Tier 2

1. **Data Layer (Tier 3):**

The Data Layer is where the actual data of the online web scraping is stored andmanaged. This layer includes the database where downloads, user profiles, document results, and other relevant data are stored. It is responsible for handling data storage, retrieval, and management operations.

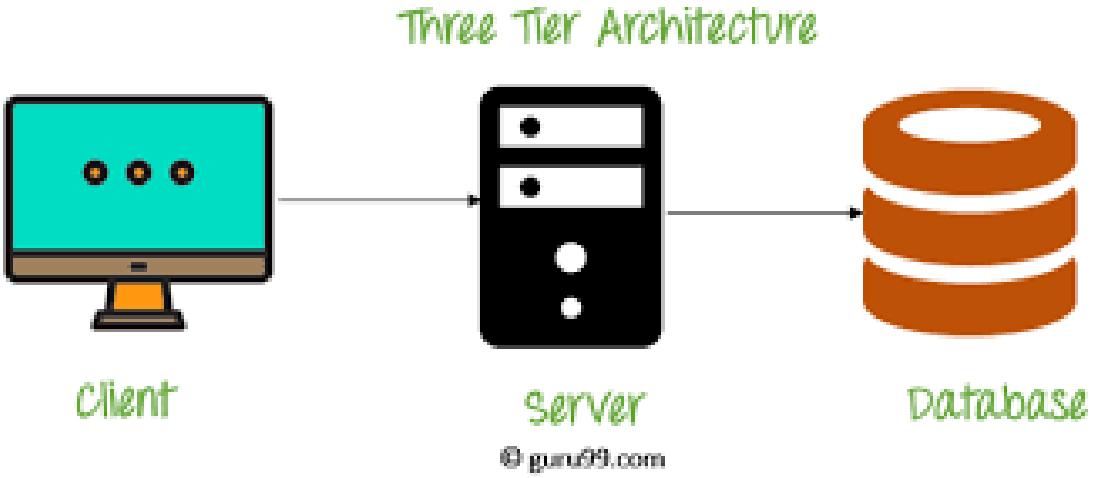


Fig 5.3.3 database architecture Tier 3

Chapter No 6: Testing (Software Quality Attributes)

6.1 Test Case Specification

The Test Case Specification for the Web scraping project outlines a concise and targeted set of test cases to verify the system's functionalities. It includes test scenarios for user authentication, web scraping and user profile management. Each test case specifies input data, expected outcomes, and steps for execution, ensuring a comprehensive testing process to identify and rectify potential issues. The specification aims to ensure a reliable and user-friendly online Web scraping that meets all project requirements.

6.2 Black Box Test Cases:

Black box testing focuses on testing the system's functionality without considering its internal structure. The following black box testing techniques will be used:

6.2.1 BVA or Boundary Value Analysis

Boundary Value Analysis (BVA) is a software testing technique used to test the boundaries or extreme values of input data. It helps identify errors that are likely to occur at the boundaries of acceptable ranges. For an online Web scraping, BVA can be applied to test various input fields, timers, and other boundary conditions.

6.2.2 Equivalence Class Partitioning

Equivalence Class Partitioning (ECP) is a software testing technique that involves dividing the input domain into groups or classes, where each class represents a set of equivalent inputs that should be treated in the same way by the system. The goal is to reduce the number of test cases while still providing adequate test coverage. For an online Web scraping, we can apply ECP to various input fields and scenarios.

6.2.3 State Transition Testing

State Transition Testing is a software testing technique that focuses on testing the behavior of a system as it transitions between different states. For an online Web scraping, state transition testing can be applied to various user interactions and system responses.

6.2.4 Decision Table Testing

Decision Table Testing is a software testing technique used to test the combinations of different inputs and conditions in a system. It helps identify and test all possible combinations of inputs

and their associated outcomes. For an Web scraping, decision table testing can be applied to various business rules and conditions.

