LAB REPORT

Submitted by

Ishaan Manhas (RA2111027010031)
Ansab Aalim(RA2111027010030)
Aryan Chaudhary (RA2111027010041)

Under the Guidance of

Dr Arthi K
Associate Professor, DSBS

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING

with specialization in Big Data Analytics

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

MAY 2023

COLLEGE OF ENGINEERING & TECHNOLOGY
SRM INSTITUTE OF SCIENCE & TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUR – 603 203
Chengalpattu District

BONAFIDE CERTIFICATE

Register No. _____ Certified to be the bonafide work done by _____of II Year/IV Sem B.Tech Degree Course in the **Practical Software Software Engineering and Project Management 18CSC206J** in **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY,** Kattankulathur during the academic year 2022 – 2023.

LAB INCHARGE **Head of the Department**

Dr Arthi K

Professor

Department of Computing Technologies

SRMIST – KTR.

ABSTRACT

A file manager allows users to manage files and directories through a user interface. While files can be managed through the commandline, not all users know how to do that. With a file manager, users can arrange, access, and administer their files and directories properly without knowing how to use the command line. Some of the tasks a file manager allows users to perform includes copying, moving, and renaming files or directories. These tasks can be done using the modulation concept.

TABLE OF CONTENTS

PROBLEM STATEMENT

The main objective of the file manager project is to give users an interface to manage their files.
Users want a file manager that has a file management tool that looks good and is easy to use.
You can use the PySimpleGUI library to create unique user interfaces with a powerful widget,
without having to deal with a lot of complexity.
Your users should be able to perform simple tasks like creating new directories or empty text files.
They should also be able to copy and move files or directories.
The sys, os, and shutil libraries will be quite useful for this project, as they can be used to execute
actions on the files in the background, while the user clicks away.
The grid and list views are popular views today, so you can implement both in the application. This
gives the user the option to choose which view option is suitable for them.
.

STAKEHOLDERS & PROCESS MODELS

OUR STAKEHOLDERS:-

STAKEHOLDER	ACTIVITY/AREA /PHASE	INTEREST	INFLUENCE	PRIORITY
Business owners	Designing the online shopping platform interface	Effective user interface design to improve user engagement and increase revenue	High influence on the website's design as the primary user	High priority as the website's success directly affects their business
Web developers	Integrating the interface into the platform	An interface that is easy to integrate and customize	Moderate influence as they may have technical knowledge to provide input	Moderate priority as their primary concern is integration rather than the interface design
Customers	Using the online shopping platform	An interface that is easy to use and navigate to make purchases	Low influence as they are not involved in the development process	High priority as their satisfaction with the interface affects user engagement and revenue
Investors	Funding the development of the website	A successful website that generates revenue and profit	High influence as they have a financial interest in the success of the website	High priority as they have invested capital and want a return on investment
Designers	Creating the tools and resources for the website	A design that is visually appealing and provides necessary functionality	High influence as they are responsible for designing the website	Moderate priority as their primary concern is designing the website and providing resources
Quality assurance teams	Ensuring the website and platform function correctly	An interface that functions correctly and efficiently	Low influence as their role is to ensure the website functions correctly	High priority as they ensure the interface is working correctly for customers

OUR PROCESS MODEL:-

Our chosen process model is Rapid Application Development Model(RAD).

RAD is a linear sequential software development process model that emphasizes a concise development cycle using an element based construction approach. If the requirements are well understood and described, and the project scope is a constraint, the RAD process enables a development team to create a fully functional system within a concise time period.

RAD (Rapid Application Development) is a concept that products can be developed faster and of higher quality through:

Gathering requirements using workshops or focus groups
Prototyping and early, reiterative user testing of designs
The re-use of software components
A rigidly paced schedule that refers design improvements to the next product version
Less formality in reviews and other team communication

IDENTIFYING REQUIREMENTS

Requirement analysis is a crucial phase of software development, including for a website that designs an online shopping platform interface. It helps ensure that the final product meets stakeholder needs, prevents potential issues early in the project, defines project scope, and enables resource allocation. In short, it ensures project success by identifying requirements, potential issues, and providing a clear understanding of the project scope.

System Requirements:

System requirements are the technical specifications needed for the online shopping platform to function correctly. Here are some key system requirements that the platform should meet:

1) Hardware:

a) Server infrastructure to host the platform and manage database resources.
b) Sufficient network infrastructure to support high volumes of traffic.
c) Backup and disaster recovery systems to ensure data safety.

2) Software:

a) Operating system and middleware to run the platform (e.g., Linux, Apache, MySQL, PHP,Windows)
b) Development tools and frameworks for building the platform.
c) Integration with third-party APIs for payment processing and shipping services
d) Data analytics and reporting tools to track key metrics and performance.

3) Security:

a) Secure authentication and authorization systems for users and administrators
b) Encryption of sensitive data, such as credit card details and personal information
c) Regular security testing and updates to identify and address vulnerabilities.

4) Performance:

a) Fast loading times and smooth, responsive user interface
b) Scalability to support increasing numbers of users and transactions.
c) Reliable uptime and minimal downtime for maintenance or updates

5) Compatibility:

a) Support for multiple browsers and operating systems
b) Responsive design to optimize for different screen sizes and devices.
c) Accessibility features to make the platform usable for all users, including those with disabilities.

Functional Requirements:

Functional requirements describe the specific features and functionality that the online shopping platform should provide to its users. Here are some key functional requirements that the platform should meet:

1) Allow the User to add and remove new files.

2) Allow the user to search for files based on title, publication date, author, etc. and find their location in the library

3) **Users can request, access, or prioritize a file**

4) Users can add and manage the files.

5) The system can move files

Non-Functional Requirements:

Non-functional requirements are the qualities that the online shopping platform should have to ensure a good user experience and meet technical and operational standards. Here are some key non-functional requirements that the platform should meet:

1. **Usability -** It is the main non-functional requirement for a file management system. The UI should be simple enough for everyone to understand and get the relevant information without any special training. Different languages can be provided based on the requirements.

2. **Accuracy -** It is another important non-functional requirement for the file management system. The data stored about the files and the space calculated should be correct, consistent, and reliable.

3. **Availability -** The System should be available for the duration when the interface operates and must be recovered within a short period if it fails. The system should respond to the requests within a fraction.

4. **Maintainability-** The software should be easily maintainable and adding new features and making changes to the software must be as simple as possible. In addition to this, the software must also be portable.

5. **Scalability**

a. Ensure that the platform can scale to support increasing numbers of users and transactions without performance degradation.

6. Maintainability

a. Ensure that the platform is easy to maintain and update with minimal effort.
b. Provide documentation and support for developers and administrators.

PROJECT PLAN & EFFORT

The project plan for a website that helps design an online shopping platform interface outlines how the project will be executed, monitored, and controlled. It includes information such as the project scope, timelines, budget, resources, and risk management strategies.

