ADVANCED JAVA PROGRAMMING WITH GUI

CSE PATHSHALA

A training report

Submitted in partial fulfillment of the requirements for the award of degree of

Batchelor of Technology

Computer Science Engineering

Submitted to

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

From 06/10/25 to 07/28/25

SUBMITTED BY

Name of student: Ankit Kumar

Registration Number: 12318018

Signature of the student:

To whom so ever it may concern

I, Ankit Kumar ,12318018 , hereby declare that the work done by me on " ADVANCED JAVA PROGRAMMING WITH GUI" from June 2025 to July 2025, is a record of original work for the partial fulfillment of the requirements for the award of the degree

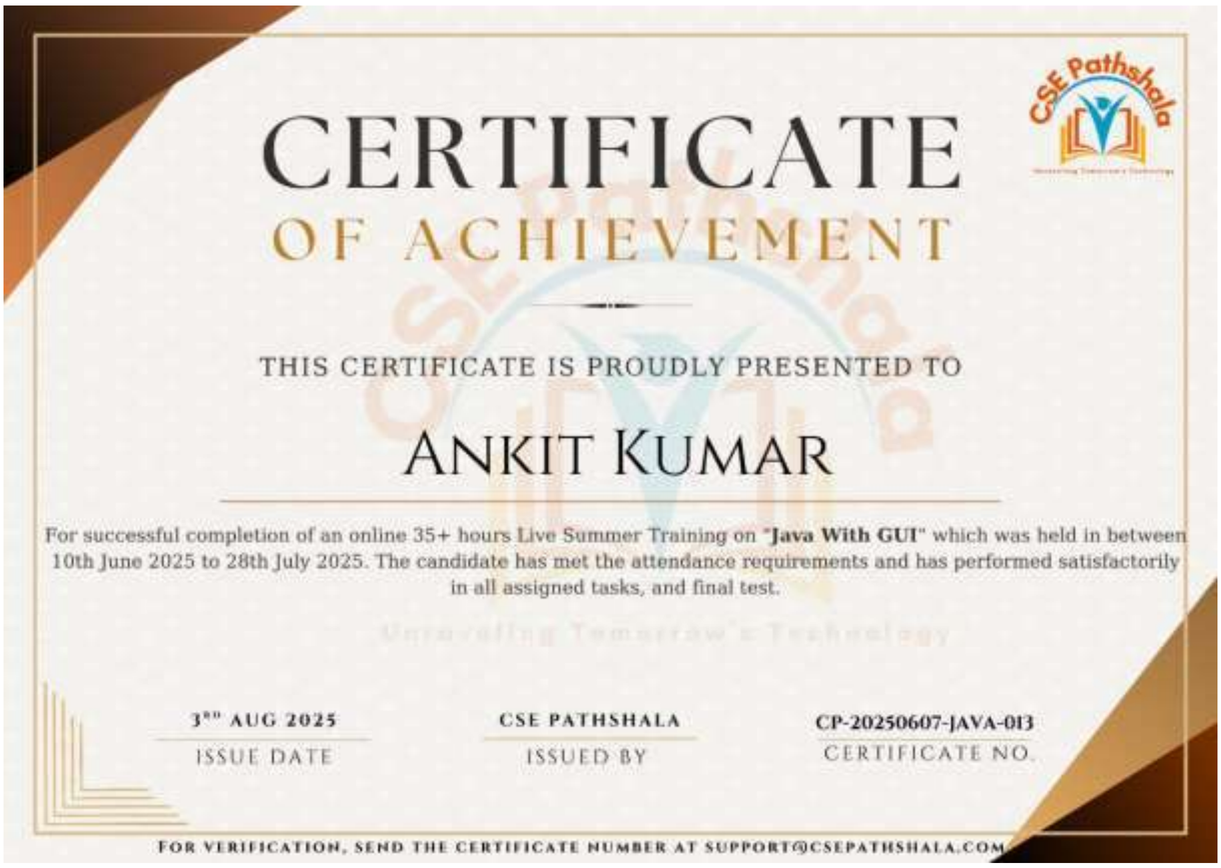
B. TECH CSE

Ankit Kumar (12318018)

Signature of the student

Dated: 15/8/25

CSE PATHSHALA CERTIFICATE:



CERTIFICATE

OF ACHIEVEMENT

THIS CERTIFICATE IS PROUDLY PRESENTED TO

ANKIT KUMAR

For successful completion of an online 35+ hours Live Summer Training on "**Java With GUI**" which was held in between 10th June 2025 to 28th July 2025. The candidate has met the attendance requirements and has performed satisfactorily in all assigned tasks, and final test.

