OOM Mini Project

Title : Document Management System
Project no. : 14
Group no. : A28

Done By:

1) Sugam Sareen (IIT2022002)
2) Jatin Soni (IIT2022003)
3) Vatsal Bhuva (IIT2022004)

A JAVA based Document Management AND Retrieval System design

AIM :

Develop an application to manage document storage and retrieval.

TECHNOLOGY USED:

JAVA, JAVA GUI, Java Swing

CONTENT :

UML diagrams:-Use Case diagram, Class diagram, CRC diagram(s), illustrating the design of the program, code snippets, code explanation

PROJECT TASKS:

Use case analysis, Design, Implementation, and User Interface

DESIGN CONCEPT:

Use of Object Oriented Methodology concepts to design the system base.

Basic Classes:

1) Category
2) Document
3) Topic
4) Tag, and have the following relationships:

- A document belongs to a category such as policy, plan, report, receipt, order, ... etc.
- A document belongs to a topic such as “CS243 Course Files in Fall 2013”, “Cluster Graduation Project in 2013”, ... etc.
- A document can have any number of tags such as: “legal”, “medical”, “administrative”, “technical”, “2013”, “reporting”, ... etc.

Provides an interface for the user to:

A. Add/Edit/Delete instances belonging to each class,
B. Retrieve document by Category,
Topic, Tag.

Working:

A user can register/sign-in. Once done, the user is redirected to the main page, where all the options are available.

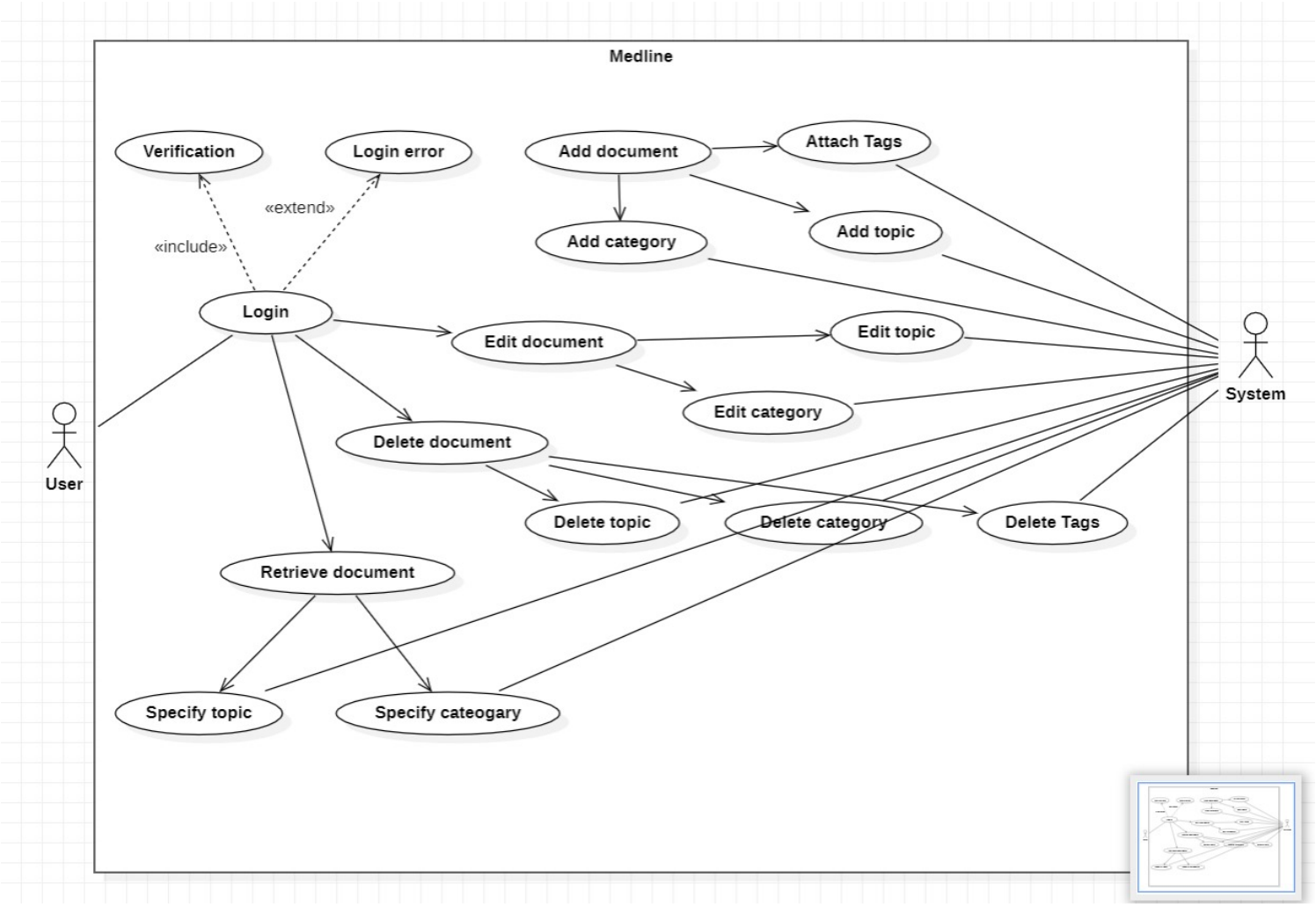
The user can add documents (with tags, topics, category, etc), delete, edit, or retrieve documents. All the added documents are displayed on the main page in a tabular format, which allows the user to view the current documents and perform operations on them.
The user also has the option to view these documents in a list form in the specific operations page.
All the changes made to the document change the document object, and cause necessary updates in the table as well, ensuring synchronization of data across all pages.

