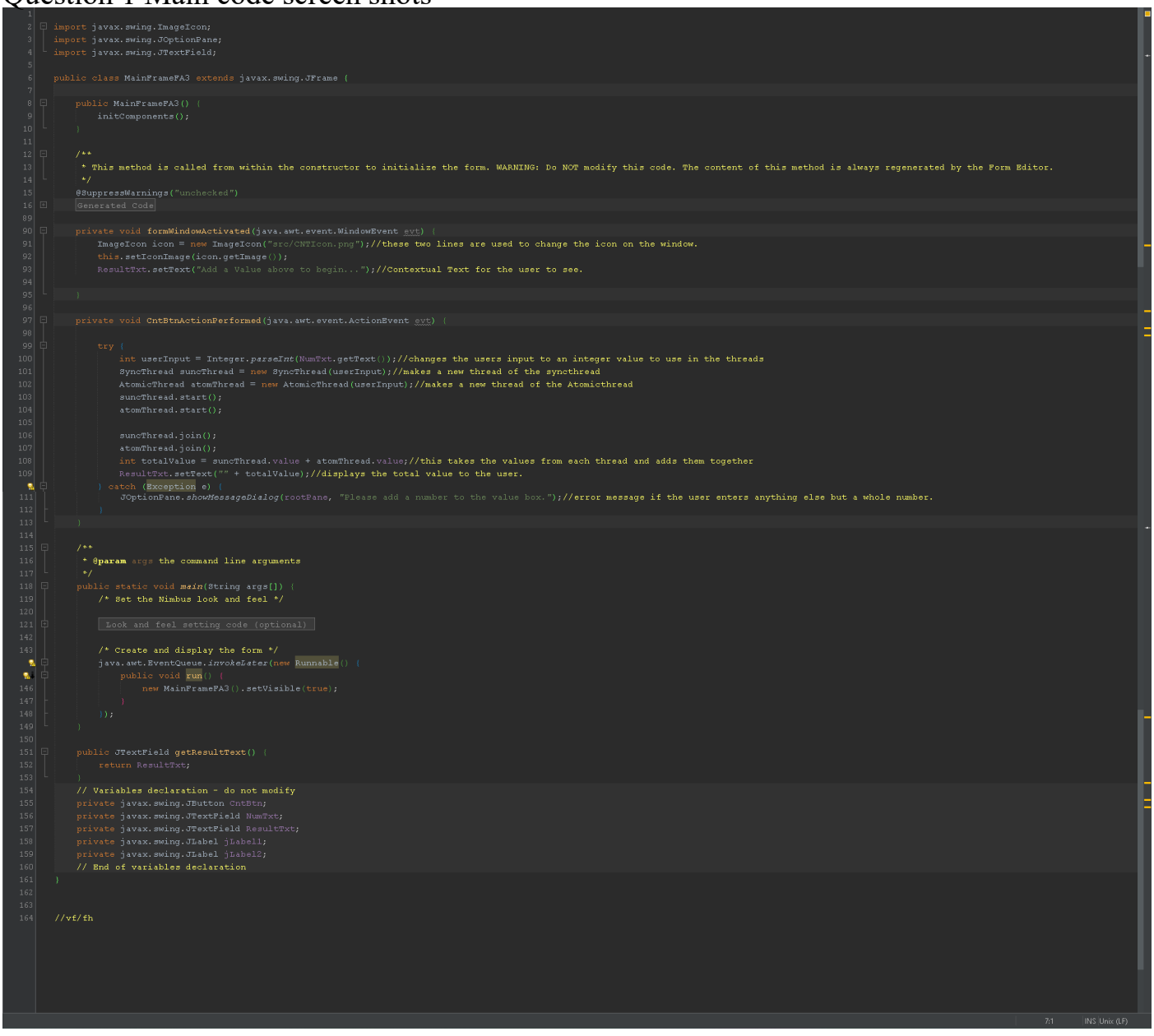
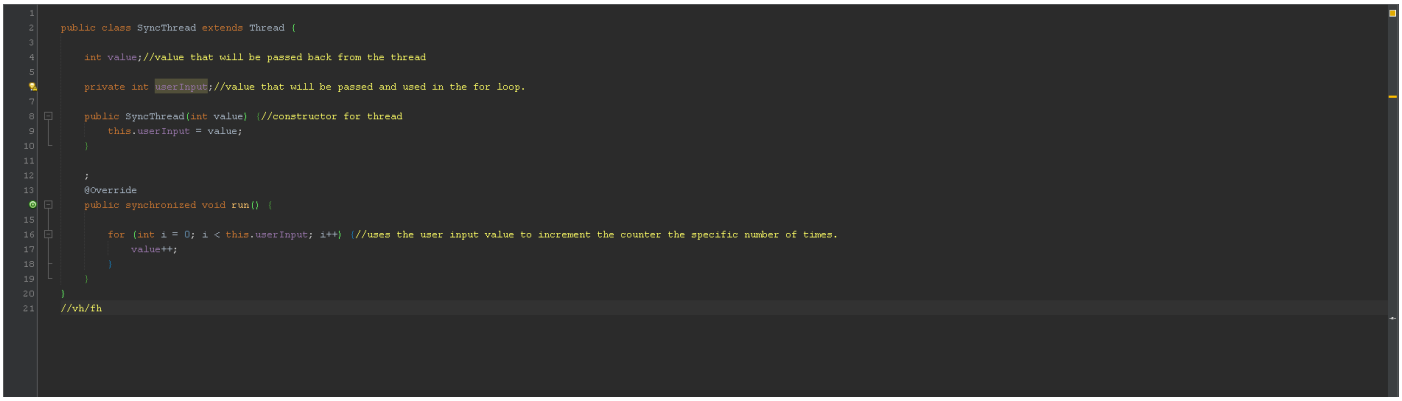
Advanced Java Semester 2
JD522
Formative 3
Jonathan Van Eyssen 20231438

