

CHAPTER 1

INTRODUCTION

1.1 PROBLEMS DEFINITIONS

The main aim or the objective of the project is to provide an implementation of a chat application like messenger using GUI. We use a simple programming language like Java and develop the modules. As java contains Swings which contain some default methods it would be much easier in designing a project with easy of code. The graphical user interface is a form of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation. The actions in a GUI are usually performed through direct manipulation of the graphical elements. Beyond computers, GUIs are used in many handheld mobile devices such as MP3 players, portable media players, gaming devices, smartphones and smaller household, office and industrial controls.

The term GUI tends not to be applied to other lower-display resolution types of interfaces, such as video games (where head-up display (HUD) is preferred), or not including flat screens, like volumetric displays because the term is restricted to the scope of two-dimensional display screens able to describe generic information, in the tradition of the computer science research at the Xerox Palo Alto Research Center. By the 1980s, cell phones and handheld game systems also employed application specific touchscreen GUIs. Newer automobiles use GUIs in their navigation systems and multimedia centers, or navigation multimedia center combinations.

1.2 OBJECTIVES

The main objective of this project is an implementation of a simple messenger like platform with the help of GUI based program. By submitting the inputs like username and password he/she gets logged into their respective account. After logging in, the list of friends present in the account will be displayed. Everyone's account is separate and by clicking on the account a chat history will be displayed. Here a person is able to communicate through messages. New members can also be added to the list by creating/ by adding their respective accounts.

1.3 METHODOLOGY TO BE FOLLOWED

In the methodology of the code we can divide the whole program into four modules. 1. Login page 2. Signup 3. SecondFrame 4. ChatBox 1. Login page: In this module we take the inputs from the user which are the username and passwords as Labels and providing their respective textfields. It also contains Reset and submit buttons. When we click on submit a new window opens where all the list of friends are displayed. 2. Create new account: If the user wishes to create a new contact we can click on signup button. After clicking it displays a form where all the details of the user are to be entered as input. And that new contact will be added to the contacts list. 3. Signin: After entering the username and password when we click on login a new frame opens up which contains the list of contacts a user have. 4. ChatHistory: When multiple contacts are being displayed we can click on any of the friend to chat with that particular person and a previous chat history will be displayed.

1.4 EXPECTED OUTCOMES

- By submitting the inputs like username and password he/she gets logged into their respective account.
- After logging in, the list of friends present in the account will be displayed. Everyone's account is separate and by clicking on the account a chat history will be displayed.

- Here a person is able to communicate through messages.
- New members can also be added to the list by creating/ by adding their respective accounts.

1.5 HARDWARE AND SOFTWARE REQUIRMENTS

1. Core i5 processor
2. RAM 2GB
3. HDD capacity 500 GB
4. Front end java
5. Operating system: windows 8
6. Speed: 1.1 GHz
7. 10TH generation
8. A CD -ROM drive
9. Minimum Windows 95 software
10. Hard drive and minimum of 8MB

CHAPTER 2

OBJECT ORIENTED AND CONCEPTS

2.1 CLASS

- A class is a user defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type. In general, class declarations can include these components, in order:
- Modifiers : A class can be public or has default access.
- Class name: The name should begin with a initial letter (capitalized by convention).
- Superclass(if any): The name of the class's parent (superclass), if any, preceded by the keyword extends. A class can only extend (subclass) one parent.
- Interfaces(if any): A comma-separated list of interfaces implemented by the class, if any, preceded by the keyword implements. A class can implement more than one interface.
- Body: The class body surrounded by braces, { }.Constructors are used for initializing new objects. Fields are variables that provides the state of the class and its objects, and methods are used to implement the behavior of the class and its objects.

- There are various types of classes that are used in real time applications such as nested classes, anonymous classes, lambda expressions.

2.2 OBJECT

It is a basic unit of Object-Oriented Programming and represents the real-life entities. A typical Java program creates many objects, which as you know, interact by invoking methods. An object consists of:

State: It is represented by attributes of an object. It also reflects the properties of an object.

Behavior: It is represented by methods of an object. It also reflects the response of an object with other objects.

Identity: It gives a unique name to an object and enables one object to interact with other objects.

