MINI PROJECT

Personal Diary Management System

Name : Selvadhirsana P

Register number : 71125BCS154

Class : I - CSE - 'A'

Subject : C Programming

Date : 01 - 12 - 2025

INTRODUCTION:

The “Personal Diary Management System” is a console-based C application that allows users to securely store, view, and delete their diary entries. The program provides password protection to ensure privacy. When the application runs for the first time, the user is asked to create a password. This password is converted into a hashed number for security and stored in a file. On every next login, the user must enter the correct password to access the diary.

Once logged in, the user can perform the following operations:

1. Write a New Diary Entry – The user can type a text entry which is saved in a file along with the current date.

2. View All Entries – All previously saved diary entries are displayed.

3. Delete an Entry – The user can select one of the saved entries to delete.

4. Exit – Quit the application.

The system uses text files (diary.txt and password.dat) to store information, making the program simple and easy to run on any computer. The program also uses safe input methods and basic hashing to ensure input correctness and password protection.

ALGORITHM:

Step 1: Start the program

Step 2:

- Check if password file exists
- Try to open password.dat.

❖ If the file does not exist:
❖ Ask the user to create a password (without spaces).
❖ Hash the password using simpleHash().
❖ Save the hash in password.dat.
❖ Display “Password created successfully”.

Step 3:

➢ If password file exists
➢ Read the stored password hash from the file.

Step 4:

✓ Ask the user to enter the password
✓ Read password input using readLine().
✓ Hash the entered password.
✓ If hashed value matches the stored hash → Login successful.
✓ Otherwise, show error and repeat until correct password is entered.

Step 5:
Show main menu

1. Write New Entry
2. View All Entries
3. Delete an Entry
4. Exit

Step 6:
Read user choice

Step 7:

❖ If choice = 1 (Write Entry)
❖ Ask user to type entry text.

- ❖ Get current date using getCurrentDate().
- ❖ Append date and entry into diary.txt.
- ❖ Add separator line (---).
- ❖ Display “Entry saved successfully”.

Step 8:

- ➢ If choice = 2 (View Entries)
- ➢ Open diary.txt in read mode.
- ➢ Display all saved lines.
- ➢ If file not found, display “No diary entries found”.

Step 9:

- ✓ If choice = 3 (Delete Entry)
- ✓ Load all diary entries into memory.
- ✓ Count how many entries exist by checking lines starting with "Date:".
- ✓ Display a numbered list of entry dates.
- ✓ Ask user which entry number to delete.
- ✓ Recreate diary.txt while skipping the selected entry block.
- ✓ Show “Entry deleted successfully”.

Step 10:

- ❖ If choice = 4 (Exit)
- ❖ Display “Goodbye!”
- ❖ Terminate the program.

Step 11:

- ➢ If invalid choice
- ➢ Show error and return to menu.

Step 12:

End program.

