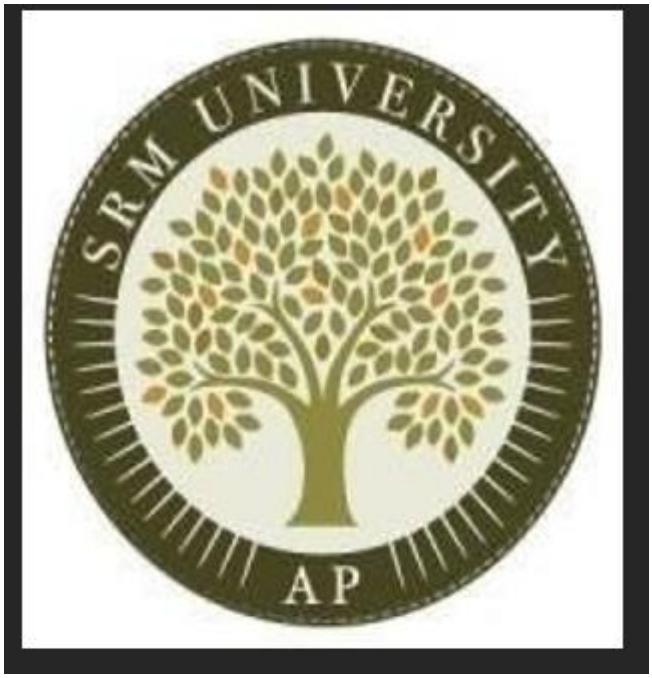
ADVANCED BROWSER NAVIGATION USING STACKS

Department: CSE — SRM University

Academic Year: 2025–2026



Submitted By:

• B. Thanishka — AP24110011830

• M. Harika — AP24110011839

• G. Sourabi — AP24110011844

• D. Bindhu — AP24110011845

Submitted To:

Prakash sir

Date: 08/12/25

Table of Contents

11. **Detailed Module Description**
    - Back Navigation Module
    - Forward Navigation Module
    - Visit New Website
    - Bookmarks Feature
    - Search History
    - Close & Reopen Last Page
    - Export Browsing History
12. **Output Description**
13. **Full Source Code**
14. **Conclusion**
15. **Future Enhancements**
16. **References**

CERTIFICATE

This is to certify that the Mini Project titled:

“ADVANCED BROWSER NAVIGATION SYSTEM USING STACKS”

submitted by

B. Thanishka (AP24110011830)

M. Harika (AP24110011839)

G. Sourabi (AP24110011844)

D. Bindhu (AP24110011845)

of B.Tech — Computer Science and Engineering, SRM University, is a bonafide record of the project work carried out during the academic year 2024–2025 under my guidance.

It has not been submitted to any other University or Institute for the award of any degree or diploma.

Project Guide

(Sign) _____

Name: _____

Head of the Department

(Sign) _____

Name: _____

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Prakash sir Department of Computer Science and Engineering, SRM University, for providing invaluable guidance and support throughout the completion of our mini project.

We extend our heartfelt thanks to all the faculty members of the Department of CSE for their continuous encouragement and constructive suggestions.

We are deeply thankful to our parents and friends for their constant motivation and support.

Finally, we thank our institution for providing excellent infrastructure and learning environment.

— Team Members

• B. Thanishka

• M. Harika

• G. Sourabi

• D. Bindhu

ABSTRACT

A web browser allows users to access and navigate web pages across the internet. An important aspect of browsing is navigation through visited pages using Back and Forward operations. Modern browsers implement this using the Stack Data Structure, ensuring the most recently visited pages are accessed first (LIFO — Last In First Out).

This project aims to simulate a Browser Navigation System using two stacks: one for backward navigation and one for forward navigation. It also features additional functionalities such as:

✓ Visiting new websites

✓ Adding and showing bookmarks

✓ Searching browsing history

✓ Reopening closed pages

✓ Exporting browsing history to a text file

✓ Clear and user-friendly output display

This mini project demonstrates how stacks support efficient memory management and navigation in real-world applications. It strengthens understanding of programming logic, data structures, and file operations in the C language.

INTRODUCTION

Web browsers are essential tools used to access information available on the World Wide Web. During browsing, users move from one webpage to another, often going back and forth between visited links.

To handle this efficiently, browsers use two stacks:

Stack

Purpose

Operation

Back Stack

Stores previously visited pages

POP last visited page

Forward Stack

Stores pages returned from back navigation

POP forward pages

When a user:

✓ Visits a new site → Push current page to Back stack & clear Forward stack

✓ Clicks Back → Move current page to Forward stack and load last Back page

✓ Clicks Forward → Move current page to Back and load last Forward page

This project implements the above logic using stack operations, demonstrating how simple structures can solve complex navigation problems.