Question 1 Main code screen shots



```
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;
import javax.swing.JTextField;

public class MainFramePA3 extends javax.swing.JFrame {

    public MainFramePA3() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code

    private void formWindowActivated(java.awt.event.WindowEvent evt) {
        ImageIcon icon = new ImageIcon("src/CNTIcon.png");//these two lines are used to change the icon on the window.
        this.setIconImage(icon.getImage());
        ResultTxt.setText("Add a Value above to begin...");//Contextual Text for the user to see.

    }

    private void CntBtnActionPerformed(java.awt.event.ActionEvent evt) {

        try {
            int userInput = Integer.parseInt(NumTxt.getText());//changes the users input to an integer value to use in the threads
            SyncThread suncThread = new SyncThread(userInput);//makes a new thread of the syncthread
            AtomicThread atomThread = new AtomicThread(userInput);//makes a new thread of the Atomicthread
            suncThread.start();
            atomThread.start();

            suncThread.join();
            atomThread.join();
            int totalValue = suncThread.value + atomThread.value;//this takes the values from each thread and adds them together
            ResultTxt.setText("" + totalValue);//displays the total value to the user.
        } catch (Exception e) {
            JOptionPane.showMessageDialog(rootPane, "Please add a number to the value box.");//error message if the user enters anything else but a whole number.
        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */

        Look and feel setting code (optional)

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MainFramePA3().setVisible(true);
            }
        });
    }

    public JTextField getResultText() {
        return ResultTxt;
    }
    // Variables declaration - do not modify
    private javax.swing.JButton CntBtn;
    private javax.swing.JTextField NumTxt;
    private javax.swing.JTextField ResultTxt;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    // End of variables declaration
}


//vf/fh
```

Question 1 Sync Thread screen shot



```
public class SyncThread extends Thread {

    int value;//value that will be passed back from the thread

    private int userInput;//value that will be passed and used in the for loop.

    public SyncThread(int value) {//constructor for thread
        this.userInput = value;
    }

    ;
    @Override
    public synchronized void run() {

        for (int i = 0; i < this.userInput; i++) {//uses the user input value to increment the counter the specific number of times.
            value++;
        }
    }
}
//vh/fh
```

