المملكة العربية السعودية
وزارة التعليم
جامعة الملك خالد
الكلية التطبيقية بمحايل عسير

Memories Hub

Alina Ahmed

Amnah Ahmed

Amira Ali

Aisha Ibrahim

Bushra Abdullah

Manar Abdullah

Rawiya Mohammed

Rwabi Mohammed

Walaa Ahmad

Under Supervision of
Dr. Iqrar Ahmad

Abstract:

Memories Hub is a web-based application designed to help users document, organize, and relive their personal memories in a structured and meaningful way. The project aims to provide a digital space where individuals can store their memorable moments with descriptions, dates, and interactive features. By using Python as the main development language along with web technologies, the system allows users to add, view, edit, search, and delete memory entries with ease.

The development of this project followed the Waterfall model, which provided a clear and structured approach through its distinct phases, including requirements analysis, system design, implementation, and testing. Each phase contributed to refining the idea and ensuring that the final product meets user expectations.

The main goal of Memories Hub is to create an intuitive and user-friendly platform for preserving personal experiences. The application focuses on simplicity, functionality, and emotional connection, enabling users to revisit special moments anytime. The outcome is a functional prototype that demonstrates the core features and lays the foundation for future enhancements and scalability.

Acknowledgment:

We begin by praising and thanking Allah, without whom this work would not have been possible. We thank Allah for easing all the difficulties and helping us reach this stage through His grace and care. We ask Allah, the Almighty, to always guide us to what is good and successful, and to grant us the strength and determination to achieve more aspirations and ambitions in the future.

We then extend our sincere gratitude and deep appreciation to Dr. [Iqrar Ahmed] for his continuous support and valuable guidance, which had a significant impact on the completion of this project.

His attention to detail, constructive scientific observations, and constant encouragement played a key role in developing and executing the idea in the best possible way.

His valuable guidance has undoubtedly been a major factor in improving the quality of the work and steering us towards success.

The time and effort he dedicated to following up on every detail and his commitment to offering the best advice were crucial in achieving these excellent results.

We offer him our sincere thanks, appreciation, and respect for his continuous support and efforts.

Table of Content:

Table of Figures:

Introduction:

In today’s fast-paced digital world, people often seek ways to preserve their most meaningful moments and memories. Whether it’s a special day, an emotional event, or a simple everyday experience, capturing and revisiting such memories adds value to our personal lives. With the rise of web technologies, creating a dedicated platform for organizing and reflecting on personal memories has become more accessible and impactful.

This project, titled Memories Hub, aims to provide users with a simple and intuitive web-based application that allows them to store, manage, and revisit their personal experiences. By offering features such as adding, editing, viewing, searching, and deleting memory entries, the system supports users in building a digital memory archive that is both functional and emotionally meaningful.

The idea for this project emerged from the growing need for a digital solution that goes beyond traditional journaling or photo albums. The application focuses not only on organization but also on interaction, allowing users to engage with their past in a personal and structured way.

This introduction provides an overview of the motivation behind the project, its importance, and the value it brings to users seeking a personalized memory management tool.

Chapter 1

Proposal

Project Overview:

The project facilitates the preservation and organization of memories in a safe and easy-to-access way. It allows users to store photos, videos, audio recordings, and notes in a secure digital environment. The system provides features such as categories, automatic reminders, and private sharing with others. The development is based on the Python language to ensure security, efficiency, and ease of use.

This project aims to offer a modern method to preserve and retrieve important memories at any time. It highlights the importance of memories in our lives and their role in documenting and sharing special moments. It also contributes to solving the problem of losing or forgetting important memories due to the absence of a structured digital system. The user experience is enhanced through a simple and easy-to-use interface.

A. Related Works:

Python is an easy-to-learn, versatile programming language used in various fields like software development. It is open-source, compatible with all operating systems like Windows and Linux, and can also be used on mobile devices. Python offers extensive libraries, supports graphical user interfaces, and is a high-level language, making it easy to work with.

Advantages of Python:

1-Easy to read and write.

2-Syntax is similar to English, making it ideal for beginners.

3-Open-source and accessible to all.

4-Works on all operating systems.

Disadvantages of Python:

1-Slower than compiled languages like C/C++.

2-Not the best option for mobile app development.

3-Some weaknesses in database access and runtime errors.

B. Project Goal:

The main idea of the Memories Hub project is to create a digital platform that enables users to save, organize, and revisit their personal memories in one central place. The platform addresses the common problem of scattered or lost memories due to the lack of a dedicated memory management system. Overall, the project aims to enhance the experience of preserving life moments, making it a unique and valuable solution for digital memory keeping.

C.Project Objectives:

This project introduces several innovative elements:

- Comprehensive Storage: Supports various types of memory inputs such as voice recordings, images, and written notes.
- Easy Access: Designed with a user-friendly interface for quick navigation and retrieval.
- Memory Customization: Allows users to organize their memories using categories or tags based on their preferences.
- Memory Sharing: Offers private sharing options with friends and family to foster emotional connection.

Chapter 2

System Analysis and Design

Initiation Phase

The development of Memories Hub begins by identifying the need for a platform that helps users store and organize their digital memories in a secure and easy way. A project team is assigned to create an initial plan, and this idea is documented in a project proposal. After approval, the next phase is the system concept development.

System Concept Development Phase

Once the need for the platform is approved, the approach for implementing the idea is reviewed and evaluated for feasibility and suitability. The scope of the project and its core functionalities—such as memory management, album organization, privacy options, and the ability to share memories—are defined.