PROJECT MANAGEMENT PLAN:-

FOCUS AREA	DETAILS
Integration Management	Project Team Structure: 1. Team Manager - Ishaan Manhas 2. Team Members - Aryan Chaudhary & Ansab Aalim Roles & Responsibilities of Team: 1. Ishaan Manhas - Module integration and developer 2. Ansab Aalim - Module code engineer and developer 3. Aryan Chaudhary - Software testing engineer and documentation lead

Quality Management	Quality Assurance: Quality assurance will be managed including governance, roles and responsibilities, tools and techniques and reporting. Application will be tested multiple times and feedback will be taken from various users. Quality Control: Survey will be taken and prototype testing will be done to ensure proper that all mechanisms are working perfectly
Stakeholder	Senior Staff - Dr. K Arthi Project Manager - Ishaan Manhas Project Members - Ansab Aalim and Aryan Chaudhary Resource Manager End Users

COST & EFFORT ESTIMATE:-

ACTIVITY DESCRIPTION	SUB-TASK	DESCRIPTION	EFFORT (IN HOURS)	COST IN INR
Design the user screen	E1R1A1T1	Confirm the user requirements	3	1,500
	E1R1A1T2	Development of frontend screen	3	1,500
Backend Development	E1R1A2T1	Development of various modules	5	2,500
	E1R1A2T2	Dcoumentation	12	6,000
Total			23	11,500

INFRASTRUCTURE / RESOURCE COST:-

INFRASTRUCTURE REQUIREMENT	QTY	COST PER QUANTITY PER ANNUM	COST PER ITEM
PCs	3	1 lakh 53 thousand	4 lakh 60 thousand
Network Routers and Connection	3	5 thousand	15 thousand

MAINTENANCE & SUPPORT COST:-

CATEGORY	DETAILS	QTY	COST PER QTY PER ANNUM	COST PER ITEM
People	Team members and manager. Developers, DB Manager and Documentation lead and STE	3	0	0
License	Windows 11 and MAC OS Database IDE	3	10,000	1,00,000
Infrastructure	Server, Storage and Network	3	5,000	15,000
Total Maintenance and Support Cost				1,15,000

WORK BREAKDOWN STRUCTURE & RISK ANALYSIS

A Work Breakdown Structure is a hierarchical decomposition of the project deliverables, which breaks down the project into manageable and achievable work packages. The importance of the WBS lies in its ability to provide a clear and structured view of the project, helping to identify and organize the tasks required to achieve project objectives.

TEAM MEMBER IDENTIFICATION:-

NAME	ROLE	RESPONSIBILITIES
Ishaan Manhas	Project Manager	Manage the project
Aryan Chaudhary	Business Analyst	Elicitation and documentation of requirements, functional and non-functional analysis, use case development
Ishaan Manhas	Technical Lead	Technology and architecture planning and implementation, software development oversight, quality assurance and testing
Ansab Aalim	UI/UX Designer	User interface and user experience design, wireframing,prototyping, usability testing
Ishaan Manhas, Ansab Aalim and Aryan Chaudhary	Frontend Developer	Develop user interface
Ishaan Manhas, Ansab Aalim and Aryan Chaudhary	Backend Developer	Design, Develop and Unit Test Services/API/DB
Ishaan Manhas, Ansab Aalim and	Cloud Architect	Design the cost effective, highly available

Aryan Chaudhary		and scalable architecture

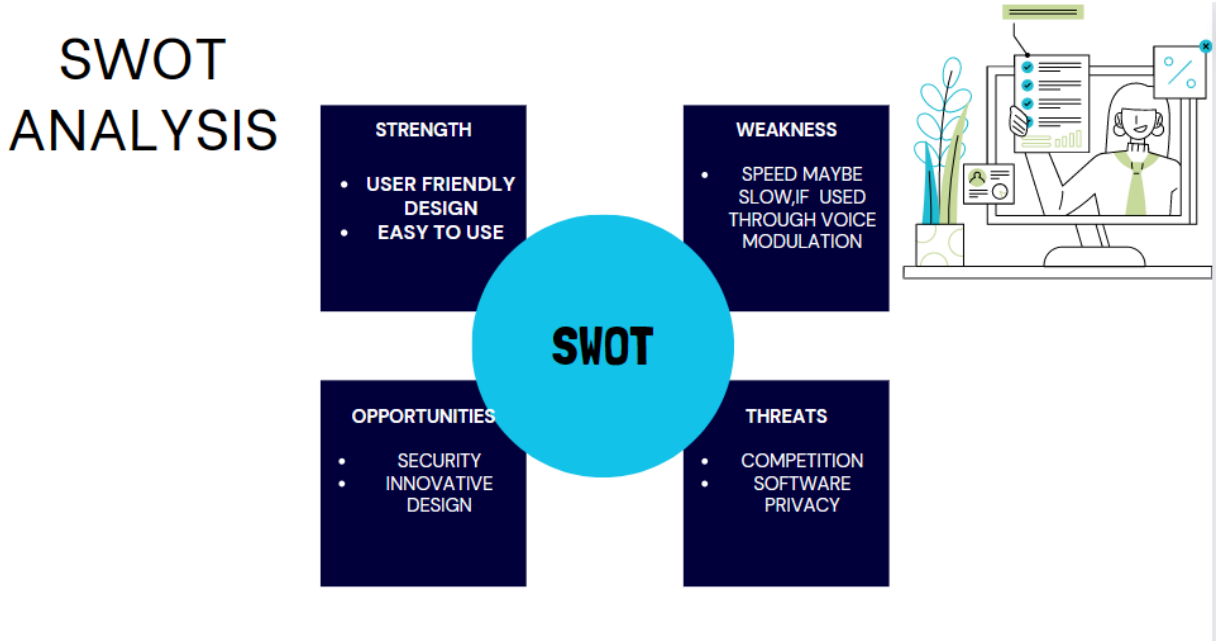
RESPONSIBILITY ASSIGNMENT MATRIX:-

RACI MATRIX	**TEAM MEMBERS**			
ACTIVITY	**Aryan Chaudhary (BA)**	**Ansab Aalim (DEVELOPERS)**	**Ishaan Manhas (PROJECT MANAGER)**	**KEY BUSINESS USER**
Requirement Gathering	R	A	C	I
Functional Design	R	A	C	I
Technical Design	R	A	C	I
Coding	I	R	A	-
Testing	A	R	C	I
Deployment	I	R	C	-
Maintenance	A	R	C	I

Risk assessment is a crucial component of project management, as it helps to identify and analyze potential risks that may impact the project's success. By conducting a risk assessment, the project team can proactively identify potential issues and develop strategies to mitigate or eliminate them.

SWOT ANALYSIS:-

SWOT analysis is a strategic planning tool used to evaluate the Strengths, Weaknesses, Opportunities, and Threats of a project or organization. It involves analyzing the internal and external factors that may impact the project's success and identifying ways to capitalize on strengths, address weaknesses, exploit opportunities, and mitigate threats.



RMMM PLAN:-

STRATEGY	DESCRIPTION	AVOIDANCE TECHNIQUE
Avoidance	Reducing the likelihood of the occurrence of risk	Bringing in a consulting expert with experience in the field
Mitigation	Reducing the impact of a risk if it occurs	Creation of exception processing to deal with unexpected situations Preparing an emergency Technical Maintenance Team

Transferring	Moving the impact of a risk(if it occurs) to an external party	Set up a low-cost-per-hour Help Desk to provide support
Accepting	Deciding to accept the consequences of an occurring risk and documentation of it for future use	Set up a Disaster Analysis and Review Team to prevent similar problems from occurring in the future

SYSTEM ARCHITECTURE, USE CASE & CLASS DIAGRAM

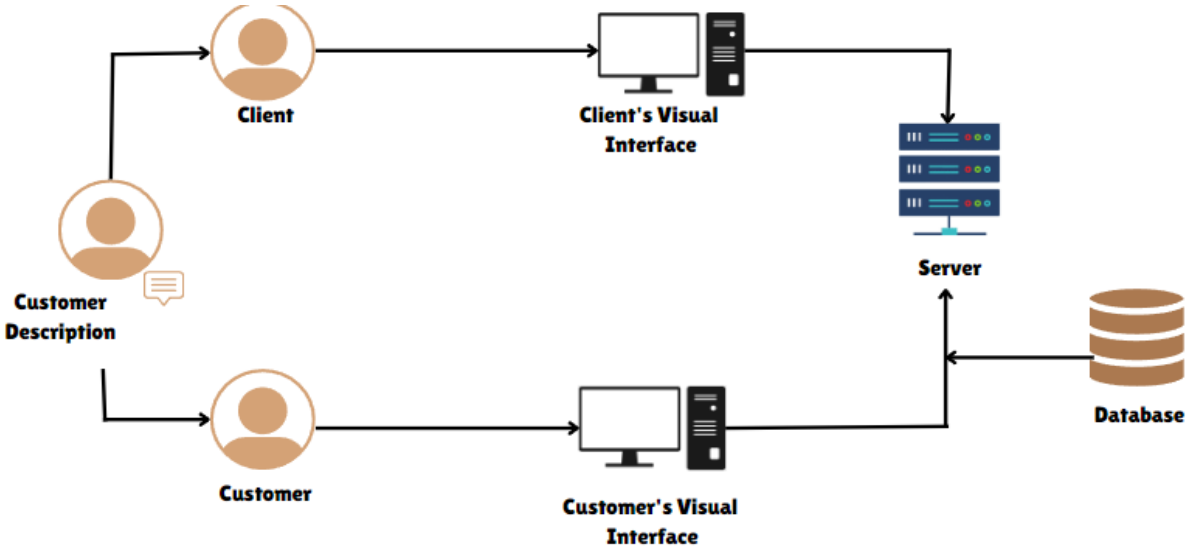
SYSTEM ARCHITECTURE:-

System architecture refers to the overall design and organization of a software or technology system. It includes the structure, components, modules, interfaces, and data flows that make up the system.