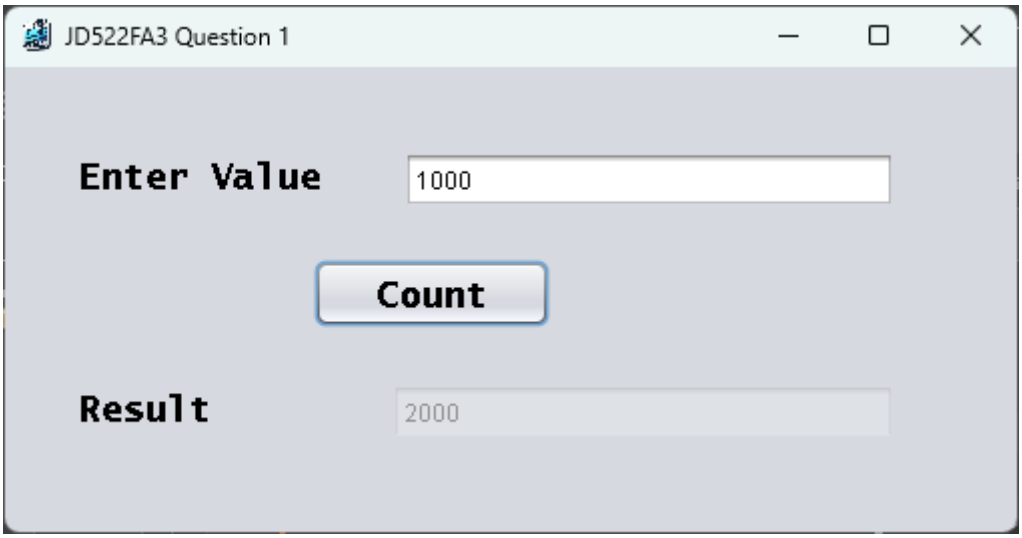
Question 1 Atomic Integer Thread

```
import java.util.concurrent.atomic.AtomicInteger;

public class AtomicThread extends Thread {

    private AtomicInteger atomCTN = new AtomicInteger(0);//makes a new atomic integer object and gives it a Zero value

    public synchronized void increment()//method to increament the atomic integer.
    {
        atomCTN.incrementAndGet();//this just increments the atomic integer
    }

    public int getCounter() {//getter for the atomic integer.
        return atomCTN.get();
    }

    int value; //value that will be passed back from the thread

    private int userInput; //value that will be passed and used in the for loop.

    public AtomicThread(int value) { //constructor for thread
        this.userInput = value;
    }
    @Override
    public synchronized void run() {

        for (int i = 0; i < this.userInput; i++) { //uses the user input value to increment the counter the specific number of times.
            increment();
        }
        value = getCounter();//this assigns the value of the atomic integer to the value so that it can be used in the main to show the user.
    }
}
//vh/fh
```