3RD AUG 2025	CSE PATHSHALA	CP-20250607-JAVA-013
ISSUE DATE	ISSUED BY	CERTIFICATE NO.

FOR VERIFICATION, SEND THE CERTIFICATE NUMBER AT SUPPORT@CSEPATHSHALA.COM

Acknowledgement

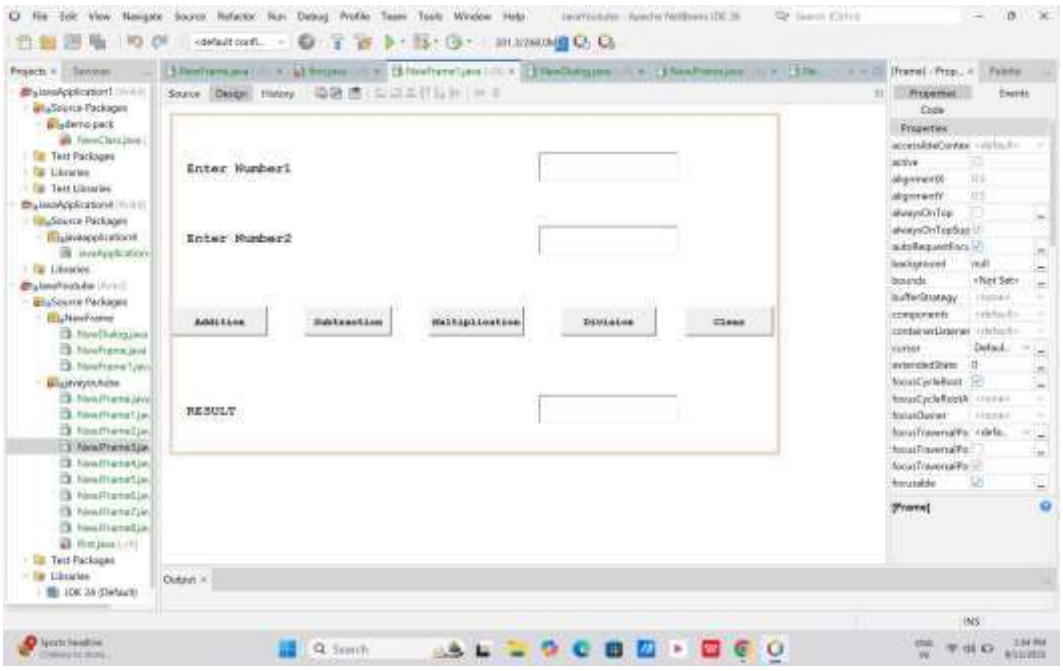
I would like to express my sincere gratitude to everyone who supported me throughout my Summer Internship Program.

I am deeply thankful to my mentors at **The CSE PATHSHALA**, for their invaluable guidance, encouragement, and insightful feedback. Their expertise was instrumental in helping me navigate the complexities of Java Programming and GUI.

Finally, I wish to thank my family and friends for their unwavering encouragement and support throughout this journey.

ANKIT KUMAR

List of figures : Arithmetic Operations GUI



Code:

```
package NewFrame;


/**
 *
 * @author ankit
 */
public class NewFrame1 extends java.awt.Frame {
```