Planning Phase

The concept is further developed to describe how the system will operate once implemented, and to assess the impact on users and administrators. To ensure that the product is delivered on-time and within budget, resources, tools, schedules, and review plans are clearly outlined. Security certification also begins here by identifying system security requirements and performing a high-level vulnerability assessment.

Requirements Analysis Phase

User needs are defined, including memory storage (photos, videos, texts), security, and performance, to ensure a smooth experience.

Design Phase

The system structure is planned, including the database, file upload module, memory management, and sorting by date, with security measures in place.

Development Phase

Modules are developed using Python, tested, integrated, and then deployed on a suitable server with technical support available.

Integration and Test Phase

In this phase, the components of Memories Hub are integrated and systematically tested. Users try out the system to ensure that all features—such as photo uploads, creating memories, and sharing content—are functioning smoothly. Before the final launch, security and performance tests are conducted to guarantee the site's stability.

Implementation Phase

Once testing is successfully completed, the site is deployed to the production environment so it's accessible to users. This involves configuring servers, setting up the database, and connecting the site to any necessary external services (e.g., cloud storage). After deployment, the site's performance is monitored to ensure a seamless user experience.

Operations and Maintenance Phase

The site remains live and is continuously monitored to maintain optimal performance. User feedback is collected to guide improvements—such as faster loading times or the addition of new features. Regular

updates are carried out to keep the site compatible with the latest technologies and to protect stored data. If significant changes are needed, some earlier planning and development steps may be revisited.

SDLC Objectives

This guide was developed to establish best practices for system developers, project managers, and stakeholders involved in the “Memories Hub” project. The main objectives include:

- Reducing the risk of data loss and ensuring secure memory storage.
- Addressing system and data requirements throughout the platform’s lifecycle.
- Identifying technical and user experience challenges early.
- Providing cost transparency for efficient decision-making.
- Setting realistic expectations regarding system capabilities and limitations.
- Balancing development efforts across technical, financial, and usability aspects.

Methodology

The “Memories Hub” project follows the Waterfall methodology, ensuring a structured and sequential development process through these stages:

- Requirements Gathering: Identify and document key features, such as image uploads, categorization, sharing, and privacy settings.
- System Design: Define the platform’s architecture, database structure, and secure data management mechanisms.
- Implementation: Develop the web-based system using Python, focusing on seamless functionality and an intuitive user experience.
- Testing: Conduct unit, system, and integration tests to ensure smooth performance across different devices and use cases.
- Deployment: Prepare the system for launch by setting up servers, databases, and necessary configurations to ensure stability.

Hardware Tools Used:

- Processor: Intel Core i5 or higher
- Memory (RAM): 8GB
- Storage: 256GB SSD

Software Tools Used:

- Python
- Django
- SQLite
- HTML, CSS, JavaScript
- Windows 10 / Linux

Software Requirement Specification

The Software Requirement Specification (SRS) serves as the foundation for the development of "Memories Hub", outlining the essential system requirements and specifications. Initially, it may be challenging to visualize all aspects of the project, but as the system grows in complexity, documenting and analyzing the requirements becomes crucial to ensure a well-structured and efficient system.

This document aims to transform the project's vision and ideas into a formal specification containing all the necessary requirements in a clear and organized manner. This provides a solid reference for the development team, reducing errors and ensuring that the final product aligns with user expectations.

Data Flow Diagram

The Data Flow Diagram (DFD) illustrates how information flows within “Memories Hub”, showcasing the movement of data between users, interfaces, and the database.

The system is represented through a graphical diagram that highlights data sources, the processes applied to the data, storage locations, and final destinations.

To ensure clarity in design, the diagram is divided into different levels of detail. Additional details, such as data volume, frequency, and user interactions, may be specified in supplementary diagrams or within a data dictionary.

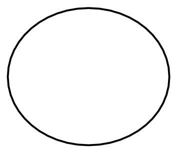
The DFD plays a crucial role in system analysis, helping define the core structure of the project and clarify relationships between its components, making the design and development process more efficient.

Terminology used in DFD

- **Process**

Processes are the foundation of building programs in Python, and are used to perform various tasks such as calculations, analysis, data processing, files, and networking.

Graphical Representation:

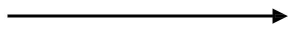


- **Data Streams**

Data streams in Python are a method of reading or writing data sequentially.

Data streams can be used to manipulate files, links, databases, and other resources.

Graphical Representation:



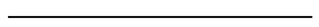
- **Actors**

An actor is an object that performs calculations on a data stream via the desired result or integer values.

- **Data store**

A data store is a useful object that stores information where information is stored for later access.

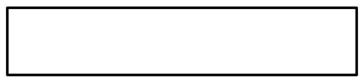
Graphical Representation:



- **External Entity**

An external entity in Python is something used for a specific function but not part of the program.

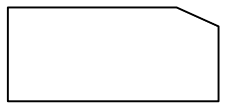
Graphical Representation:



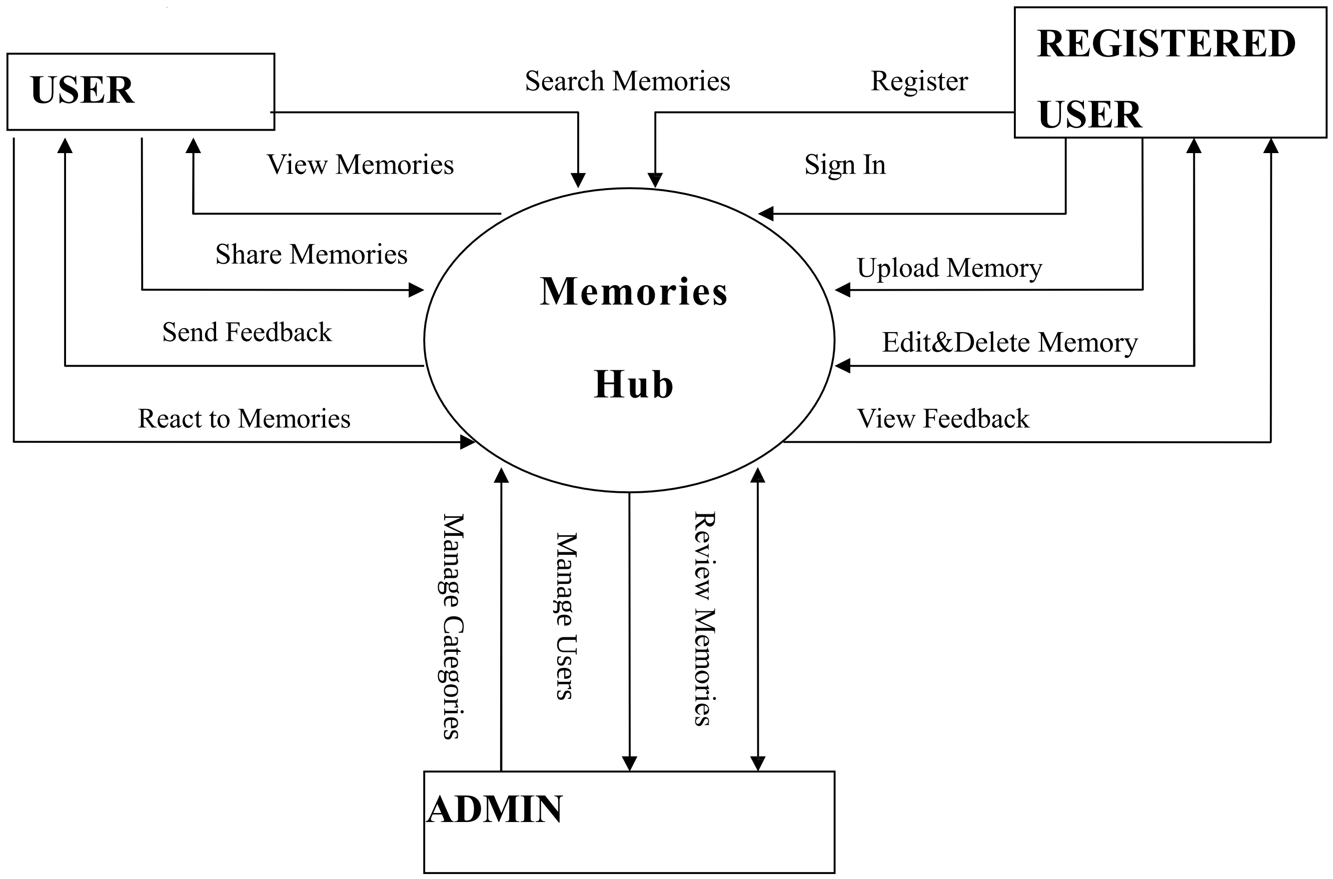
- **Output Symbol**

A value returned at program termination to indicate success or failure.

Graphical Representation:



Context Level DFD



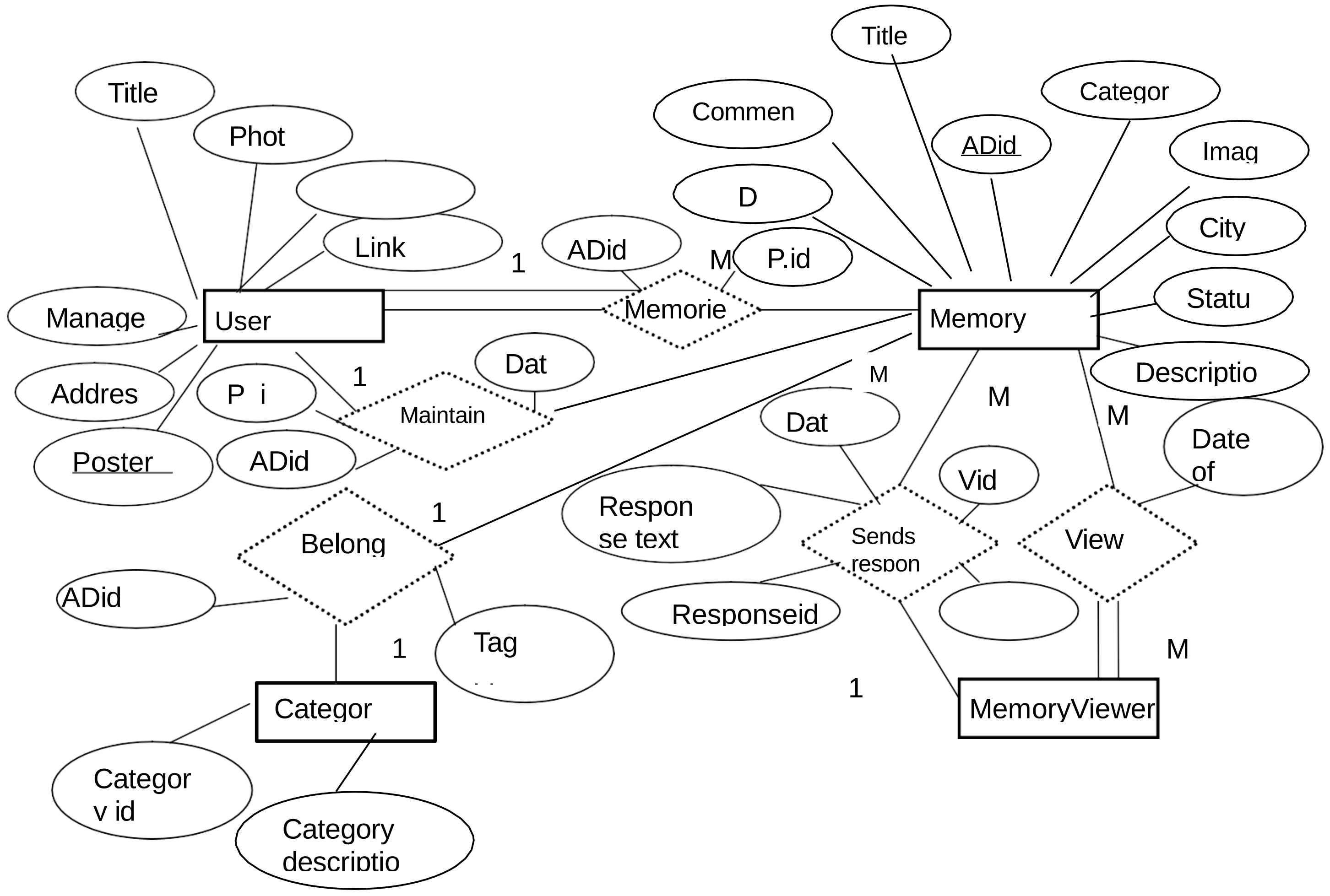
E.R Diagram

A data flow diagram (DFD) is a tool used to graphically represent a system to integrate data, data stores, processes, and data sources/destinations. It is similar to a DFD map of the road, showing a grid of type representing all the different parties, with different details for the layout of different hierarchical lines. This extends to the different levels of 'leveling' or 'splitting'.

Like a road map, there is no jumping point or stopping point, no timing or steps to get somewhere. We simply know that a data path must exist at some point that it will be required. Route map of all existing routes or to check the necessity of the route.

Details that do not appear in different levels of DFD, such as cultivation, replication, etc., appear clearly or in data evaluation. For example, you may be able to store data stores in a data dictionary.

A DFD uses numbers from type to represent the Internet, also known as a “bubble chart.” He called for clarification of the system and the main changes aimed at designing the system. So, there is a starting point for the design phase which is broken down into detail level requirements.



Title
Title
Categor
Commen
Phot
ADid
Imag
D
City
Link
1
ADid
M
P.id
Statu
Manage
User
Memorie
Memory
Dat
M
Descriptio
1
Addres
P i
Maintain
Dat
M
M
Date of
Poster
ADid
Vid
Respon se text
1
Belong
Sends respon
View
ADid
Responseid
Tag
1
M
1
Categor
MemoryViewer
Categor y id
Category descriptio

Chapter 3

Implementation

Introduction:

After completing the planning and analysis phase in the first stage, during which the project’s objectives, scope, and necessary requirements were defined, we now move on to the second phase: the implementation stage. In this phase, theoretical plans and ideas will be transformed into a tangible reality through the development and execution of various components and functions of the Memories Hub system.

This phase will focus on carrying out the practical steps needed to achieve the project’s goals, including developing software interfaces, creating databases, and implementing functions for managing and organizing users’ digital memories. Additionally, emphasis will be placed on ensuring the security and confidentiality of user data while providing a smooth and efficient user experience.

The objective of this phase is to fully and effectively build the Memories Hub system, with a strong focus on achieving the project’s goals and meeting user needs.

Implementation Steps:

1. The official Trinket website was accessed through the internet browser on the used device (laptop or iPad). This platform provides an interactive environment that supports writing and running Python code directly through the browser, without the need to install additional software.

2. After accessing the homepage of the website, a new project was created, and the Python 3 environment was selected, where the interactive code editor opened directly.

3. Within the editor, the main file was named main.py, which contains the main code responsible for implementing the system's functions.

4. The process of writing the Memories Hub project code began, including the core functions such as: adding memories, viewing stored memories, and searching for memories by title.

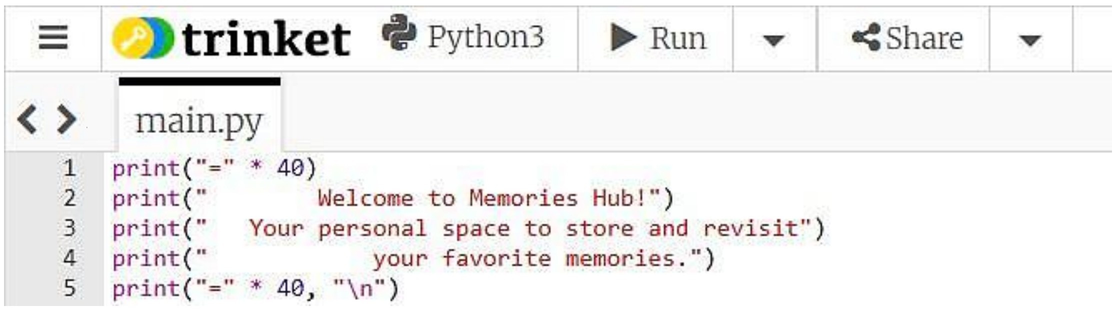
5. During the coding process, all functions were continuously tested using the Run button provided by the platform, ensuring all functions worked correctly and identifying any programming errors to fix them immediately.

6. The Trinket platform provided a suitable environment for writing and running the code with ease of modification and saving, which accelerated the development process and ensured safe storage of the work.

7. After completing the coding and implementing the core functions, the system was reviewed to ensure its compatibility with the requirements previously defined in the first phase of the project.