Question 1 Application screenshots

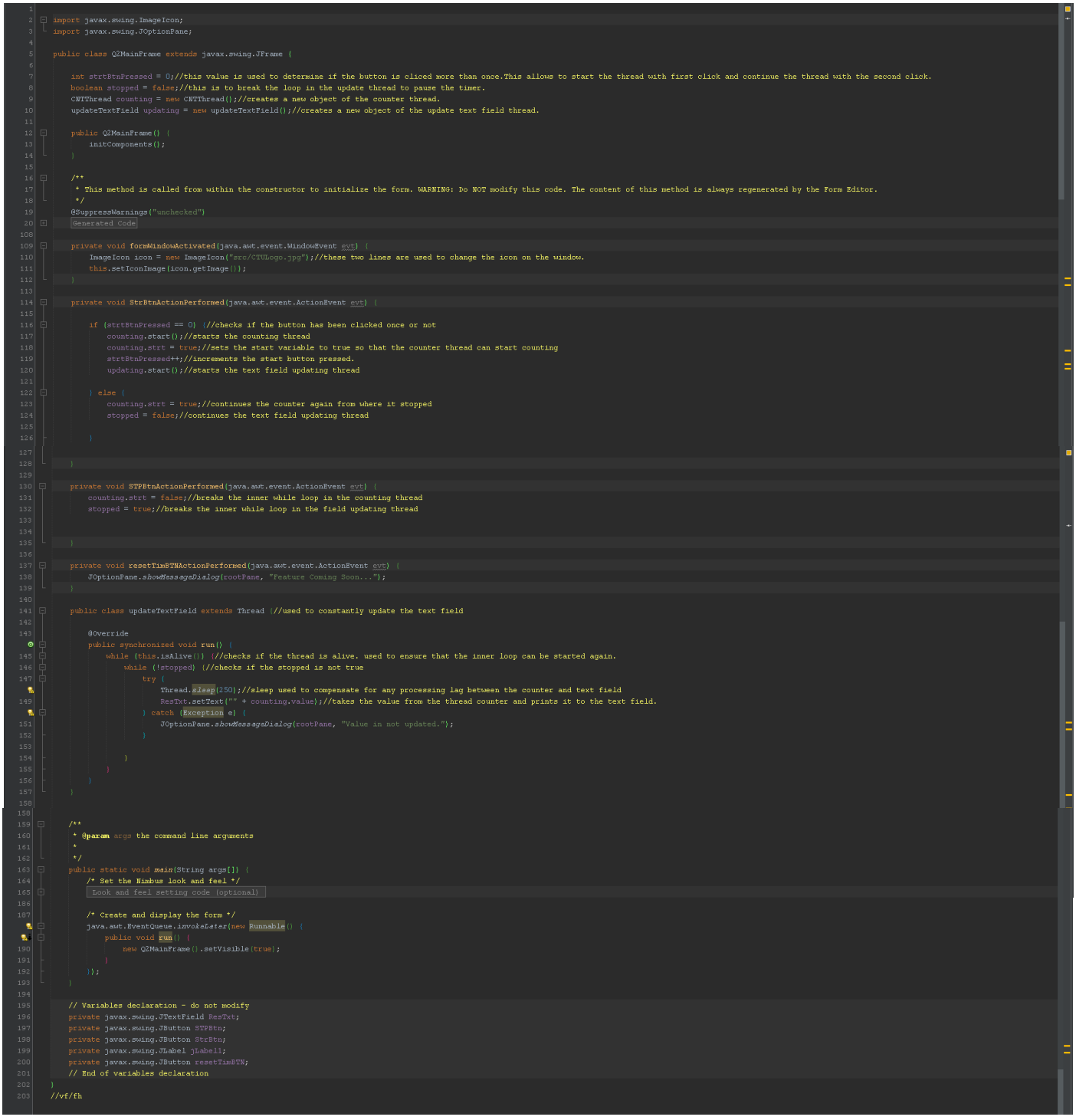
Application view on Opening

Error Message if user enters anything but a whole number



Application view once counting complete

Question 2 Main Code screenshots

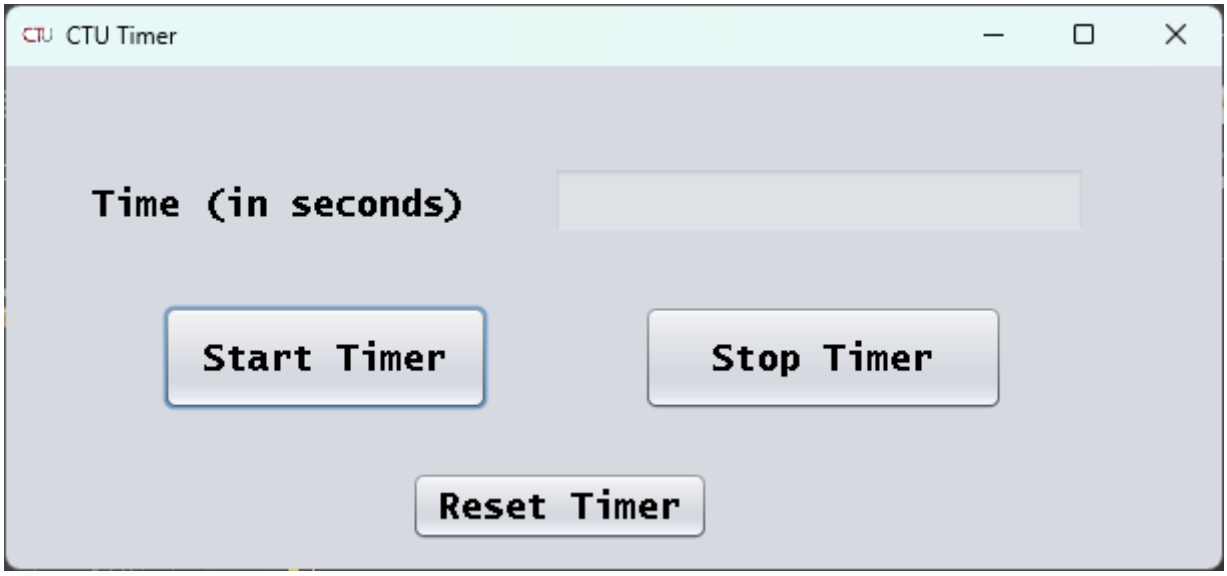


```
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;

public class Q2MainFrame extends javax.swing.JFrame {

    int strtBtnPressed = 0;//this value is used to determine if the button is cliced more than once.This allows to start the thread with first click and continue the thread with the second click.
    boolean stopped = false;//this is to break the loop in the update thread to pause the timer.
    CNTThread counting = new CNTThread();//creates a new object of the counter thread.
    updateTextField updating = new updateTextField();//creates a new object of the update text field thread.

    public Q2MainFrame() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form. WARNING: Do NOT modify this code. The content of this method is always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code

    private void formWindowActivated(java.awt.event.WindowEvent evt) {
        ImageIcon icon = new ImageIcon("src/CTULogo.jpg");//these two lines are used to change the icon on the window.
        this.setIconImage(icon.getImage());
    }

    private void StrBtnActionPerformed(java.awt.event.ActionEvent evt) {

        if (strtBtnPressed == 0) {//checks if the button has been clicked once or not
            counting.start();//starts the counting thread