The architecture of a system is crucial because it determines how the system will function and how its components will interact with each other. A well-designed system architecture should be scalable, flexible, and adaptable to changes, while also meeting the performance and security requirements of the system.

System architecture also involves considering the different stakeholders and users of the system, as well as their requirements and needs. It is essential to design a system architecture that meets the needs of all stakeholders, including end-users, developers, administrators, and other members of the system's ecosystem.

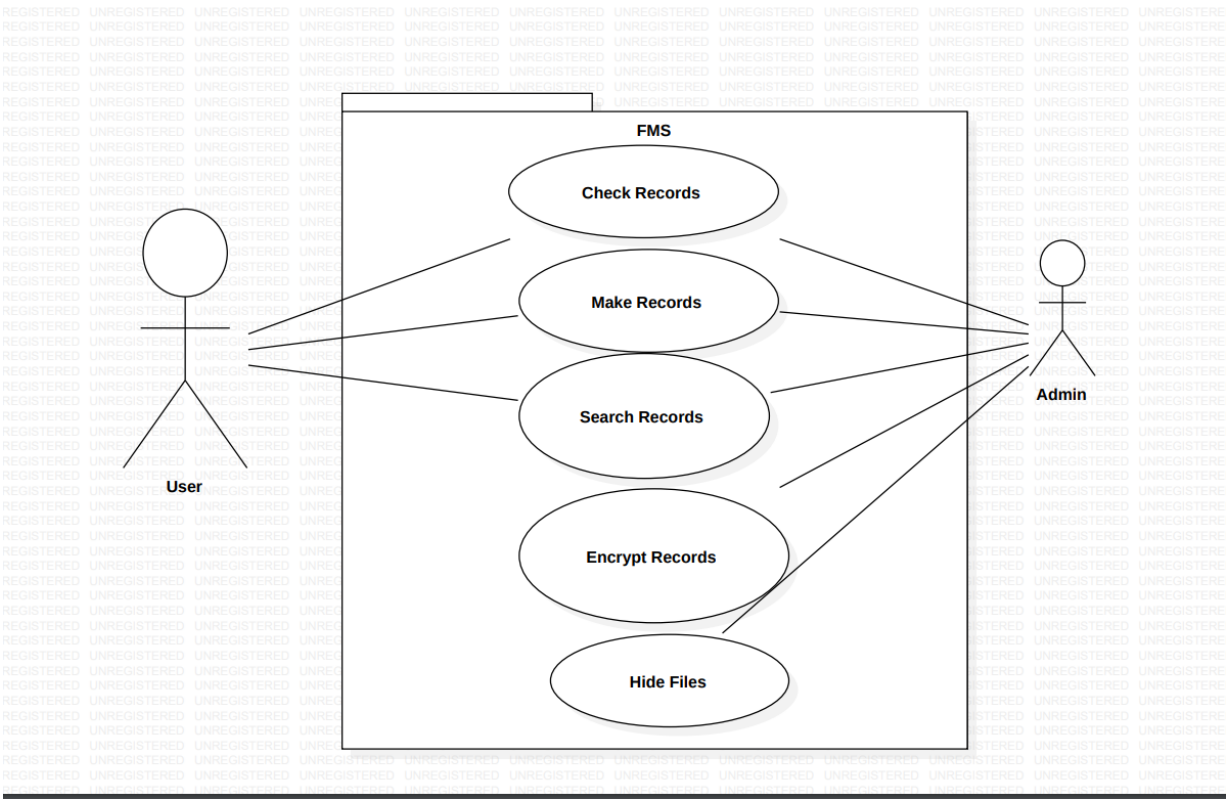
Overall, system architecture is a critical component of software engineering and technology design, and it requires careful planning, analysis, and design to create a system that meets the needs of all stakeholders while also being reliable, efficient, and secure.



- **Customer:** The customer refers to the end-user of the online shopping interface. This could be an individual consumer or a business that is purchasing products through the interface.
- **Client Description:** The client description refers to the specifications or requirements provided by the client (the organization or individual who is commissioning the development of the online shopping interface). This includes details such as the desired functionality, features, and user experience of the interface.
- **User Visual Interface:** The client's visual interface refers to the front-end component of the online shopping interface that is designed to meet the client's requirements and specifications. This may include elements such as the layout, color scheme, and branding of the interface.

- **Customer's Visual Interface:** The customer's visual interface refers to the part of the online shopping interface that the customer interacts with. This includes elements such as the product catalog, shopping cart, checkout process, and other user-facing components.
- **Server:** The server is the back-end component of the online shopping interface that is responsible for processing requests, managing data, and performing other functions that support the interface's functionality. This may include servers that handle web traffic, database management, and other tasks.
- **Database:** The database is where all the data related to the online shopping interface is stored, including product information, customer data, and order details. The database is a critical component of the system, and it is responsible for managing data in a way that ensures it is secure, reliable, and easily accessible by the interface's different components.

USE CASE DIAGRAM:-



- **Check Records:** This use case represents the process of checking if the records are present in the present working directory.

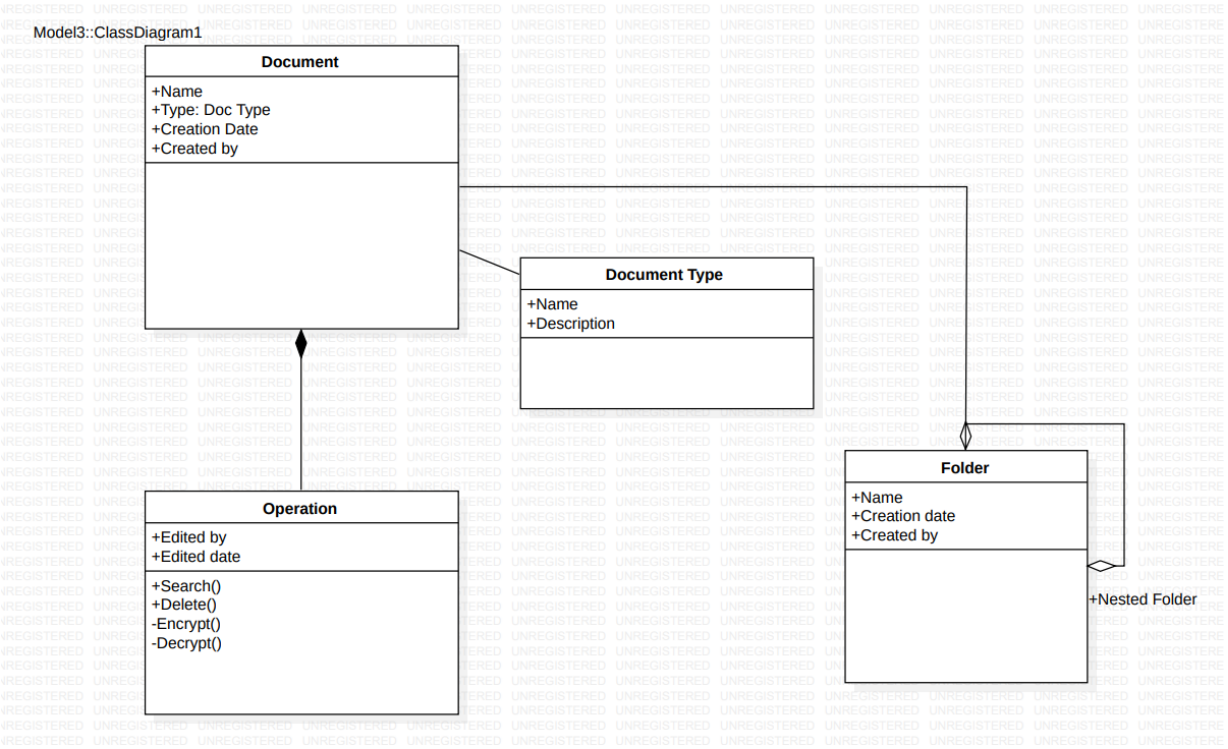
- **Make Record:** This use case represents the process of making new files in the system. It is also a standalone use case and is not directly related to any of the other use cases.