SOURCE CODE:

```
#include <stdio.h>
```

```
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <time.h>
int simpleHash(const char *password) {
   int hash = 0;
   for (int i = 0; password[i] != '\0'; i++)
      hash += password[i] * (i + 1);
   return hash;
}
void readLine(char *buf, int bufsize) {
   if (fgets(buf, bufsize, stdin) == NULL) {
      buf[0] = '\0';
      return;
   }
   size_t len = strcspn(buf, "\n");
   if (len < strlen(buf)) {
      buf[len] = '\0';
   } else {
      /* No newline found -> input may have overflowed buffer.
Discard rest. */
      int c;
      while ((c = getchar()) != '\n' && c != EOF) ;
   }
}

void getCurrentDate(char *buffer, int buflen) {
   time_t t = time(NULL);
   struct tm *tm_info = localtime(&t);
   strftime(buffer, buflen, "%d-%m-%Y", tm_info);
}

void writeEntry() {
   struct DiaryEntry { char date[20]; char entry[500]; };
   char entry[500];
```

```
    char datebuf[20];

    printf("\nEnter your diary entry (max 500 chars):\n");
    readLine(entry, sizeof(entry));
    if (entry[0] == '\0') {
        printf("No text entered. Entry cancelled.\n");
        return;
    }

    getCurrentDate(datebuf, sizeof(datebuf));
    FILE *fp = fopen("diary.txt", "a");
    if (!fp) {
        printf("Error opening diary file for writing.\n");
        return;
    }
    fprintf(fp, "Date: %s\n%s\n---\n", datebuf, entry);
    fclose(fp);
    printf("Entry saved successfully!\n");
}

void readEntries() {
    FILE *fp = fopen("diary.txt", "r");
    if (!fp) {
        printf("\nNo diary entries found!\n");
        return;
    }

    printf("\n---- YOUR DIARY ENTRIES ----\n\n");
    char line[600];
    while (fgets(line, sizeof(line), fp)) {
        fputs(line, stdout);
    }
    fclose(fp);
}
```

```
void deleteEntry() {
   FILE *fp = fopen("diary.txt", "r");
   if (!fp) {
      printf("\nNo diary entries found!\n");
      return;
   }

   #define MAX_LINES 2000
   #define MAX_LINE_LEN 700
   char (*lines)[MAX_LINE_LEN] = malloc(MAX_LINES *
MAX_LINE_LEN);
   if (!lines) {
      fclose(fp);
      printf("Memory error.\n");
      return;
   }
   int totalLines = 0;
   while (totalLines < MAX_LINES && fgets(lines[totalLines],
MAX_LINE_LEN, fp)) {
      totalLines++;
   }
   fclose(fp);

   /* Find and display entries (lines that start with "Date:") */
   int entryCount = 0;
   for (int i = 0; i < totalLines; i++) {
      if (strncmp(lines[i], "Date:", 5) == 0) {
         entryCount++;
         /* print the date line (without extra newline handling) */
         printf("%d. %s", entryCount, lines[i] + 6); /* +6 to skip
"Date: " */
      }
   }
   if (entryCount == 0) {
      printf("No entries to delete!\n");
```

```
    free(lines);
    return;
  }

  /* Ask which entry to delete */
  char input[32];
  int del = -1;
  while (1) {
    printf("\nEnter entry number to delete (or 0 to cancel): ");
    readLine(input, sizeof(input));
    if (input[0] == '\0') {
      printf("Invalid input. Please enter a number.\n");
      continue;
    }
    int valid = 1;
    for (size_t i = 0; input[i] != '\0'; i++) if (!isdigit((unsigned
char)input[i])) { valid = 0; break; }
    if (!valid) { printf("Invalid input. Enter whole number
only.\n"); continue; }
    del = atoi(input);
    if (del == 0) {
      printf("Delete cancelled.\n");
      free(lines);
      return;
    }
    if (del < 1 || del > entryCount) {
      printf("Invalid entry number. Choose between 1
and %d.\n", entryCount);
      continue;
    }
    break;
  }

  /* Re-write file excluding the selected entry block */
  fp = fopen("diary.txt", "w");
```

```
    if (!fp) {
        printf("Error opening diary file for writing.\n");
        free(lines);
        return;
    }

    int currentEntry = 0;
    for (int i = 0; i < totalLines; i++) {
        if (strncmp(lines[i], "Date:", 5) == 0) {
            currentEntry++;
        }

        if (currentEntry == del) {

            while (i < totalLines) {
                if (strncmp(lines[i], "---", 3) == 0) {
                    break;
                }
                i++;
            }

            if (i < totalLines && strncmp(lines[i], "---", 3) == 0)
continue;
        } else {
            fprintf(fp, "%s", lines[i]);
        }
    }

    fclose(fp);
    free(lines);
    printf("\nEntry deleted successfully!\n");
}

int main() {
```