            counting.strt = true;//sets the start variable to true so that the counter thread can start counting
            strtBtnPressed++;//increments the start button pressed.
            updating.start();//starts the text field updating thread

        } else {
            counting.strt = true;//continues the counter again from where it stopped
            stopped = false;//continues the text field updating thread

        }

    }

    private void STPBtnActionPerformed(java.awt.event.ActionEvent evt) {
        counting.strt = false;//breaks the inner while loop in the counting thread
        stopped = true;//breaks the inner while loop in the field updating thread


    }

    private void resetTimBTNActionPerformed(java.awt.event.ActionEvent evt) {
        JOptionPane.showMessageDialog(rootPane, "Feature Coming Soon...");
    }

    public class updateTextField extends Thread {//used to constantly update the text field

        @Override
        public synchronized void run() {
            while (this.isAlive()) {//checks if the thread is alive. used to ensure that the inner loop can be started again.
                while (!stopped) {//checks if the stopped is not true
                    try {
                        Thread.sleep(250);//sleep used to compensate for any processing lag between the counter and text field
                        ResTxt.setText("" + counting.value);//takes the value from the thread counter and prints it to the text field.
                    } catch (Exception e) {
                        JOptionPane.showMessageDialog(rootPane, "Value in not updated.");
                    }

                }
            }
        }
    }


    /**
     * @param args the command line arguments
     *
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        Look and feel setting code (optional)

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Q2MainFrame().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JTextField ResTxt;
    private javax.swing.JButton STPBtn;
    private javax.swing.JButton StrBtn;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JButton resetTimBTN;
    // End of variables declaration
}
//vf/fh
```

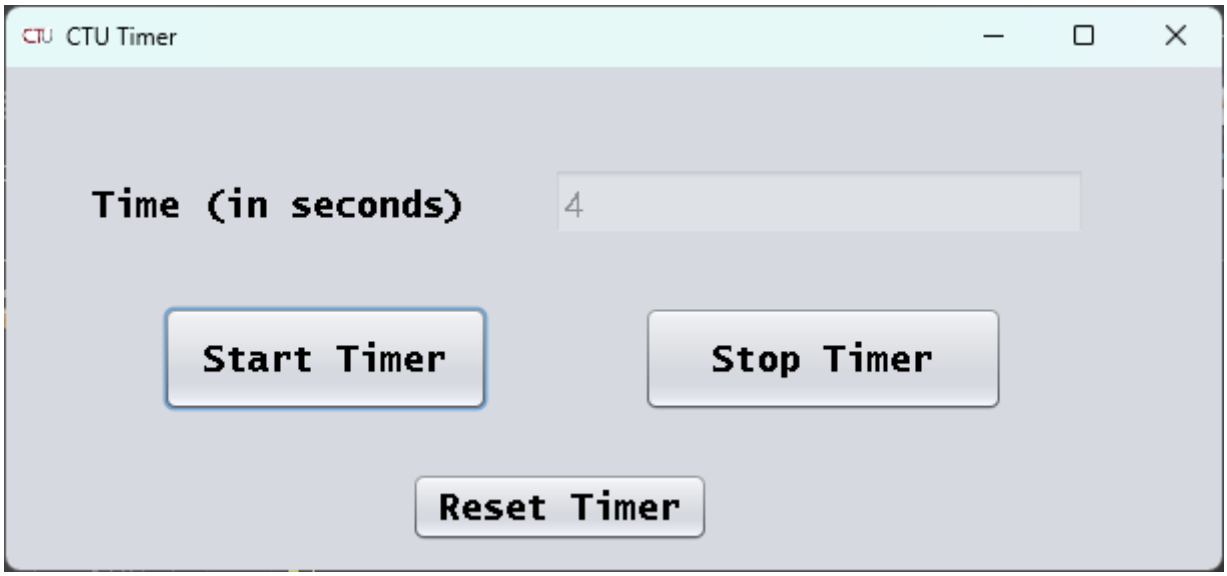
Question 2 Counter Thread Code screenshot

```java
import java.util.logging.Level;
import java.util.logging.Logger;

public class CNTThread extends Thread {

    int value;//value that will be passed back from the thread
    boolean strt = false;

    @Override
    public synchronized void run() {
        while (this.isAlive()) {//this while loop is to check if the thread is active and serves as the restart for the loop once stopped.
            while (strt) {//this while loop is to check if the loop should be started or not.
                try {
                    value++;//this increments the value which is retrieved by the update thread
                    this.sleep(1000);//this makes the counter wait for a second so that each increment is a second appart.

                } catch (InterruptedException ex) {
                    Logger.getLogger(CNTThread.class.getName()).log(Level.SEVERE, null, ex);
                }

            }
        }
    }
}
//vf/fh
```