- **Search Records:** This use case represents the ability for customers to browse and search for the required file.

- **Delete Records:** This use case represents the ability for customers to delete a file from the system.

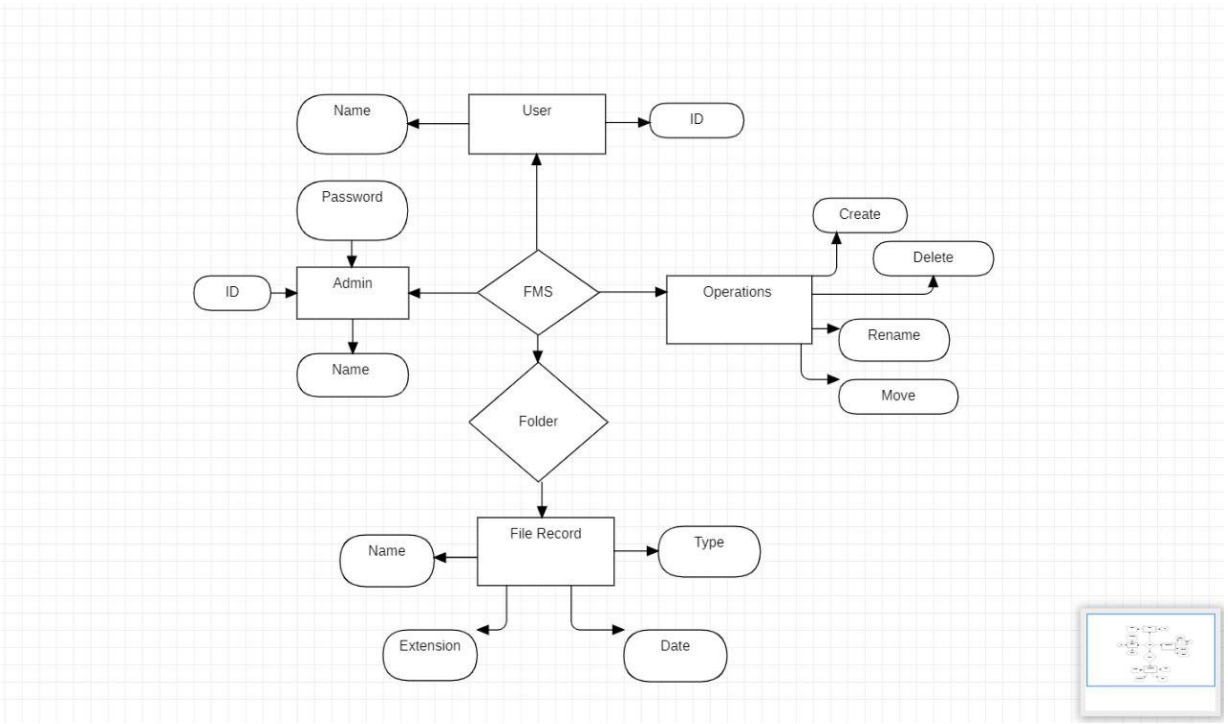
- **Move Records:** This use case represents the process of moving one folder in the system to another.

CLASS DIAGRAM:-



- **Document Class:** Contains various attributes that define the document created.
- **Document Type Class:** Subclass of Class Document. Contains the attributes that define the document type.
- **Operation Class:** Class containing all the methods responsible for execution of modular operations.
- **Folder Class:** Class containing folder attributes required to define doc format while creation.

ENTITY RELATIONSHIP DIAGRAM



- **User:** Represents the clients who use FMS to store files. Customers can create, delete, rename and move files in their system
- **Folder :** Represents the interface available after creating multiple files interface.

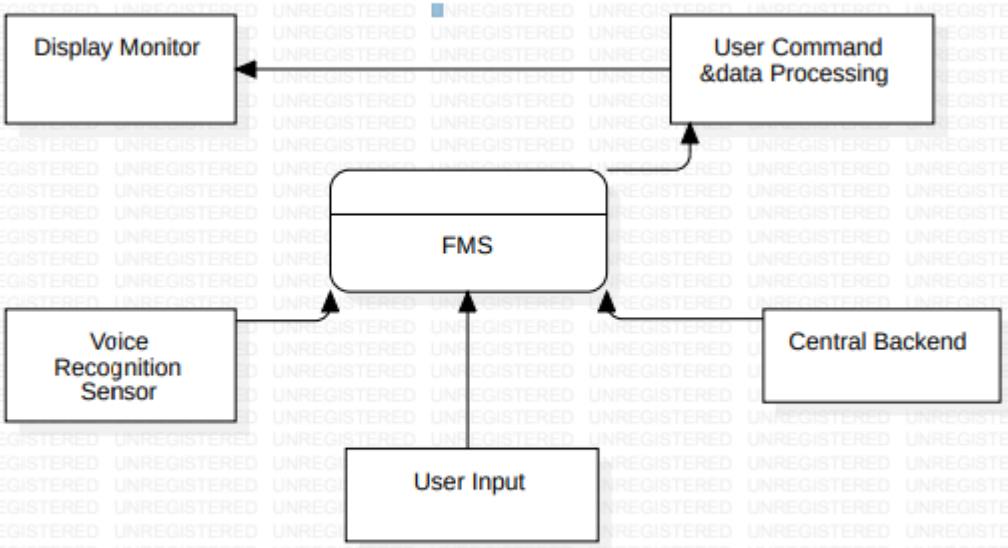
- **Operations:** Represents the collection of all modules present in the software.
- **File Record:** Represents the various attributes of the file after its creation.

The relationships between these entities are as follows:

- A user accesses FMS software directly (one-to-one relationship between User and FMS).
- A Folder can contain multiple File Records (one-to-many relationship between Folder and File Record).
- A User can perform multiple operations through FMS (one-to-many relationship between User and Operations).
- An order is associated with a single customer (one-to-one relationship between Customer and Order).

