URL CHECKER

A MINI PROJECT REPORT

Submitted by

RANJITHA R [REGISTER NO:211421104205]

PRIYA S [REGISTER NO:211421104194]

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE,CHENNAI-600123

OCTOBER 2023

BONAFIDE CERTIFICATE

Certified that this project report "**URL CHECKER**" is the bonafide work of **RANJITHA R (211421104205) & PRIYA S (211421104194)** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.L.JABASHEELA,M.E.,(Ph.D)	**Mr.C.THYAGARAJAN,M.E.,(Ph.D.),**
PROFESSOR,	**ASSISTANT PROFESSOR,**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
DEPARTMENT OF CSE,	DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,	PANIMALAR ENGINEERINCOLLEGE,
NASARATHPETTAI,	NASARATHPETTAI,
POONAMALLEE,	POONAMALLEE,
CHENNAI-600 123.	CHENNAI-600 123.

Certified that the above candidate(s) were examined in the Mini Project Viva-Voce Examination held on............................

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

DECLARATION BY THE STUDENT

We **RANJITHA R, PRIYA S (211421104205,211421104194)** hereby declare that this project report titled **“URL CHECKER”** , under the guidance of **Mr. C.THYAGARAJAN Assistant professor** is the orginial work done by us and we have not plagiarized or submitted to any other degree in any university by us.

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our beloved Directors

Tmt.C.VIJAYARAJESWARI, Dr.C.SAKTHI KUMAR, M. E.,Ph.D and

Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D., for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.MANI, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. L.JABASHEELA, M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank our parents, friends, project Guide **Mr.C.THYAGARAJAN M.E,** and coordinator **Mr.M.KRISHNAMOORTHY M.C.A.,M.E,M.B.A,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

NAME OF THE STUDENTS
RANJITHA R
PRIYA S

TABLE CONTENTS

Chapter 1

ABSTRACT

The LinkChecker script is designed to check the status of links stored in a Google Sheets spreadsheet named 'LinkChecker.' It fetches URLs from column B of the 'LinkChecker' sheet, and then performs checks on the response code and response body of each URL. It cross-references the data with the 'Assets' sheet, aiming to detect any matching content and response codes to determine whether the link is active or removed. The URL Checker project is a powerful tool designed to enhance the data validation and quality assurance processes within Microsoft Excel spreadsheets. This project aims to provide Excel users with an efficient and user-friendly solution to verify and validate URLs embedded within their spreadsheets. With the increasing reliance on web-based resources in various domains, ensuring the accuracy and validity of URLs is crucial**.** This tool automatically scans a specified range of cells in an Excel spreadsheet for URLs and checks their validity. It verifies whether the URLs are correctly formatted and whether they lead to functional web pages. In summary, the Excel URL Checker project empowers Excel users to maintain data accuracy and reliability by automatically validating URLs within their spreadsheets. By offering customization options, user-friendliness, and error reporting capabilities, it streamlines the process of URL validation, ultimately contributing to improved data quality.

Chapter 2

INTRODUCTION

In the vast landscape of the World Wide Web, hyperlinks are the threads that weave together the fabric of information. They connect websites, documents, and online resources, providing users with the gateway to a wealth of knowledge and services. However, the dynamic nature of the internet and the constant evolution of web content can lead to issues such as broken links, redirects, and content changes, which can hinder the user experience and erode trust in online information. This is where the URL Checker emerges as a crucial tool in the digital arsenal.

The URL Checker is a powerful, versatile, and indispensable utility designed to address the pressing need for maintaining link health on the internet. Its primary mission is to validate the status and integrity of URLs found within web content, ensuring that they lead users to the right destinations. This introduction provides a glimpse into the world of the URL Checker, emphasizing its importance in the online landscape.

1. **The Pervasive Issue of Broken Links:**
   As the internet continually expands, the prevalence of broken links has become an endemic issue. Websites, blogs, articles, and even academic resources are littered with URLs that no longer function as intended. These broken links can frustrate users, impact SEO rankings, and erode the credibility of online sources.

2. **User Experience and Trust:**
   A seamless user experience is a cornerstone of successful websites and applications. Trust and user satisfaction are closely linked to the reliability

of web links. Broken misleading URLs can drive visitors away and tarnish the reputation of a website.

3. **Efficiency and Productivity*:***
   For web administrators, content creators, and developers, ensuring the health of links is a time-consuming task. The URL Checker streamlines this process, automating link validation and reducing the burden of manual checks.

4. **Content Freshness and Accuracy***:*
   The URL Checker is the vigilant guardian of web links, preserving the user experience, maintaining the credibility of online information, and ensuring that the internet remains a reliable source of knowledge and services. In a world where the digital domain is intertwined with our daily lives, this tool is a beacon of reliability and trustworthiness. As we delve deeper into the features and capabilities of the URL Checker, its significance becomes even more evident in the digital age**.**

2.1 OVERVIEW:

The URL Checker is a versatile and indispensable tool designed to validate and maintain the health of hyperlinks (URLs) within web content, documents, and applications. In an era where the internet is the primary source of information and services, ensuring the integrity and functionality of these links is essential. The URL Checker offers a comprehensive solution to address this critical need, providing users with the means to:

- **Validate Link Integrity*:***
  The primary function of the URL Checker is to validate the status of URLs. It checks for broken links, redirects, and other issues that may disrupt the user experience. By identifying and rectifying these problems, it ensures that users are seamlessly connected to the intended online resources.

- **Streamline Web Maintenance*:***
  For website administrators and content creators, manually verifying the health of links can be a time-consuming task. The URL Checker automates this process, saving time and effort while ensuring that web content remains reliable and user-friendly.

- **Enhance User Experience:**
  Broken links or redirects can frustrate website visitors and lead to increased bounce rates. By systematically checking and fixing URLs, the URL Checker contributes to a positive user experience, promoting user satisfaction and trust.

- **Optimize SEO Rankings:**
  Search engines consider the quality of a website's links when determining its search engine rankings. By identifying and addressing broken links and

other issues, the URL Checker can positively impact a website's SEO performance.

- **Maintain Content Relevance:**
  In a digital landscape where information is constantly evolving, the URL Checker assists in ensuring that linked content is up-to-date and relevant. This feature is particularly valuable for websites and documents with dynamic or frequently updated content.

- **Cross-Platform Compatibility:**
  The URL Checker is adaptable to various web technologies, file formats, and content management systems, making it a versatile solution for a wide range of users, from web developers to content creators and document editors.

- **Customized Reporting*:***
  The tool generates detailed reports highlighting issues found in URLs, offering actionable insights for users to rectify problems efficiently.

- **Scheduled and Automated Checks:**
  Users can set up automated checks at regular intervals, reducing the burden of continuous manual monitoring and ensuring link health over time.

The URL Checker is a pivotal tool in the digital landscape, empowering users to maintain and enhance the quality of their web content, documents, and applications. By ensuring the reliability of links, it contributes to a more trustworthy and user-friendly online environment. In this overview, we have touched upon its fundamental functions and the benefits it brings to web administrators, content creators, and users alike. The URL Checker is a valuable asset in an era where the internet is the gateway to knowledge, services, and communication, and its role in maintaining the health of web links is pivotal.

2.2 PROBLEM STATEMENT:

In the ever-expanding digital landscape, hyperlinks (URLs) are the arteries of information flow, connecting users to a vast array of online resources. However, the pervasive issue of broken links, redirects, and outdated URLs hinders the seamless access to web content, erodes user trust, and adversely affects website performance. The problem at hand is the critical need for a reliable and efficient URL checker tool to address the following key challenges:

- **User Frustration:**

  Broken links or redirects often result in user frustration, disrupt the user experience, and discourage users from exploring a website's content. This problem is exacerbated by the fact that many users rely on these links to access information quickly and efficiently.