We've also ensured error-handling at all points of the project. For example, a user cannot add a document without mentioning a title. A user cannot delete/edit/retrieve documents if there are no documents available. We've also ensured a login/registration system to allow only authenticated users to make use of the portal. Currently, the registration only lasts for that session of the portal, but this can be made more efficient if connected with a database

USE CASE DIAGRAM:

A UML use case diagram is the primary form of system/software requirements for a new software program underdeveloped. A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

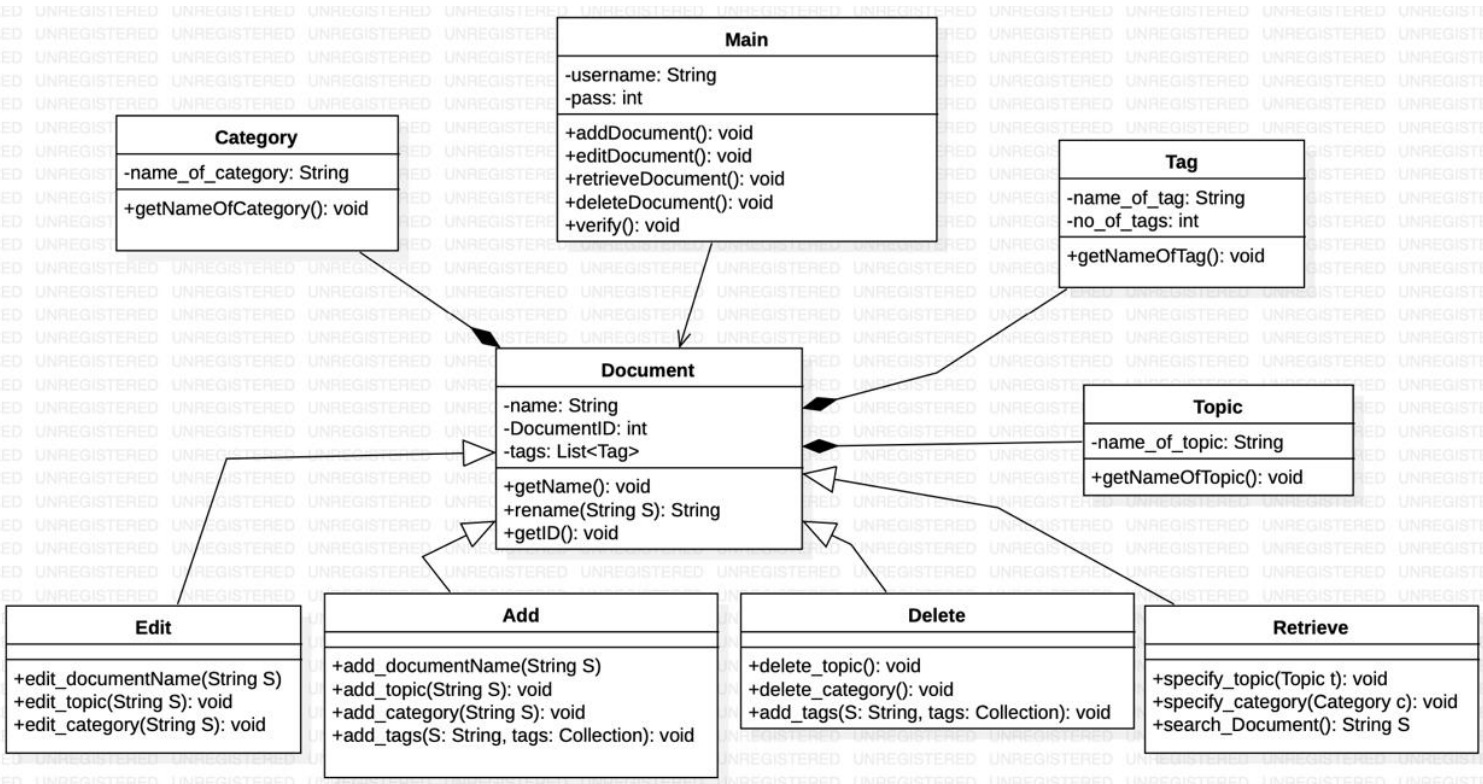
Image: The Use-Case diagram for our project