DATA FLOW DIAGRAM

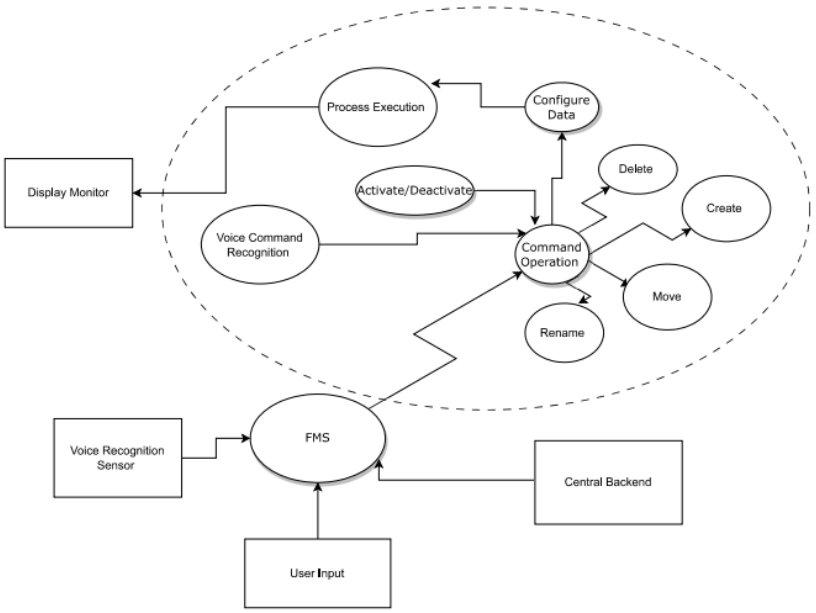
DFD LEVEL 0 DIAGRAM



It involves 4 main processes which are as follows:-

- **FMS:** Represents the user interface of the FMS software. This process accepts user inputs and performs necessary actions.
- **Central Backend:** Represents the database and source code where the main execution of a task is performed.
- **Voice Recognition:** Represents the Voice Recognition module responsible to execute tasks based on the voice commands.
- **Display Monitor:** Represents the means of output.

DFD LEVEL 1 DIAGRAM



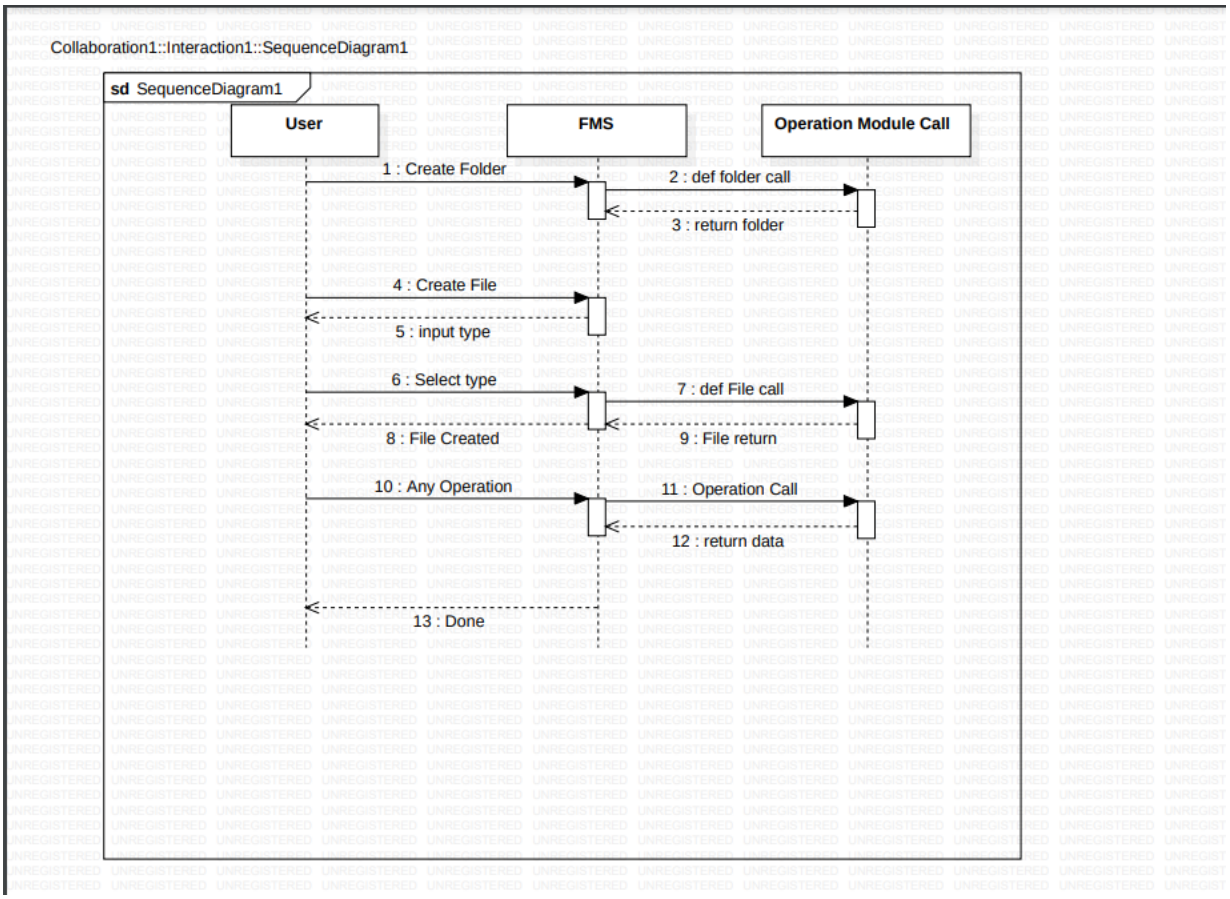
FMS: Represents the user interface of the FMS software. Takes in user inputs in the form of event triggers or voice commands for execution of various tasks.

Command Operation: Contains the various operation modules required for execution of tasks. According to the user's requirements it can perform operations like delete, create, move, rename. It can perform these operations using voice commands as well. After process execution the data is sent to the Display/Monitor entity.

Display/Monitor: Represents the means of output. Required to display the desired output.