PROBLEM STATEMENT

Design and implement a Text-based Browser Navigation System in C language that:

🔷 Simulates Back and Forward buttons using Stacks

🔷 Stores browsing history and bookmarks

- 🔷 Allows reopening closed pages
- 🔷 Exports history to a text file
- 🔷 Displays clean and clear status of navigation

The system must efficiently manage memory and allow seamless browsing experience similar to modern browsers.

OBJECTIVES

Implement a browser navigation model using Stack data structure

Efficiently manage Back and Forward operation

Maintain browsing history & support searching

Implement bookmarks and restore last closed tab

Provide export functionality using file handling

Apply Data Structures knowledge to real-world situations

SYSTEM REQUIREMENTS

Hardware Requirements

Component

Specification

Processor

Dual Core / i3 or above

RAM

Minimum 4GB

Storage

10GB free space

Input Devices

Keyboard, Mouse

Display

Standard Monitor

Software Requirements

Item

Specification

Operating System

Windows / Linux

Programming Language

C

Compiler

GCC / Turbo C / Code::Blocks / VS Code

Text Editor

Any (VS Code recommended)

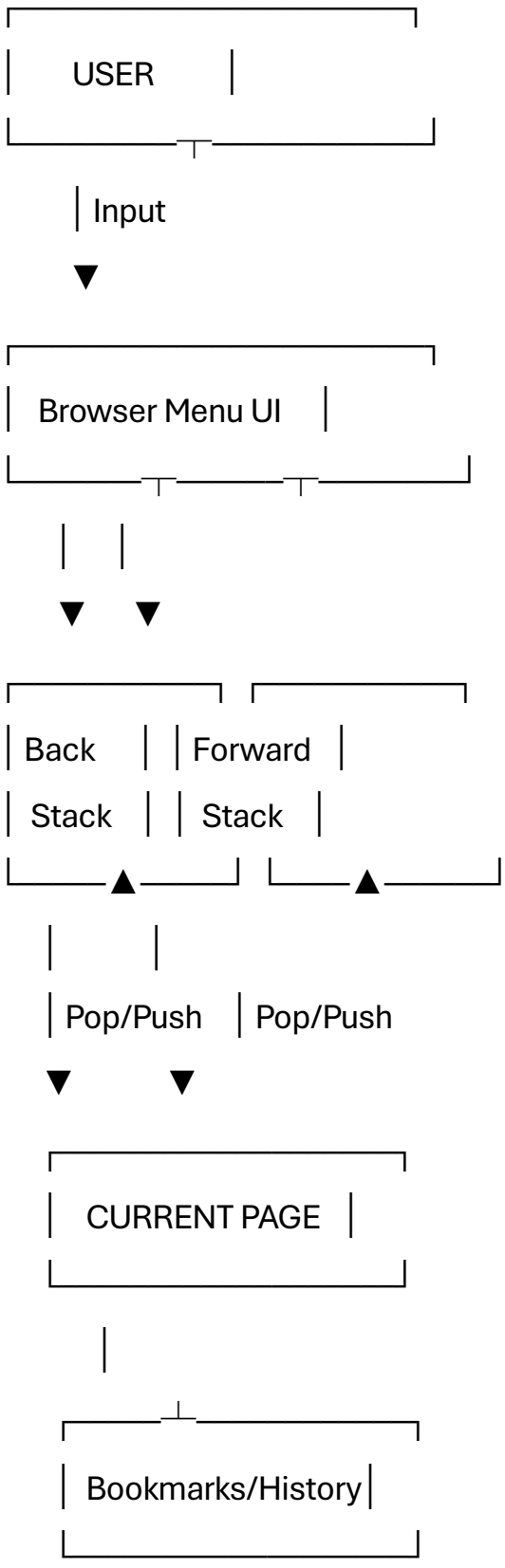
METHODOLOGY

The main idea is to use two stacks for navigation:

Logic Behind Navigation:

1. When a new URL is opened →
   - Push current page into Back stack
   - Clear Forward stack
2. When Back is pressed →
   - Push current page into Forward stack
   - Pop top of Back stack → Make it current page
3. When Forward is pressed →
   - Reverse operation of Back button
4. Bookmarks stored separately in array
5. Search is done using string comparison in both stacks
6. History can be saved to a file

SYSTEM ARCHITECTURE DIAGRAM



ALGORITHMS & PSEUDOCODE

🔷 Algorithm — Visit New Site()