CLASS DIAGRAM:

The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object- oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Image: The Class-Diagram for our project, depicting the possible classes that can be used.



CRC DIAGRAMS:

Class-responsibility-collaboration (CRC) cards are a brainstorming tool used in the design of object-oriented software. To create a CRC card, you can begin by writing out a scenario which identifies the major actors and actions which the actors do. Only write out actions and actors specific to that particular scenario. Nouns should turn into the classes of the card, verbs typically turn into the responsibilities of the card, and collaborators are the other cards with which the card will be interacting.

Topic	
• Responsible for attaching a topic to a document	• Document

AddDocument	
• Responsible for adding topic, category, and tags for each document added	• Document • Topic • Category • Tag

EditDocument	
• Responsible for updating and changing topic, category and tags for each document	• Document • Topic • Category • Tag

DeleteDocument	
• Responsible for deleting all contents of a document	• Document • Topic • Category • Tag

RetrieveDocument	
• Responsible for retrieving a document by specifying it's topic, category and tags	• Document • Topic • Category • Tag

Main	
• Resposible for verification of user (login) • Responsible for choosing between options of actions on Document (add/edit/delete/retrieve)	• Document • AddDocument • EditDocument • DeleteDocument • RetrieveDocument

Document	
• Responsible for holding data • Responsible for having tags, topic, and category • Responsible for actions on documents	• Tag • Category • Topic

Tag	
• Responsible for attaching tags to a document • Responsible for holding record of each tag for each document	• Document

Category	
• Responsible for defining the category of each Document	• Document

CODE SNIPPETS:

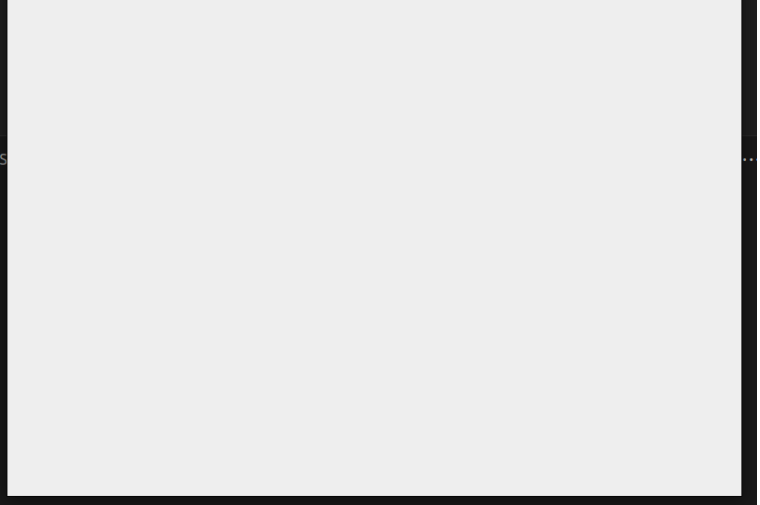
1) Login/Register (with error-handling): (Main.java)