SEQUENCE AND COLLABORATION DIAGRAM

SEQUENCE DIAGRAM

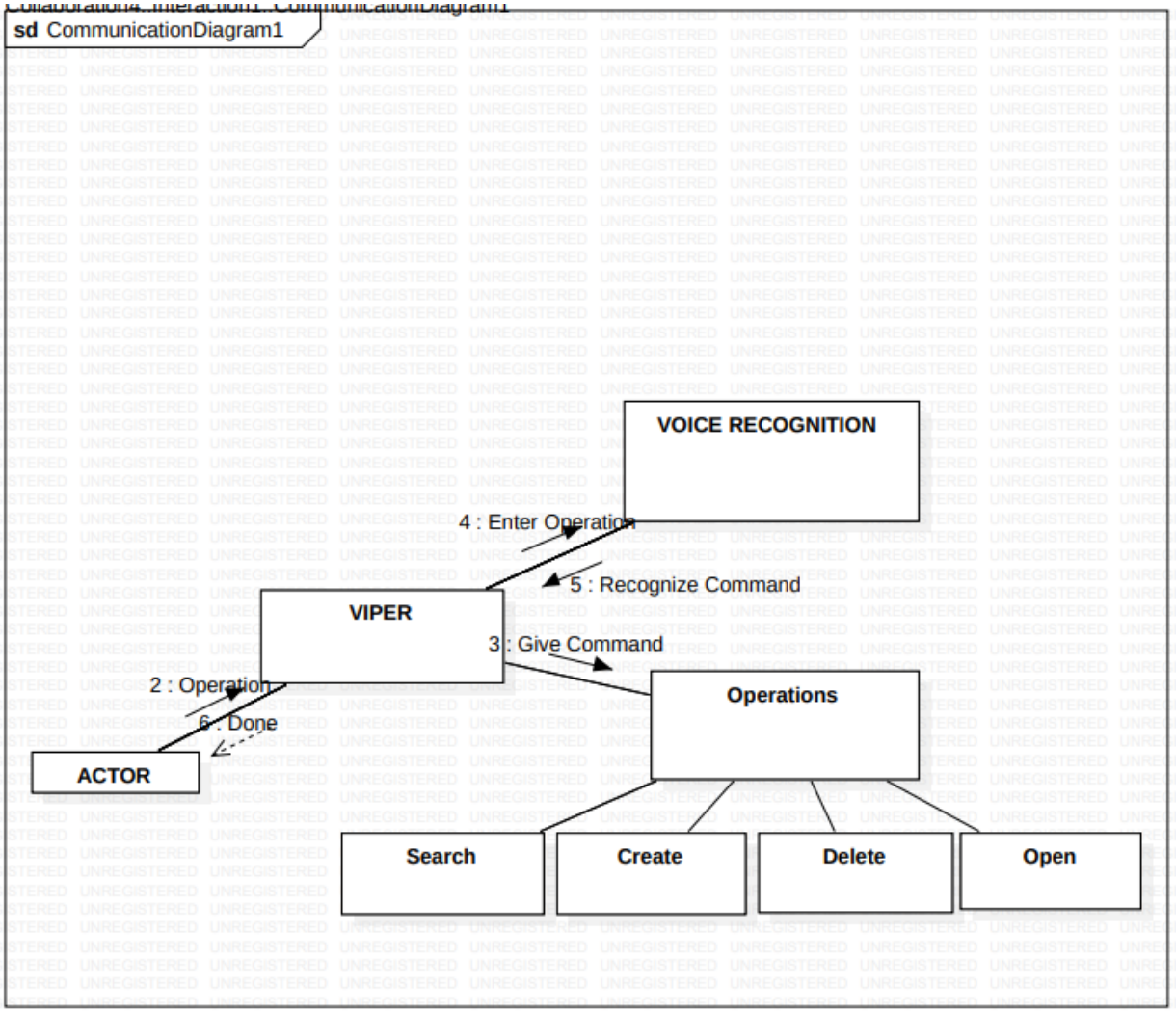


In this diagram, we have three participants. They collaborate with each other as:

1. **User:**
    - Sends requests to work (create,delete,rename,move)on a file .
    - Sends requests to the FMS to select the type of file to be worked on
2. **FMS:**
    - Receives requests from the User to work on the file .
    - Sends requests to the Operation Module Call to call the appropriate function.
    - Sends requests to the Operation Module Call to execute proper Operation Call.
3. **Operation Module Call:**
    - Sends requests to FMS to return the folder for the operation called.

- Receives requests from FMS to retrieve and update details of the file.

COLLABORATION DIAGRAM



DEVELOPMENT OF TESTING FRAMEWORK/ USER INTERFACE

EXECUTIVE SUMMARY

Scope:

The scope of testing the software application includes verifying and validating its functionality, usability, and performance. This involves testing the different features and functionalities of the application, such as file creation, file deletion, file renaming, file moving, and user interface interactions. It may also involve testing the application's compatibility with different operating systems and environments.

Objective:

The objective of testing the software application is to ensure that it works as intended, meets the requirements, and is free from defects and issues that may affect its performance, usability, or reliability. The testing process aims to identify and fix any bugs, errors, or inconsistencies in the application to ensure its quality and reliability for end users..

Approach:

The testing approach for the software application can follow a structured and systematic approach, such as the following steps:

a) Test Planning: Define the testing objectives, scope, and test requirements. Identify the testing resources, such as tools and equipment, needed for testing. Develop a detailed test plan that outlines the testing strategy, test scenarios, test cases, and test data.

b) Test Design: Create test scenarios and test cases based on the application's requirements and functionalities. Define the expected results for each test case.

c) Test Execution: Execute the test cases using appropriate test data and inputs. Record the test results, including any issues or defects identified during testing.

d) Usability and Performance Testing: Evaluate the application's usability, including user interface interactions, navigation, and overall user experience. Also, conduct performance testing to verify the application's responsiveness, resource usage, and scalability under different conditions.

e) Acceptance Testing: Once the application is tested and fixed, conduct acceptance testing to ensure that it meets the defined acceptance criteria and is ready for deployment.

f) Documentation: Document the testing process, including the test plan, test scenarios, test cases, test results, and any issues or defects found during testing.

The scope of testing includes verifying and validating the functionality, usability, and performance of the software application. This involves testing features such as file creation, deletion, renaming, and moving, as well as user interface interactions. It may also involve testing the application's compatibility with different operating systems and environments to ensure it works as intended and meets the requirements. The objective is to identify and fix any bugs or issues that may affect the application's performance, usability, or reliability, and ensure its quality and reliability for end users.

STEP	PROCESS	DESCRIPTION
1	Define the testing scope and objectives	This includes identifying the features and functionalities that need to be tested and the expected outcomes.
2	Identify the testing approach	Used to test the user interface of the online shopping interface creator. This can include manual testing, automated testing, or a combination of both.
3	Create test cases	Based on the testing scope and objectives, create a set of test cases that can be used to test the user interface of the online shopping interface creator. The test cases should cover all the features and functionalities of the system.
4	Develop test scripts	If automated testing is used, develop test scripts that can be used to automate the testing process. This includes selecting the right tools and frameworks for test automation.
5	Execute the tests	Execute the tests to identify any defects or issues in the user interface of the online shopping interface creator.
6	Report and track defects	Report any defects or issues found during testing and track them until they are resolved.
7	Evaluate the results	Once the testing is complete, evaluate the results to identify areas for improvement and provide recommendations for future testing.

TEST CASES & REPORTING

FUNCTIONAL TEST CASES

TEST ID	TEST SCENARIO	TEST CASE	EXECUTION TEST	EXPECTED OUTCOME	ACTUAL OUTCOME	STATUS	REMARKS
1	User can add a file using voice command	Create File	1. Say "Create "	A new file should be created and the user should be prompted to give it a name	A new file was created	PASS	Success

2	User can delete a file using voice command	Delete File	1. Say "Delete" 2. Select the file to be deleted	The file should be deleted from the system	The File was deleted from the system	PASS	Success
3	User can rename a file using voice command	Rename File	1. Say "Rename " 2. Select the file to be renamed 3. Write the new name of the file	The file should be renamed with the new name	The file was renamed	PASS	Success
4	User can move a file to a specific folder using voice command	Move File	1. Say "Move " 2. Select file to be moved 3. Select the folder to which the file should be moved	The file should be moved to the specified folder	The file was moved from one folder to another	PASS	Success

NON FUNCTIONAL TEST CASES

TEST ID	TEST SCENARIO	TEST CASE	EXECUTION TEST	EXPECTED OUTCOME	ACTUAL OUTCOME	STATUS	REMARKS
1	Voice Recogniti on	The system should be able to accurately recognize the user's voice commands	1. Voice command given 2. System Identifies the command and executes task	The system should execute the task successfully	Tasks were executed successfully	PASS	SUCCESS

2	Response Time	The system should respond to the user's voice commands within an acceptable time limit	1. After a voice command is given, response time is calculated.	Time taken to execute a command should be low	Commands were executed quickly	PASS	SUCCESS
3	Error Handling	The system should provide appropriate error messages and feedback to the user in case of incorrect voice commands or system errors	After a wrong command is entered, the system should show errors	Errors should be reflected on the screen	Errors were shown when wrong command was entered	PASS	SUCCESS

TEST CASE REPORTING

CATEGORY	PROGRESS AGAINST PLAN	STATUS
Functional Testing	Green	Completed
Non-Functional Testing	Green	Completed

Functional Test Cases	Coverage	Status
Create File	100%	Completed
Delete File	100%	Completed
Rename File	100%	Completed
Move File	100%	Completed

DESIGN IMPLEMENTATION

Backend Program

V.I.P.E.R.py

SEPM Project > V.I.P.E.R.py > FileManagerApp > execute_command

```python
import os
import speech_recognition as sr
import tkinter as tk
from tkinter import filedialog
from tkinter import messagebox

class FileManagerApp:
    def __init__(self, master):
        self.master = master
        self.master.title("File Manager App")

        self.create_widgets()

    def create_widgets(self):
        # Create command label and entry
        self.command_label = tk.Label(self.master, text="Enter Command:")
        self.command_label.grid(row=0, column=0, padx=5, pady=5)

        self.command_entry = tk.Entry(self.master, width=50)
        self.command_entry.grid(row=0, column=1, padx=5, pady=5)

        # Create buttons for actions
        self.open_button = tk.Button(self.master, text="Open File", command=self.open_file)
        self.open_button.grid(row=1, column=0, padx=5, pady=5)

        self.create_button = tk.Button(self.master, text="Create File", command=self.create_file)
        self.create_button.grid(row=1, column=1, padx=5, pady=5)

        self.delete_button = tk.Button(self.master, text="Delete File", command=self.delete_file)
        self.delete_button.grid(row=1, column=2, padx=5, pady=5)

        self.rename_button = tk.Button(self.master, text="Rename File", command=self.rename_file)
        self.rename_button.grid(row=1, column=3, padx=5, pady=5)

        # Create status label
        self.status_label = tk.Label(self.master, text="")
        self.status_label.grid(row=2, column=0, columnspan=4, padx=5, pady=5)
```