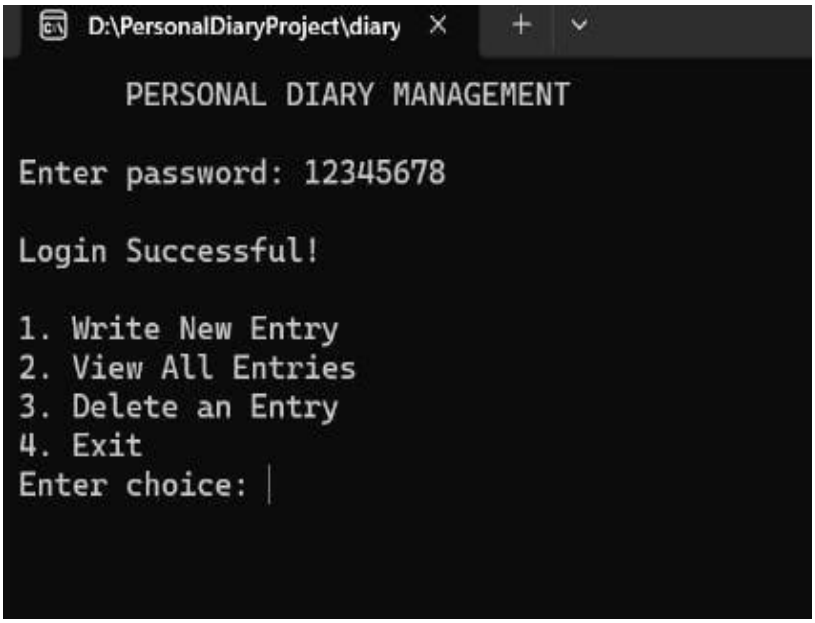
```
    char password[256];
    int savedHash = 0;

    printf("      PERSONAL DIARY MANAGEMENT      \n");


    FILE *pf = fopen("password.dat", "r");
    if (!pf) {

        while (1) {
            printf("No password set. Create a new password (no
spaces): ");
            readLine(password, sizeof(password));
            if (password[0] == '\0') {
                printf("Password cannot be empty. Try again.\n");
                continue;
            }

            int hasSpace = 0;
            for (size_t i = 0; password[i] != '\0'; i++) if
(isspace((unsigned char)password[i])) { hasSpace = 1; break; }
            if (hasSpace) {
                printf("Password must not contain spaces. Try
again.\n");
                continue;
            }
            break;
        }
        savedHash = simpleHash(password);
        pf = fopen("password.dat", "w");
        if (!pf) {
            printf("Error creating password file.\n");
            return 1;
        }
        fprintf(pf, "%d", savedHash);
```

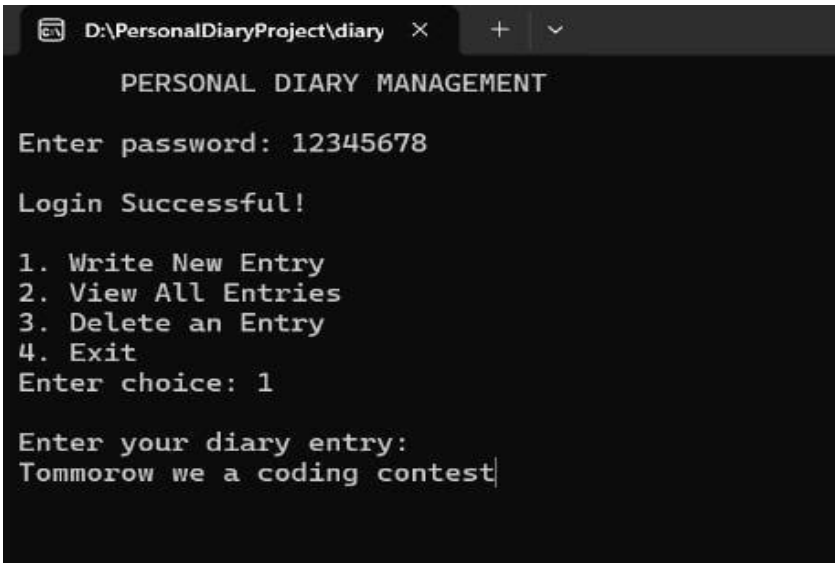
```
    fclose(pf);
    printf("Password created successfully!\n");
} else {
    if (fscanf(pf, "%d", &savedHash) != 1) {
        fclose(pf);
        printf("Password file corrupted.\n");
        return 1;
    }
    fclose(pf);
}


while (1) {
    printf("\nEnter password: ");
    readLine(password, sizeof(password));
    if (password[0] == '\0') {
        printf("Incorrect Password! Try again.\n");
        continue;
    }
    if (simpleHash(password) == savedHash) {
        printf("\nLogin Successful!\n");
        break;
    } else {
        printf("Incorrect Password! Try again.\n");
    }
}


while (1) {
    printf("\n1. Write New Entry\n");
    printf("2. View All Entries\n");
    printf("3. Delete an Entry\n");
    printf("4. Exit\n");

    char input[32];
```

```
printf("Enter choice: ");
readLine(input, sizeof(input));

if (input[0] == '\0') {
   printf("Invalid input! Enter a whole number only.\n");
   continue;
}

int valid = 1;
for (size_t i = 0; input[i] != '\0'; i++) {
   if (!isdigit((unsigned char)input[i])) { valid = 0; break; }
}
if (!valid) {
   printf("Invalid input! Enter a whole number only.\n");
   continue;
}

int choice = atoi(input);
switch (choice) {
   case 1:
      writeEntry();
      break;
   case 2:
      readEntries();
      break;
   case 3:
      deleteEntry();
      break;
   case 4:
      printf("Exiting... Goodbye!\n");
      return 0;
   default:
      printf("Invalid choice! Try again.\n");
      break;
}
```