6.2.5 Graph Base Testing

Graph-based testing is a software testing technique that involves representing the relationships between various components of the system as a graph and then using graph traversal algorithms to design and execute test cases. In the context of an online Web scraping, graphbased testing can be applied to model the interactions between different modules, pages, and functionalities of the system.

6.3 White Box Testing:

White box testing focuses on examining the internal structure of the system, including the code, to ensure that all paths and conditions are tested. The following white box testing techniques will be used:

6.3.1 Statement Coverage

Statement coverage is a metric used in software testing to measure the percentage of code statements that have been executed during the testing process. It provides an indication of how much of the source code has been tested. For an online Web scraping project, statement coverage can be achieved by executing test cases that exercise different paths and functionalities in the code base.

6.3.2 Branch Coverage

Branch coverage is a software testing metric that measures the percentage of decision branches that have been executed during the testing process. A decision branch is a point in the code where the program can take one of two or more paths based on a condition or decision. For an online Web scraping project, branch coverage can be achieved by designing test cases that exercise different possible outcomes and decision points in the code.

6.3.3 Path Coverage

Path coverage is a software testing metric that measures the percentage of unique paths through the source code that have been executed during the testing process. A path is a unique sequence of statements and branches from the entry point of a function to its exit point. For an online Web scraping project, path coverage can be achieved by designing test cases that exercise all possible paths in the code.

Test Cases:

Register Test Case 1:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 1	Sign upwithdata entered	• First Name(empty) • Last Name(empty) • Email(empty) • Password(empty) • Confirm Password(empty) • Timezone(empty)	Role:User First Name: Last Name: Email: Password: Confirm Password: Timezone	Successfully	AsExpected	P

Table4.1 register testcase

Register Test Case 2:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 2	Sign upwithdata entered	• First Name(empty) • Last Name • Email • Password(empty) • Confirm Password(empty) • Timezone(emp	Role:User First Name: Last Name: Sikandar Email: sania@gmail.com Password: Confirm Password: Timezone	Successfully	AsExpected	P

		ty)				

Table4. 2 register testcase

Register Test Case 3:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 3	Sign upwithdata entered	• First Name • Last Name • Email • Password(empty) • Confirm Password(empty) • Timezone(empty)	Role:User First Name: Sania Last Name: Sikandar Email: sania@gmail.com Password: Confirm Password: Timezone	Successfully	AsExpected	P

Table4. 3 register testcase

Register Test Case 4:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 4	Sign upwithdata entered	• First Name • Last Name • Email • Password • Confirm Password • Timezone	Role:User First Name: Sania Last Name: Sikandar Email: sania@gmail.com Password: Sania123 Confirm Password: Sania123 Timezone Tuesday, 1 August 2023,6:45 pm	Successfully	AsExpected	P

Table4. 4 register testcase

User Login Test Case 1:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 1	Sign inwithdata entered	• Email(empty) • Password(empty)	Role:User Email: Password :	Successfully	AsExpected	P

Table4.5user login testcase

User Login Test Case 2:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 2	Sign inwithdata entered	• Email(empty) • Password	Role:User Email: Password : Sania45	Successfully	AsExpected	P

Table4.6 user login testcase

User Login Test Case 3:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 3	Sign inwithdata entered	• Email • Password(empty)	Role:User Email: sania@gmail.com Password :	Successfully	AsExpected	P

Table4.7userlogin testcase

User Login Test Case 4:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 4	Sign inwithdata entered	• Email • Password	Role:User Email: sania@gmail.com Password : Sania45	Successfully	AsExpected	P

Table4.8userlogin testcase

Download Test Case 1:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 1	Download withdataentered	• URL(empty)	Role:User URL:	Successfully	AsExpected	P

Table4.9download testcase

Download Test Case 2:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 2	Download withdataentered	• URL	Role:User URL: www.insgr/images.com	Successfully	AsExpected	P

Table4.9download testcase

Download Test Case 3:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 3	Download withdataentered	• URL(incorrect)	Role:User URL: www.images.com	Successfully	AsExpected	P

