IBM Project Report

On

Travel rate Scraping

Developed by:

Hem Kamli (20162101005)

Guided By:-

Prof. Neha Rajput

Mr. Harsh Bantuya

Submitted to

Department of Computer Science & Engineering Institute of Computer Technology

Year: 2024

ACKNOWLEDGEMENT

IBM project is a golden opportunity for learning and self-development. I consider myself very lucky and honored to have so many wonderful people lead me through in completion of this project. First and foremost, I would like to thank Dr. Rohit Patel, Principal, ICT, and Prof. Dharmesh Darji, Head, ICT who gave us an opportunity to undertake this project. My grateful thanks to Prof. Ravi Patel & Mr.Palwinder S (Internal & External Guides) for their guidance in project work Online Blockchain based certificate generation and validation system for government organization, who despite being extraordinarily busy with academics, took time out to hear, guide and keep us on the correct path. We do not know where would have been without his/her help. CSE department monitored our progress and arranged all facilities to make life easier. We choose this moment to acknowledge their contribution gratefully.

Hem Kamli (20162101005)

ABSTRACT

The travel rate scraping project proposes an automated system to streamline flight data collection from various B2B and B2C websites into a centralized database. It aims to offer users comprehensive flight information, including the cheapest fares and all available options for specific routes, accessible through an intuitive dashboard. Future work includes expanding data sources, integrating advanced data analysis techniques, and incorporating user preferences for personalized recommendations. Real-time updates and mobile application development are suggested to enhance accessibility, while partnerships with travel agencies and airlines can improve service quality and access to exclusive deals. User feedback mechanisms will drive iterative improvements, ensuring the system remains responsive to evolving user needs and industry trends. Through these enhancements, the system can evolve into a robust platform, providing travelers with efficient and informed flight booking solutions.

INDEX

CHAPTER 1: INTRODUCTION

CHAPTER: 1 INTRODUCTION

The flight data scraping project introduces an innovative solution for efficiently gathering flight information from various online platforms, both business-to-business (B2B) and business-to-consumer (B2C). The goal is to centralize this data into a single database, providing users with a comprehensive view of available flights, including pricing details and route options.

The project's primary objective is to simplify the process of finding the best flight deals by presenting all relevant information in an easy-to-use dashboard. By automating data collection and aggregation, users can quickly compare fares and make informed decisions.

Future developments for the project include expanding the sources of data to ensure coverage of a wider range of airlines and travel agencies. Additionally, plans involve incorporating advanced data analysis techniques to offer users personalized recommendations based on their preferences and travel history.

To enhance accessibility, the project proposes the development of a mobile application, providing real-time updates and allowing users to access flight information on the go. Furthermore, partnerships with travel agencies and airlines are envisaged to improve service quality and provide users with access to exclusive deals and promotions.

Continuous feedback mechanisms will be implemented to gather user input and refine the system based on evolving needs and industry trends. This iterative approach ensures that the system remains responsive and adaptable to changing requirements.

Overall, the project aims to evolve into a robust platform that offers travelers efficient and informed solutions for booking flights. By leveraging technology and user feedback, the system will continue to enhance the travel booking experience, making it easier and more convenient for users to find the best flight options available.

CHAPTER 2: OBJECTIVE

CHAPTER 2: OBJECTIVE

2.1 Primary Objective

The primary objective of this project is to develop a travel rate scraping system that collects flight data from diverse sources and stores it into a database. Specifically, the objectives include:

- Implementing web scraping algorithms to extract flight information from B2B and B2C websites.
- Designing a database schema to efficiently store and manage the scraped data.
- Developing a user-friendly dashboard interface to display the cheapest flights and all available options for a given route.
- Implementing algorithms to analyze and compare flight prices, enabling users to make informed decisions.
- Ensuring the scalability and reliability of the system to accommodate future expansions and updates.

2.2 Literature review

Previous research in the field of travel technology has explored various approaches to automate the process of collecting and analyzing flight data. Web scraping techniques have been widely adopted to extract information from websites efficiently. Additionally, database management systems have been utilized to store and organize large volumes of data collected from disparate sources. Several studies have also focused on developing user interfaces that provide intuitive ways for travelers to search and compare flight options. By building upon existing methodologies and technologies, this project aims to contribute to the advancement of travel technology by providing a comprehensive and user-centric solution for accessing flight information.