1. Ask user to enter URL
2. Push current page into Back Stack
3. Set currentPage = new URL
4. Clear Forward Stack
5. Display the page

🔷 Algorithm — Back()

1. If Back stack empty → show “No history”
2. Push current page to Forward Stack
3. Pop last page from Back → set as currentPage

🔷 Algorithm — Forward()

1. If Forward empty → show message
2. Push current page to Back Stack
3. Pop from Forward → set as currentPage

🔷 Algorithm — Add Bookmark()

1. Store currentPage into bookmarks array
2. Display success message

🔷 Algorithm — Search History()

1. Input search keyword
2. Compare with currentPage, backStack & forwardStack
3. Display whether found or not

🔷 Algorithm — Export History()

1. Create/Write to a text file
2. Save Current, Back & Forward pages list
3. Display success message

DETAILED MODULE DESCRIPTION

1 Back Navigation Module

Uses pushBack() to save previous pages

popBack() retrieves the latest page

Ensures LIFO behavior

2 Forward Navigation Module

Restores pages after using Back

Similar push/pop mechanism

3 Visit New Website

Updates current page

Clears forward history

Adds old page to Back Stack

4 Bookmarks Feature

Stores frequently visited pages

Displays via menu option

5 Search History

Linearly searches in:

Current page

Back Stack

Forward Stack

6 Close & Reopen Last Page

Last closed page stored separately

Can be restored anytime unless overwritten

7 Export Browsing History

Creates BrowsingHistory.txt

Stores complete navigation data

OUTPUT DESCRIPTION

Feature

What Happens in Output

Visit New Page

Displays “Browsing: <URL>”

Back

Shows previous page as current

Forward

Restores page from Forward history

Add Bookmark

Shows confirmation message

Search History

Displays “Found” or “Not found”

Reopen Closed

Shows restored page name

Export History

Shows “History exported successfully”

