School of Computer Science and Engineering

Lovely Professional University

Phagwara, Punjab (India)

Java Programming

Clock App

PROJECT REPORT

Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

Narayan Gupta (12110966)

Aditya Narayan Pandey(12107845)

Farid Mohammed Farid Saleh(12009180)

Under the guidance of

Ms. Chandani Bhasin

Table of Contents

S.No	CONTENTS	PAGES
1.	Acknowledgement	3
2.	Introduction	4
3.	Uses	4
4.	Objectives	4
5.	Basic Functions	5 - 11
6.	Roles and Responsibility	11
7.	Conclusion	11-12

ACKNOWLEDGMENT

Place : Lovely Professional University

Date : 18th April , 2023

We would like to thank Ms. Chandani Bhasin for assigning us with this Java project. Through this project we are able to grasp more technical and have a hands-on practical experience with Java projects. Through it we are able to learn how a project is created and how necessary and crucial technical knowledge is. We are really grateful to the faculty that has provided us with the necessary guidelines.

Name	Registration No.
Narayan Gupta	**12110966**
Aditya Narayan Pandey	**12107845**
Farid Mohammed Farid Saleh	**12009180**

Introduction:

Introducing my Clock App, designed to provide accurate timekeeping and convenience for users. With multiple time zone support(World Clock) and customizable stop clock, this app is perfect for those who need to keep track of time in different parts of the world. The sleek and modern design of this project is both visually appealing and easy to use, making it a great addition to anyone's phone. I'm excited to present this project with you all in thus class.

Uses:

- Accurate timekeeping: A Clock App is primarily designed to provide accurate timekeeping. It ensures that you always know the time, no matter where you are in the world.
- Multiple time zone support: If you frequently travel or work with people in different time zones, a Clock App with multiple time zone support can be incredibly useful. It allows you to quickly and easily switch between different time zones, so you can stay on top of your schedule.
- Stopwatch and timer: Many Clock Apps also include a stopwatch and timer. This can be useful for tracking your workout or timing your cooking.

Overall, a Clock App is a versatile tool that can help you manage your time and stay on top of your schedule, whether you're at home or on the go.

Objectives:

The objective of this project is to implement a fully featured Clock App by using different Java swing components and awt components. Some of the basic functionality that we have implemented are Thread concept to change the time of our Clock every millisecond

We have also used Thread to change the time in stop watch every 100 millisecond

Events of Button and Mouse are handled by extending Listener classes. Our project is :

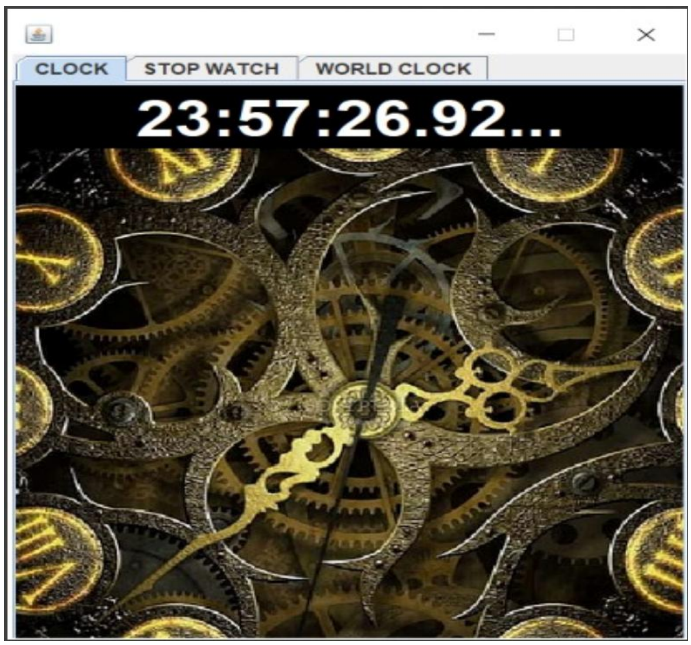
- User friendly
- Helps in viewing your data and privileges
- Easy to handle changes.
- Efficient and fast in response
- Can work on various systems like, windows, Mac etc.

BASIC FUNCTIONS:

Our programmed clock app have some basic function which make our project different from others. This Java code creates a clock application that consists of three tabs: Current Time, Stop Watch, and World Clock.

- **Current Time:**

The "Current Time" tab displays an image of a clock face, with a label below it showing the current time. A thread updates the time every millisecond.