CHAPTER 3: METHODOLOGY

CHAPTER 3: METHODOLOGY

3.1 Website selection

Flight data will be scraped from a diverse selection of B2B and B2C travel websites, considering factors like popularity, reliability, and relevance to users. Websites with large user bases, trusted data accuracy, and catering to various traveler preferences will be prioritized for data extraction.

3.2 Technology selection

Choose appropriate technologies for web scraping, database management, and frontend development.

Node.js with Crawlee (Playwright) is selected for web scraping due to its robustness and flexibility.

MySQL is chosen for database management due to its reliability and scalability.

Next.js is selected for building the frontend dashboard due to its efficiency in creating interactive and responsive web applications.

3.3 Web scraping

Utilizing Crawlee (Playwright), a Node.js library, for web scraping offers several advantages. Playwright's comprehensive API allows for seamless automation of web interactions, enabling the scraping of flight information such as prices, departure times, airlines, and available routes from selected websites.

Developing scraping scripts with error handling mechanisms is essential to ensure the reliability and robustness of the scraping process. Error handling mechanisms should address potential issues such as website changes, page loading errors, and data inconsistencies to ensure smooth operation and accurate data extraction.

By implementing these strategies, the project can effectively scrape flight data from a variety of B2B and B2C travel websites, providing users with accurate and up-to-date information to facilitate informed decision-making.

3.4 Database management

Creating a MySQL database schema to store scraped flight data involves designing tables to organize and manage the information effectively. Key tables may include those for flights, airlines, routes, and additional relevant data.

For the flights table, attributes such as flight number, departure and arrival airports, departure and arrival times, and ticket prices can be included. Additionally, the airlines table can store information about different airlines, such as their names and contact details.

Routes can be stored in a separate table, containing details about specific flight paths, including departure and arrival airports and distances. Other relevant information, such as aircraft types or departure terminals, can be stored in additional tables as needed.

Utilizing the Node.js MySQL library, a connection with the database can be established to execute SQL queries for tasks such as data insertion, retrieval, and management. This allows for seamless integration between the backend and the database, enabling efficient handling of scraped flight data.

Implementing data validation and normalization techniques is crucial to maintain data integrity and consistency within the database. This involves validating incoming data to ensure it meets predefined criteria and normalizing the data structure to eliminate redundancy and improve efficiency.

3.5 Frontend development

Develop the frontend dashboard using Next.js, a React framework for server-side rendering and client-side routing.

Design intuitive user interfaces with responsive layouts and interactive components for displaying flight information.

Implement features such as search functionality, sorting options, and filters to facilitate user navigation and exploration of flight data.

CHAPTER 4: SOFTWARE AND HARDWARE REQUIREMENTS

CHAPTER 4: SOFTWARE AND HARDWARE REQUIREMENTS

4.1 Hardware and Software Requirements:

4.1.1 Hardware Requirements:

Processor: Intel Core i5 or equivalent processor (or higher) for optimal performance.

RAM: Minimum 8GB RAM for smooth operation of development tools and database management.

Storage: At least 256GB SSD for storing project files, databases, and related data.

Network: Stable internet connection for web scraping tasks and accessing online resources.

4.1.2. Software Requirements:

Operating System: Any modern operating system supported by Node.js, such as Windows, macOS, or Linux.

- Development Environment:

Node.js: Latest stable version of Node.js installed for running JavaScript-based applications.

Visual Studio Code or any preferred code editor for writing and editing project code.

- Web Scraping Tools:

Crawlee with Playwright: Install Crawlee library along with Playwright for web scraping tasks.

- Database Management System:

MySQL: Latest version of MySQL Community Server for storing and managing scraped flight data.

- Frontend Development Framework:

Next.js: Install Next.js framework for building the frontend dashboard interface.

React.js: Next.js is built on top of React.js, so ensure React.js is also installed.