Ln 140, Col 41 Spaces: 4 UTF-8 CRLF Python 3.10.0 64-bit

V.I.P.E.R.py

SEPM Project > V.I.P.E.R.py > FileManagerApp > execute_command

```python
    def recognize_speech(self):
        r = sr.Recognizer()
        with sr.Microphone() as source:
            self.status_label.config(text="Speak...")
            self.master.update()
            audio = r.listen(source)

        try:
            command = r.recognize_google(audio)
            self.status_label.config(text="You said: " + command)
            self.master.update()
            return command
        except sr.UnknownValueError:
            self.status_label.config(text="Google Speech Recognition could not understand audio")
            self.master.update()
        except sr.RequestError as e:
            self.status_label.config(text="Could not request results from Google Speech Recognition service; {0}".format(e))
            self.master.update()

    def open_file(self):
        filename = filedialog.askopenfilename(title="Select File", filetypes=(("All files", "*.*"),))
        if filename:
            try:
                os.startfile(filename)
                self.status_label.config(text="File opened successfully")
                self.master.update()
            except FileNotFoundError:
                self.status_label.config(text="File not found")
                self.master.update()
```

V.I.P.E.R.py

SEPM Project > V.I.P.E.R.py > FileManagerApp > execute_command

```
    def create_file(self):
        filename = filedialog.asksaveasfilename(title="Create File", defaultextension=".txt")
        if filename:
            try:
                with open(filename, 'w'):
                    self.status_label.config(text="File created successfully")
                    self.master.update()
                    file_ext = os.path.splitext(filename)[1].lower()
                    self.move_file_to_folder(filename, file_ext)
            except FileExistsError:
                self.status_label.config(text="File already exists")
                self.master.update()

    def delete_file(self):
        filename = filedialog.askopenfilename(title="Select File to Delete", filetypes=(("All files", "*.*"),))
        if filename:
            confirm = messagebox.askyesno(title="Confirm Deletion", message=f"Are you sure you want to delete {os.path.basename(filename)}?")
            if confirm:
                try:
                    os.remove(filename)
                    self.status_label.config(text="File deleted successfully")
                    self.master.update()
                except FileNotFoundError:
                    self.status_label.config(text="File not found")
                    self.master.update()
```

V.I.P.E.R.py

SEPM Project > V.I.P.E.R.py > FileManagerApp > execute_command

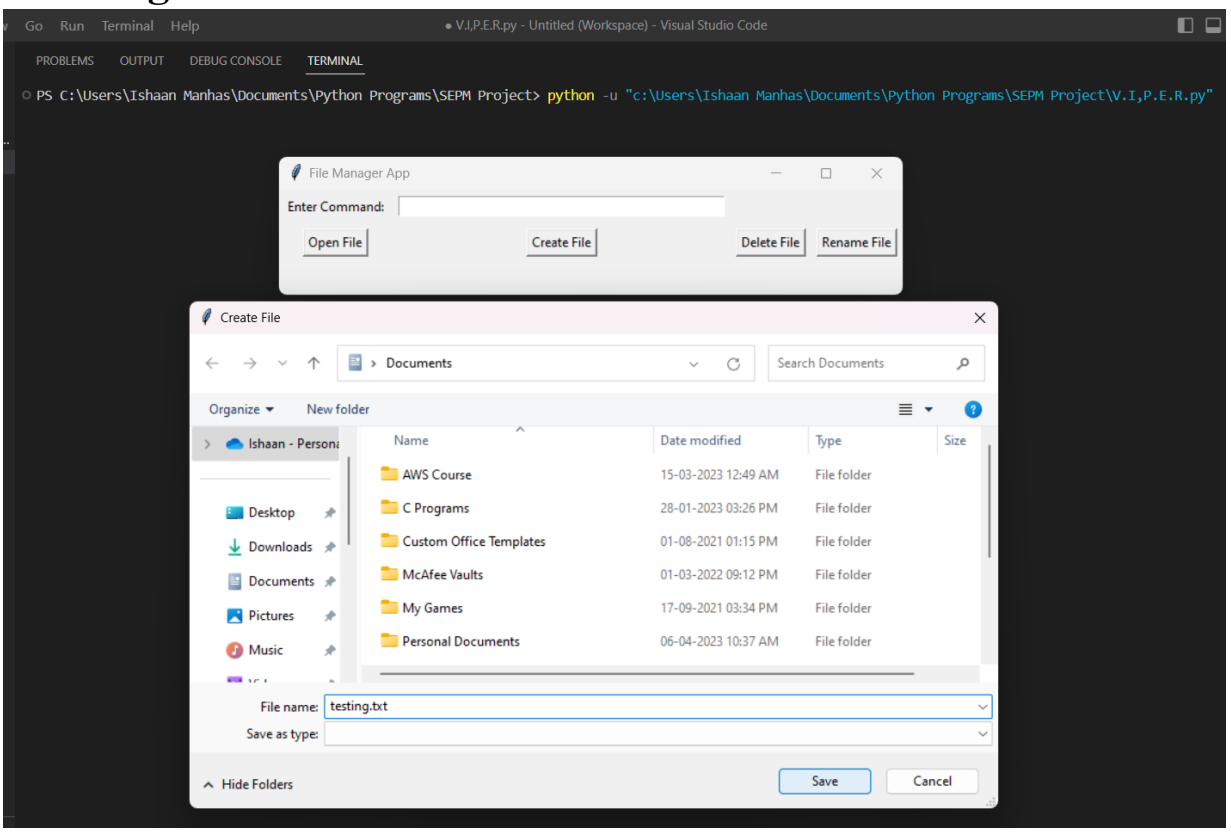
```
    def rename_file(self):
        filename = filedialog.askopenfilename(title="Select File to Rename", filetypes=(("All files", "*.*"),))
        if filename:
            new_filename = filedialog.asksaveasfilename(title="Rename File", defaultextension=os.path.splitext(filename)[1])
            if new_filename:
                try:
                    os.rename(filename, new_filename)
                    self.status_label.config(text="File renamed successfully")
                    self.master.update()
                except FileExistsError:
                    self.status_label.config(text="File already exists with that name")
                    self.master.update()

    def move_file_to_folder(self, filename, file_ext):
        folder_path = os.path.join(os.getcwd(), file_ext[1:].upper() + "_Files")
        if not os.path.exists(folder_path):
            os.mkdir(folder_path)
        new_file_path = os.path.join(folder_path, os.path.basename(filename))
        os.replace(filename, new_file_path)

    def move_file_to_location(self):
        filename = filedialog.askopenfilename(title="Select File to Move", filetypes=(("All files", "*.*"),))
        if filename:
            new_location = filedialog.askdirectory(title="Select Destination Folder")
            if new_location:
                try:
                    _, file_ext = os.path.splitext(filename)
                    new_file_path = os.path.join(new_location, os.path.basename(filename))
                    os.replace(filename, new_file_path)
                    self.status_label.config(text="File moved successfully")
                    self.master.update()
                except FileNotFoundError:
                    self.status_label.config(text="File not found")
                    self.master.update()
```

```
    def execute_command(self):
        command = self.command_entry.get().lower()
        if command == "open":
            self.open_file()
        elif command == "create":
            self.create_file()
        elif command == "delete":
            self.delete_file()
        elif command == "rename":
            self.rename_file()
        elif command == "move":
            self.move_file_to_location()
        elif command == "speech":
            recognized_command = self.recognize_speech()
            if recognized_command:
                self.command_entry.delete(0, tk.END)
                self.command_entry.insert(0, recognized_command)
                self.execute_command()
        else:
            self.status_label.config(text="Invalid command entered")
            self.master.update()

    def on_key_press(self, event):
        if event.keysym == 'Return':
            self.execute_command()

def main():
    root = tk.Tk()
    app = FileManagerApp(root)
    root.bind('<Key>', app.on_key_press)
    root.mainloop()

if __name__ == '__main__':
    main()
```

