Industrial Internship Report on

" URL Shortener"

Prepared by

[Manchala SriNithin]

Executive Summary

This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner UniConverge Technologies Pvt Ltd (UCT).

This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 6 weeks' time.

My project was "The URL shortener" project is a web application built to simplify long URL links by generating shorter, more manageable URLs. It features an intuitive user interface allowing users to input lengthy URLs and receive corresponding shortened versions. Utilizing Flask for backend development and HTML, CSS, and Bootstrap for frontend styling, the application securely stores URL mappings in an SQLite database. Users can easily access original URLs by navigating through shortened links, enhancing convenience and usability across various online platforms.

This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solution for that. It was an overall great experience to have this internship.

TABLE OF CONTENTS

1 Preface

In a span of six weeks, I participated in an internship program offered by USC/UCT IoT Academy in collaboration with Upskill Campus and Uni Converge Technologies. This journey was marked by exploration, learning, and professional growth as I delved into real world projects, honed my technical skills, and gained insights into the evolving tech landscape.

Weekly Overview:

Week 1: I began by understanding the problem statement and program objectives, setting the stage for subsequent weeks.

Week 2: Following project instructions, I formulated a plan to address challenges, focusing on brainstorming solutions and outlining strategies.

Weeks 34: With a solid plan, I immersed myself in implementation, leveraging technical skills to make steady progress.

Week 5: Validating my implementation became a priority, involving rigorous testing and finetuning for optimal performance.

Week 6: I submitted the project report and underwent verification, reflecting on my accomplishments and growth throughout the internship.

About the Project:

The project focused on developing a userfriendly URL shortener web application, streamlining the process of generating and accessing shortened URLs.

Acknowledgments:

I extend gratitude to mentors and program organizers for their guidance and support, shaping my learning experience significantly.

Message to Juniors and Peers:

Embrace opportunities for growth, approach challenges with resilience, and never stop pursuing your passions.

Overall Experience:

This internship has been transformative, fostering personal and professional growth while equipping me with practical skills for future endeavors

2 Introduction

2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various **Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.

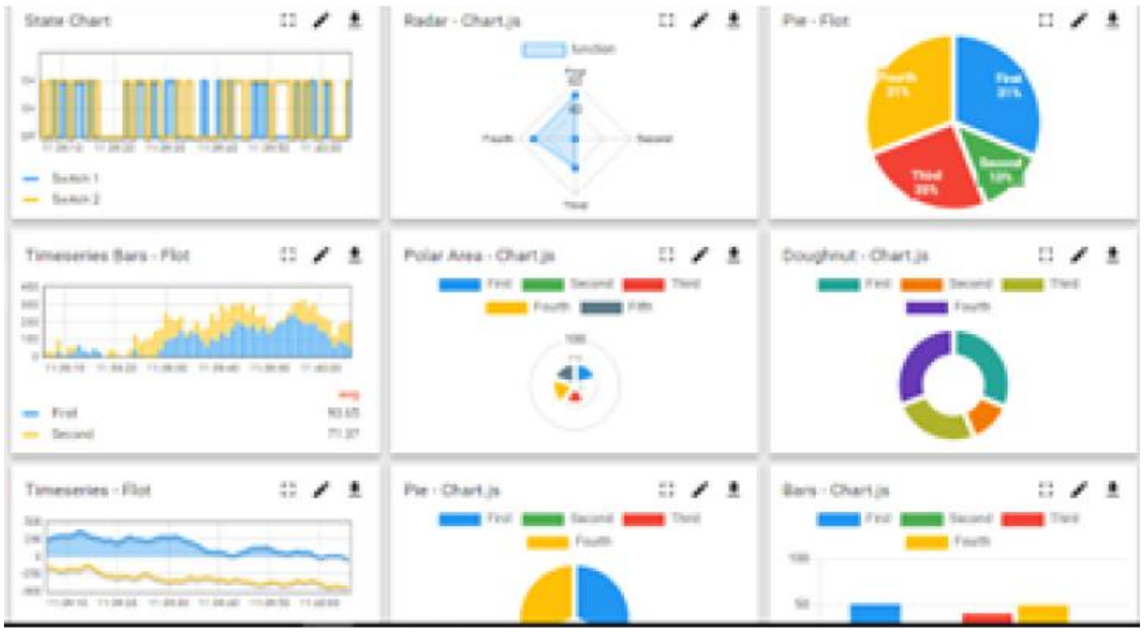
uct
Uniconverge Technologies
IIOT Products
We offer product ranging from Remote IOs, Wireless IOs, LoRaWAN Sensor Nodes/ Gateways, Signal converter and IoT gateways
IIOT Solutions
We offer solutions like OEE, Predictive Maintenance, LoRaWAN based Remote Monitoring, IoT Platform, Business Intelligence…
OEM Services
We offer solutions ranging from product design to final production we handle everything for you..

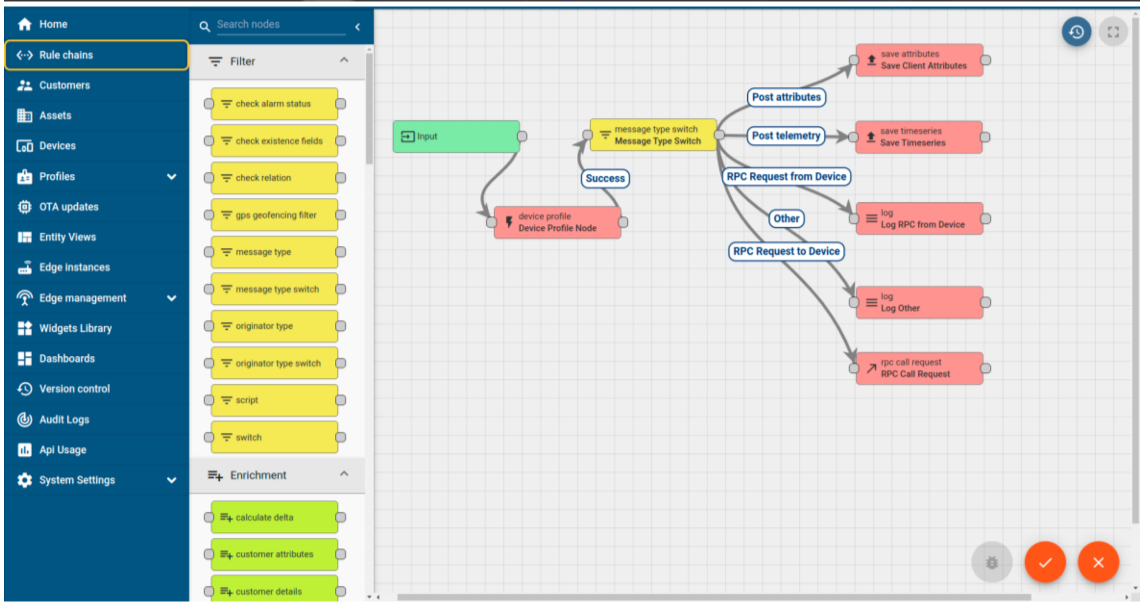
i. UCT IoT Platform (uct Insight)

UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- It enables device connectivity via industry standard IoT protocols MQTT, CoAP, HTTP, Modbus TCP, OPC UA
- It supports both cloud and onpremises deployments.