Objects correspond to things found in the real world. For example, a graphics program may have objects such as “circle”, “square”, “menu”. An online shopping system might have objects such as “shopping cart”, “customer”, and “product”.

Declaring Objects (Also called instantiating a class):

When an object of a class is created, the class is said to be instantiated. All the instances share the attributes and the behavior of the class. But the values of those attributes, i.e. the state are unique for each object. A single class may have any number of instances.

As we declare variables like (type name;). This notifies the compiler that we will use name to refer to data whose type is type. With a primitive variable, this declaration also reserves the proper amount of memory for the variable. So for reference variable, type must be strictly a concrete class name. In general, we can't create objects of an abstract class or an interface.

Initializing an object

The new operator instantiates a class by allocating memory for a new object and returning a reference to that memory. The new operator also invokes the class constructor.

- This class contains a single constructor. We can recognize a constructor because its declaration uses the same name as the class and it has no return type. The Java compiler differentiates the constructors based on the number and the type of the arguments.

Note: All classes have at least one constructor. If a class does not explicitly declare any, the Java compiler automatically provides a no-argument constructor, also called the default constructor. This default constructor calls the class parent's no-argument constructor (as it contains only one statement i.e., `super();`), or the Object class constructor if the class has no other parent (as Object class is parent of all classes either directly or indirectly).

Ways to create object of a class

There are four ways to create objects in java. Strictly speaking there is only one way (by using new keyword), and the rest internally use new keyword.

- Using new keyword: It is the most common and general way to create object in java.
- Using `Class.forName(String className)` method: There is a pre-defined class in `java.lang` package with name `Class`. The `forName(String className)` method returns the `Class` object associated with the class with the given string name. We have to give the fully qualified name for a class. On calling `new Instance()` method on this `Class` object returns new instance of the class with the given string name.
- Using `clone()` method: `clone()` method is present in `Object` class. It creates and returns a copy of the object.

- Deserialization: De-serialization is technique of reading an object from the saved state in a file.

Creating multiple objects by one type only

- In real-time, we need different objects of a class in different methods. Creating a number of references for storing them is not a good practice and therefore we declare a static reference variable and use it whenever required. In this case, wastage of memory is less. The objects that are not referenced anymore will be destroyed by Garbage Collector of java.
- In inheritance system, we use parent class reference variable to store a sub-class object. In this case, we can switch into different subclass objects using same referenced variable.
- Anonymous objects are the objects that are instantiated but are not stored in a reference variable.
 1. They are used for immediate method calling.
 2. They will be destroyed after method calling.
 3. They are widely used in different libraries. For example, in AWT libraries, they are used to perform some action on capturing an event

2.3 INHERITANCE

Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object. It is an important part of OOPs (Object Oriented programming system).

The idea behind inheritance in Java is that you can create new classes that are built upon existing classes. When you inherit from an existing class, you can reuse methods and fields of the parent class. Moreover, you can add new methods and fields in your current class also.

2.4 POLYMORPHISM

- Polymorphism in Java is a concept by which we can perform a single action in different ways. Polymorphism is derived from 2 Greek words: poly and morphs. The word "poly" means many and "morphs" means forms. So polymorphism means many forms.
- There are two types of polymorphism in Java: compile-time polymorphism and runtime polymorphism. We can perform polymorphism in java by method overloading and method overriding.
- If you overload a static method in Java, it is the example of compile time polymorphism. Here, we will focus on runtime polymorphism in java.
- Runtime polymorphism or Dynamic Method Dispatch is a process in which a call to an overridden method is resolved at runtime rather than compile-time.
- In this process, an overridden method is called through the reference variable of a superclass. The determination of the method to be called is based on the object being referred to by the reference variable.
- Compile time polymorphism or static method dispatch is a process in which a call to an overloading method is resolved at compile time rather than at run time. In this process, we done overloading of methods is called through the reference variable of a class here no need to superclass.
- Method Overloading in Java:
 - If a class have multiple methods by same name but different parameters, it is known as Method Overloading.
 - If we have to perform only one operation, having the same name of the methods increases the readability of the program. Suppose you have to perform addition of the given numbers but there can be any number of

arguments, if you write the method such as `sum(int,int)` for two parameters, and `sum2(int,int,int)` for three parameters then it may be difficult for you as well as other programmers to understand the behavior of the method because its name differs. So, we perform method overloading to figure out the program quickly.