Main.java > Main > main(String[])

```java
Run | Debug
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    do {
        System.out.println();
        System.out.println("Create a new user or log in or
        int choice = sc.nextInt();
        if (choice == 1) {
            System.out.println("Enter your user name: ");
            String name = sc.next();
            System.out.println("Enter the password: ");
            String password = sc.next();
            User u = new User(name, password);

            int flag = 1;
```

PROBLEMS 56 OUTPUT TERMINAL PORTS GITLENS POSTMAN CONSOLE DEBUG CONS

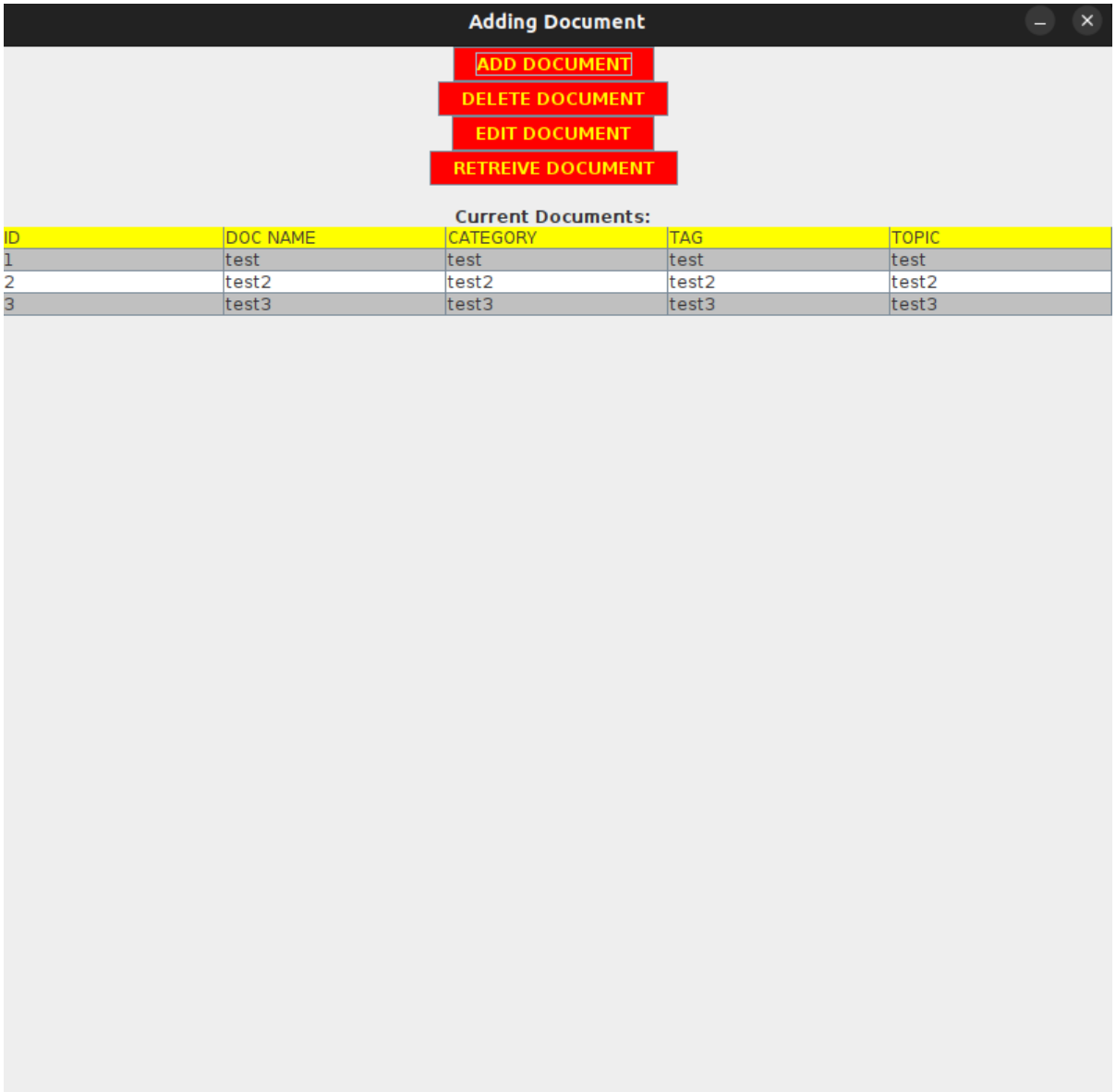
```
Create a new user or log in or exit? (1/2/3) :
1
Enter your user name:
Vatsal
Enter the password:
test
User created successfully. You can now choose to log in.

Create a new user or log in or exit? (1/2/3) :
2
Enter your user name:
test
Enter the password:
test
Welcome test
```




```java
import javax.swing.*;
import java.util.*;
import java.awt.*;

sugam, 3 days ago | 2 authors (You and others)
public class Main {
    static int running = 1;

    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        do {
            System.out.println();
            System.out.println("Create a new user or log in or exit? (1/2/3) : ");
            int choice = sc.nextInt();
            if (choice == 1) {
                System.out.println("Enter your user name: ");
                String name = sc.next();
                System.out.println("Enter the password: ");
                String password = sc.next();
                User u = new User(name, password);

                int flag = 1;
                for (User check : User.users) {
                    if (check.username.equals(name)) {
                        System.out.println("Username already exists. Please try again.");
                        flag = 0;
                        break;
                    }
                }
                if (flag == 1)
                    User.users.add(u);
```

```java
            } else if (choice == 2) {
                System.out.println("Enter your user name: ");
                String name = sc.next();
                System.out.println("Enter the password: ");
                String password = sc.next();

                int flag = 1;
                if (name.equals("test") && password.equals("test")) {
                    System.out.println("Welcome " + name);
                    flag = 0;