It has features to
• Build Your own dashboard
• Analytics and Reporting
• Alert and Notification
• Integration with third party application(Power BI, SAP, ERP)
• Rule Engine





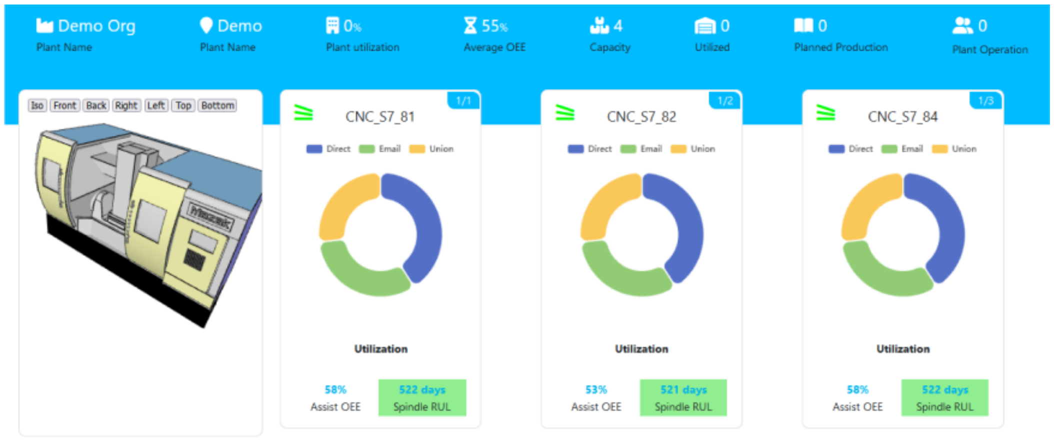
ii. Smart Factory Platform (FACTORY WATCH)

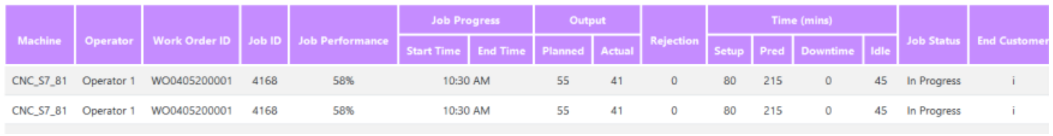
Factory watch is a platform for smart factory needs.

It provides Users/ Factory

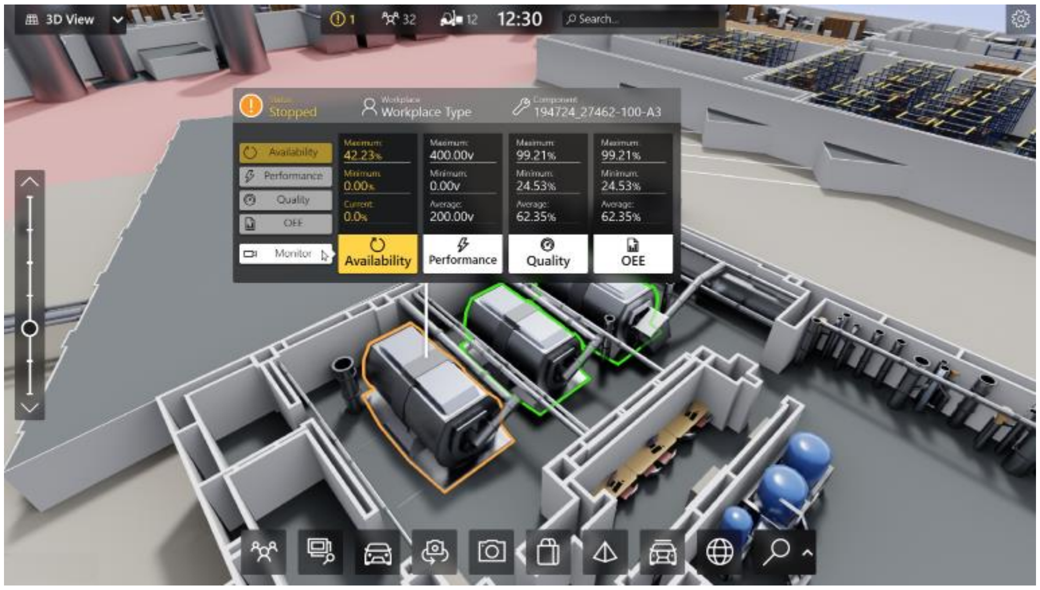
- with a scalable solution for their Production and asset monitoring
- OEE and predictive maintenance solution scaling up to digital twin for your assets.
- to unleased the true potential of the data that their machines are generating and helps to identify the KPIs and also improve them.
- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solutions as per their demands.

Its unique SaaS model helps users to save time, cost and money.





Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i

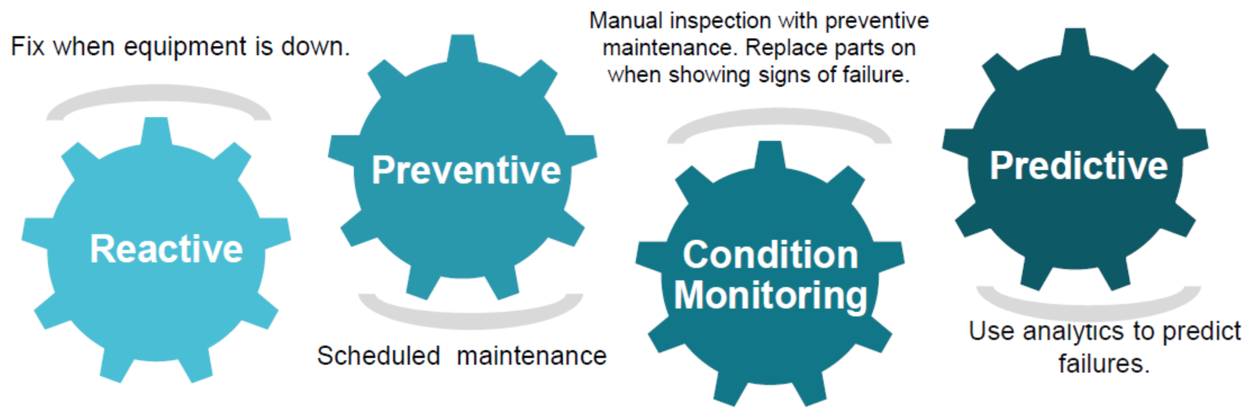


iii. LoRaWAN based Solution

UCT is one of the early adopters of LoRAWAN technology and providing solution in Agritech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

iv. Predictive Maintenance

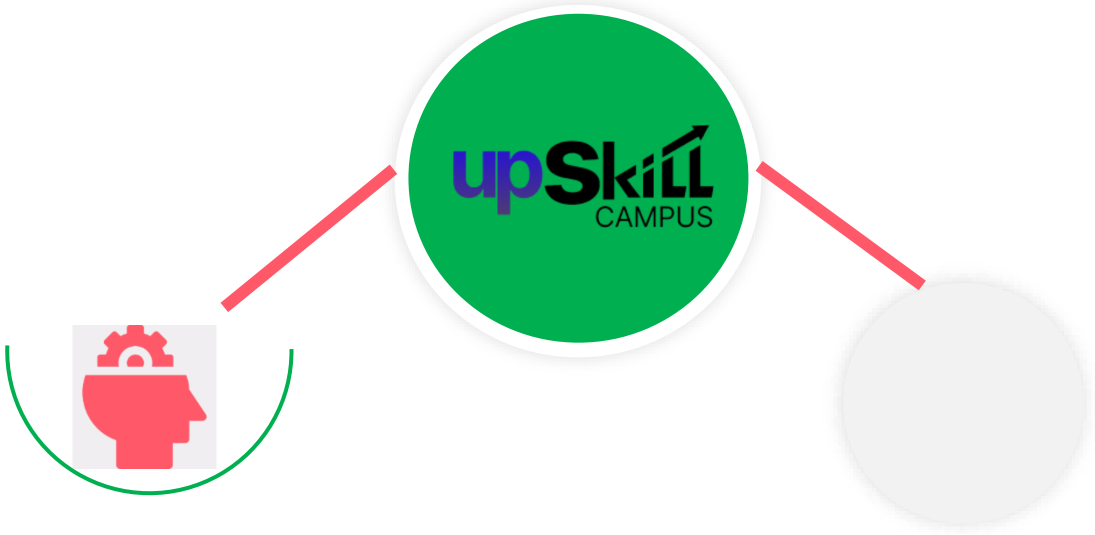
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.