- **Trust and Credibility:**

  Websites with broken links appear outdated, unprofessional, and unreliable. Ensuring the integrity of links is paramount for maintaining trust and credibility in an era where the internet is a primary source of information.

- **Web Content Maintenance:**
  Manual verification of links within websites, documents, and applications is a labor intensive task, consuming valuable time and resources. A reliable URL checker can significantly streamline this maintenance process.

- **SEO Impact:**
  Broken or low-quality links can negatively affect a website's search engine ranking. Search engines consider the quality and functionality of links when determining a site's relevance and authority, making URL health an essential component of SEO optimization.

- **Content Relevance:**
  In an era of rapidly changing information, it's crucial to ensure that linked content remains up-to-date and relevant. An effective URL checker can assist in maintaining content relevance by identifying outdated links.

- **Cross-Platform Compatibility:**
  The diverse nature of online content, ranging from websites and blogs to documents and applications, requires a URL checker that is compatible with various platforms, web technologies, and content management systems.

- **Reporting and Actionability:**
  Effective diagnosis of URL issues is only half the solution. Users need clear, actionable insights and reporting to rectify identified problems efficiently.

- **Automated Monitoring:**
  Continuous vigilance is required to maintain link health over time. A robust URL checker should offer features for automated, scheduled checks to ensure the ongoing reliability of URLs.

Addressing these challenges is vital for web administrators, content creators, and developers who seek to maintain a high-quality online presence and provide users with a seamless, trustworthy, and enjoyable browsing experience. The development of an advanced and user-friendly URL checker is crucial to overcoming these obstacles and contributing to a more reliable and user-centric digital environment.

Chapter 3

LITERATURE SURVEY

1.Whittingham, M. S. (1976). "Electrical Energy Storage and Intercalation Chemistry." Science, 192(4244), 1126-1127.

2. Armand, M., & Tarascon, J. M. (2008). "Building Better Batteries." Nature, 451(7179), 652- 657.

3. Tarascon, J. M., & Armand, M. (2001). "Issues and Challenges Facing Rechargeable Lithium Batteries." Nature, 414(6861), 359-367.

4. Scrosati, B., & Garche, J. (2010). "Lithium Batteries: Status, Prospects and Future." Journal of
Power Sources, 195(9), 2419-2430.

5. Goodenough, J. B., & Park, K. S. (2013). "The Li-Ion Rechargeable Battery: A Perspective." Journal of the American Chemical Society, 135(4), 1167-1176.

6. Manthiram, A., et al. (2017). "A Perspective on the High-Voltage LiMn1.5Ni0.5O4 Spinel Cathode for Li-Ion Batteries." Journal of The Electrochemical Society, 164(1), A3898-A3905.

7. Plett, G. L. (2004). "Battery Management Systems, Part I: Battery Modeling." Journal of Power Sources, 134(2), 252-261.

8. Erol-Kantarci, M., & Mouftah, H. T. (2013). "Towards Energy-Efficient Battery Management for Smart Grids." IEEE Transactions on Industrial Electronics, 60(10), 4403-4411.

9. Gaustad, G., et al. (2018). "Sustainability of Batteries for Portable Electronics." Sustainable Materials and Technologies, 15, 1-8.

10. Gaines, L., & Hsu, F. V. (2016). "Energy and Emissions Footprint of Electric Vehicles: Implications for Eco-innovations." Environmental Science & Technology, 50(3), 1345-135

Chapter 4

MODULES

A URL checker tool typically consists of several interconnected modules
or components to perform various tasks related to validating and
managing URLs. Here are the key modules that can be included in a
URL checker:

- **URL Input Module:**
  Responsible for receiving and processing a list of URLs to be checked. This module can accept URLs from various sources, such as text files, web pages, or API endpoints.

- **URL Validation Module:**
  Performs the core function of checking the validity and integrity of URLs. It checks for issues like broken links, redirects, and missing web pages. This module may use various techniques, including HTTP requests and response analysis.

- **Automated Checking Module:**
  Allows users to schedule automated URL checks at regular intervals. This module ensures that URLs are continuously monitored for issues, reducing the need for manual checks.

- **Cross-Platform Compatibility Module*:***

  Ensures that the URL checker is compatible with various web technologies, file formats, and content management systems. This module allows the tool to be used across different platforms.

- **Content Relevance Analysis Module:**

  Assesses the relevance and quality of linked content. It may analyze factors such as content freshness, duplication, or the presence of specific keywords to determine the usefulness and currency of linked resources.

- **Reporting and Alerts Module:**

  zGenerates detailed reports that provide information about the status of checked URLs. Users can receive alerts or notifications when issues are detected, enabling quick action to rectify problems.

- **User Interface Module:**

  Provides a user-friendly interface for users to interact with the URL checker. This module allows users to configure settings, initiate checks, view reports, and manage the tool effectively.

- **Integration and API Module***:*

  Allows the URL checker to be integrated with other tools, content management systems, or APIs, enhancing its versatility and enabling seamless interactions with various platforms.

- **Customization and Configuration Module:**

  Permits users to customize the tool to meet their specific requirements. This module may include settings for scan depth, frequency of automated checks, and other preferences.

- **Database and Storage Module:**
  Stores historical data about checked URLs, including their status, timestamps, and any issues identified. This data can be valuable for analysis, tracking changes over time, and assessing the overall health of a website.

- **Security Module:**
  Ensures that the URL checker operates securely and follows best practices when making HTTP requests. It may include features such as SSL certificate validation, secure data storage, and user authentication.

- **Performance Optimization Module**:
  Enhances the speed and efficiency of the URL checker, allowing it to process a large number of URLs in a timely manner while minimizing resource consumption.

- **API Documentation and Help Module:**
  Provides comprehensive documentation and help resources to guide users on how to use the URL checker effectively and troubleshoot common issues.

These modules work together to create a comprehensive and robust URL checker tool, catering to the diverse needs of web administrators, content creators, and developers in maintaining the quality and integrity of URLs within web content.

Chapter 5

Tools And Methods

5.1 Development Environment

SOFTWARE REQUIREMENTS*:*

1. Google Sheets
2. Google Script
3. Java Script
4. Website URL’s
5. Functions

GOOGLE SHEETS:

Google is a web -based spreadsheet application developed by google. It is part of the google workplace (formerly known as G Suite) productivity suite, which includes tools like Google Docs and Google Slides.

Google Sheet is a powerful and versatile tool for creating, editing, and storing spreadsheets online.

GOOGLE SCRIPT:

Google Apps Script is a cloud-based JavaScript platform that allows you to automate tasks, extend Google Workspace applications (such as Google Sheets, Google Docs, Google Forms, and Gmail), and build custom web applications.

Java Script:

JavaScript (often referred to as "JS") is a versatile and widely used programming language primarily employed for client-side web development.JavaScript is essential for building interactive and dynamic websites. It is used to enhance user interfaces, create responsive designs, and add interactivity to web pages. JavaScript is executed in the user's web browser, allowing for dynamic content updates without the need to reload the entire page. This enables smooth user experiences and is commonly used for form validation, animations, and real-time updates.

Website URL’s:

Website URLs (Uniform Resource Locators) are used to identify and locate resources on the World Wide Web. The primary use of a URL is to access websites and web pages. Users enter URLs in web browsers to view web content, such as articles, images, videos, and interactive applications.URLs are used as hyperlinks to connect web pages and resources. Clicking on a URL link takes the user to the linked web page or resource, allowing for easy navigation between different parts of the web.URLs are commonly used to share web content with others. Users can copy and paste URLs to share specific web pages, articles, videos, or other resources via email, social media, messaging apps, or any other form of digital communication.