8. At the end of this phase, the project was saved on the Trinket platform for easy future access, whether for further development or additional enhancements in the upcoming project phases.

1. Welcome code in the program



```python
print("=" * 40)
print("        Welcome to Memories Hub!")
print("   Your personal space to store and revisit")
print("            your favorite memories.")
print("=" * 40, "\n")
```

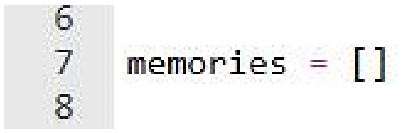
This section is responsible for displaying the welcome message when the program starts.

It uses print("=" * 40) to create a decorative border to make the interface visually appealing.

The message welcomes the user to “Memories Hub” and explains that it’s a personal space to save and revisit memories.

This is purely for design and user engagement purposes, with no functional role in memory management.

2. Memories List Code



```
memories = []
```

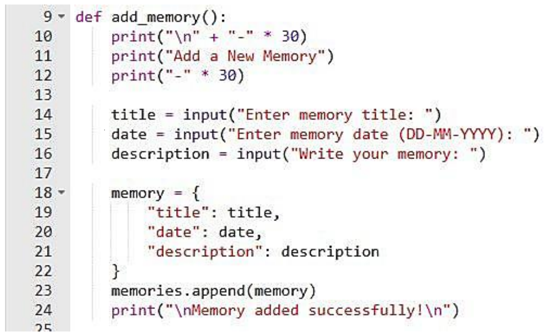
This is the initialization part where an empty list called memories is created.

This list will act as the temporary database during the program’s execution.

Every time the user adds a memory, it gets stored in this list as a dictionary (title, date, and description).

Since this program does not use files or external databases, this list holds all the data until the program is closed.

3. Code to add a new memory



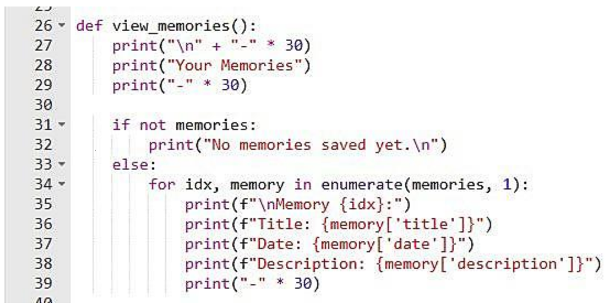
```
def add_memory():
    print("\n" + "-" * 30)
    print("Add a New Memory")
    print("-" * 30)

    title = input("Enter memory title: ")
    date = input("Enter memory date (DD-MM-YYYY): ")
    description = input("Write your memory: ")

    memory = {
        "title": title,
        "date": date,
        "description": description
    }
    memories.append(memory)
    print("\nMemory added successfully!\n")
```

This is the add_memory() function.

- It starts by displaying a section header to show the user they are in the “Add Memory” section.
- It asks the user to enter:
- Title of the memory.
- Date in the format DD-MM-YYYY.
- Description to capture the details of the memory.
- All of this data is stored in a dictionary with keys: title, date, description.
- This dictionary is appended to the memories list.
- Finally, it prints a confirmation message that the memory was added successfully.

This code handles the main data entry process in the program.

4. Memories Display Code



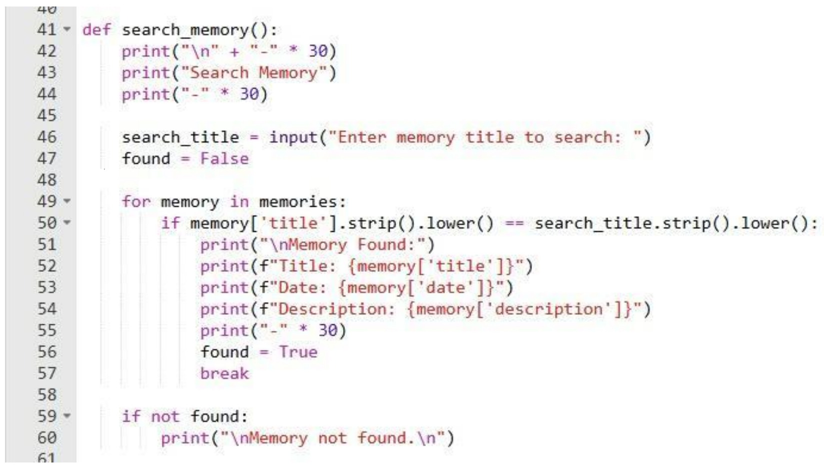
```python
def view_memories():
    print("\n" + "-" * 30)
    print("Your Memories")
    print("-" * 30)

    if not memories:
        print("No memories saved yet.\n")
    else:
        for idx, memory in enumerate(memories, 1):
            print(f"\nMemory {idx}:")
            print(f"Title: {memory['title']}")
            print(f"Date: {memory['date']}")
            print(f"Description: {memory['description']}")
            print("-" * 30)
```

This is the view_memories() function.

- It prints a section header to indicate that the memories will be displayed.
- If the memories list is empty, it prints: No memories saved yet.
- If there are memories, it loops through the list using enumerate() so that each memory gets a number (1, 2, 3…).
- Each memory's title, date, and description is printed.
- A separator line is printed after each memory to make the output clear.

This function is responsible for showing all saved memories in an organized format.

5. Search code for a memory


```
def search_memory():
    print("\n" + "-" * 30)
    print("Search Memory")
    print("-" * 30)

    search_title = input("Enter memory title to search: ")
    found = False

    for memory in memories:
        if memory['title'].strip().lower() == search_title.strip().lower():
            print("\nMemory Found:")
            print(f"Title: {memory['title']}")
            print(f"Date: {memory['date']}")
            print(f"Description: {memory['description']}")
            print("-" * 30)
            found = True
            break

    if not found:
        print("\nMemory not found.\n")
```

This is the search_memory() function.