                    running = 0;
                } else {
                    for (User u : User.users) {
                        if (u.username.equals(name) && u.password.equals(password)) {
                            System.out.println("Welcome " + name);
                            flag = 0;
                            running = 0;

                        }
                    }
                }

                if (flag == 1) {
                    System.out.println("Sorry !! " + name + " \nEither username or password is wrong ");
                } else {
                    JFrame jFrame = new JFrame();
                    jFrame.setTitle("Home Page");

                    Add a = new Add();
                    Container cPane = jFrame.getContentPane();
                    Newclass template = new Newclass(a, cPane);
                    jFrame.setSize(template.getSize());
                    jFrame.setResizable(false);
                    cPane.add(template);

                    jFrame.setVisible(true);
                    jFrame.setLocationRelativeTo(null);
                    jFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                }

            } else {
                System.out.println("Thank you for using our application");
                running = 0;
            }

            sc.nextLine();

        } while (running == 1);
        sc.close();

    }

}
```

2) Main Page: (loaded from Newclass.java)




```java
public class Newclass extends JPanel {

    private JLabel showDocuments;

    public Newclass(Add a, Container cPane) {

        setSize(800, 800);
        setLayout(new BorderLayout());
        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new BoxLayout(buttonPanel, BoxLayout.Y_AXIS));

        JButton addDocument = new JButton("ADD DOCUMENT");
        addDocument.setPreferredSize(new Dimension(100, 100)); // Set preferred size
        addDocument.setAlignmentX(Component.CENTER_ALIGNMENT); // Center the button horizontally
        addDocument.setBackground(Color.RED);
        addDocument.setForeground(Color.YELLOW);

        buttonPanel.add(addDocument);

        You, 4 days ago | 1 author (You)
        addDocument.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                Add add = new Add();
                add.setSize(getSize());
                next(cPane);
                cPane.add(add);
                cPane.revalidate();
                cPane.repaint();
            }
        });
```