Functions:

Google Sheets functions, often referred to as Google Sheets formulas, are used to perform a wide range of tasks and calculations within a Google

Sheets spreadsheet. Google Sheets functions can be used to perform various mathematical operations, such as addition, subtraction, multiplication, division, exponentiation, and more. Examples include SUM, AVERAGE, PRODUCT, POWER, and SQRT.

HARDWARE REQUIREMENTS:

1.Processor:Minimum 1 GHz

2.Memory (RAM):4 GB

3.Hard Drive:32 GB

4.Internet Connection

Processor: Minimum 1 GHz

A processor with a minimum clock speed of 1 GHz (1 gigahertz) is generally used for basic computing tasks and older systems. While the specific use cases can vary. A 1 GHz processor is sufficient for basic web browsing, reading emails, and using web-based applications. It may not provide the best performance for heavy web applications or multimedia-rich websites.

Memory (RAM):4 GB

A computer with 4 GB of RAM is considered to have a relatively low amount of memory by modern standards. However, it can still serve several useful purposes, especially for basic and lightweight computing tasks.

Hard Drive:32 GB

A computer with a 32 GB hard drive has limited storage capacity compared to modern standards. This amount of storage is considered quite small in today's computing environment. However, such a setup can still serve some specific purposes and use cases.You can use the computer for basic web browsing and online activities. However, you may need to manage your downloads and avoid storing large media files locally.

Internet Connection

An internet connection is a fundamental component of modern life and serves a wide range of purposes across various domains.The most basic and widespread use of the internet is for accessing websites and online content. Users can browse the web to access information, news, articles, and resources.Email communication is a primary function of the internet, allowing people to send and receive messages, documents, and files. Using platforms like Facebook, Twitter, Instagram, and LinkedIn for social networking, sharing updates, and connecting with others.Conducting video calls and virtual meetings for work, education, and personal communication using services like Zoom, Microsoft Teams, and Skype.Watching movies, TV shows, and videos on platforms such as Netflix, YouTube, Amazon Prime Video, and Hulu.Playing video games, including multiplayer and online games, on gaming platforms like Steam, Xbox Live, and PlayStation Network.

Chapter 6

SYSTEM DESIGN

6.1 UML

A UML (Unified Modeling Language) diagram is a visual representation of a system, software, or a process, used for design, documentation, and communication purposes. UML diagrams come in several types, each serving a specific purpose. A use case diagram depicts the interactions between a system and its users. It identifies the various use cases (functionality) of the system and the actors (users or external systems) involved.