- A section header is printed.
- The user is asked to enter the title of the memory they want to find.
- The function loops through the memories list.
- It compares each memory’s title (case insensitive) with the search term.
- If a match is found, it prints the title, date, and description.
- If no match is found, it prints: Memory not found.

This code allows the user to quickly find a specific memory by searching for its title.

Chapter 4

Testing

Introduction:

In this stage of the “Memories Hub” project, the practical implementation of the system — which was planned and documented in the previous phases — has been completed. The main goal of the project is to allow users to record their personal memories by entering the title, date, and description of each memory, and to access them later in an organized and simple way.

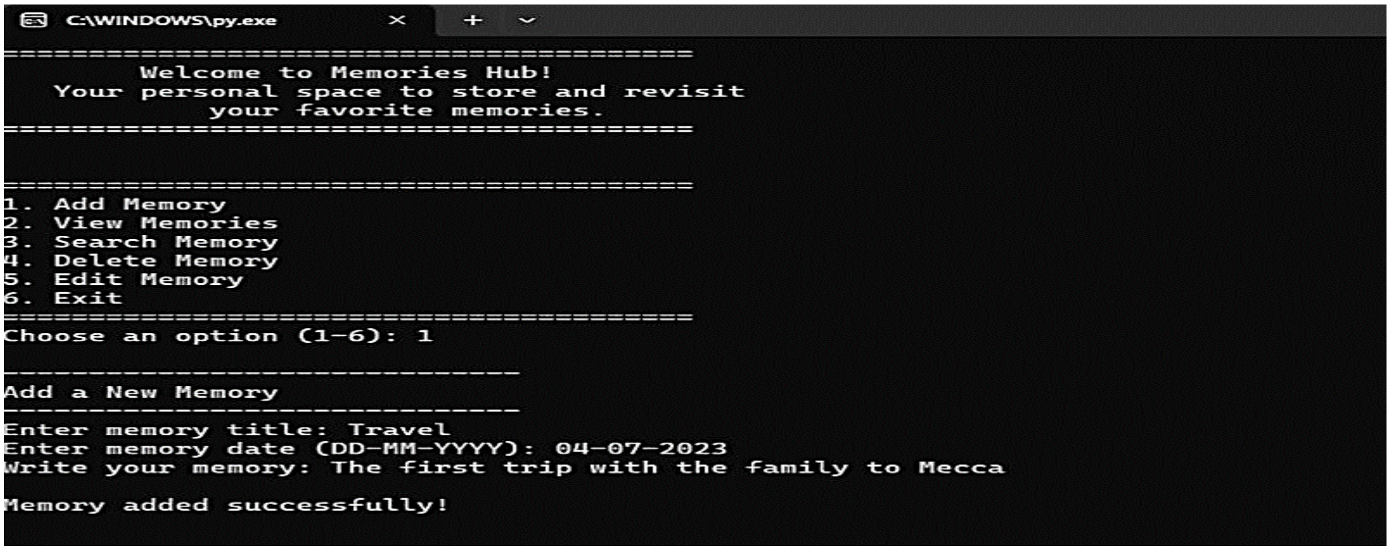
The system was developed using the Python programming language and executed entirely through the Trinket platform, without relying on any graphical user interfaces. The implementation includes all essential features such as adding memories, viewing them, searching, deleting, and editing.

The system was also tested using different data sets to ensure its correct functionality and stability. Additionally, various error scenarios were tested, such as entering invalid data or unsupported commands, to evaluate how well the program handles errors.

Output Screenshots:

Below are the output screenshots captured during the testing phase. Each image demonstrates a key functionality of the system, including adding, viewing, editing, deleting, and searching for memories.

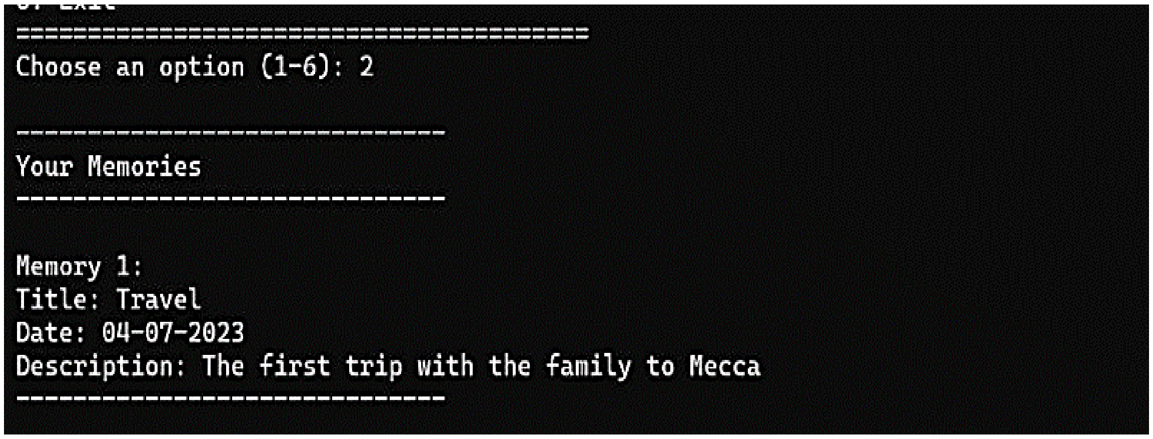
6. Adding a New Memory



```
========================================
        Welcome to Memories Hub!
   Your personal space to store and revisit
            your favorite memories.
========================================


========================================
1. Add Memory
2. View Memories
3. Search Memory
4. Delete Memory
5. Edit Memory
6. Exit
========================================
Choose an option (1-6): 1

------------------------------
Add a New Memory
------------------------------
Enter memory title: Travel
Enter memory date (DD-MM-YYYY): 04-07-2023
Write your memory: The first trip with the family to Mecca

Memory added successfully!
```

This screenshot demonstrates the process of adding a new memory to the system.The user is guided to input the memory's title, date, and a brief description. Once all required fields are filled in, the system stores the memory successfully and displays a confirmation message. This functionality ensures that users can easily document their personal moments for future reference.