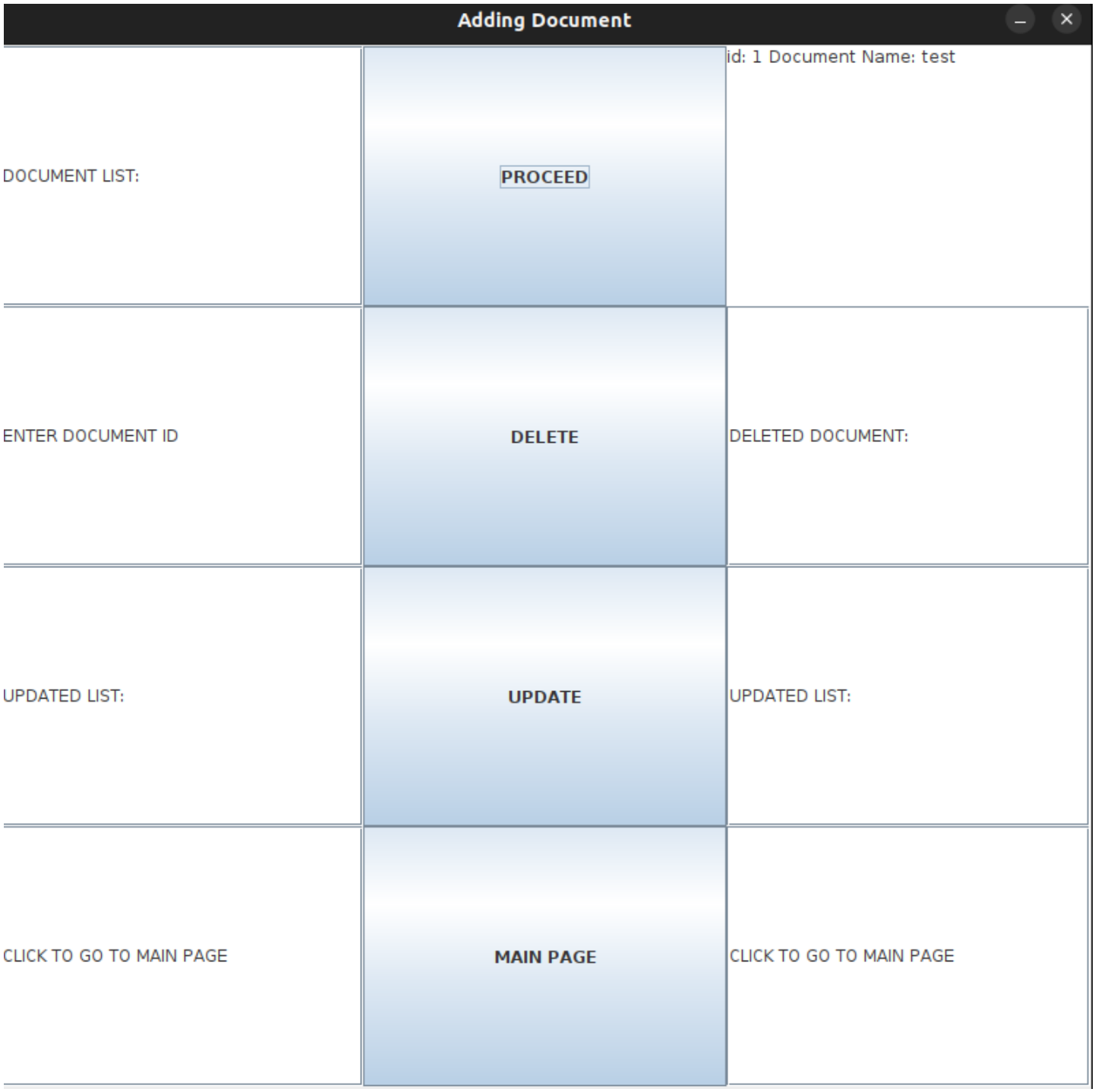
```java
        JButton deleteDocument = new JButton("DELETE DOCUMENT");
        deleteDocument.setPreferredSize(new Dimension(100, 100));
        deleteDocument.setAlignmentX(Component.CENTER_ALIGNMENT);
        deleteDocument.setBackground(Color.RED);
        deleteDocument.setForeground(Color.YELLOW);

        buttonPanel.add(deleteDocument);

        You, 4 days ago | 1 author (You)
        deleteDocument.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                Delete delete = new Delete();
                delete.setSize(getSize());
                next(cPane);
                cPane.add(delete);
                cPane.revalidate();
                cPane.repaint();
            }
        });      You, 4 days ago • Add files
```