Table4.10download testcase

Viral FinderTest Case 1:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 3	Viral Finder dataentered	• Search(empty)	Role:User Search:	Successfully	AsExpected	P

Table4.11viral finder testcase

Viral Finder Test Case 2:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 2	Viral Finder dataentered	• Search	Role:User Search: #something	Successfully	AsExpected	P

Table4.12viral finder testcase

Viral Finder Test Case 3:

Test CaseID	TestScenarios	TestSteps	TestData	Expected Result	Actual Result	P/F
TC– 3	Viral Finder dataentered	• Search(incorrect)	Role:User Search: #skuch344	Successfully	AsExpected	P

Table4.13viral finder testcase

Chapter No 7: Tools and Technologies

7.1 Programming Languages

- Javascript
- html5
- CSS
- Bootstrap

7.2 Operating Environment

- Linux
- Windows
- macOS

Appendix A: User documentation

Check list	YES	NO
Title and Description	YES	
Introduction to the Problem	YES	
Software Requirement Specification	YES	
Analysis(Use case model)	YES	
Design	YES	
Implementation	YES	
Testing(Software Quality Attributes)	YES	
Tools and Technologies	YES	

Appendix B: Source Code

```
namespace App\Classes;


use App\Post;
use App\Utils\Globals\Statuses;
use GuzzleHttp\Client;
use Illuminate\Support\Facades\Auth;



class InstagramAPI
{
    private $base_uri;
    private $client;
    private $access_token;

    public function __construct($base_uri = null)
    {
        //        $this->base_uri = $base_uri;
        //        if ($base_uri == null){
        $this->base_uri = 'https://graph.facebook.com/v10.0';
        //        }
        $this->client = new Client([
            'base_uri' => $this->base_uri,
        ]);
    }
```

```
public function setAccessToken($token)
{
    $this->access_token = $token;
}

public function getUser()
{
    $this->access_token = Auth::user()->instagram->access_token;
    if ($this->access_token) {
        $response = $this->client->request('GET', '/me', [
            'query' => [
                'fields' => 'id,username',
                'access_token' =>  $this->access_token
            ]
        ]);
        return json_decode($response->getBody()->getContents());
    }
    return [];
}

public function autoPost($post)
{

    $hashtags = str_replace(['[', ']'], '', $post->tags);
```

```
$container = null;

$caption = $post->caption;
$caption = $caption . ' ' . $hashtags;

$image_url = $post->media_name;

$user_id = $post->user->instagram->instagram_id;
$uri = $user_id . '/media';

$this->access_token = $post->user->instagram->access_token;

if ($this->access_token) {
    $response = $this->client->request('POST', $uri, [
        'query' => [
            'caption' => $caption,
            'image_url' => $image_url, // actual image not working on local
            'access_token' =>  $this->access_token,
        ]
    ]);
    $container = json_decode($response->getBody()->getContents());
}
$post->status = Statuses::POSTED;
$post->update();
```

```
    return $this->createPost($container->id, $post->user);
}

public function createPost($container_id, $user)
{

    $user_id = $user->instagram->instagram_id;
    $uri = $user_id . '/media_publish';

    if ($container_id != null) {
        $response = $this->client->request('POST', $uri, [
            'query' => [
                'creation_id' => $container_id,
                'access_token' =>  $this->access_token,
            ]
        ]);
        return json_decode($response->getBody()->getContents());
    }
}

public function getAccount($username)
{
    $user_id = Auth::user()->instagram->instagram_id;
    $uri = $user_id . '/';
```

```
        $fields = 'business_discovery.username(' . $username .
')){username,website,name,profile_picture_url,biography,followers_count,follows_count,media_
count,media{id,caption,media_type,comments_count,like_count,media_url,permalink}}';

        $this->access_token = Auth::user()->instagram->access_token;


        if ($this->access_token) {
            $response = $this->client->request('GET', $uri, [
                'query' => [
                    'fields' => $fields,
                    'access_token' =>  $this->access_token
                ]
            ]);
            return json_decode($response->getBody()->getContents());
        }
        return [];
    }