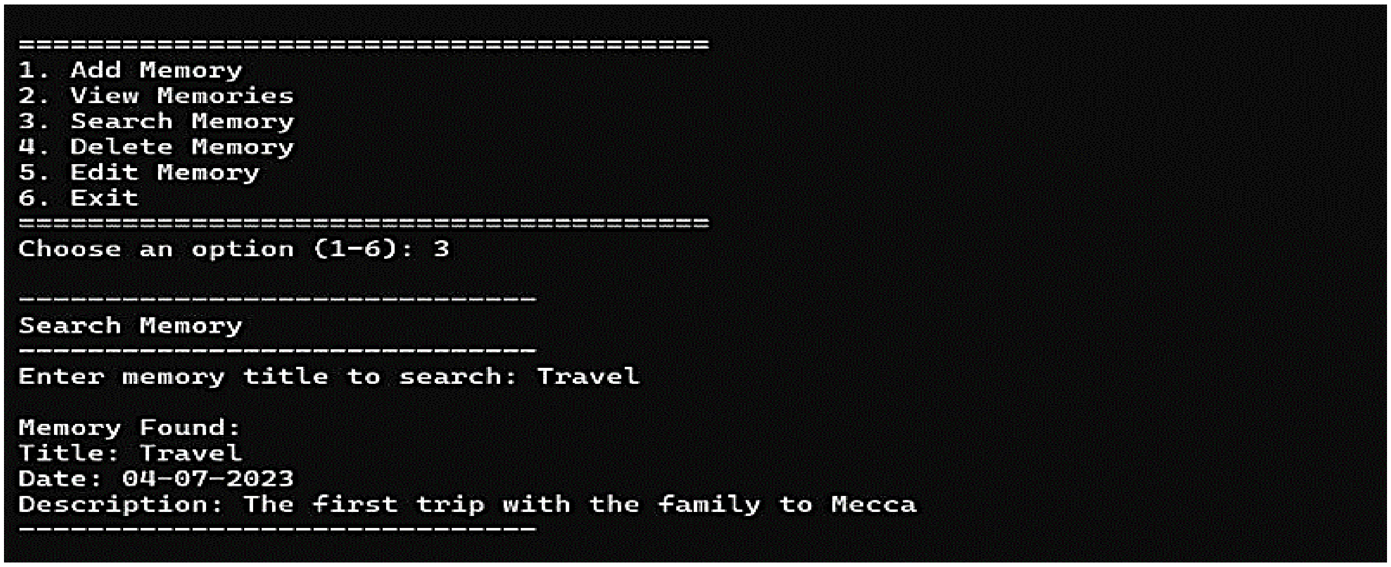
7. Viewing Stored Memories



```
========================================
Choose an option (1-6): 2

------------------------------
Your Memories
------------------------------

Memory 1:
Title: Travel
Date: 04-07-2023
Description: The first trip with the family to Mecca
------------------------------
```

The output in this image shows a list of all stored memories. Each memory is displayed with its corresponding title, date, and description in a clean and organized format. This feature allows users to review their past memories at any time and verify the accuracy of the stored data.

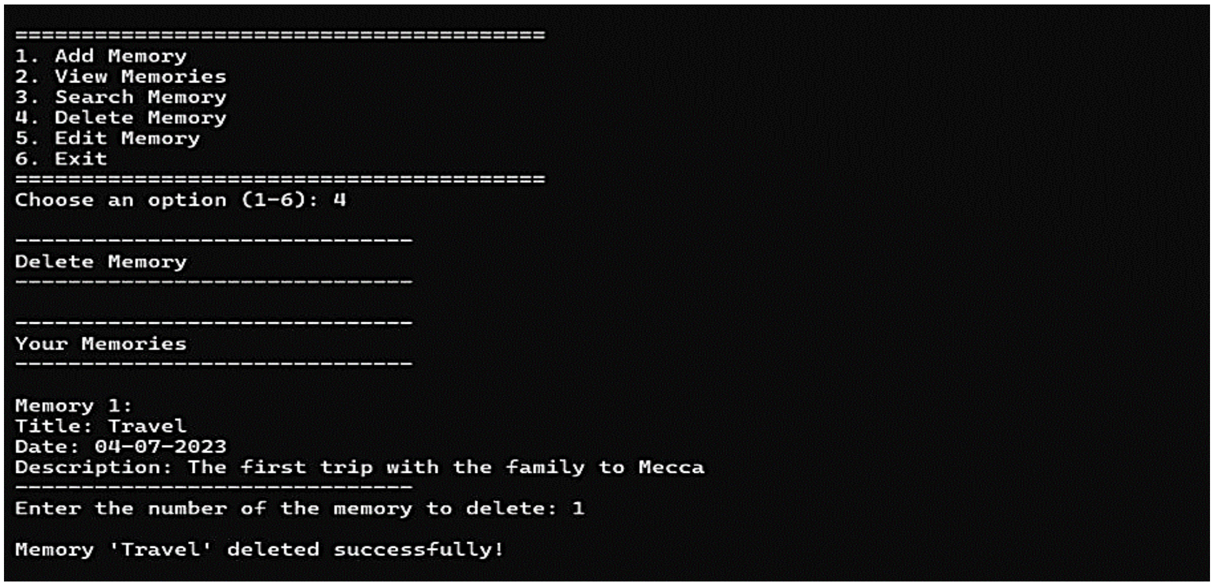
8. Searching for a Specific Memory



```
========================================
1. Add Memory
2. View Memories
3. Search Memory
4. Delete Memory
5. Edit Memory
6. Exit
========================================
Choose an option (1-6): 3

------------------------------
Search Memory
------------------------------
Enter memory title to search: Travel

Memory Found:
Title: Travel
Date: 04-07-2023
Description: The first trip with the family to Mecca
------------------------------
```