3) Add Document (with error-handling): (Add.java)


```java
    public static int[] checks = { 0, 0, 0 }; // name, topic, category

    public void addDocumentToList() {
        if (checks[0] == 1 && checks[1] == 1 && checks[2] == 1) {
            String name = t.getText();
            String topic = t2.getText();
            String category = t4.getText();
            String tag = t6.getText();        You, 3 days ago • add checks to

            Document.doc_name[Document.counter] = name;
            Topic.topic[Document.counter] = topic;
            Category.category[Document.counter] = category;
            Tag.tags[Document.counter] = tag;

            Document.counter = Document.counter + 1;
        }
    }

    public Add() {

        setLayout(new GridLayout(5, 3));

        // get input of doc name
        t = new JTextField("ENTER DOCUMENT NAME");
        t.setSize(300, 300);
        t.setEditable(true);

        // display doc name
        t1 = new JTextField("DOCUMENT NAME: NONE");
        t1.setSize(300, 300);
        t1.setEditable(true);

        JButton j1 = new JButton("PROCEED");

        t2 = new JTextField("ENTER TOPIC");
        t2.setSize(300, 300);
        t2.setEditable(true);

```

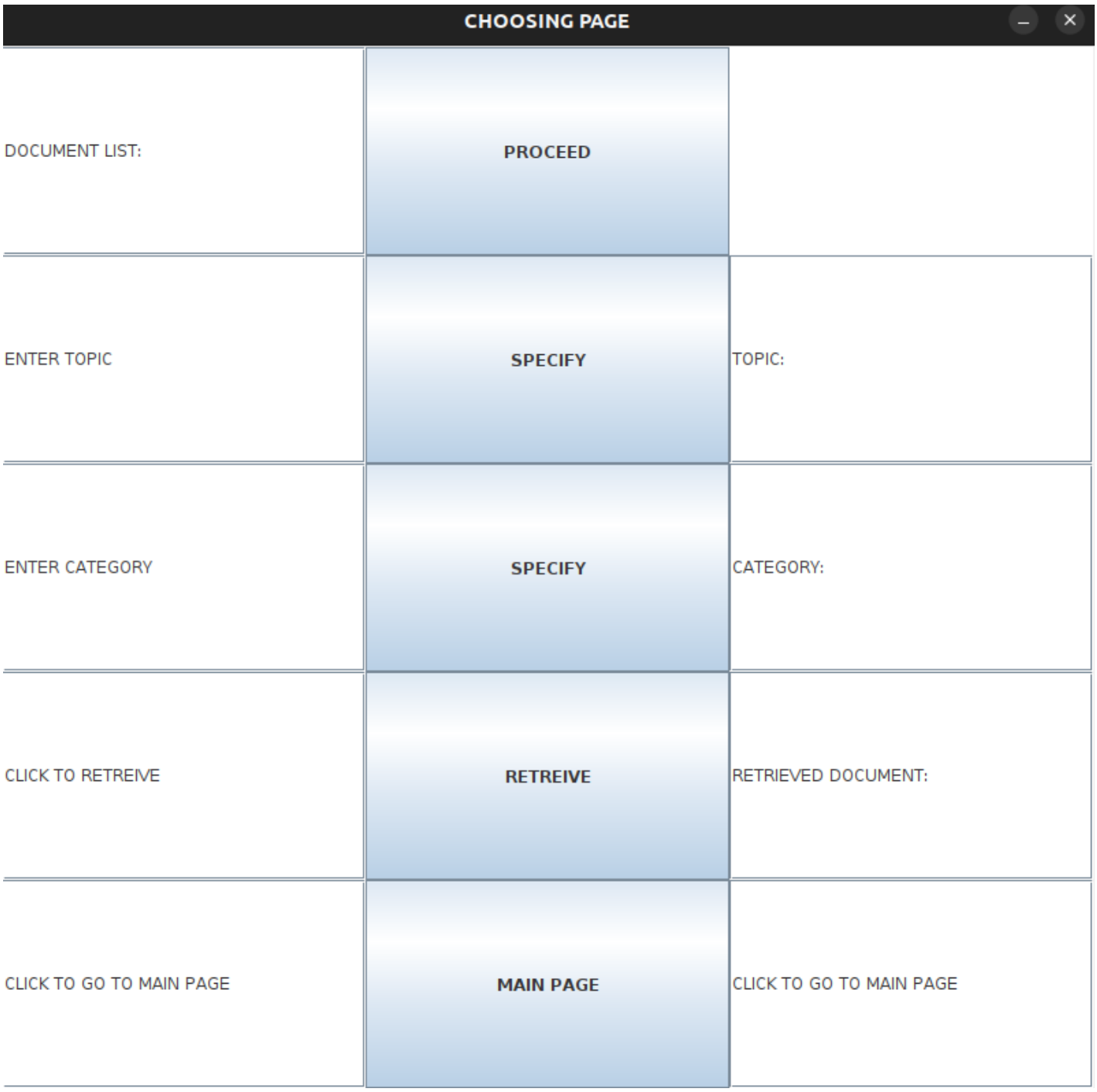
```java
You, 3 days ago | 1 author (You)
j1.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

        String s1 = t.getText();
        // displau doc name on rhs
        t1.setText("DOCUMENT NAME: " + s1);
        checks[0] = 1;
    }
});

You, 3 days ago | 1 author (You)
j2.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

        String s1 = t2.getText();
        t3.setText("TOPIC: " + s1);
        checks[1] = 1;
    }
});

You, 3 days ago | 1 author (You)
j3.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {

        String s1 = t4.getText();
        t5.setText("CATEGORY: " + s1);
        checks[2] = 1;
```