- Backend Framework:

Express.js: Install Express.js for creating API endpoints to interact with the MySQL database.

- Dependency Management:

npm (Node Package Manager): Used for installing and managing project dependencies.

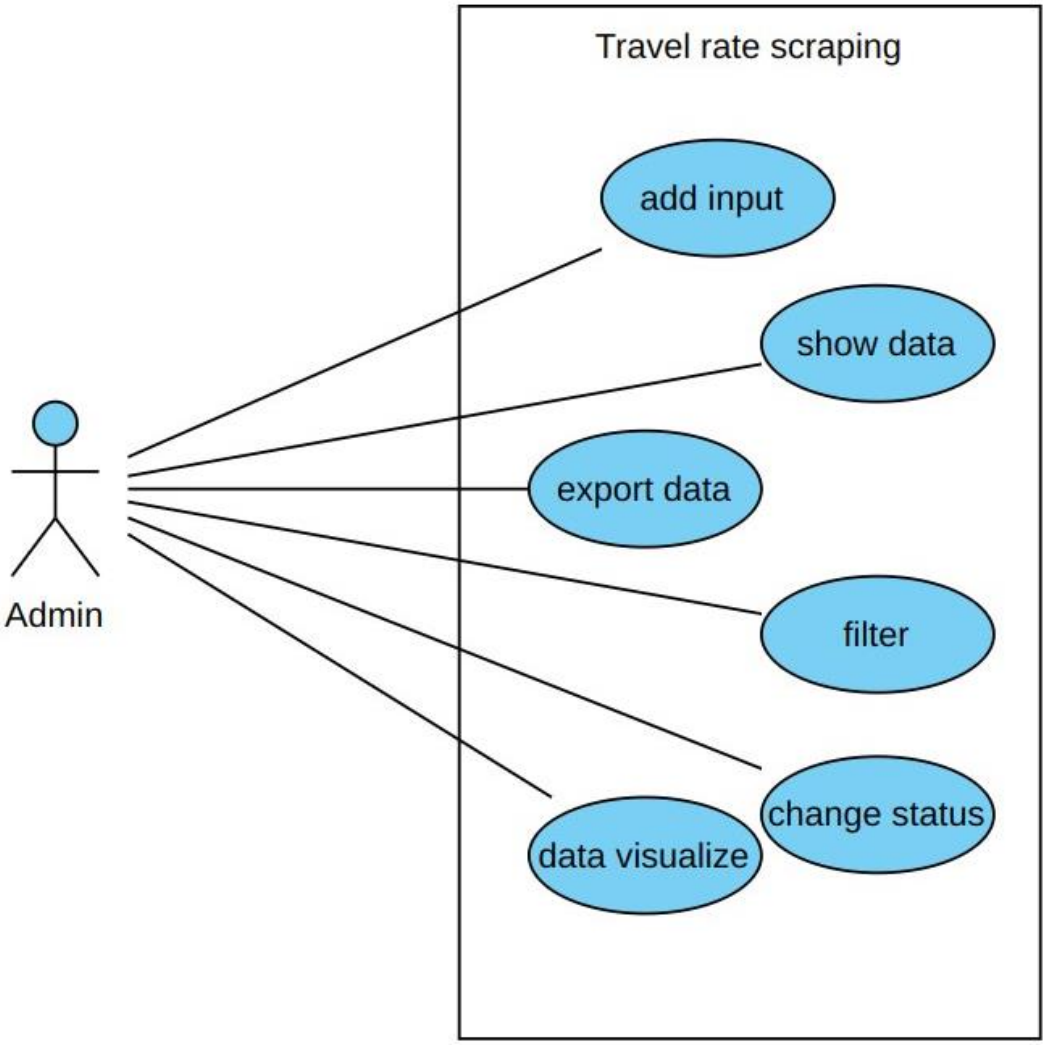
- Version Control:

Git: Version control system for tracking changes to project code and collaborating with team members.