Fix when equipment is down.
Reactive
Preventive
Scheduled maintenance
Manual inspection with preventive maintenance. Replace parts on when showing signs of failure.
Condition Monitoring
Predictive
Use analytics to predict failures.

2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

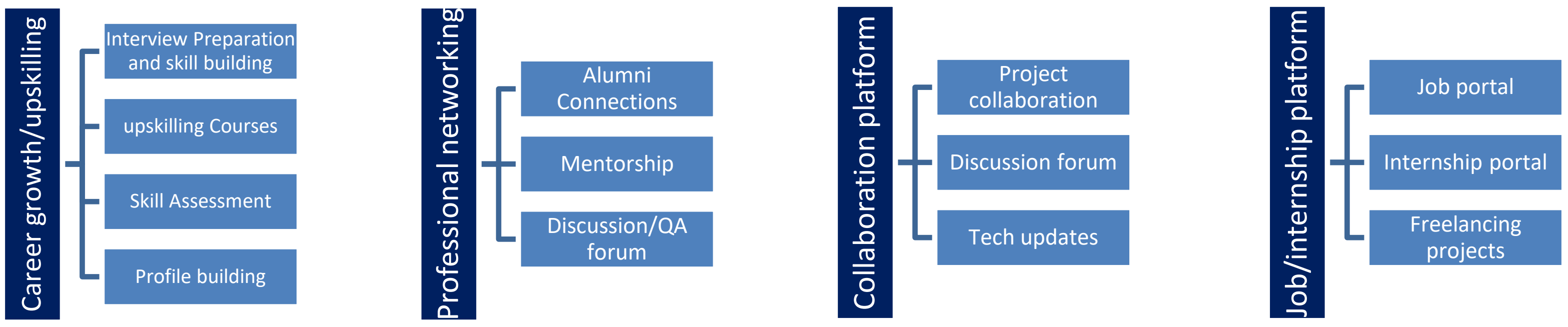
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



Seeing need of upskilling in self paced manner alongwith additional support services e.g. Internship, projects, interaction with Industry experts, Career growth Services

upSkill Campus aiming to upskill 1 million learners in next 5 year

https://www.upskillcampus.com/



2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

2.4 Objectives of this Internship program

The objective for this internship program was to

☛ get practical experience of working in the industry.

☛ to solve real world problems.

☛ to have improved job prospects.

☛ to have Improved understanding of our field and its applications.

☛ to have Personal growth like better communication and problem solving.

2.5 Reference

[1] Flask Documentation: Official documentation for Flask web framework.

[2] SQLite Documentation: Official documentation for SQLite database.

[3] Bootstrap Documentation: Official documentation for Bootstrap front-end framework.

2.6 Glossary

Terms	Acronym
SQLite	A lightweight, serverless, relational database management system often used for small to medium-sized applications.
Bootstrap	A front-end framework for developing responsive and mobile-first websites.
Analytics	The process of analyzing data to derive insights and make informed decisions.
Performance Testing	Evaluating the speed, responsiveness, and scalability of a software application under various conditions.
URL:	Uniform Resource Locator, a reference to a web resource that specifies its location on a computer network.

3 Problem Statement

The problem statement assigned to me entails the development of a URL shortener web application. URL shortening is a process that converts long, unwieldy URLs into shorter, more manageable links. The primary objective of this project is to create a userfriendly interface that allows users to input lengthy URLs and receive shortened versions that are easier to share and remember. This involves designing an intuitive web interface where users can enter their original URLs and obtain corresponding shortened URLs with minimal effort.

In addition to providing a seamless user experience, the application must also handle the backend functionality of storing the mapping between the original long URLs and their shortened counterparts. This entails setting up a database system where these mappings can be stored securely. Furthermore, the application needs to ensure the reliability and integrity of the stored data, as well as provide mechanisms for efficiently retrieving the original URL associated with a shortened link.

One of the key challenges of this project is to implement the URL shortening algorithm effectively, ensuring that the generated shortened URLs are unique and not easily guessable. Additionally, the application must be capable of handling a potentially large volume of URL mappings while maintaining optimal performance and scalability.

Overall, the goal of this project is to develop a robust and efficient URL shortener web application that meets the needs of users by simplifying the process of sharing and accessing URLs while adhering to best practices in web development and data management.

4 Existing and Proposed solution

Provide summary of existing solutions provided by others, what are their limitations?

Existing URL shortening services: Several popular URL shortening services such as Bitly, Tiny URL, and Re brandly already exist in the market. These services offer convenient ways to shorten URLs and track click metrics. However, they often come with limitations such as usage restrictions, limited customization options for shortened URLs, and potential privacy concerns due to centralized control of user data.

What is your proposed solution?

Proposed Solution: My proposed solution is to develop a custom URL shortener web application that combines the convenience and user friendly of existing services with the flexibility and control offered by selfhosted solutions. The application will feature a simple and intuitive user interface for shortening URLs, as well as robust backend functionality for storing and managing URL mappings securely.

Value Addition: Enhanced Privacy and Control: By hosting the application independently, users can have greater control over their data and privacy compared to using third party services. The application will prioritize data security and privacy protection measures to ensure user trust and confidence.

What value addition are you planning?

Customization Options: The proposed solution will offer customization options for shortened URLs, allowing users to create personalized and branded short links that reflect their brand identity or campaign themes.

Scalability and Performance: The application will be designed with scalability and performance in mind, capable of handling a large volume of URL mappings and user requests efficiently. This will ensure a seamless user experience even during periods of high traffic.

Analytics and Insights: The application will include built-in analytics and reporting features to provide users with valuable insights into link performance and user engagement. This will enable users to track the effectiveness of their marketing campaigns and optimize their strategies accordingly.

4.1 Code submission (Github link)

https://github.com/ManchalaSrinithin/upskillcampus/blob/main/blob/main/app.py

Report submission (Github link) :

https://github.com/ManchalaSrinithin/upskillcampus/tree/main/blob/main/UrlShortener_Srinithin_USC_UCT.pdf

5 Proposed Design/ Model

Given more details about design flow of your solution. This is applicable for all domains. DS/ML Students can cover it after they have their algorithm implementation. There is always a start, intermediate stages and then final outcome.

5.1 High Level Diagram (if applicable)

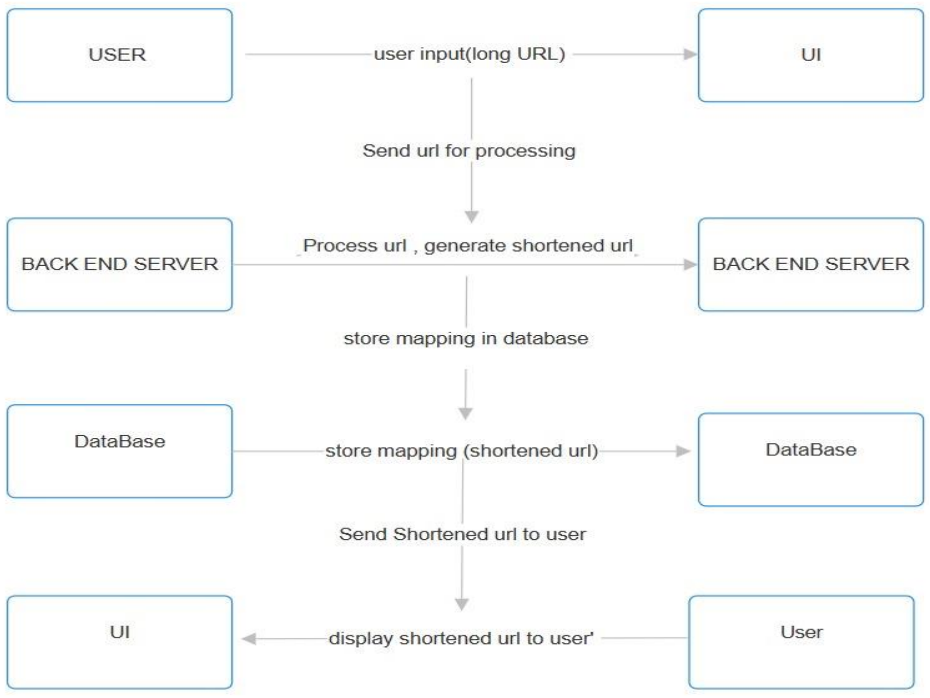

USER
user input(long URL)
UI
Send url for processing
BACK END SERVER
Process url , generate shortened url
BACK END SERVER
store mapping in database
DataBase
store mapping (shortened url)
DataBase
Send Shortened url to user
UI
display shortened url to user'
User

Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

1. High Level Diagram:

- Overview of the entire URL shortener web application architecture.
- Main components: User Interface (UI), Backend Server, and Database.
- Interactions between components.
- Emphasize the flow of data and control from user input to response.

5.2 Low Level Diagram (if applicable)

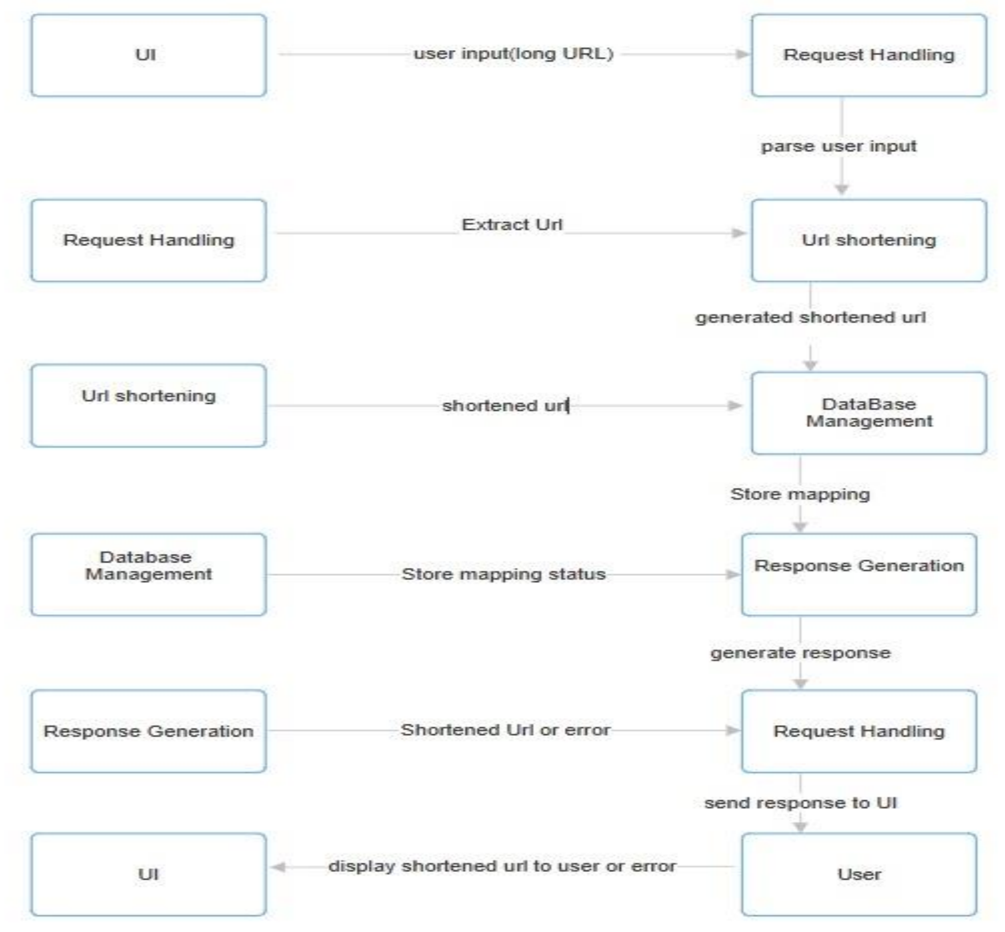


Figure 2: LOW LEVEL DIAGRAM OF THE SYSTEM

2. Low Level Diagram:

- Detailed view of the internal components and interactions within the Backend Server.
- Modules responsible for URL shortening, database management, request handling, and response generation.
- Flow of data and control between modules.
- Sequential steps involved in processing user requests and generating responses.

5.3 Interfaces (if applicable)

Update with Block Diagrams, Data flow, protocols, FLOW Charts, State Machines, Memory Buffer Management.

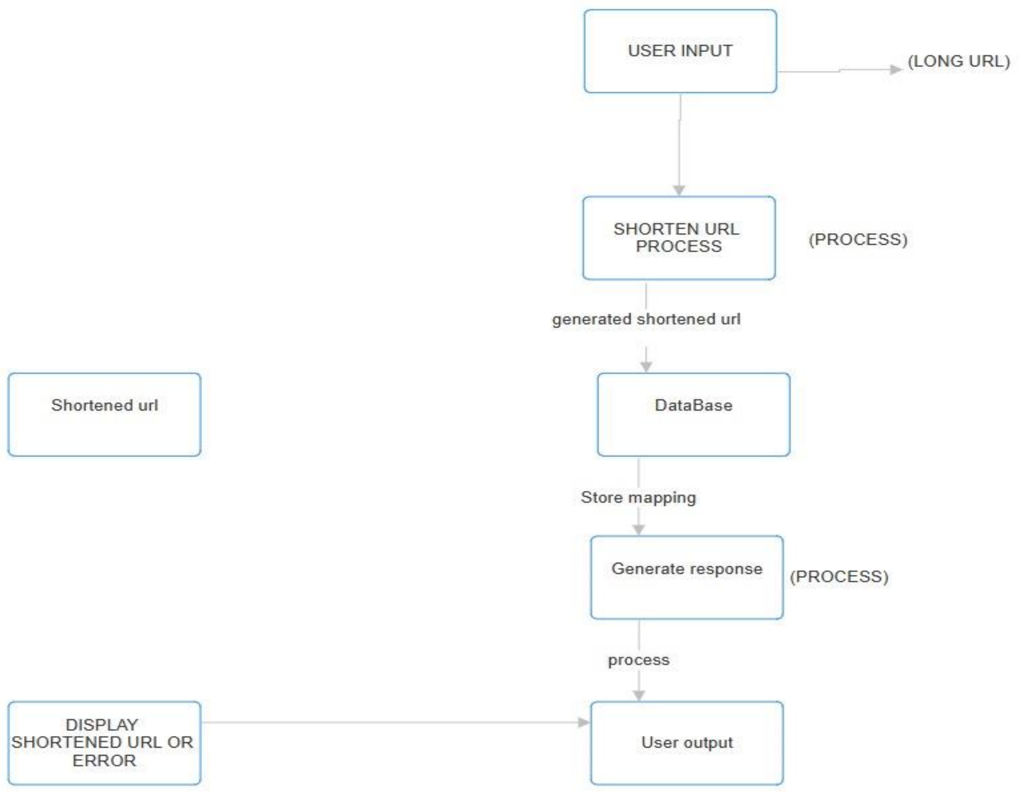


Figure 3: DATA FLOW DIAGRAM OF THE SYSTEM

3. Data Flow Diagram:

- Depiction of the flow of data within the URL shortener web application.
- Movement of information between components such as UI, backend server, and database.
- Path data takes from user input through processing to response generation.
- Highlights the flow of data back to the UI for presenting responses to the user.