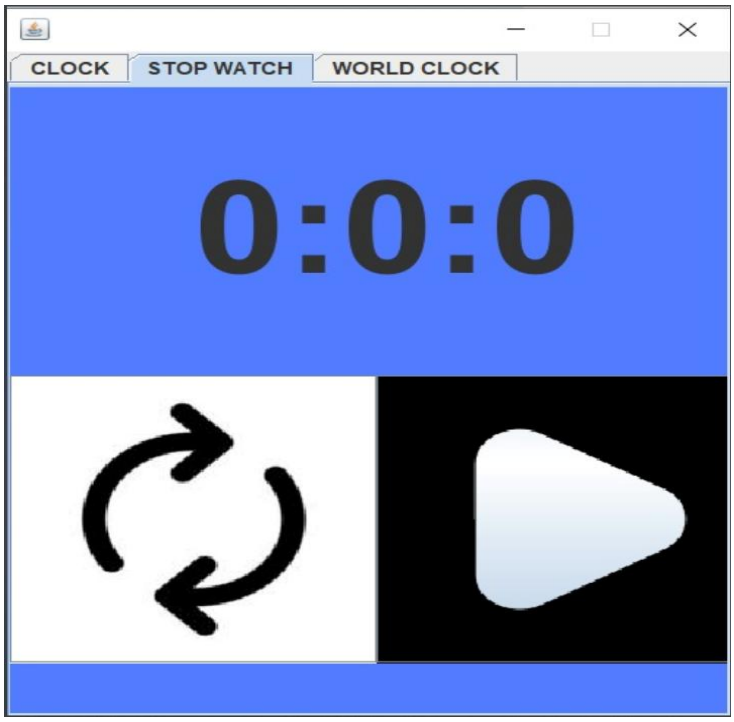
Code for showing current time:

```
//Current time
JPanel jp1=new JPanel();
//Image
ImageIcon ic=new ImageIcon("clock1.jfif");
jp1.setBackground(Color.black);
JLabel ima=new JLabel(ic);
jp1.add(ima);
jp1.setLayout(null);
JLabel jl1=new JLabel();
jl1.setForeground(Color.white);
Font f1=new Font("SansSerif",1,40);
jl1.setFont(f1);
```

```
// Thread
    Thread t=new Thread(()-> {
       while(true) {
          jl1.setText(""+LocalTime.now());
          try {
             Thread.sleep(1);
          } catch(Exception e) {};
       }
    });
    t.start();
    jl1.setText(""+LocalTime.now());
```

- **Stop Watch:**

The "Stop Watch" tab displays a panel with a label that shows the current elapsed time. Two buttons, "Pause" and "Reset," are included. When the user clicks the "Pause" button, the timer stops counting up, and when the user clicks the "Reset" button, the timer resets to zero.



Required Code:

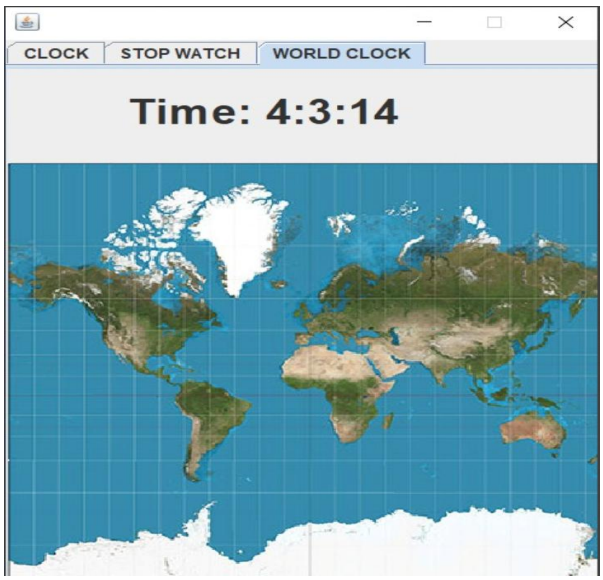
```
JPanel jp2=new JPanel();
jp2.setBackground(new Color(82, 124, 255));
jp2.setLayout(null);
```

```
ImageIcon pa=new ImageIcon("pause.png");
ImageIcon re=new ImageIcon("reset.jfif");
JButton pause=new JButton(pa);
JButton reset=new JButton(re);
JLabel jl2=new JLabel();
Font f2=new Font("SansSerif",1,90);
jl2.setFont(f2);
jp2.add(pause);
jp2.add(reset);
class MyListener implements ActionListener {
    int x=0,y=0;
    int h=0,m=0,s=0;
    public void actionPerformed(ActionEvent ae) {
        y++;
        if(ae.getSource()==pause) {
            if(x==0)x=1;
            else x=0;
        } else if(ae.getSource()==reset) {
            h=0;
            m=0;
            s=0;
            jl2.setText(h+":"+m+":"+s);
        }
     if(y==1) {
            Thread nt=new Thread(()-> {
                while(true) {
                    if(x==1) {
                        jl2.setText(h+":"+m+":"+s);
                        s++;
                        if(s==60) {
                            m++;
                            s=0;
```

```
                }
                if(m==60) {
                    h++;
                    m=0;
                }
            }
            try {
                Thread.sleep(1000);
            } catch(Exception e) {}
        }
    });
    nt.start();
  }
 }
}
jl2.setText(0+":"+0+":"+0);
```

- **World Clock:**

The "World Clock" tab displays a map image and labels that show the current time for various time zones. When the user moves the mouse over a particular region of the map, the label updates to show the current time in that time zone.