- Advantage of method overloading:
- Method overloading increases the readability of the program.
- Different ways to overload the method:
- There are two ways to overload the method in java
- By changing number of arguments.
- By changing the data type

2.5 ABSTRACT CLASS

- A class which is declared with the abstract keyword is known as an abstract class in [Java](#). It can have abstract and non-abstract methods (method with the body).
- Before learning the Java abstract class, let's understand the abstraction in Java first.
- A class which is declared with the abstract keyword is known as an abstract class in [Java](#). It can have abstract and non-abstract methods (method with the body)
- Another way, it shows only essential things to the user and hides the internal details, for example, sending SMS where you type the text and send the message. You don't know the internal processing about the message delivery

2.6 MULTI THREADING

- Multithreading in [Java](#) is a process of executing multiple threads simultaneously.
- A thread is a lightweight sub-process, the smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking. we

use multithreading than multiprocessing because threads use a shared memory area. They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process. It does not block the user because threads are independent and you can perform multiple operations at the same time. You can perform many operations together, so it saves time. Threads are independent, so it doesn't affect other threads if an exception occurs in a single thread.

2.7 I/O FUNCTIONS

- Java I/O (Input and Output) is used *to process the input and produce the output*.
- Java uses the concept of a stream to make I/O operation fast. The java.io package contains all the classes required for input and output operations. We can perform file handling in by Java I/O API.
- A stream is a sequence of data. In Java, a stream is composed of bytes. It's called a stream because it is like a stream of water that continues to flow.
- In Java, 3 streams are created for us automatically. All these streams are attached with the console.
- **System.out:** standard output stream
- **System.in:** standard input stream
- **System.err:** standard error stream
- Let's see the code to print **output and an error** message to the console.
- `System.out.println("simple message");`
- `System.err.println("error message");`

2.8 JAVA PACKAGES

- A java package is a group of similar types of classes, interfaces and sub-packages.

- Package in java can be categorized in two form, built-in package and user-defined package.
- There are many built-in packages such as java, lang, awt, javax, swing, net, io, util, sql etc.
- Here, we will have the detailed learning of creating and using user-defined packages.
- **Advantage of Java Package**
- Java package is used to categorize the classes and interfaces so that they can be easily maintained.
- Java package provides access protection.
- Java package removes naming collision.

2.9 EXCEPTIONAL HANDLING

The Exception Handling in Java is one of the powerful *mechanism to handle the runtime errors* so that normal flow of the application can be maintained. In this page, we will learn about Java exceptions, its type and the difference between checked and unchecked exceptions.

CHAPTER 3

DESIGN

3.1 DESIGN GOALS

The main objective of this project is an implementation of a simple messenger like platform with the help of GUI based program. By submitting the inputs like username and password he/she gets logged into their respective account. After logging in, the list of friends present in the account will be displayed. Everyone's account is separate and by clicking on the account a chat history will be displayed. Here a person is able to communicate through messages. New members can also be added to the list by creating/ by adding their respective accounts. The main aim or the objective of the project is to provide an implementation of a chat application like messenger using GUI. We use a simple programming language like Java and develop the modules. As java contains Swings which contain some default methods it would be much easier in designing a project with easy of code

3.2 ALGORITHM

- In the implementation of the code we can divide the whole program into four modules.
- Login page 2. Signup 3. SecondFrame 4. ChatBox 1. Login page: In this module we take the inputs from the user which are the username and passwords as Labels and providing their respective text fields. It also contains Reset and submit buttons. When we click on submit a new window opens where all the list of friends is displayed. 2. Create new account:
 - If the user wishes to create a new contact, we can click on signup button. After clicking it displays a form where all the details of the user are to be entered as input

- And that new contact will be added to the contacts list. 3. Sign-in: After entering the username and password when we click on login a new frame opens up which contains the list of contacts a user has.
- 4. ChatHistory: When multiple contacts are being displayed, we can click on any of the friend to chat with that particular person and a previous chat history will be displayed.