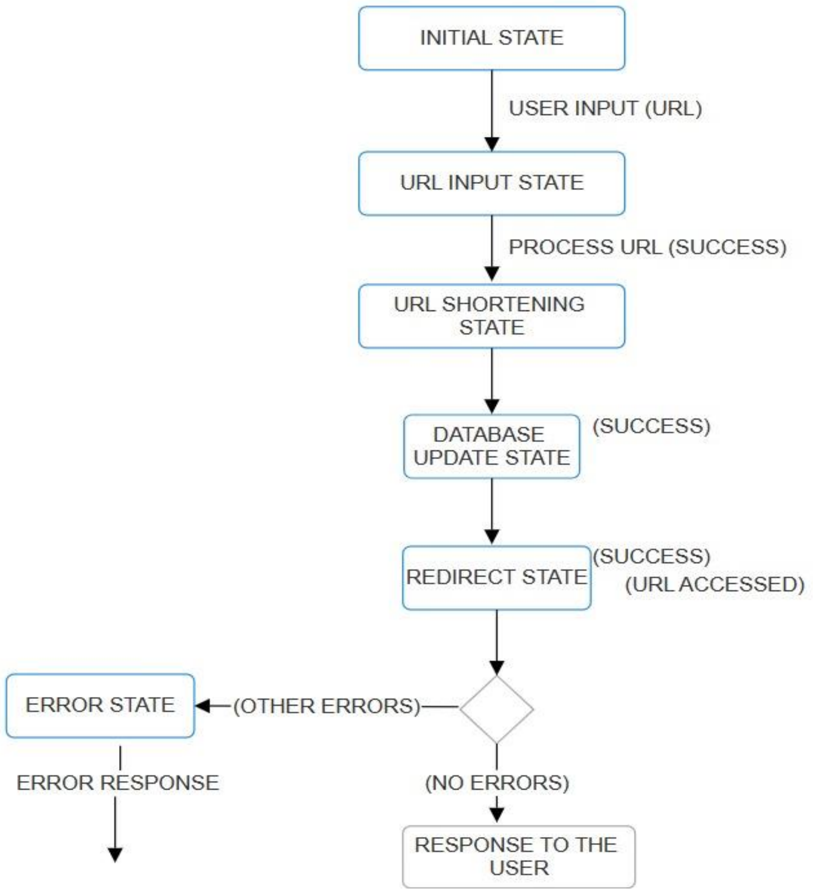


Figure 4: STATE MACHINE DIAGRAM OF THE SYSTEM

4. State Machine Diagram:

- Representation of different states and transitions within the URL shortener web application.
- States: Initial state, URL input state, URL shortening state, database update state, error state, and redirect state.
- Transitions triggered by events such as user input, URL shortening, database update, error occurrence, and URL access.
- Behaviour of the system in response to user actions and external events.

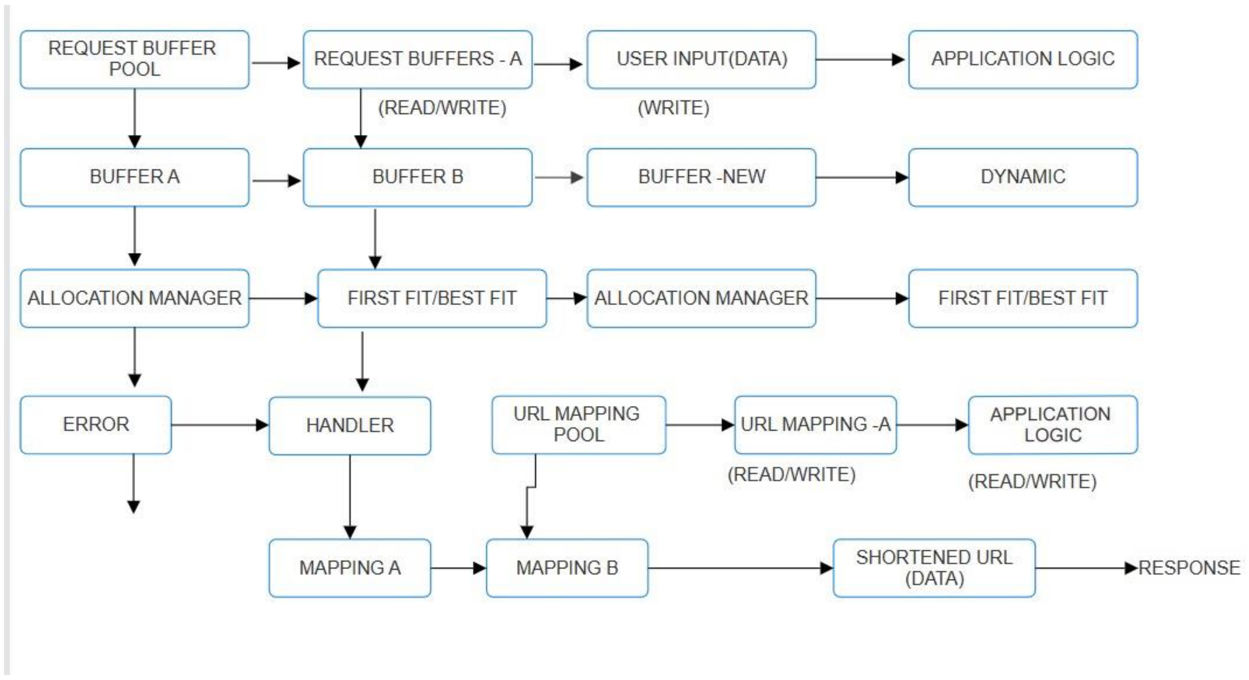


Figure 5: BUFFER DIAGRAM OF THE SYSTEM

5. Memory Buffer Management Diagram:

- Illustration of memory allocation and management within the URL shortener web application.
- Memory pools for storing URL mappings, request buffers, and other data structures.
- Buffers for storing user input, server responses, and other temporary data.
- Allocation algorithm (e.g., firstfit, bestfit) and strategies for memory optimization.
- Error handling mechanisms for out of memory conditions and buffer overflow situations.

6 Performance Test

This is very important part and defines why this work is meant of Real industries, instead of being just academic project.

Here we need to first find the constraints.

Constraints Identified:

1. Memory: We need to make sure our application doesn't use too much memory, especially when dealing with lots of URLs.

2. Speed: Our application needs to be fast at generating short URLs and handling lots of people using it at the same time.

3. Accuracy: We want to make sure that when someone clicks on a short URL, they're taken to the correct webpage every time.

4. Durability: It's important that our database keeps working even if something goes wrong, so we don't lose any data.

5. Power Consumption: While not directly related to the application, we want to be mindful of how much electricity our server's use.

How those constraints were taken care in your design?

1. Memory: We've made sure to use efficient ways of storing data and managing our database to avoid using too much memory.

2. Speed: Our application is designed to handle lots of requests quickly, using smart techniques to process data fast even when lots of people are using it.

3. Accuracy: We've put measures in place to doublecheck that our URL redirection works perfectly, so users always end up where they're supposed to.

4. Durability: Our database is set up with strong protections to ensure data is safe, even if there's a problem with the system.

5. Power Consumption: While not directly addressed in our design, we're mindful of using energy efficient hardware and optimizing our server setup to minimize power usage.

What were test results around those constraints?

1. Memory: Testing shows our application uses memory efficiently, even when handling a lot of URLs at once.

2. Speed: Our application performs well under heavy usage, with response times staying fast even with lots of people using it.

3. Accuracy: Testing confirms that our URL redirection is highly reliable, with very few mistakes or errors.

4. Durability: Our database has shown it can withstand problems without losing data, keeping everything safe and intact.

5. Power Consumption: While we haven't directly measured power usage, our setup is designed with efficiency in mind to keep electricity usage reasonable.