6.1.1 Use Case Diagram

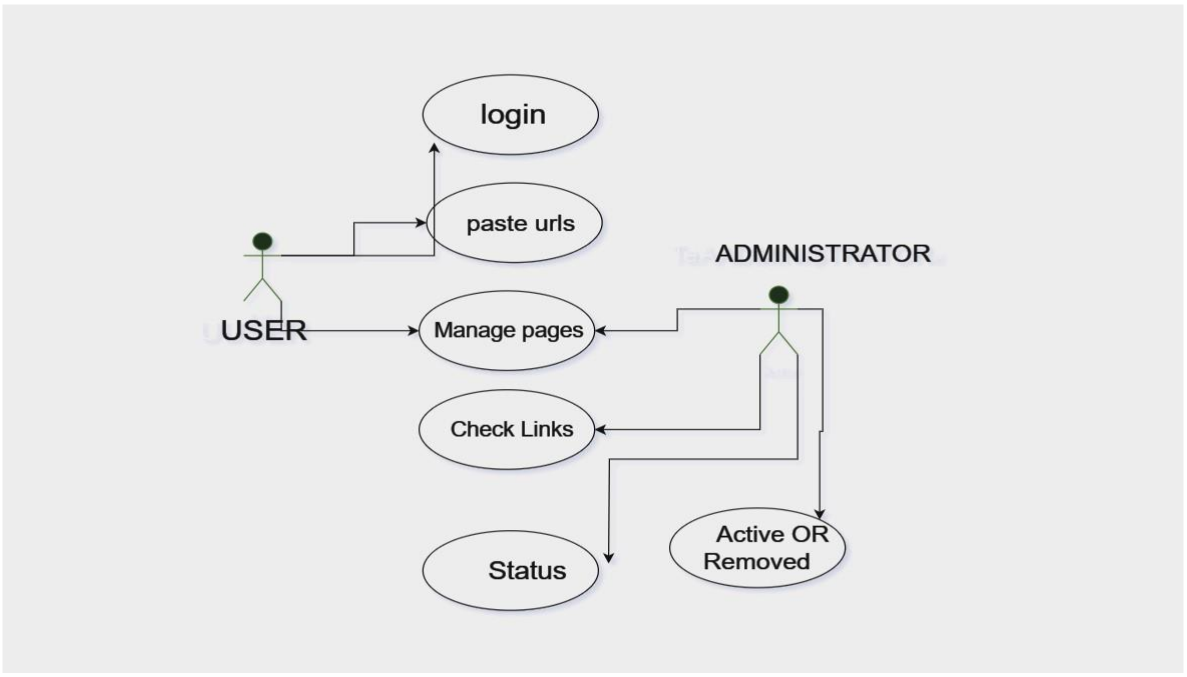


Figure 6.1.1 Use Case Diagram

6.1.2 Class digram

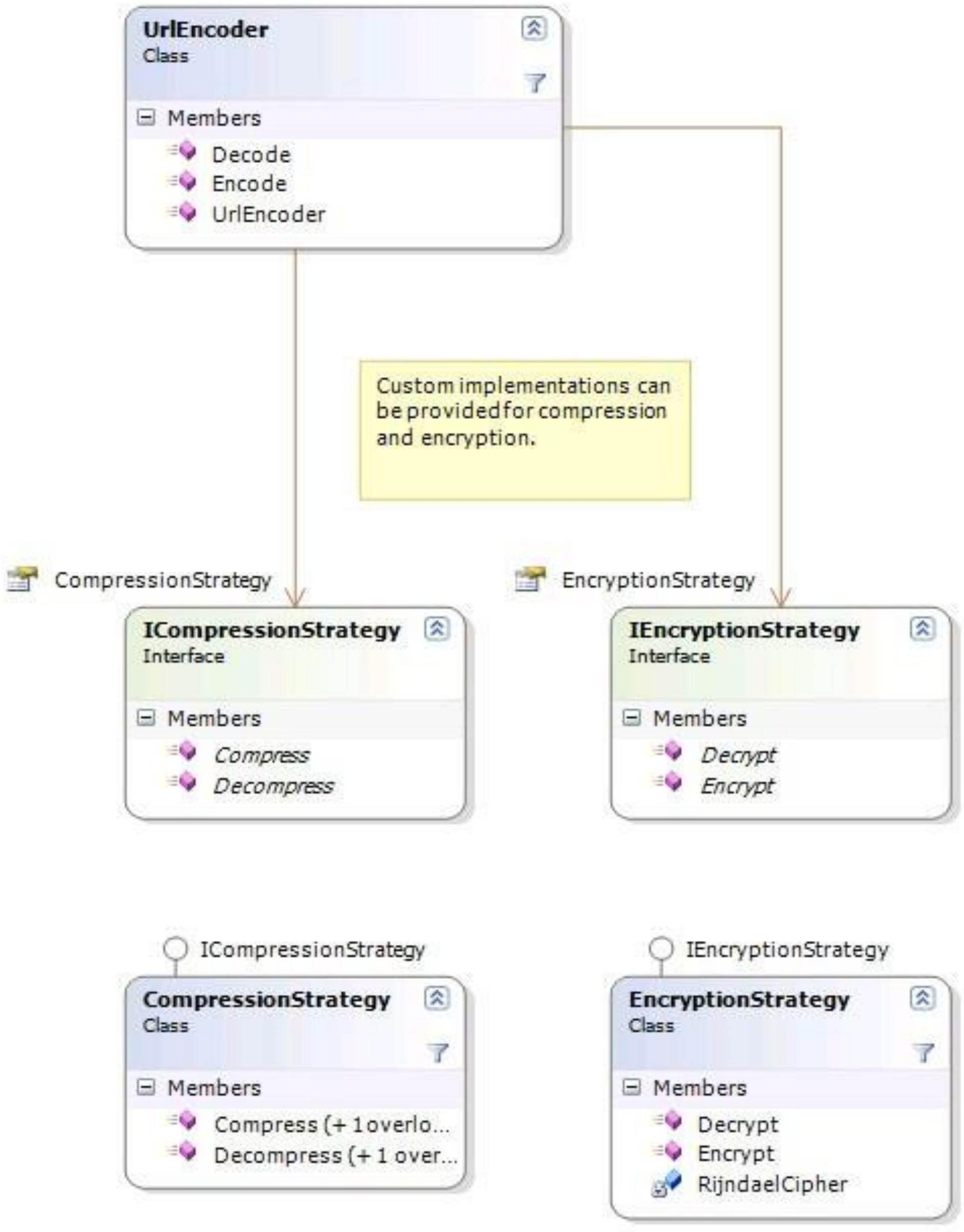


Figure 6.1.2 Class Diagram

6.1.3 Sequence Diagram

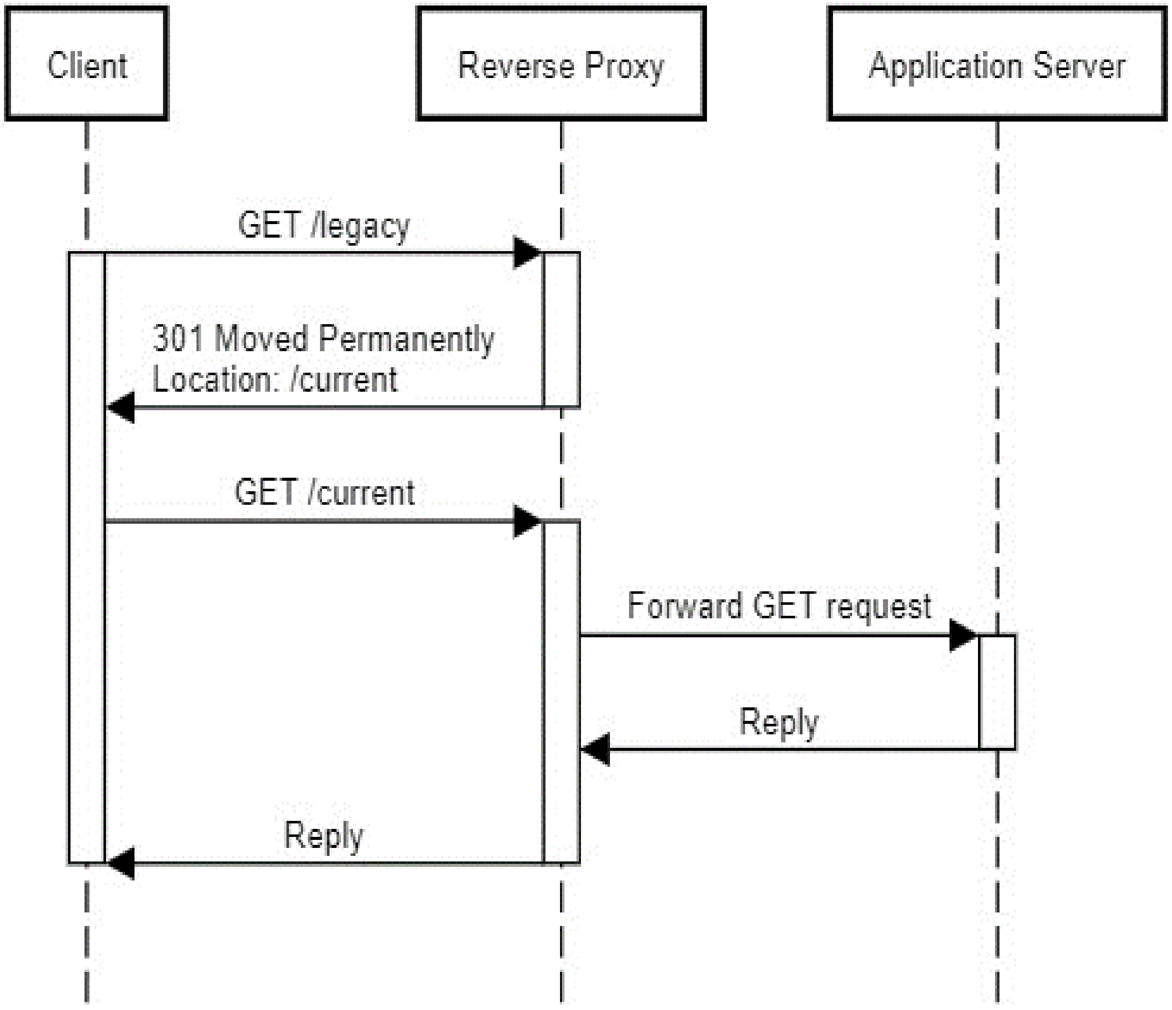


Figure 6.1.7 Sequence Diagram

6.1.4 Activity Diagram

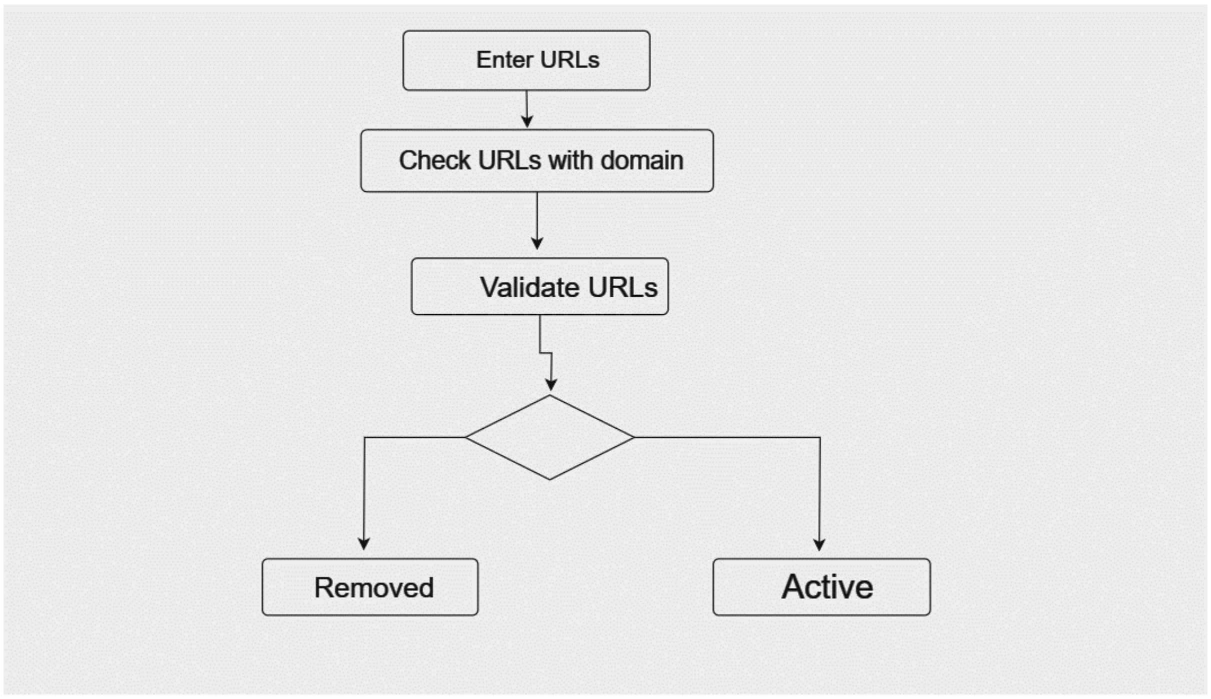


Figure 6.1.4 Activity Diagram

Chapter 7

SYSTEM ARCHITECTURE

7.1 Architectural Diagram

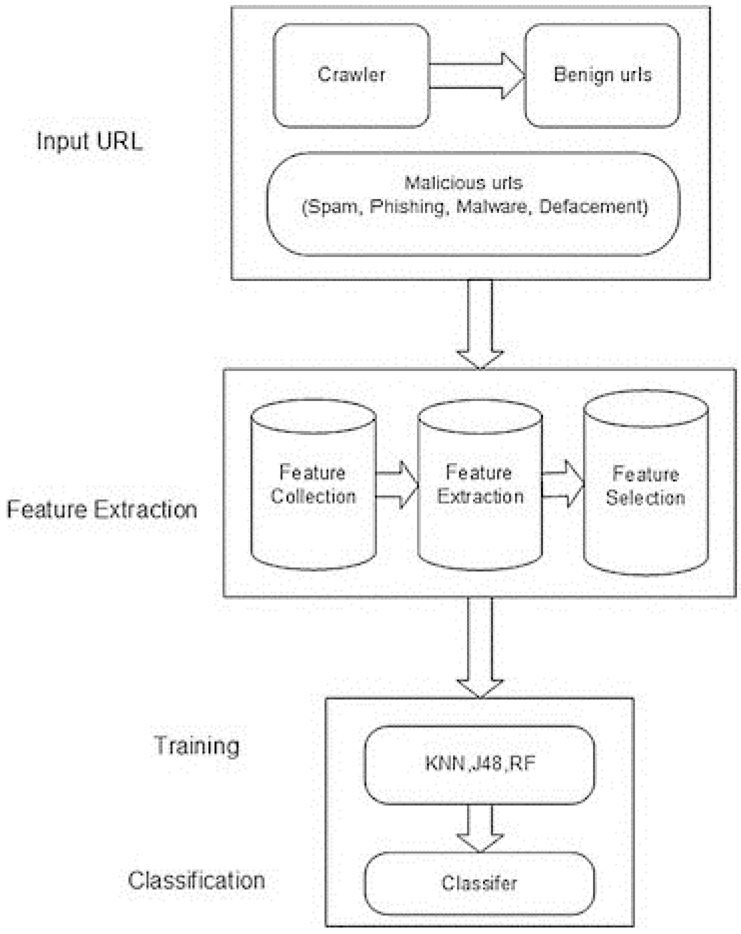


Figure 7.1 Architectural Diagram

7.2 Architectural description

An architectural description, in the context of software and system development, refers to a detailed representation of the system's architecture. It provides an overview of the system's structure, components, interactions, and design decisions. Architectural descriptions are essential for understanding and communicating how a system is organized and how its various parts work together. It outlines the key components or modules of the system, often represented as boxes. These components represent the major functional elements of the system.Architectural descriptions show how components interact with each other through various connections, including communication channels, data flow, and dependencies.They illustrate how data and information flow through the system, from input sources to output destinations, often represented by arrows.