CHAPTER-4

4.1 FUNCTIONALITY

```
import java.awt.*;

import javax.swing.*;

import java.awt.event.*;

import java.util.*;

import java.io.*;

class SAKETH extends JFrame

{

    static JFrame f;

    public static void main(String[] args)

    {

        f=new JFrame("MESSENGER");

        JPanel p=new JPanel();

        JLabel n=new JLabel("USER NAME");

        n.setBounds(10,10,100,30);

        JLabel i=new JLabel("PHONE NUMBER");

        i.setBounds(10,50,100,30);

        JLabel b=new JLabel("G-MAIL");

        b.setBounds(10,100,100,30);
```

```
JLabel y=new JLabel("D.O.B");  
  
y.setBounds(10,150,100,30);  
  
JLabel nb=new JLabel("ADD CONTACT");  
  
nb.setBounds(10,200,100,30);  
  
JTextField nt=new JTextField();  
  
nt.setBounds(180,10,100,30);  
  
JTextField it=new JTextField();  
  
it.setBounds(180,50,100,30);  
  
JTextField bt=new JTextField();  
  
bt.setBounds(180,100,100,30);  
  
JTextField yt=new JTextField();  
  
yt.setBounds(180,150,100,30);  
  
JTextField nbt=new JTextField();  
  
nbt.setBounds(180,200,100,30);  
  
JButton sl=new JButton("LOGIN");  
  
sl.setBounds(85,250,100,30);  
  
JLabel bi=new JLabel("USER NAME");  
  
bi.setBounds(10,10,100,30);  
  
JLabel bn=new JLabel("PASSWORD");  
  
bn.setBounds(10,50,100,30);
```

```
JLabel an=new JLabel("G-MAIL");  
an.setBounds(10,100,100,30);  
  
JLabel pr=new JLabel("PASSWORD");  
pr.setBounds(10,150,100,30);  
  
JTextField bit=new JTextField();  
bit.setBounds(180,10,100,30);  
  
JTextField bnt=new JTextField();  
bnt.setBounds(180,50,100,30);  
  
JTextField ant=new JTextField();  
ant.setBounds(180,100,100,30);  
  
JTextField pt=new JTextField();  
pt.setBounds(180,150,100,30);  
  
JButton sa=new JButton("SUBMIT");  
sa.setBounds(85,200,100,30);  
  
JLabel ebn=new JLabel("MESSAGES");  
ebn.setBounds(10,10,150,30);  
  
JLabel ebi=new JLabel("UPLOAD VEDIOS");  
ebi.setBounds(10,50,150,30);  
  
JLabel ean=new JLabel("UPLOAD IMAGES");  
ean.setBounds(10,100,150,30);
```



```

JTextField ebnt=new JTextField();

ebnt.setBounds(200,10,100,30);

JTextField ebit=new JTextField();

ebit.setBounds(200,50,100,30);

JTextField eant=new JTextField();

eant.setBounds(200,100,100,30);

JTextField box=new JTextField();

box.setBounds(10,200,360,50);

JButton sd=new JButton("SUBMIT");

sd.setBounds(85,150,100,30);

//.....MAIN PAGE NAMES.....

JLabel l=new JLabel("CREATE");

l.setBounds(10,10,100,60);

JLabel a=new JLabel("LOGIN");

a.setBounds(10,50,100,30);

JLabel d=new JLabel("CHAT HISTORY");

d.setBounds(10,90,100,30);

//.....MAIN PAGE BUTTONS.....

JButton lb=new JButton("CREATE NEW ACCOUNT");

lb.setBounds(110,10,180,30);
```

```

JButton ab=new JButton("LOGIN");

ab.setBounds(110,50,100,30);

JButton db=new JButton("CHAT HISTORY");

db.setBounds(110,90,180,30);

p.add(l);

p.add(a);

p.add(d);

p.add(lb);

p.add(ab);

p.add(db);

f.add(p);

f.setSize(300,300);

p.setLayout(null);

f.setVisible(true);

lb.addActionListener(new ActionListener(){

    public void actionPerformed(ActionEvent e){

        f=new JFrame("CREATE NEW ACCOUNT");

        JPanel p=new JPanel();

        p.add(n);

        p.add(i);
    }
});