Required code:

```
JPanel jp3=new JPanel();
```

```
        jp3.setLayout(null);
        JLabel jl3=new JLabel();
        ImageIcon map=new ImageIcon("map1.jpg");
        JLabel jl4=new JLabel(map);
        jp3.add(jl3);
        jp3.add(jl4);
        Font f3=new Font("SansSerif",1,30);
        jl3.setFont(f3);
      class Mouse extends MouseAdapter {
          int x,y;
          public void mouseMoved(MouseEvent me) {
             x=me.getX();
             y=me.getY();
                    jl3.setText(x+" "+y);
             if(x>272&&x<304&&y>239&&y<273) {
                jl3.setText("Time:
"+LocalTime.now().getHour()+":"+LocalTime.now().getMinute()+":"+LocalTime.now().getSecond());
             }
             if(x>327&&x<369&&y>295&&y<323) {
                ZoneId z=ZoneId.of("Australia/Melbourne");
                jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
             }
             if(x>64&&x<111&&y>217&&y<243) {
                ZoneId z=ZoneId.of("America/Los_Angeles");
                jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
             }
             if(x>127&&x<157&&y>273&&y<312) {
                ZoneId z=ZoneId.of("Brazil/East");
                jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
             }
             if(x>117&&x<131&&y>311&&y<353) {
```

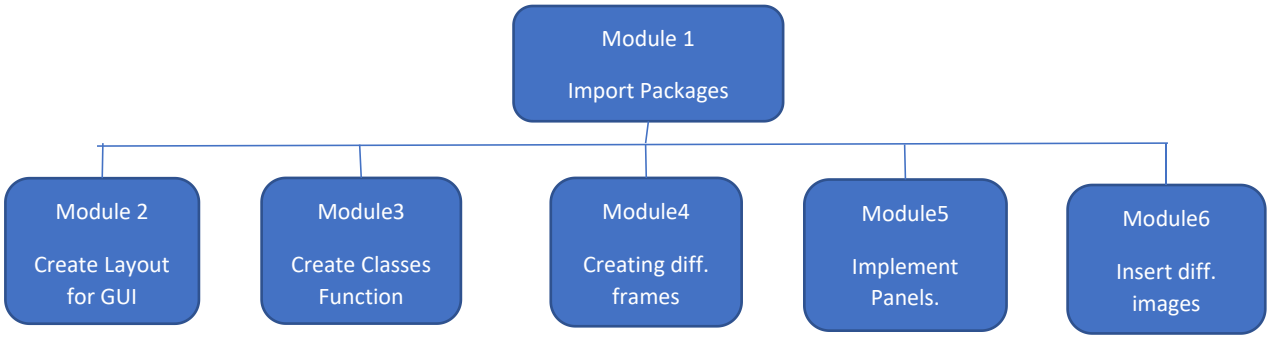
```
            ZoneId z=ZoneId.of("America/Argentina/Mendoza");
            jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
        }
        if(x>344&&x<361&&y>222&&y<250) {
            ZoneId z=ZoneId.of("Japan");
            jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
        }
        if(x>213&&x<236&&y>210&&y<230) {
            ZoneId z=ZoneId.of("Europe/Athens");
            jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
        }
        if(x>293&&x<346&&y>170&&y<207) {
            ZoneId z=ZoneId.of("Europe/Moscow");
            jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
        }
        if(x>203&&x<241&&y>247&&y<286) {
            ZoneId z=ZoneId.of("Africa/Algiers");
            jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
        }
        if(x>22&&x<377&&y>386&&y<431) {
            ZoneId z=ZoneId.of("Antarctica/South_Pole");
            jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
        }
            if(x>36&&x<109&&y>171&&y<211) {
            ZoneId z=ZoneId.of("Canada/Yukon");
            jl3.setText("Time:
"+LocalTime.now(z).getHour()+":"+LocalTime.now(z).getMinute()+":"+LocalTime.now(z).getSecond());
        }
    }
```

}

Roles and Responsibility :

Names(Reg. No.)	Roles and Responsibility
Narayan Gupta(12110966)	GUI by using diff. Components of swing and awt packages.
Aditya Narayan Pandey(12107845)	Basic functions implementation (like clock, stop watch and world timing)
Farid Mohammed Farid Saleh(12009180)	Testing and Debugging

Work Division:



Conclusion:

The program creates a JFrame with a fixed size and sets its location to be in the center of the screen. The frame's default close operation is set to EXIT_ON_CLOSE, and its visibility is set to true. The frame is not resizable.

Next, a JTabbedPane is created, which will hold the three different clocks. The first tab contains a clock showing the current time, which is displayed using a JLabel. The time is updated every millisecond using a separate thread that runs continuously in the background.

The second tab contains a stopwatch that allows the user to start, stop, and reset the timer. The stopwatch also uses a JLabel to display the elapsed time, which is updated every second using another thread.

The third tab displays a world map image, and when the user hovers over certain regions of the map, the program displays the current time in that region. The time is determined using the LocalTime class and the ZoneId class.

Overall, this program demonstrates some of the basic features of Java Swing, including creating a JFrame, using a JTabbedPane, creating labels and buttons, using separate threads for background tasks, and handling user input with ActionListeners and MouseAdapters.

References:

- Google.com
- Short Term Course on Desktop Application Development using JAVA.