    public function getPosts()
    {
        if ($this->access_token) {
            $response = $this->client->request('GET', 'users/self/media/recent/', [
                'query' => [
                    'access_token' =>  $this->access_token
                ]
            ]);
            return json_decode($response->getBody()->getContents())->data;
```

```
    }
    return [];
}

public function getTagPosts($tags)
{
    if ($this->access_token) {
        $response = $this->client->request('GET', 'tags/' . $tags . '/media/recent/', [
            'query' => [
                'access_token' =>  $this->access_token
            ]
        ]);
        return json_decode($response->getBody()->getContents())->data;
    }
    return [];
}

public function findTagId($hashtag)
{

    $this->access_token = Auth::user()->instagram->access_token;
    $user_id = Auth::user()->instagram->instagram_id;

    if ($this->access_token) {
        $response = $this->client->request('GET', 'ig_hashtag_search', [
```

```
            'query' => [
                'user_id' => $user_id,
                'q' => $hashtag,
                'access_token' =>  $this->access_token,
            ]
        ]);
        return json_decode($response->getBody()->getContents())->data;
    }
}

public function getTopMedia($hashtag_id)
{

    $this->access_token = Auth::user()->instagram->access_token;
    $user_id = Auth::user()->instagram->instagram_id;
    $uri = $hashtag_id . '/top_media';
    $fields =
'id,caption,media_type,comments_count,like_count,engagement,media_url,children,permalink,ti
mestamp';

    if ($this->access_token) {
        $response = $this->client->request('GET', $uri, [
            'query' => [
                'user_id' => $user_id,
                'fields' => $fields,
                'limit' => 50,
```

```
                'access_token' =>  $this->access_token,
            ]
        ]);
        return json_decode($response->getBody()->getContents())->data;
    }
}

public function getHashtagStats($keyword)
{
    $uri = 'https://www.instagram.com/explore/tags/' . $keyword . '/?__a=1';
    $json = file_get_contents($uri);
    $json_data = json_decode($json);

    // returning only the hashtag volume
    $hashtags_posts_volume = $json_data->graphql->hashtag->edge_hashtag_to_media->count;
    return $hashtags_posts_volume;
}

public function test()
{
    $image_url = "../storage/images/test_image.jpg";

    // user_id => 17841404710463491
    //       GET graph.facebook.com/ig_hashtag_search?user_id=17841405309211844&q=coke
```

```
//      `GET https://graph.instagram.com/ig_hashtag_search?user_id=12314&q=music`

$this->access_token = Auth::user()->instagram->access_token;
//      if($this->access_token){
//          $response = $this->client->request('GET', 'me/media', [
//              'query' => [
//                  'fields' => 'id,caption',
//                  'access_token' =>  $this->access_token
//              ]
//          ]);
//          dd(json_decode($response->getBody()->getContents()));
//          return json_decode($response->getBody()->getContents());
//      }

//      if($this->access_token){
//          $response = $this->client->request('GET', '17859385379444302', [
//              'query' => [
//                  'fields' => 'id,media_type,media_url,username,timestamp',
//                  'access_token' =>  $this->access_token
//              ]
//          ]);
//          dd(json_decode($response->getBody()->getContents()));
//          return json_decode($response->getBody()->getContents());
//      }
```

```
        //      $data = null;
        //      if($this->access_token){
        //          $response = $this->client->request('GET', 'me/media', [
        //              'query' => [
        //                  'fields' => 'id,media_type,media_url,username,timestamp',
        //                  'access_token' =>  $this->access_token
        //              ]
        //          ]);
        //          $data = json_decode($response->getBody()->getContents());
        ////            return json_decode($response->getBody()->getContents());
        //      }
        //
        //      $media_url = $data->data[1]->media_url;
        //      echo '<img src="'.$media_url.'">';

        dd('work in progress');
        return [];
    }
}
```