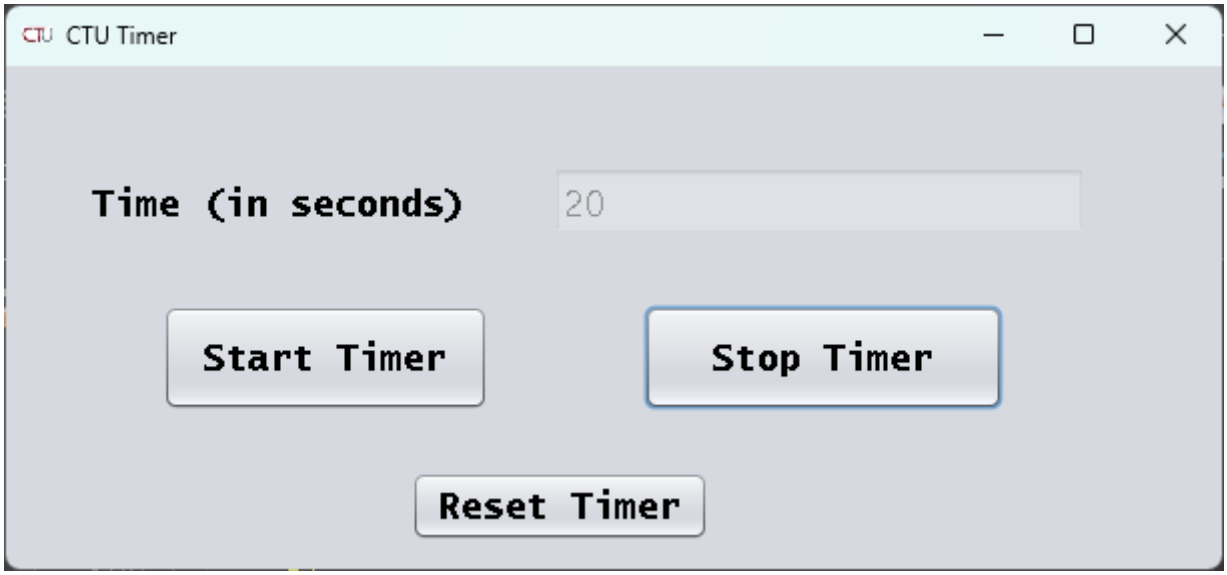
Question 2 Application screenshots



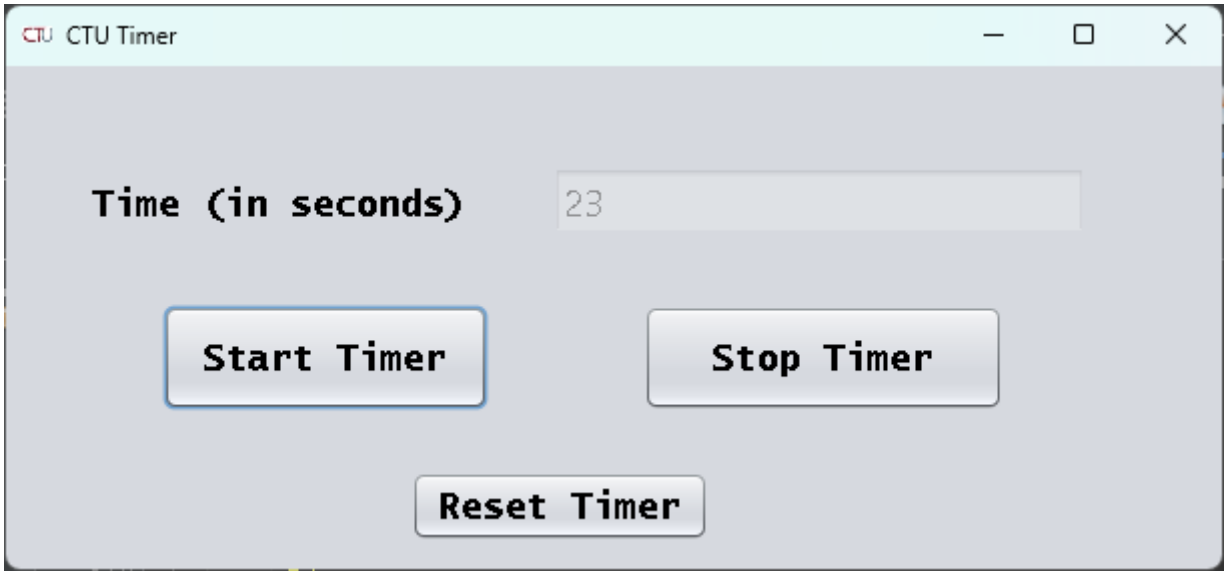
Application view on Opening



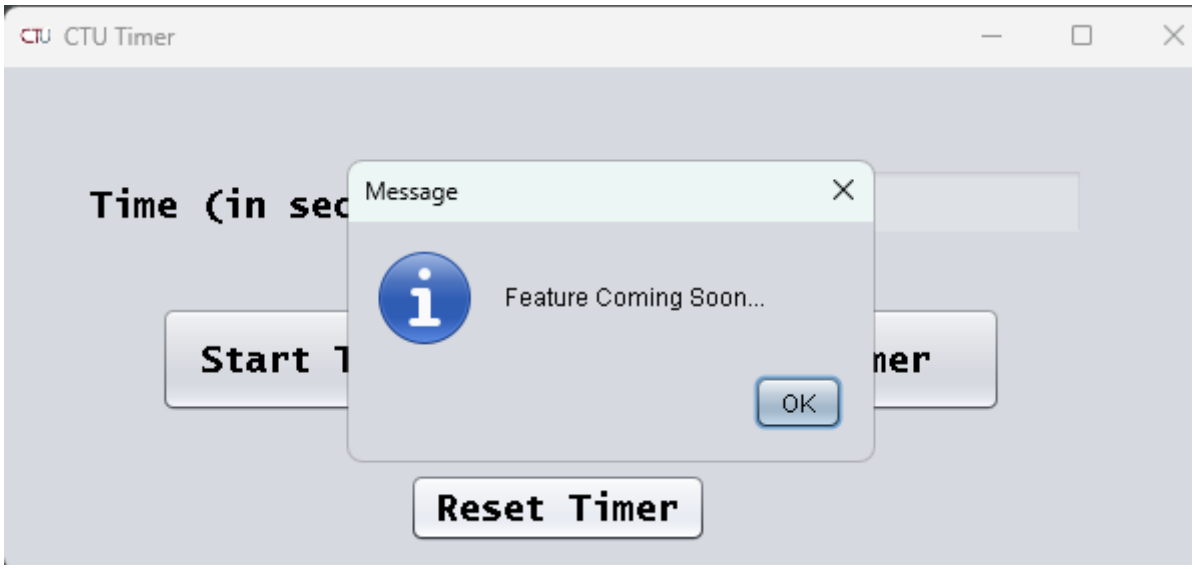
Application view on Staring the Timer



Application view on Stopping



Application view on Start Again



Reset feature Coming Soon