This image captures the system’s search functionality. The user is able to enter a keyword or a specific title to locate a particular memory. If the memory exists, the system retrieves and displays the details. This feature improves user experience by offering quick access to desired content without needing to scroll through the entire list.

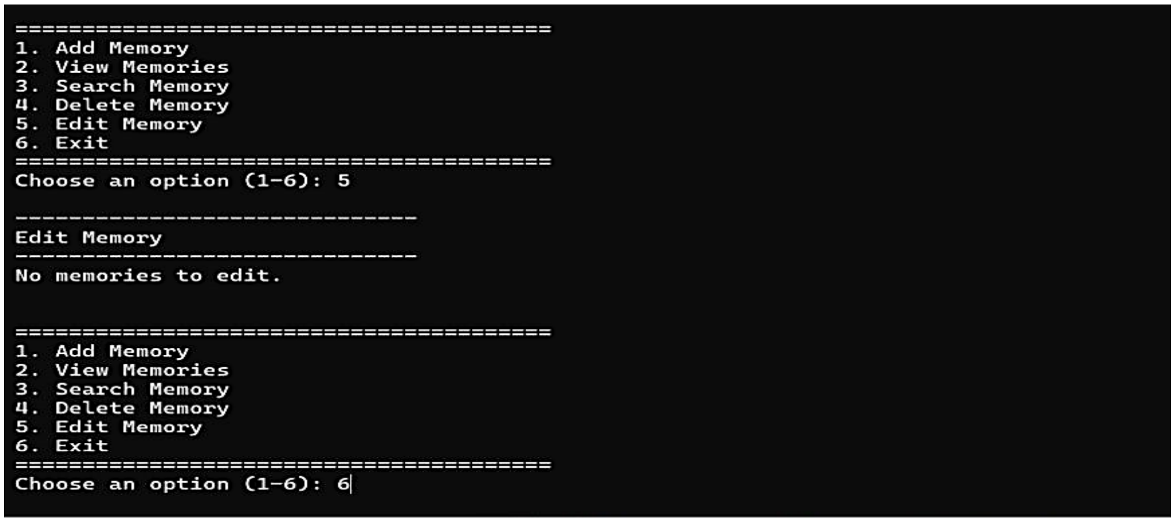
9. Deleting a Memory from the List



```
========================================
1. Add Memory
2. View Memories
3. Search Memory
4. Delete Memory
5. Edit Memory
6. Exit
========================================
Choose an option (1-6): 4

------------------------------
Delete Memory
------------------------------

------------------------------
Your Memories
------------------------------

Memory 1:
Title: Travel
Date: 04-07-2023
Description: The first trip with the family to Mecca
------------------------------
Enter the number of the memory to delete: 1

Memory 'Travel' deleted successfully!
```

In this screenshot, the delete operation is performed. The user enters the title of the memory they wish to remove, and the system confirms the deletion by removing the corresponding entry from storage. This functionality is essential for maintaining up-to-date and relevant data in the system.

10. Editing an Existing Memory



```
========================================
1. Add Memory
2. View Memories
3. Search Memory
4. Delete Memory
5. Edit Memory
6. Exit
========================================
Choose an option (1-6): 5

------------------------------
Edit Memory
------------------------------
No memories to edit.


========================================
1. Add Memory
2. View Memories
3. Search Memory
4. Delete Memory
5. Edit Memory
6. Exit
========================================
Choose an option (1-6): 6
```

This section illustrates the edit feature of the system. The user selects a memory and is allowed to modify its title, date, or description. After updating the details, the system successfully saves the changes and displays a confirmation. This ensures flexibility for the user in managing their recorded memories as needed.

Conclusion:

In conclusion, the “Memories Hub” project successfully achieved its primary goal of providing users with a digital platform to preserve and manage their personal memories in a secure and organized manner. Through the implementation of core features such as adding, editing, deleting, viewing, and searching for memories, the system proved to be functional and user-friendly.

The development process followed the Waterfall methodology, ensuring a structured and systematic approach across all stages of the Software Development Life Cycle. From initial planning and design to testing and deployment, each phase contributed to building a reliable and effective system.

This project not only enhanced technical skills in Python programming and system analysis but also highlighted the importance of digital memory preservation in today’s fast-paced world. Future improvements may include adding a graphical user interface, integrating cloud storage, and expanding sharing capabilities to further enhance the user experience.

References:

1. Python Official Documentation. https://docs.python.org

2. Trinket – Python in the Browser. https://trinket.io

3. W3Schools – Python Tutorial. https://www.w3schools.com/python/

4. Real Python – Learn Python Programming. https://realpython.com

5. Stack Overflow – Community Help and Code Snippets. https://stackoverflow.com