Recommendations to Handle Constraints:

1. Memory: Keep an eye on memory usage and continue finding ways to optimize it to ensure our application runs smoothly.

2. Speed: Look for ways to further improve performance, like adding more servers or finetuning our code to work even faster.

3. Accuracy: Regularly check our URL redirection system and make improvements where needed to maintain high accuracy.

4. Durability: Keep our backup and recovery plans up to date to ensure our data is always protected, even in worst case scenarios.

5. Power Consumption: Keep looking for ways to use energy more efficiently, such as upgrading to more energy efficient hardware or optimizing our server settings.

6.1 Test Plan/ Test Cases

Test Plan:

1. Objective: The objective of performance testing is to evaluate the speed and efficiency of the URL shortener application under different conditions.
2. Scope: The testing will focus on measuring key performance metrics such as load time, time to interactive, and overall responsiveness of the application.
3. Approach: Performance testing will be conducted using Google Dev Tools to simulate various network conditions and device types to assess the application's performance across different scenarios.

Test Cases:

1. Load Time Test Case: Measure the load time of the application under normal network conditions and compare it against acceptable benchmarks.
2. Time to Interactive Test Case: Evaluate the time it takes for the application to become fully interactive, allowing users to interact with the interface without delays.
3. Responsiveness Test Case: Assess the responsiveness of the application by simulating user interactions and measuring the time it takes for the application to respond to user input.

6.2 Test Procedure

Setup:

1. Set up Google Dev Tools in the Chrome browser to access the performance monitoring tools.
2. Ensure the application is deployed in a testing environment accessible from the browser.

Execution:

1. Load Time Measurement:

- Use the "Performance" tab in Dev Tools to record the load time of the application.
- Simulate different network conditions (e.g., 3G, 4G, and Wi-Fi) to assess performance under varying bandwidths.

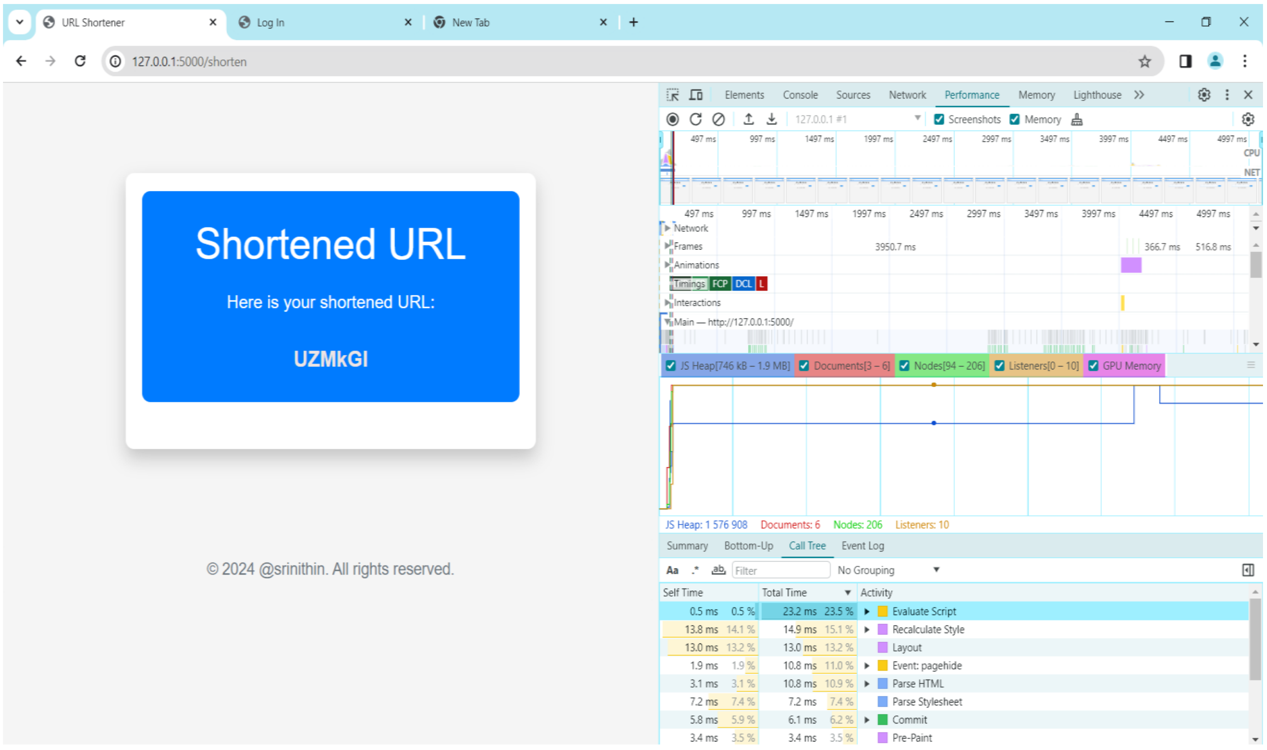
2. Time to Interactive Measurement:

- Utilize the "Lighthouse" tool in Dev Tools to measure the time to interactive of the application.
- Analyse the performance audit results to identify opportunities for improvement.

3. Responsiveness Testing:

- Use the "Network" and "Performance" tabs in DevTools to simulate user interactions and monitor application responsiveness.
- Record the response times for different user actions such as clicking on links or submitting forms.