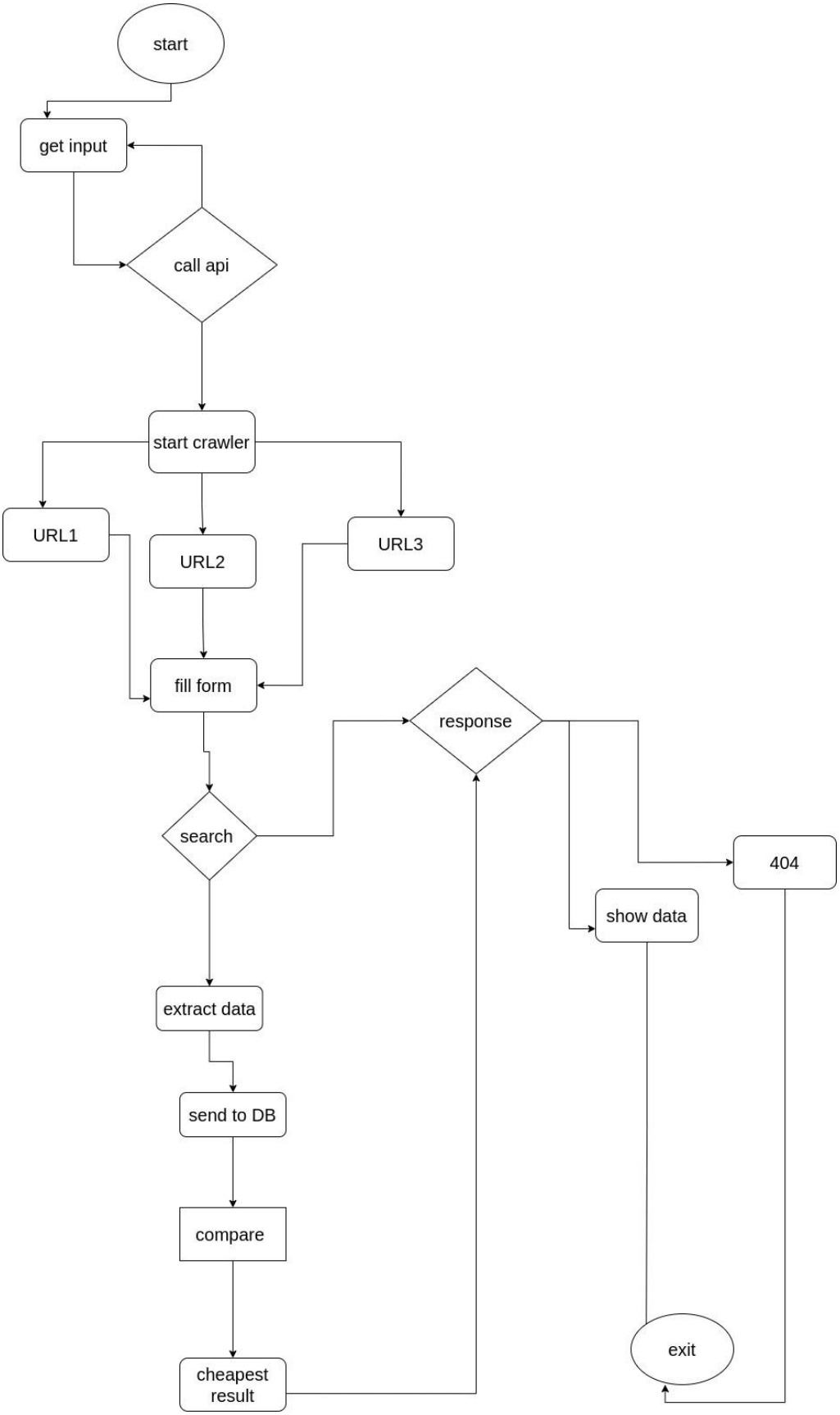
GitHub or GitLab: Online platforms for hosting project repositories and facilitating collaborative development.