```

```
p.add(sl);

p.add(b);

p.add(y);

p.add(nb);

p.add(nt);

p.add(it);

p.add(bt);

p.add(yt);

p.add(nbt);

f.add(p);

f.setSize(300,400);

p.setLayout(null);

f.setVisible(true);

}

}}

);

ab.addActionListener(new ActionListener(){

    public void actionPerformed(ActionEvent e){

        f=new JFrame("LOGIN");

        JPanel p=new JPanel();
```

```
p.add(bi);

p.add(bn);

p.add(an);

p.add(pr);

p.add(bit);

p.add(bnt);

p.add(ant);

p.add(pt);

p.add(sa);

f.add(p);

f.setSize(300,400);

p.setLayout(null);

f.setVisible(true);

}

});

db.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e)

{

f=new JFrame("CHAT HISTORY");

JPanel p=new JPanel();
```

```
p.add(ebn);

p.add(ebi);

p.add(ean);

p.add(ebnt);

p.add(ebit);

p.add(eant);

p.add(box);

p.add(sd);

f.add(p);

f.setSize(400,300);

p.setLayout(null);

f.setVisible(true);

}

});

sl.addActionListener(new ActionListener(){

    public void actionPerformed(ActionEvent e){

        f=new JFrame("LOG OUT");

        JPanel p=new JPanel();

    }

});
```

```
sa.addActionListener(new ActionListener(){  
    public void actionPerformed(ActionEvent e){  
  
    }  
  
});  
  
sd.addActionListener(new ActionListener(){  
    public void actionPerformed(ActionEvent e){  
  
    }  
  
});  
  
}  
  
}
```

CHAPTER 5

RESULT

As soon as you enter the app

 MESSENGER

CREATE

CREATE NEW ACCOUNT

LOGIN

LOGIN

CHAT HISTORY

CHAT HISTORY

If we want to create new account



A screenshot of a web application window titled "CREATE NEW ACCO...". The window contains a form with five input fields: "USER NAME", "PHONE NUMBER", "G-MAIL", "D.O.B", and "ADD CONTACT". Each field is represented by a text box. Below the input fields is a blue button labeled "LOGIN". The window has standard OS controls (minimize, maximize, close) in the top right corner.

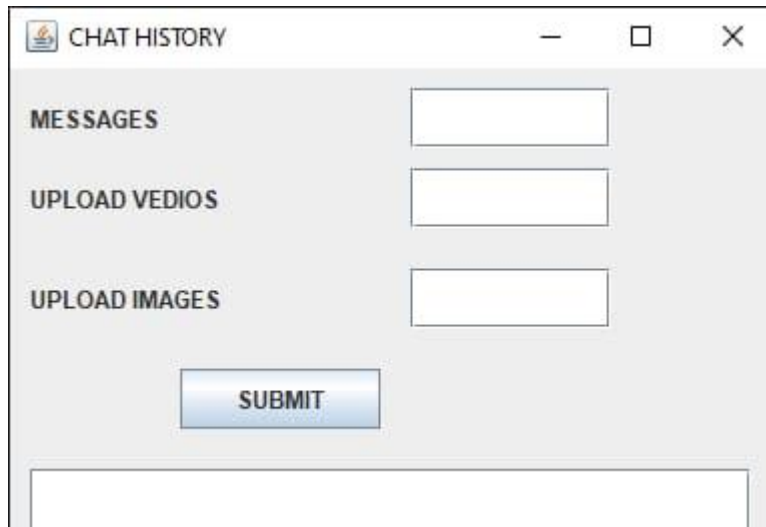
If we want to login to the already existing account



A screenshot of a web application window titled "LOGIN". The window contains a form with four input fields: "USER NAME", "PASSWORD", "G-MAIL", and "PASSWORD". Each field is represented by a text box. Below the input fields is a blue button labeled "SUBMIT". The window has standard OS controls (minimize, maximize, close) in the top right corner.

THE SMART COMMUNICATION

If we want to see the chat history



A web form titled "CHAT HISTORY" with a standard window header (minimize, maximize, close buttons). The form contains three input fields and a submit button:

- MESSAGES**: A text input field.
- UPLOAD VEDIOS**: A text input field.
- UPLOAD IMAGES**: A text input field.
- SUBMIT**: A blue button with white text.

CHAPTER 6

CONCLUSION

By using GUI, we can easily develop many user interface applications with ease. When we use only a language like java which contains some predefined methods it would be easier. We can even develop applications quite easily and more number of options and designs are available in GUI.

CHAPTER 7

REFERENCE