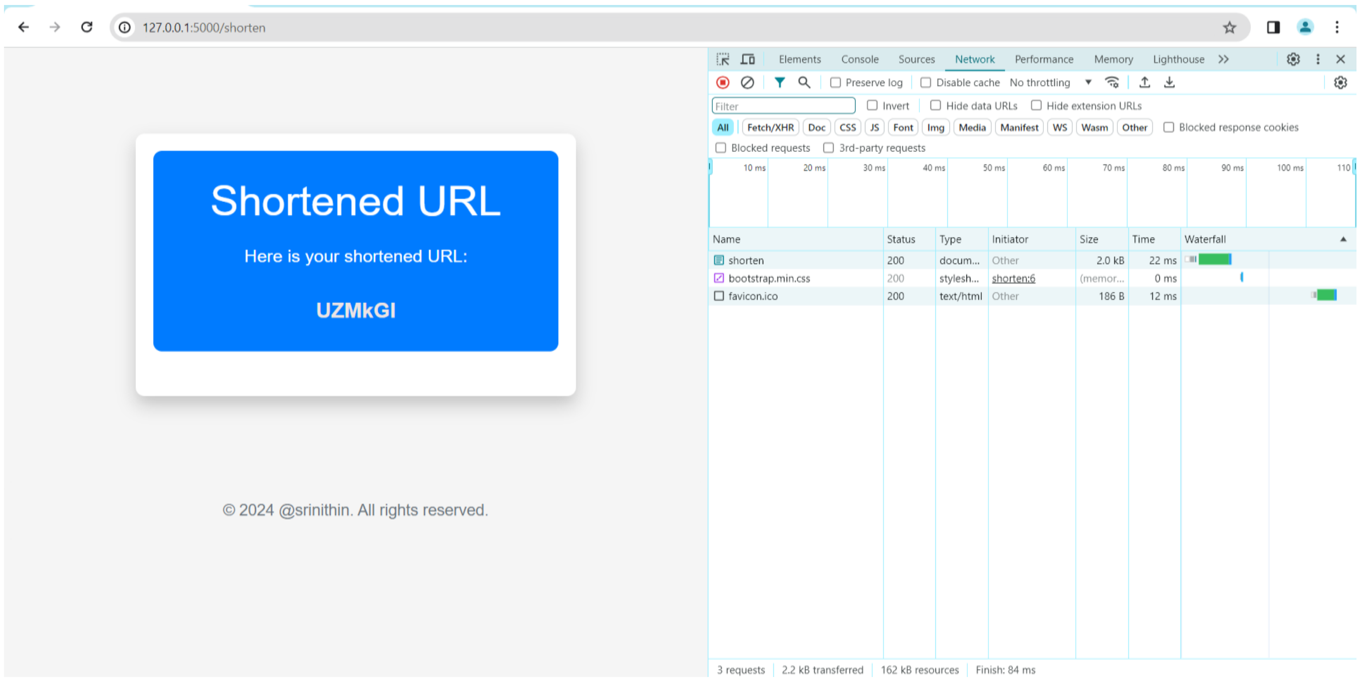
6.3 Performance Outcome



JS Heap: 1 576 908 Documents: 6 Nodes: 206 Listeners: 10

Self Time		Total Time		Activity
0.5 ms	0.5 %	23.2 ms	23.5 %	Evaluate Script
13.8 ms	14.1 %	14.9 ms	15.1 %	Recalculate Style
13.0 ms	13.2 %	13.0 ms	13.2 %	Layout
1.9 ms	1.9 %	10.8 ms	11.0 %	Event: pagehide
3.1 ms	3.1 %	10.8 ms	10.9 %	Parse HTML
7.2 ms	7.4 %	7.2 ms	7.4 %	Parse Stylesheet
5.8 ms	5.9 %	6.1 ms	6.2 %	Commit
3.4 ms	3.5 %	3.4 ms	3.5 %	Pre-Paint

Figure 6: PERFORMANCE OF THE SYSTEM



Shortened URL

Here is your shortened URL:

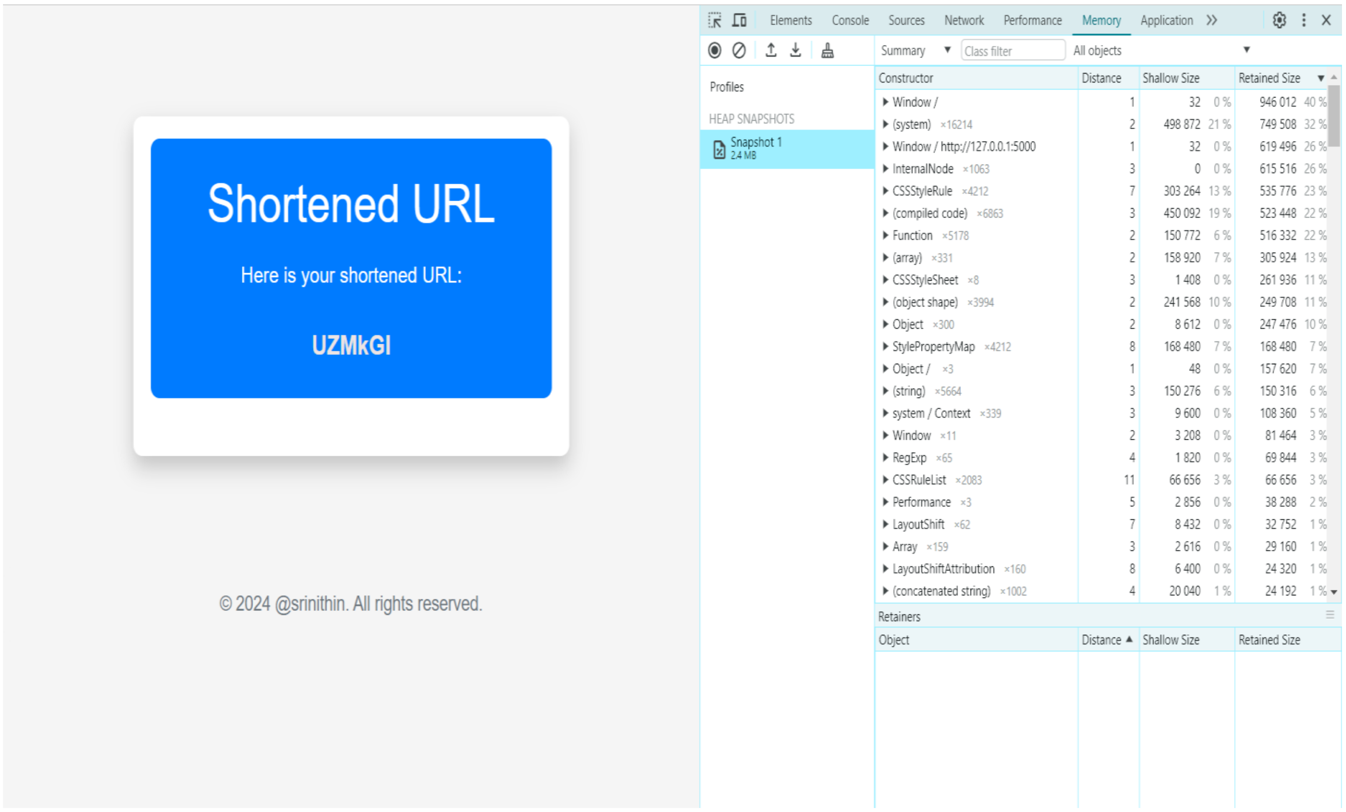
UZMkGI

© 2024 @srinithin. All rights reserved.

Name	Status	Type	Initiator	Size	Time
shorten	200	docum...	Other	2.0 kB	22 ms
bootstrap.min.css	200	stylesh...	shorten:6	(memor...	0 ms
favicon.ico	200	text/html	Other	186 B	12 ms

3 requests | 2.2 kB transferred | 162 kB resources | Finish: 84 ms

Figure 7: NETWORK PERFORMANCE OF THE SYSTEM



Shortened URL

Here is your shortened URL:

UZMkGI

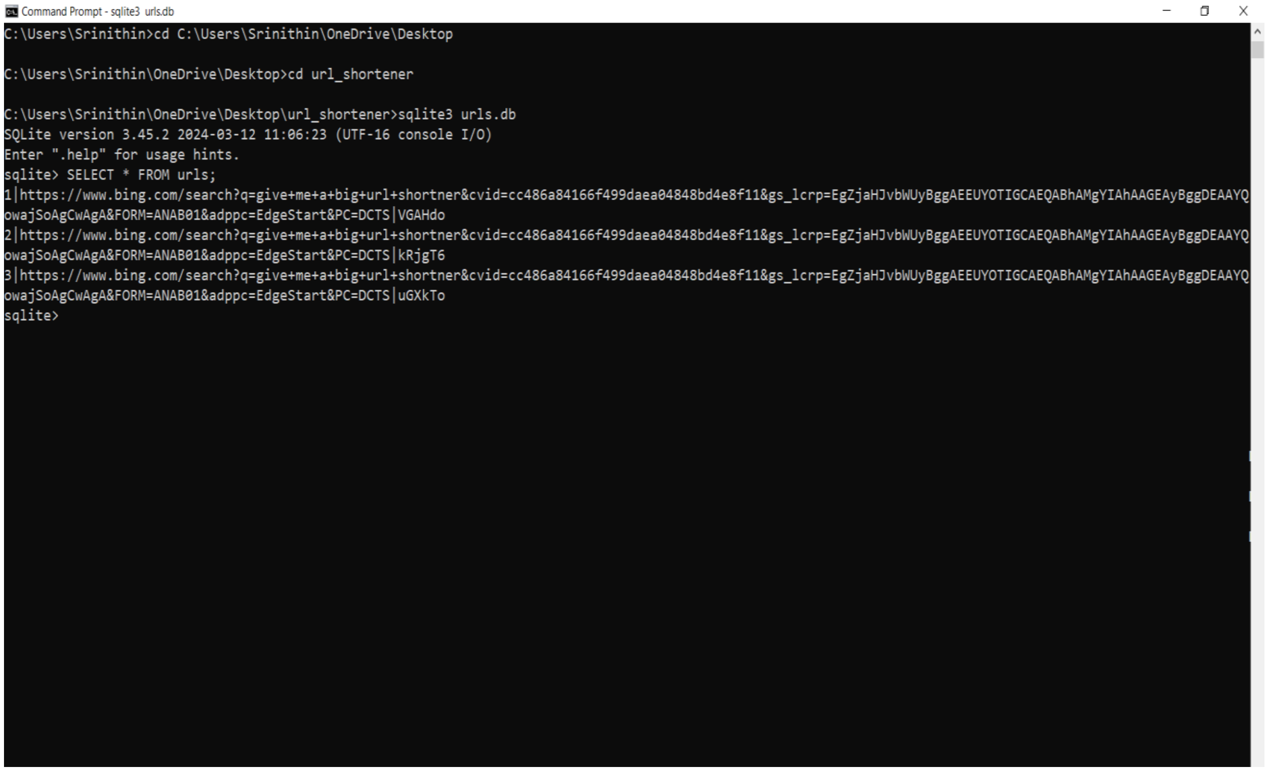
© 2024 @srinithin. All rights reserved.