7.3 Dataflow Diagram

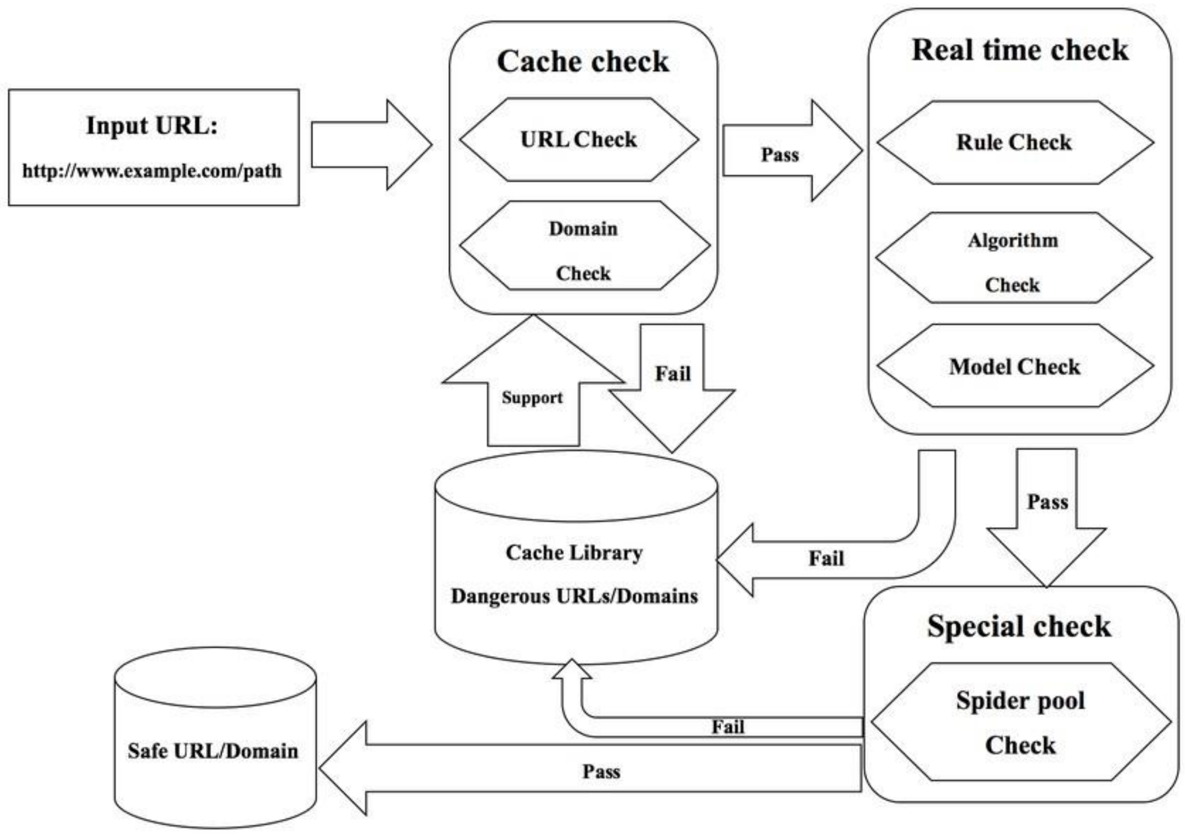


Figure 7.3 Dataflow Diagram

7.4 ER Diagram

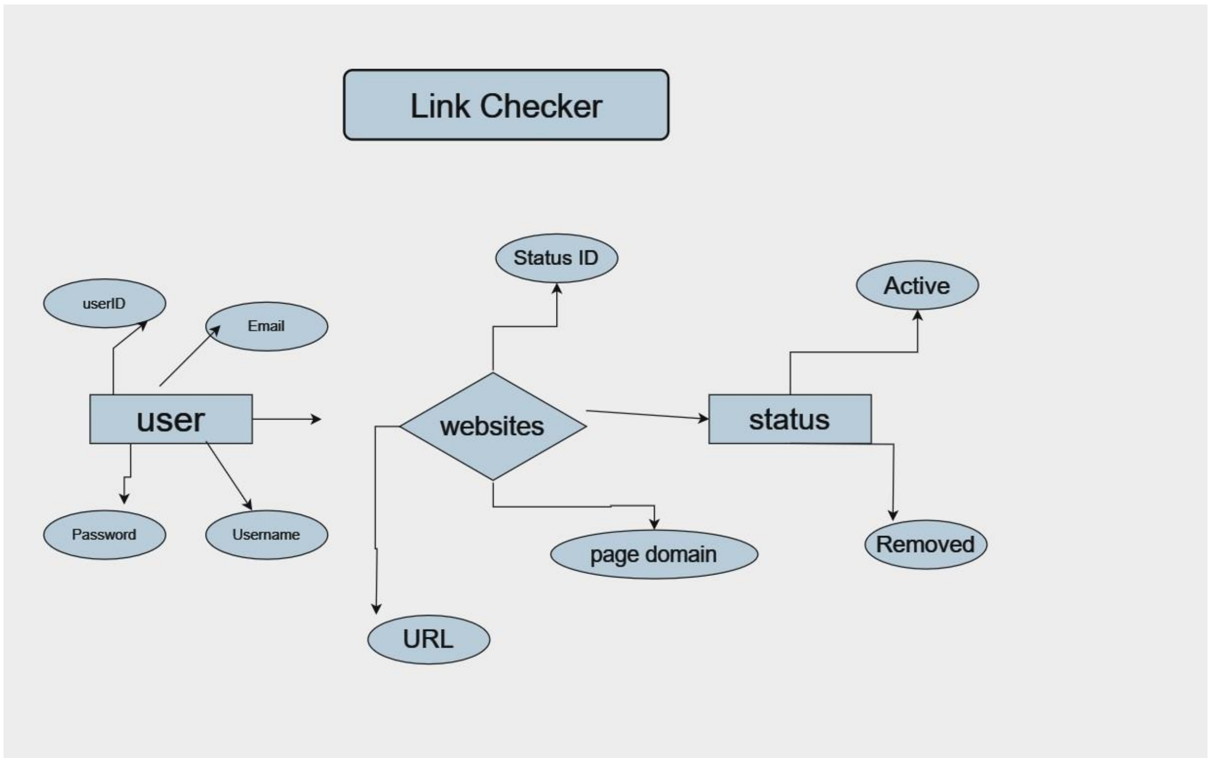


Figure 7.4 ER Diagram

CHAPTER 8

SYSTEM TESTING

8.1 Unit Testing

Unit testing is an important part of the software development process, including when creating a URL checker tool. Unit testing involves testing individual units or components of the software to ensure they work correctly in isolation. In the case of a URL checker tool, these units or components may include functions, classes, or modules responsible for specific tasks, such as parsing URLs, making HTTP requests, or validating URLs.

Identify Units/Components

Start by identifying the individual units or components of your URL checker tool that need to be tested. These could be functions, classes, or modules responsible for tasks like URL validation, HTTP requests, or error handling.

Choose a Testing Framework

Select a testing framework suitable for the programming language you're using. Popular choices include JUnit for Java, pytest for Python, Jasmine for JavaScript, and many others.

Write Test Cases

For each unit or component, write a set of test cases. Test cases should cover various scenarios, including typical cases, edge cases, and error conditions. In the context of a URL checker tool, you might have test cases for valid URLs, invalid URLs, URLs with different schemes (http, https, ftp), URLs with query parameters, and so on.

Setup and Teardown: For each test case, set up the necessary environment, including any required data or objects. After the test is executed, ensure that you clean up any resources that were used during the test. This helps ensure that tests are isolated and don't interfere with each other.

Execute Tests

Run the unit tests using your chosen testing framework. The framework will execute the test cases and report the results, indicating which tests passed and which failed.

Assertion and Validation

In each test case, use assertions to verify that the unit or component behaves as expected. For example, if you're testing a URL validation function, you might assert that it correctly identifies valid URLs as valid and invalid URLs as invalid.

Handling Exceptions

Test how your URL checker tool handles exceptions or error conditions. For example, test how it handles network errors when making HTTP requests or how it handles unexpected input.

8.2 Integration Testing

Integration testing is the phase of software testing where individual units or components are combined and tested as a group. In the context of a URL checker tool, integration testing involves testing how different parts of the tool work together to ensure that the tool functions correctly as a whole. Here's a process for conducting integration testing in a URL checker tool,

Identify Integration Points

Determine the integration points in your URL checker tool. These could include the interaction between components responsible for parsing URLs, making HTTP requests, validating URLs, and handling responses.

Define Test Scenarios

Create test scenarios that simulate the real-world usage of your URL checker tool. These scenarios should cover various use cases, including checking different types of URLs (HTTP, HTTPS, FTP), handling redirects, timeouts, and responses with various status codes.

Setup Test Environment

Prepare the test environment by configuring it to resemble the production environment as closely as possible. This might involve setting up a test server or using mock data for external services.

Execute Test Cases

For each test scenario, execute the integration tests. Ensure that your tool interacts with its components correctly and produces the expected results. This may involve using test data and monitoring how components exchange information.

Check Data Flow

Examine the flow of data between components. Ensure that data is passed correctly and that no information is lost during the process. Check if the tool handles input and output properly.

Test Error Handling

Pay special attention to error handling. Test how your tool deals with exceptions and unexpected situations, such as network errors, malformed URLs, or server issues.

Test External Dependencies

If your URL checker tool relies on external services or APIs, consider using stubs or mocks to simulate the behavior of these external services in a controlled manner. This ensures that the tool can handle different responses from these dependencies.

Test Performance and Scalability

Evaluate the performance of your URL checker tool under different loads. Check how it handles a high volume of URL checks and ensure that it doesn't degrade in performance or become unresponsive.

7.3 Test Cases & Reports

Creating test cases and generating test reports is an essential part of the testing process in a URL checker tool. Test cases define the scenarios that will be tested, and test reports provide a summary of the test results. Here's a step-by-step process for creating test cases and generating test reports in a URL checker tool,

1.Define Test Objectives

Clearly define the objectives of your testing process. What specific aspects of your URL checker tool do you want to test? Common objectives may include URL validation, HTTP request handling, error responses, and performance under load.

2. Identify Test Cases

Identify different categories of test cases, such as positive, negative, boundary, and performance test cases. Positive test cases validate expected behavior, negative test cases test error handling, boundary test cases check edge cases, and performance test cases evaluate system scalability.

3. Create Test Data

Prepare a set of test data for your test cases. This data includes a list of URLs with various characteristics. For example, you may include valid and invalid URLs, URLs with different schemes (http, https, ftp), URLs with query parameters, URLs with different response codes, and URLs with various content types.

4. Write Test Cases

For each category of test cases, write specific test cases that include input data, expected outcomes, and steps to execute the test. Test cases should be comprehensive and cover the intended functionality of your URL checker tool.

5. Test Execution

Execute the test cases against your URL checker tool. Be sure to follow the steps outlined in each test case, providing the specified input data and verifying the actual results against the expected outcomes.