Ln 140, Col 41 Spaces: 4 UTF-8 CRLF Python 3.10.0 64-bit

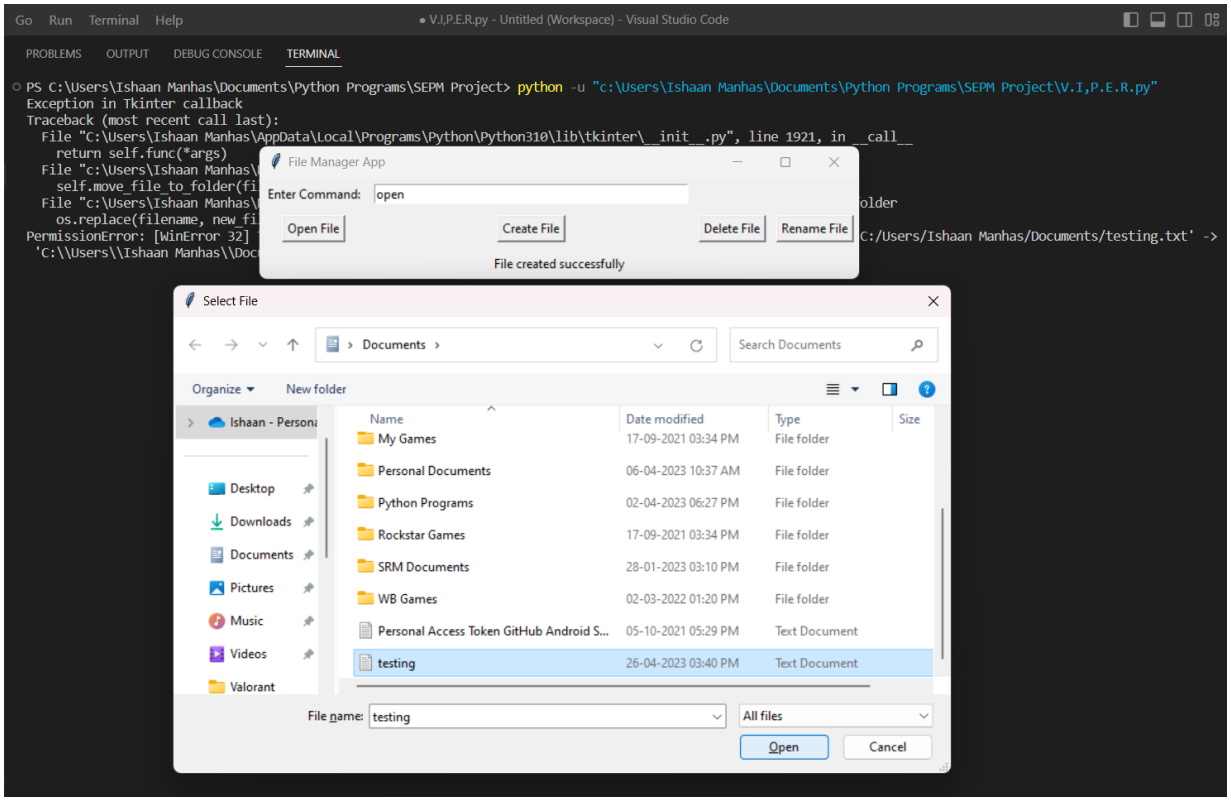
Module Implementations(Voice Integrated)

1. Creating File



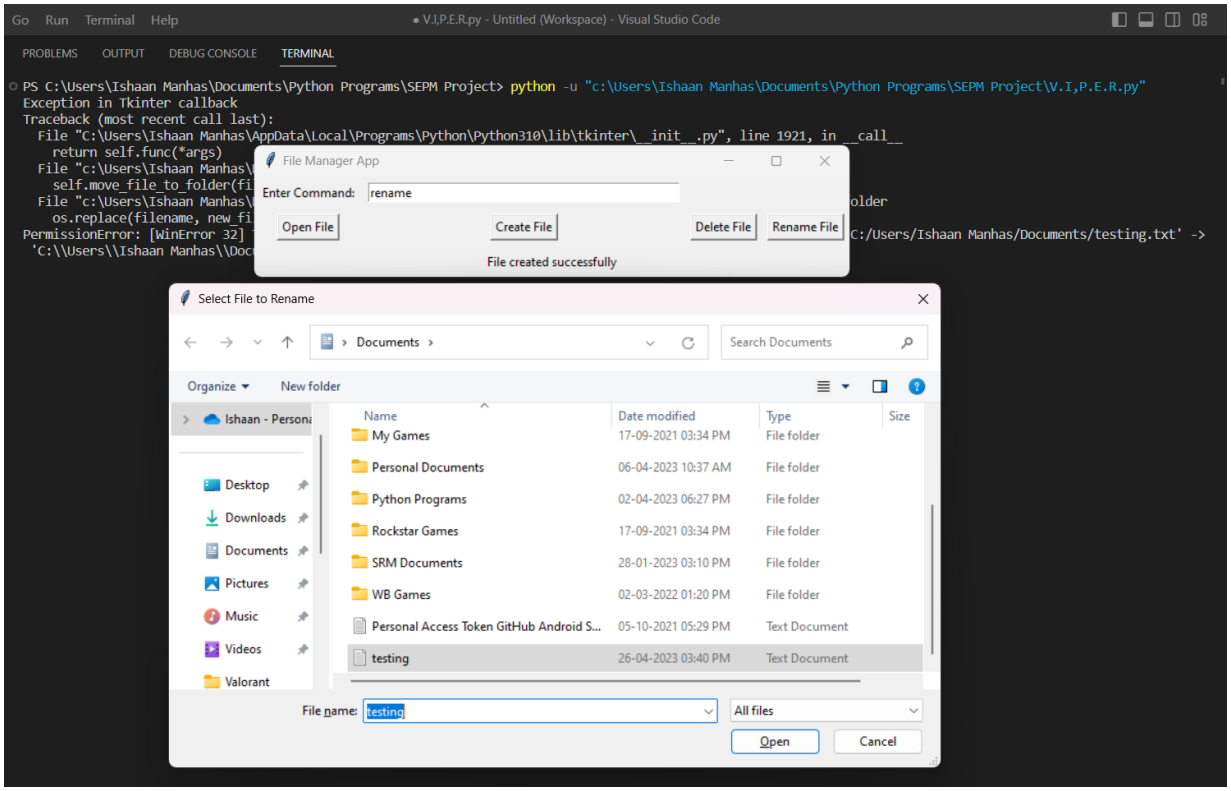
Enter “speech” as command and speak “create”. Create file window will open. Enter the file name with extension and save.

2. Opening File



Enter “speech” as command and speak “open”. Select File window will open. Select the file and click on “Open”.

3. Renaming File



Enter “speech” as command and speak “rename”. Select File to Rename window will open. Select the file and click on the “File name” field to edit its existing name. After editing click on enter.

CONCLUSION

The voice integrated file management system project has been successfully implemented, providing users with a more intuitive and efficient way of managing their files and folders. The use of voice recognition technology has made the system more user-friendly and accessible, allowing users to perform tasks without the need for a keyboard or mouse. The project has also demonstrated the potential of voice recognition technology in other areas, such as home automation and digital assistants. Future work could include further improvements to the accuracy and speed of the voice recognition system and the integration of additional file management features. Overall, the project has been a success, and it has shown the potential of integrating voice recognition technology into file management systems.