4) Delete Document (Delete.java)



5) Edit Document:
(Edit.java)

6) Retrieve Document: (Retrieve.java)



7) Document.java (for Document objects):

```java
You, 4 days ago | 1 author (You)
import java.io.*;
import java.util.*;
import java.applet.*;
import java.awt.*;
import java.awt.BorderLayout;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JTextField;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.Scanner;

You, 4 days ago | 1 author (You)
@SuppressWarnings("serial")
public class Document {

    public String name;
    public int DocumentID;
    public String category;
    public String tag;
    public String topic;
    public static int counter;
    public static String[] doc_name = new String[100];

    public Document(String S, int id, String topic, String category, String tag) {
        this.name = S;
        this.DocumentID = id;
        this.category = category;
        this.tag = tag;
        this.topic = topic;
    }

    public void getName() {
        System.out.println(name);
    }

    public void getID() {
        System.out.println(DocumentID);
    }

    public String rename(String s) {
        this.name = s;
        return s;      You, 4 days ago • Add files
    }
```

8) User.java (for storing users):

User.java

User.java > ...

You, 4 days ago | 1 author (You)

```java
import java.util.ArrayList;

public class User {
    String username;
    String password;
    public static ArrayList<User> users = new ArrayList<User>();

    public User(String username, String password) {
        this.username = username;
        this.password = password;
        System.out.println("User created successfully. You can now choose to log in.");
    }
}
```

—----------

On a closing note, we would like to thank Prof. O.P. Vyas and Prof. Ranjana Vyas, and all the TAs for their constant support and guidance, and for giving us a chance to explore our potentials and this world of Object-Oriented Programming by implementing the fundamentals through a GUI-based application.

This has given us hands-on experience with Java and UML-diagrams, and it has helped us gain valuable knowledge to help develop efficient UML Diagrams for an efficient Java codebase.

—----------X—----------

Thank You!