CHAPTER 5: IMPLEMENTATION DETAILS

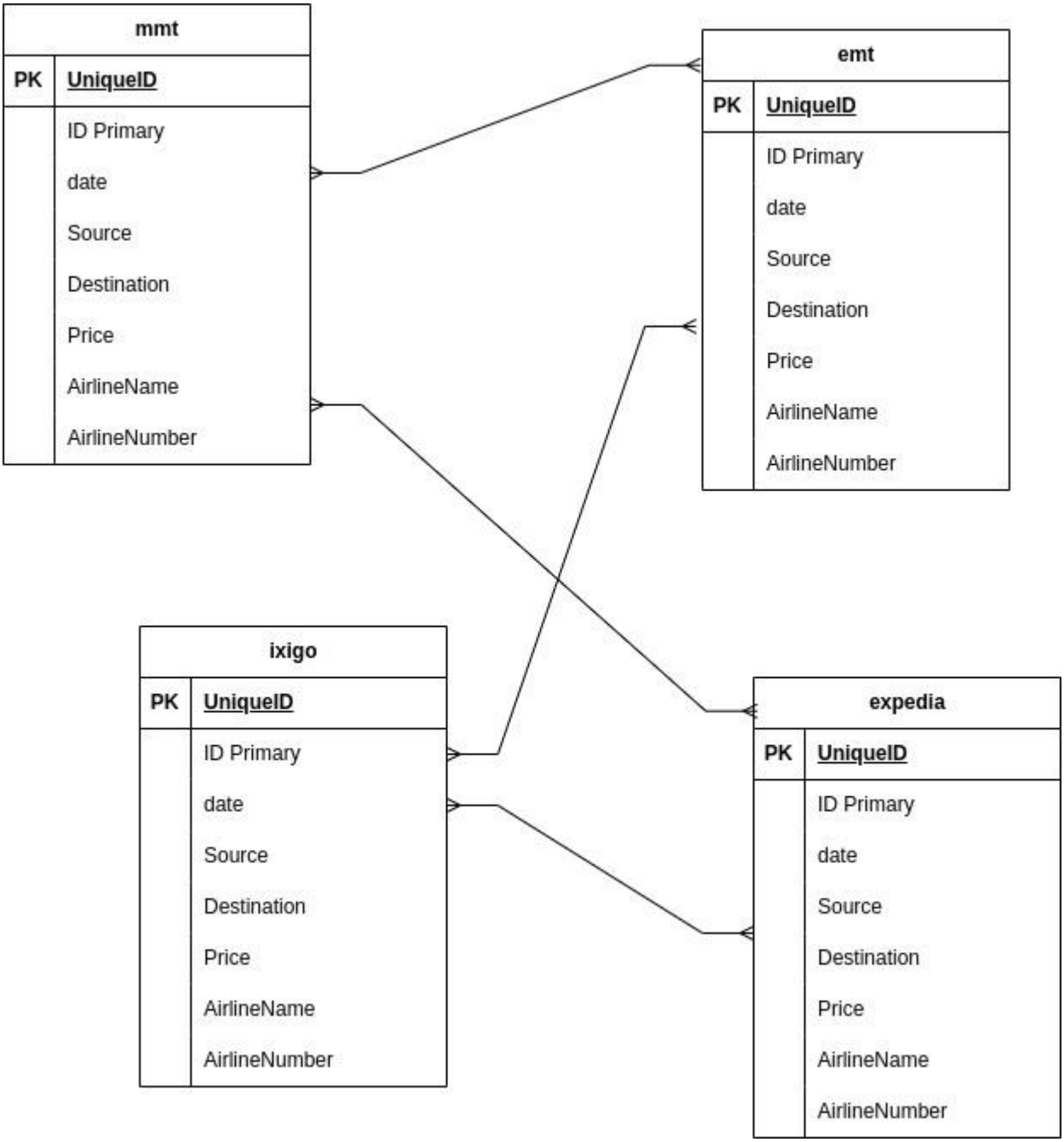
5.1 Use case


Travel rate scraping
add input
show data
export data
filter
change status
data visualize
Admin

5.2 Flowchart



5.3 ER Diagram



5.4 Web scraping

Node.js with Crawlee (Playwright): Utilized Node.js with Crawlee, a web scraping library built on top of Playwright, to automate the extraction of flight data from various B2B and B2C travel websites.