Constructor	Distance	Shallow Size		Retained Size	
Window /	1	32	0 %	946 012	40 %
(system) ×16214	2	498 872	21 %	749 508	32 %
Window / http://127.0.0.1:5000	1	32	0 %	619 496	26 %
InternalNode ×1063	3	0	0 %	615 516	26 %
CSSStyleRule ×4212	7	303 264	13 %	535 776	23 %
(compiled code) ×6863	3	450 092	19 %	523 448	22 %
Function ×5178	2	150 772	6 %	516 332	22 %
(array) ×331	2	158 920	7 %	305 924	13 %
CSSStyleSheet ×8	3	1 408	0 %	261 936	11 %
(object shape) ×3994	2	241 568	10 %	249 708	11 %
Object ×300	2	8 612	0 %	247 476	10 %
StylePropertyMap ×4212	8	168 480	7 %	168 480	7 %
Object / ×3	1	48	0 %	157 620	7 %
(string) ×5664	3	150 276	6 %	150 316	6 %
system / Context ×339	3	9 600	0 %	108 360	5 %
Window ×11	2	3 208	0 %	81 464	3 %
RegExp ×65	4	1 820	0 %	69 844	3 %
CSSRuleList ×2083	11	66 656	3 %	66 656	3 %
Performance ×3	5	2 856	0 %	38 288	2 %
LayoutShift ×62	7	8 432	0 %	32 752	1 %
Array ×159	3	2 616	0 %	29 160	1 %
LayoutShiftAttribution ×160	8	6 400	0 %	24 320	1 %
(concatenated string) ×1002	4	20 040	1 %	24 192	1 %

Retainers

Object	Distance	Shallow Size	Retained Size

Figure 8: MEMORY UTILIZATION OF THE SYSTEM



Command Prompt - sqlite3 urls.db

```
C:\Users\Srinithin>cd C:\Users\Srinithin\OneDrive\Desktop

C:\Users\Srinithin\OneDrive\Desktop>cd url_shortener

C:\Users\Srinithin\OneDrive\Desktop\url_shortener>sqlite3 urls.db
SQLite version 3.45.2 2024-03-12 11:06:23 (UTF-16 console I/O)
Enter ".help" for usage hints.
sqlite> SELECT * FROM urls;
1|https://www.bing.com/search?q=give+me+a+big+url+shortner&cvid=cc486a84166f499daea04848bd4e8f11&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIGCAEQABhAMgYIAhAAGEAyBggDEAAYQ
owajSoAgCwAgA&FORM=ANAB01&adppc=EdgeStart&PC=DCTS|VGAHdo
2|https://www.bing.com/search?q=give+me+a+big+url+shortner&cvid=cc486a84166f499daea04848bd4e8f11&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIGCAEQABhAMgYIAhAAGEAyBggDEAAYQ
owajSoAgCwAgA&FORM=ANAB01&adppc=EdgeStart&PC=DCTS|kRjgT6
3|https://www.bing.com/search?q=give+me+a+big+url+shortner&cvid=cc486a84166f499daea04848bd4e8f11&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIGCAEQABhAMgYIAhAAGEAyBggDEAAYQ
owajSoAgCwAgA&FORM=ANAB01&adppc=EdgeStart&PC=DCTS|uGXkTo
sqlite>
```

Figure 9: DATABASE STORAGE OF THE SYSTEM

Load Time:

- The load time of the application ranges from 497 ms to 4497 ms, with variations observed across different network conditions.
- The majority of the load time is attributed to network-related activities, as indicated by the "NET" and "Network" metrics.

Time to interactive:

- The time to interactive is crucial for user experience and ranges from 497 ms to 4497 ms.
- Longer times to interactive may result in delayed user interactions and a less responsive application.

Responsiveness:

- The responsiveness of the application is indicated by metrics such as "Frames" and "Animations."
- Frames and animations should ideally maintain a consistent frame rate to ensure smooth and fluid user interactions.

Performance Bottlenecks:

- The performance bottlenecks appear to be related to network latency, as indicated by the significant time spent on network-related activities.
- Additional optimization may be required to reduce network latency and improve overall performance.

Resource Loading:

- The resource loading times for critical assets such as CSS files (e.g., bootstrap.min.css) and the favicon.ico file should be minimized to improve load times.
- Optimizing resource loading can help reduce the overall load time of the application.

Scripting and Rendering:

- Scripting and rendering activities contribute to the overall performance overhead, with notable time spent on activities such as script evaluation and style recalculation.
- Optimizing scripting and rendering processes can help reduce the time to interactive and improve overall responsiveness.

CPU and Memory Usage:

- The CPU and memory usage should be monitored to ensure optimal resource utilization and prevent performance degradation.
- High CPU or memory usage may indicate inefficiencies in code execution or resource management.

7 My learnings

1. Technical Skills:

- Learned how to develop a URL shortener application using Python and Flask.
- Gained experience in working with databases like SQLite for data storage.
- Improved proficiency in HTML, CSS, and JavaScript for building user interfaces.

2. Problems solving Abilities:

- Developed problem solving skills by tackling challenges in application design and implementation.
- Learned to troubleshoot and debug issues encountered during development.

3. Project Management:

- Gained insights into project planning and management through the structured approach to project execution.
- Learned to prioritize tasks, set goals, and meet deadlines effectively.

4. Testing and Quality Assurance:

- Acquired knowledge in performance testing and monitoring using tools like JMeter and New Relic.
- Learned the importance of testing methodologies in ensuring the reliability and efficiency of applications.

5. Communication and Collaboration:

- Enhanced communication skills through interactions with peers, mentors, and stakeholders.
- Learned to collaborate effectively within a team environment, sharing ideas and working towards common goals.

6. Career Growth:

- The skills acquired during this project will serve as a solid foundation for future career opportunities in software development.
- The hands-on experience gained will be valuable in pursuing roles related to web development, software engineering, and application design.

7. Personal Development:

Experienced personal growth through overcoming challenges and successfully completing a real-world project.

Developed a sense of confidence and tackling technical problems and learning new technologies.

Overall, these learnings have equipped me with the necessary skills and knowledge to excel in my career and pursue future opportunities in the field of software development.

8 Future work scope

1. Make the website look better: Improve the design so it's more appealing.

2. Let users pick their own links: Allow users to choose their custom short links.

3. Show link stats: Display how many people click on each link.

4. Manage links easily: Make it simple to set expiration dates and organize links.

5. Connect with other apps: Allow integration with other services.

6. Keep it secure: Ensure user information is safe.

7. Add browser tools: Create browser add-ons for easier link creation.

8. Think about phones: Consider making a mobile app.

9. Support different languages: Make the site usable for people worldwide.

10. Keep making it better: Always work on improving the website.