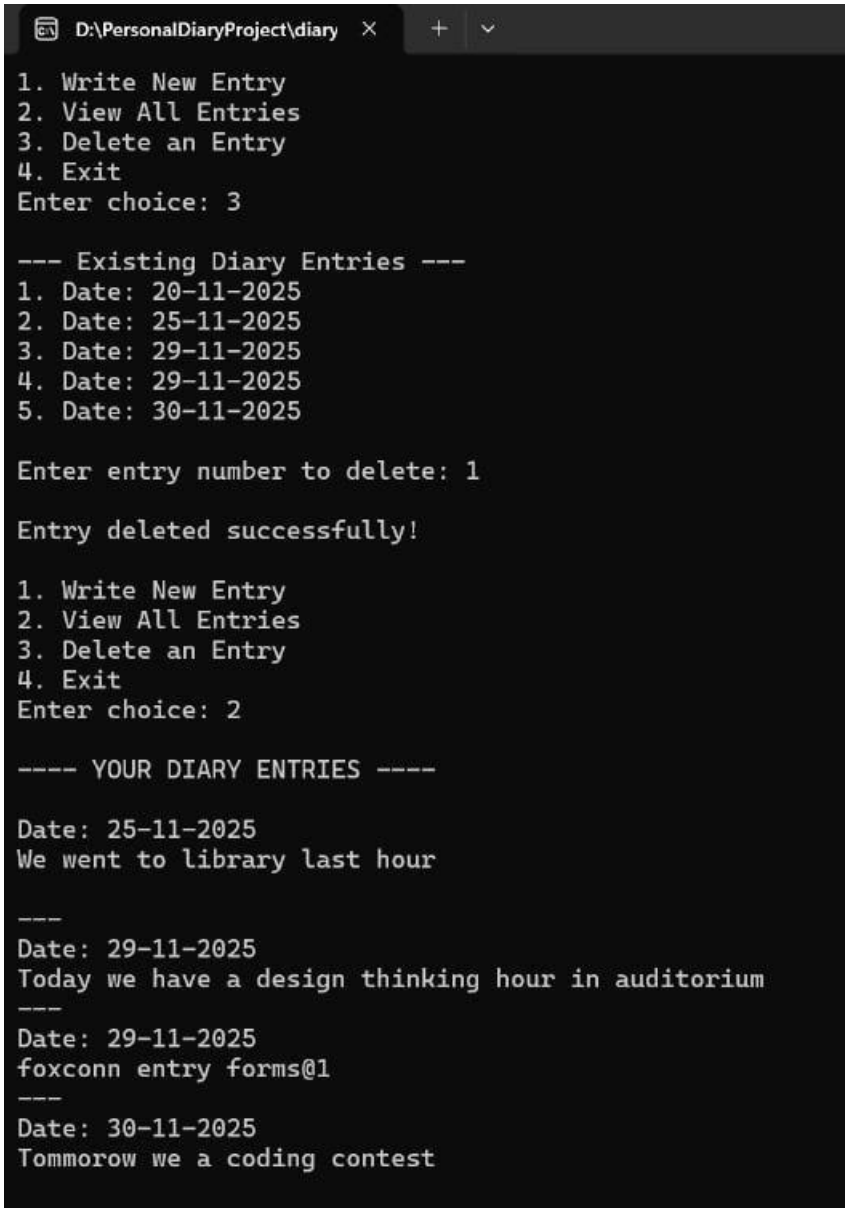
```
    return 0;
}
```

OUTPUT:





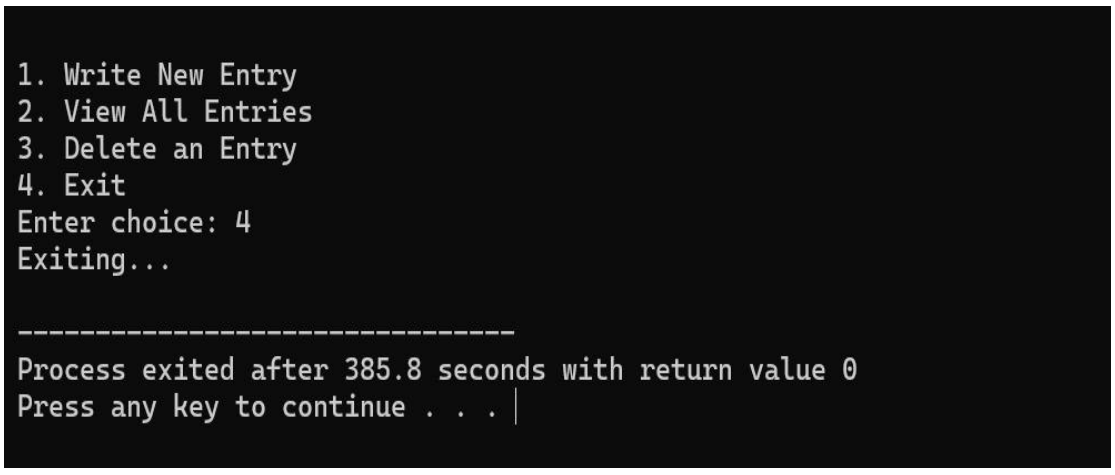
```
1. Write New Entry
2. View All Entries
3. Delete an Entry
4. Exit
Enter choice: 2

---- YOUR DIARY ENTRIES ----

Date: 20-11-2025
Today we learn about oracle

---
Date: 25-11-2025
We went to library last hour

---
Date: 29-11-2025
Today we have a design thinking hour in auditorium
---
Date: 29-11-2025
foxconn entry forms@1
---
Date: 30-11-2025
Tommorow we a coding contest
---
```


D:\PersonalDiaryProject\diary
1. Write New Entry
2. View All Entries
3. Delete an Entry
4. Exit
Enter choice: 3
--- Existing Diary Entries ---
1. Date: 20-11-2025
2. Date: 25-11-2025
3. Date: 29-11-2025
4. Date: 29-11-2025
5. Date: 30-11-2025
Enter entry number to delete: 1
Entry deleted successfully!
1. Write New Entry
2. View All Entries
3. Delete an Entry
4. Exit
Enter choice: 2
---- YOUR DIARY ENTRIES ----
Date: 25-11-2025
We went to library last hour

Date: 29-11-2025
Today we have a design thinking hour in auditorium

Date: 29-11-2025
foxconn entry forms@1

Date: 30-11-2025
Tommorow we a coding contest


1. Write New Entry
2. View All Entries
3. Delete an Entry
4. Exit
Enter choice: 4
Exiting...

Process exited after 385.8 seconds with return value 0
Press any key to continue . . .