Playwright Crawler: Developed custom crawlers using Playwright to navigate through web pages, interact with elements, and extract flight information such as airline names, flight numbers, prices, and source-destination pairs.

Scraping Multiple Websites: Implemented scraping scripts for multiple websites to collect comprehensive flight data. Each script was tailored to the structure and layout of the respective website, ensuring accurate extraction of relevant information.

Integration of Playwright and Crawlee: Leveraged Playwright's browser automation capabilities within Crawlee's framework to orchestrate scraping tasks efficiently. Crawlee abstracted away the complexities of Playwright, providing a higher-level interface for defining scraping logic and managing concurrency.

How Playwright and Crawlee Work Together: Integrated Playwright's browser automation capabilities within Crawlee's framework to automate browser interactions, navigation, and data extraction efficiently. Crawlee abstracted away the complexities of Playwright, providing a higher-level interface for defining scraping tasks and managing concurrency.

5.5 Database management

MySQL for Data Storage: Employed MySQL as the relational database management system (RDBMS) for storing the scraped flight data. Designed a database schema comprising tables for flights, airlines, routes, and pricing information to organize the data efficiently.

Express App for API Endpoint: Implemented an Express.js application to serve as an API endpoint for interacting with the MySQL database. Defined RESTful routes and endpoints to handle CRUD operations for retrieving, updating, and deleting flight data.

Comparison of Prices: Developed algorithms to compare prices obtained from different airlines across various websites. Utilized unique airline numbers as identifiers to match and compare prices with corresponding entries in the database tables.

5.6 JSON Date handling

Storage in JSON Files: Stored JSON data representing request bodies in files within the project directory. These JSON files contained structured data representing the parameters required for initiating scraping tasks.

Utilization in Crawlee: Loaded the JSON data from files and passed them as input parameters to Crawlee's scraping tasks. This facilitated the customization and configuration of scraping tasks based on specific requirements and user preferences.

5.7 Error handling and scalability

Error Handling: Implemented robust error handling mechanisms within the web scraping scripts to gracefully handle exceptions, such as network errors, page timeouts, and unexpected changes in website structure. Employed retry strategies and logging functionalities to capture and manage errors effectively, ensuring the reliability and resilience of the scraping process.

Scalability: Designed the architecture of the web scraping system to be scalable, capable of handling large volumes of data and concurrent scraping tasks. Utilized asynchronous programming techniques in Node.js to optimize performance and resource utilization, enabling the system to scale horizontally by adding more scraping instances or vertically by leveraging powerful hardware resources. Employed load balancing and distributed processing strategies to distribute scraping tasks across multiple servers or cloud instances, further enhancing the system's scalability and responsiveness to varying workloads.

CHAPTER 6: CONCLUSION AND FUTURE WORK

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 Conclusion

In conclusion, the flight data scraping project presents a promising solution for streamlining the process of gathering and analyzing flight information. By centralizing data from various online platforms, the project aims to provide users with a convenient and comprehensive view of available flights, including pricing details and route options.

The primary objective of simplifying the search for the best flight deals through an easy-to-use dashboard aligns with the needs of modern travelers. Future developments, such as expanding data sources and incorporating advanced analysis techniques, promise to further enhance the system's capabilities and user experience.

The proposed mobile application and partnerships with travel agencies and airlines signify a commitment to improving accessibility and service quality. Continuous feedback mechanisms will ensure that the system remains responsive to evolving user needs and industry trends.

Overall, the project endeavors to evolve into a robust platform that empowers travelers with efficient and informed solutions for booking flights. By leveraging technology and building upon existing methodologies, the project contributes to the advancement of travel technology and enhances the travel booking experience for users.

6.2 Future work

Future work for the travel rate scraping project includes expanding data sources, integrating advanced data analysis techniques, and incorporating user preferences for personalized recommendations. Real-time updates and mobile application development are suggested to enhance accessibility, while partnerships with travel agencies and airlines can improve service quality and access to exclusive deals. User feedback mechanisms will drive iterative improvements, ensuring the system remains responsive to evolving user needs and industry trends. Through these enhancements, the system can evolve into a robust platform, providing travelers with efficient and informed flight booking solutions.