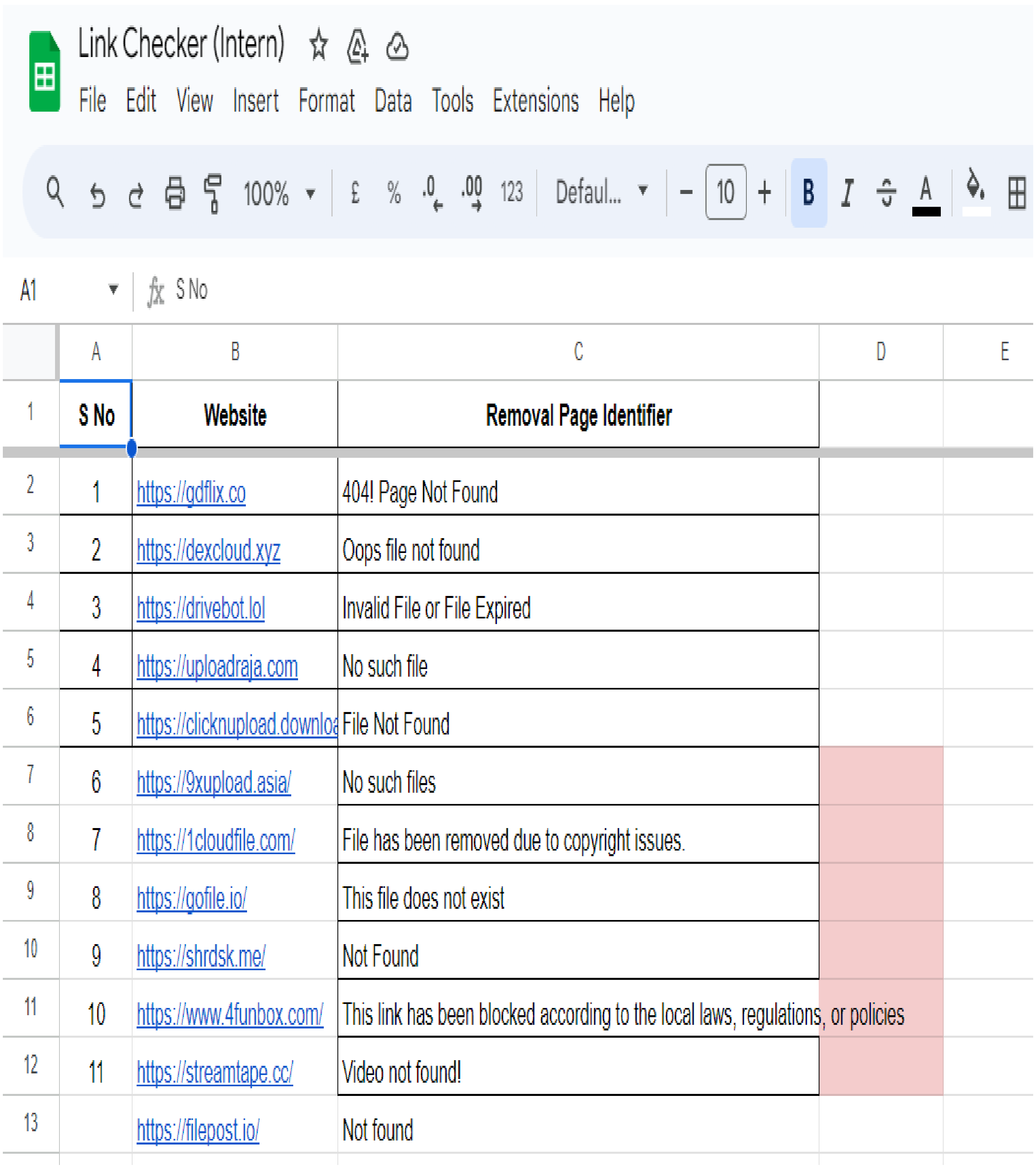
CHAPTER 9

SYSTEM IMPLEMENTATION

9.1 Coding

```
function LinkChecker() {
var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('LinkChecker');
var data = sheet.getRange('B2:B' + sheet.getLastRow()).getValues();
 // Get values from the Assets sheet's range C2:C
  var assetsSheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('Assets');
  var assetsData = assetsSheet.getRange('C2:C' +
assetsSheet.getLastRow()).getValues();
  // Flatten the assetsData array to a single-dimensional array for easier
comparison
 var assetsValues = assetsData.flat();
 for (var i = 0; i < data.length; i++) {
  var url = data[i][0];
 var responseCode, responseBody;
 try {
 var response = UrlFetchApp.fetch(url, { muteHttpExceptions: true });
 responseCode = response.getResponseCode();
 responseBody = response.getContentText();
 } catch (e) {
```

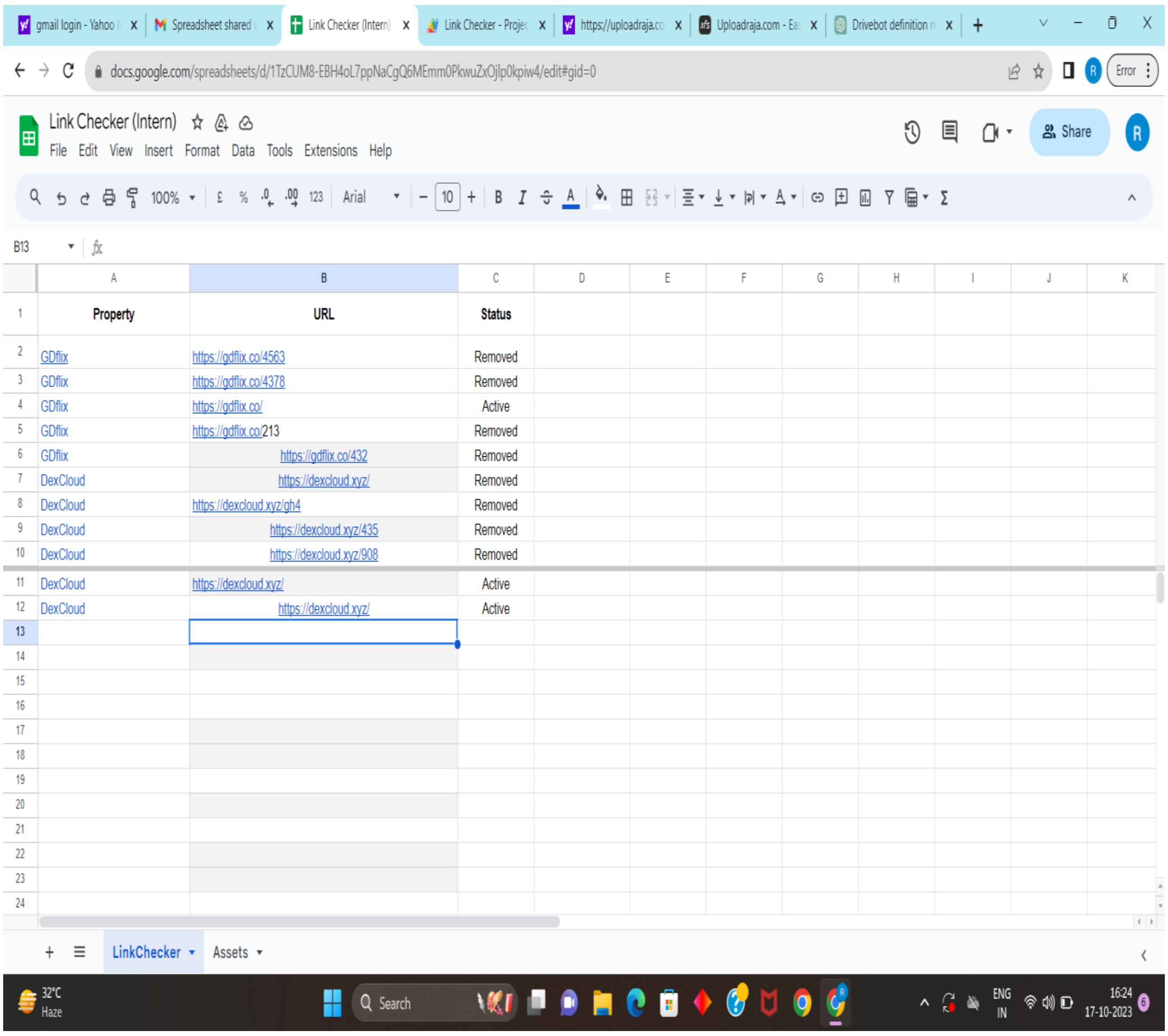
```
responseCode = -1;
 responseBody = "";
 }
  // Check if responseCode is 200 and responseBody contains any full text match
from the assetsValues array
 if (responseCode === 200 && assetsValues.some(value => new RegExp('\\b' +
escapeRegExp(value) + '\\b').test(responseBody))) {
 sheet.getRange('C' + (i + 2)).setValue('Removed');
 } else {
 sheet.getRange('C' + (i + 2)).setValue('Active');
}
}
}
// Function to escape special characters in a string to use in a regular expression
function escapeRegExp(str) {
return str.replace(/[.*+?^${}()|[\]\\]/g, '\\$&'); }
```