Show Status

Displays Back list, Forward list & Current Page

FULL SOURCE CODE

```
#include <stdio.h>
#include <string.h>
#define MAX 50
char backStack[MAX][50];
char forwardStack[MAX][50];
```

```
char bookmarks[MAX][50];
char lastClosedPage[50];

int topBack = -1, topForward = -1, topBookmark = -1;

char currentPage[50] = "Home";

// ---------------- STACK BASIC FUNCTIONS ----------------
void pushBack(char page[]) {
   if (topBack < MAX - 1)
      strcpy(backStack[++topBack], page);
}

void pushForward(char page[]) {
   if (topForward < MAX - 1)
      strcpy(forwardStack[++topForward], page);
}

char* popBack() {
   if (topBack >= 0)
      return backStack[topBack--];
   return NULL;
```

```
}

char* popForward() {
    if (topForward >= 0)
        return forwardStack[topForward--];
    return NULL;
}

// ---------------- MAIN FEATURES ----------------
void visitNewSite() {
    char site[50];
    printf("Enter website URL: ");
    scanf("%49s", site);

    pushBack(currentPage);
    strcpy(currentPage, site);
    strcpy(lastClosedPage, "");
    topForward = -1;

    printf("👉 Browsing: %s\n", currentPage);
}
```

```
void goBack() {
    if (topBack < 0) {
        printf(" ⚠ No pages in Back history.\n");
        return;
    }
    pushForward(currentPage);
    strcpy(currentPage, popBack());
    printf(" 🔙 Now browsing: %s\n", currentPage);
}

void goForward() {
    if (topForward < 0) {
        printf(" ⚠ No pages in Forward history.\n");
        return;
    }
    pushBack(currentPage);
    strcpy(currentPage, popForward());
    printf(" ➡ Now browsing: %s\n", currentPage);
}

void addBookmark() {
```

```
    if (topBookmark < MAX - 1) {
        strcpy(bookmarks[++topBookmark], currentPage);
        printf("⭐ Added to bookmarks: %s\n", currentPage);
    }
}

void showBookmarks() {
    int i;
    if (topBookmark < 0) {
        printf("⚠️ No bookmarks saved.\n");
        return;
    }
    printf("\n📒 BOOKMARKS:\n");
    for (i = 0; i <= topBookmark; i++)
        printf("%d) %s\n", i + 1, bookmarks[i]);
}

void searchHistory() {
    char key[50];
    int found = 0, i;
    printf("Enter website to search: ");
    scanf("%49s", key);
```

```
    if (strcmp(currentPage, key) == 0) found = 1;

    for (i = 0; i <= topBack; i++)
        if (strcmp(backStack[i], key) == 0) found = 1;

    for (i = 0; i <= topForward; i++)
        if (strcmp(forwardStack[i], key) == 0) found = 1;

    if (found)
        printf(" ✔ '%s' was found in history.\n", key);
    else
        printf(" ❌ '%s' not found in history.\n", key);
}

void closeCurrent() {
    strcpy(lastClosedPage, currentPage);
    printf(" ❌ Closed page: %s\n", currentPage);

    if (topBack >= 0)
        strcpy(currentPage, popBack());
    else
```

```
        strcpy(currentPage, "Home");
}

void reopenLastClosed() {
    if (strlen(lastClosedPage) == 0) {
        printf(" ⚠ No recently closed page.\n");
        return;
    }
    pushBack(currentPage);
    strcpy(currentPage, lastClosedPage);
    printf(" 🔁 Reopened last closed page: %s\n", currentPage);
    strcpy(lastClosedPage, "");
}

void exportHistory() {
    FILE *f = fopen("BrowsingHistory.txt", "w");
    int i;

    fprintf(f, "Current Page: %s\n\n", currentPage);

    fprintf(f, "Back History:\n");
    for (i = topBack; i >= 0; i--)
```

```
        fprintf(f, "%s\n", backStack[i]);

    fprintf(f, "\nForward History:\n");
    for (i = topForward; i >= 0; i--)
        fprintf(f, "%s\n", forwardStack[i]);

    fprintf(f, "\nBookmarks:\n");
    for (i = 0; i <= topBookmark; i++)
        fprintf(f, "%s\n", bookmarks[i]);

    fclose(f);
    printf("📂 Browsing history exported successfully.\n");
}

void showStatus() {
    int i;

    printf("\n----------------------------------------\n");
    printf("🌐 Current Page: %s\n", currentPage);

    printf("🔙 Back History: ");
    if (topBack < 0) printf("None");
```

```
    else {
        for (i = topBack; i >= 0; i--)
            printf("%s%s", backStack[i], (i > 0 ? " | " : ""));
    }
    printf("\n");

    printf(" ➡️ Forward History: ");
    if (topForward < 0) printf("None");
    else {
        for (i = topForward; i >= 0; i--)
            printf("%s%s", forwardStack[i], (i > 0 ? " | " : ""));
    }
    printf("\n----------------------------------------\n");
}
// ---------------- MAIN PROGRAM ----------------
int main() {
    int choice;

    while (1) {
        printf("\n================= BROWSER MENU
=================\n");
        printf("1. Visit New Website\n");
```

```
printf("2. Back\n");
printf("3. Forward\n");
printf("4. Add Bookmark\n");
printf("5. Show Bookmarks\n");
printf("6. Search History\n");
printf("7. Close Current Page\n");
printf("8. Reopen Last Closed Page\n");
printf("9. Export Browsing History\n");
printf("10. Show Status\n");
printf("11. Exit\n");
printf("Enter choice: ");
scanf("%d", &choice);

switch (choice) {
    case 1: visitNewSite(); break;
    case 2: goBack(); break;
    case 3: goForward(); break;
    case 4: addBookmark(); break;
    case 5: showBookmarks(); break;
    case 6: searchHistory(); break;
    case 7: closeCurrent(); break;
    case 8: reopenLastClosed(); break;
```

```
            case 9: exportHistory(); break;
            case 10: showStatus(); break;
            case 11: return 0;
            default: printf(" ⚠ Invalid choice. Try again.\n");
        }
    }
}
```

CONCLUSION

This project demonstrates how Stacks can be used to simulate real-life browser navigation.

The implementation proves that LIFO operations can efficiently manage browsing actions such as:

✓ Back & Forward Navigation

✓ History Management

✓ Bookmarking System

✓ File Handling Application

✓ Restoring Recently Closed Pages

Through this project, we gained hands-on experience in:

🔷 Data structures

🔷 C programming

- Algorithmic thinking
- System design & functionality

It successfully fulfills the objectives and provides a strong foundation for advanced software development.

FUTURE ENHANCEMENTS

Feature	Description
GUI Web Browser Prototype	A visual and interactive software using graphics
Multi-tab browsing	Independent tabs for multiple sessions
Auto-complete & search suggestions	Smarter browsing experience
Database support	Save history permanently in SQL database
Privacy mode	Temporary session with auto-clear cache
Synchronization	Cloud-based backup of browsing data

REFERENCES

1 Reema Thareja — Data Structures Using C

2 Ellis Horowitz — Fundamentals of Data Structures

3 GeeksforGeeks — Data Structures & C Programming

4 TutorialsPoint — C Language Basics

5 Stack Overflow Discussions

for this give tables of content