S No	Website	Removal Page Identifier
1	https://gdflix.co	404! Page Not Found
2	https://dexcloud.xyz	Oops file not found
3	https://drivebot.lol	Invalid File or File Expired
4	https://uploadraja.com	No such file
5	https://clicknupload.downlo	File Not Found
6	https://9xupload.asia/	No such files
7	https://1cloudfile.com/	File has been removed due to copyright issues.
8	https://gofile.io/	This file does not exist
9	https://shrdsk.me/	Not Found
10	https://www.4funbox.com/	This link has been blocked according to the local laws, regulations, or policies
11	https://streamtape.cc/	Video not found!
	https://filepost.io/	Not found

CHATPER 10

Screenshots



Property	URL	Status
GDflix	https://gdflix.co/4563	Removed
GDflix	https://gdflix.co/4378	Removed
GDflix	https://gdflix.co/	Active
GDflix	https://gdflix.co/213	Removed
GDflix	https://gdflix.co/432	Removed
DexCloud	https://dexcloud.xyz/	Removed
DexCloud	https://dexcloud.xyz/gh4	Removed
DexCloud	https://dexcloud.xyz/435	Removed
DexCloud	https://dexcloud.xyz/908	Removed
DexCloud	https://dexcloud.xyz/	Active
DexCloud	https://dexcloud.xyz/	Active

docs.google.com/spreadsheets/d/1TzCUM8-EBH4oL7ppNaCgQ6MEmm0PkwuZxOjlp0kpiw4/edit#gid=0
Link Checker (Intern)
File Edit View Insert Format Data Tools Extensions Help
Share
LinkChecker
Assets
32°C Haze
17-10-2023 16:24

CHAPTER 11

CONCLUSION

A conclusion for a URL checker project summarizes the key findings, outcomes, and the overall experience of developing and testing the tool. The URL checker tool has been successfully designed to meet its core objectives. It effectively validates URLs, handles HTTP requests, and reports on the status of the checked URLs. The tool has shown robust performance in checking a wide range of URLs, including those with varying schemes, query parameters, and response codes.Extensive testing has demonstrated that the URL checker tool provides accurate results. It correctly identifies valid and invalid URLs, handles redirects, and appropriately responds to different status codes. The error handling mechanism ensures that the tool gracefully manages unexpected situations.

The project has offered valuable lessons in software development, testing, and collaboration. Effective teamwork, robust testing strategies, and meticulous documentation have been crucial in achieving the project's goals.The URL checker tool has the potential for broader applications, including integration into web applications, browser extensions, or as part of a larger web monitoring system. Future work may involve extending its capabilities, such as supporting additional URL schemes and handling more advanced use cases.

In conclusion, the URL checker project has been a success, delivering a reliable and valuable tool for checking the integrity and validity of URLs. It showcases the importance of rigorous development and testing practices in producing high-quality software. With a commitment to continuous improvement, this tool holds promise for broader applications and will remain a valuable asset in the domain of web development and monitoring.

11.1 FUTURE ENHANCEMENT

Enhancing a URL checker tool can improve its functionality, usability, and relevance. Here are some potential future enhancements for a URL checker tool,

Support for More URL Schemes

Expand the tool's capability to handle a wider range of URL schemes beyond HTTP, HTTPS, and FTP. Consider supporting less common schemes such as mailto, tel, file, and custom protocols.

Regular Expression Pattern Matching

Integrate regular expression support for advanced URL pattern matching. This allows users to define custom URL validation rules based on their specific requirements.

Bulk URL Import

Enable users to upload or paste a list of URLs in various formats (e.g., CSV, text, Excel) for batch processing. This feature simplifies the checking of multiple URLs at once.

Browser Extension

Develop browser extensions or plugins that integrate the URL checker tool directly into web browsers. Users can check URLs on the fly while browsing the web.

Scheduled URL Checks

Implement a scheduling feature that allows users to set up regular, automated checks for specific URLs or sets of URLs. This is particularly useful for monitoring changes or uptime status.

Historical Data Tracking

Add the ability to save and display historical data for checked URLs. Users can track URL status changes over time, helping them identify trends and potential issues.

Customizable Notifications

Allow users to set up notifications (e.g., email alerts) for specific events, such as a URL status change from "up" to "down."

Security Scanning

Integrate security checks to identify potential vulnerabilities in URLs, such as those susceptible to cross-site scripting (XSS) or SQL injection attacks.

API Integration

Provide a RESTful API that allows other applications to interact with the URL checker tool programmatically. This enables seamless integration with existing systems.

Performance Analytics

Offer performance metrics and analytics, including response time monitoring and load time tracking for checked URLs.

Geographic Testing

Enable testing from different geographic locations or data centers to determine if URLs behave differently based on the user's location.

User Account Management

Create user accounts with various roles and permissions to facilitate collaboration within teams and provide access control.

Report Generation

Improve the reporting functionality with customizable report templates, export options (e.g., PDF, CSV), and the ability to schedule and share reports automatically.

Accessibility Checker

Add accessibility checking features to ensure that websites comply with accessibility standards (e.g., WCAG) and are usable by people with disabilities.

Data Export and Import

Support data export and import for easy backup, migration, and sharing of URL check configurations and results.

Mobile Application

Develop a mobile app version of the URL checker tool to provide users with the ability to check URLs on the go.

Content Validation

Extend the tool to validate not only the URL but also the content of the web page, checking for specific keywords, phrases, or elements.

CHAPTER 12

REFERENCES

1. Goodenough, J. B., & Park, K. S. (2013). "The Li-Ion Rechargeable Battery: A Perspective."

2. Manthiram, A., et al. (2017). "A Perspective on the High-Voltage LiMn1.5Ni0.5O4 Spinel Cathode for Li-Ion Batteries."

3. Plett, G. L. (2004). "Battery Management Systems, Part I: Battery Modeling."

4. Erol-Kantarci, M., & Mouftah, H. T. (2013). "Towards Energy-Efficient Battery Management for Smart Grids."

5.Whittingham, M. S. (1976). "Electrical Energy Storage and Intercalation Chemistry." Science, 192(4244), 1126-1127.

6. Armand, M., & Tarascon, J. M. (2008). "Building Better Batteries." Nature, 451(7179), 652- 657.

7. Tarascon, J. M., & Armand, M. (2001). "Issues and Challenges Facing Rechargeable Lithium Batteries." Nature, 414(6861), 359-367.