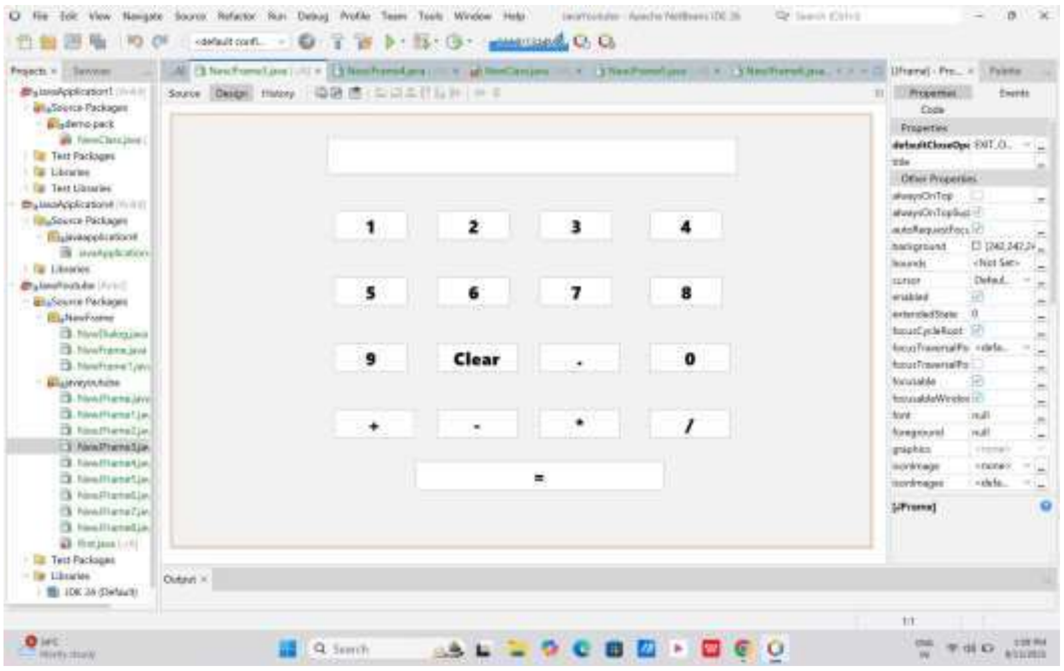
```
public NewFrame1() {
    initComponents();
}

// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    label1 = new java.awt.Label();
    label2 = new java.awt.Label();
    label3 = new java.awt.Label();
    textField1 = new java.awt.TextField();
    textField2 = new java.awt.TextField();
    textField3 = new java.awt.TextField();
    button1 = new java.awt.Button();
    button2 = new java.awt.Button();
    button3 = new java.awt.Button();
    button4 = new java.awt.Button();
    button5 = new java.awt.Button();

    addWindowListener(new java.awt.event.WindowAdapter() {
        public void windowClosing(java.awt.event.WindowEvent evt) {
            exitForm(evt);
        }
    });
```

```
setLayout(null);

label1.setFont(new java.awt.Font("DialogInput", 1, 18)); // NOI18N
label1.setText("Enter Number1");
add(label1);
label1.setBounds(20, 50, 200, 40);

label2.setFont(new java.awt.Font("DialogInput", 1, 18)); // NOI18N
label2.setText("Enter Number2");
add(label2);
label2.setBounds(20, 140, 190, 50);

label3.setFont(new java.awt.Font("DialogInput", 1, 18)); // NOI18N
label3.setText("RESULT");
add(label3);
label3.setBounds(20, 380, 200, 40);

textField1.setFont(new java.awt.Font("DialogInput", 1, 36)); // NOI18N
textField1.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      textField1ActionPerformed(evt);
   }
});
add(textField1);
```

```
textField1.setBounds(500, 50, 190, 40);

textField2.setFont(new java.awt.Font("DialogInput", 1, 36)); // NOI18N
add(textField2);
textField2.setBounds(500, 150, 190, 40);

textField3.setFont(new java.awt.Font("DialogInput", 1, 36)); // NOI18N
add(textField3);
textField3.setBounds(500, 380, 190, 40);

button1.setFont(new java.awt.Font("DialogInput", 1, 14)); // NOI18N
button1.setLabel("Addition");
button1.addActionListener(new java.awt.event.ActionListener() {
  public void actionPerformed(java.awt.event.ActionEvent evt) {
    button1ActionPerformed(evt);
  }
});
add(button1);
button1.setBounds(0, 260, 130, 40);

button2.setFont(new java.awt.Font("DialogInput", 1, 14)); // NOI18N
button2.setLabel("Subtraction");
button2.addActionListener(new java.awt.event.ActionListener() {
  public void actionPerformed(java.awt.event.ActionEvent evt) {
        button2ActionPerformed(evt);
    }
});
add(button2);
button2.setBounds(180, 260, 120, 40);

button3.setFont(new java.awt.Font("DialogInput", 1, 14)); // NOI18N
button3.setLabel("Multiplication");
button3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        button3ActionPerformed(evt);
    }
});
add(button3);
button3.setBounds(350, 260, 130, 40);

button4.setFont(new java.awt.Font("DialogInput", 1, 14)); // NOI18N
button4.setLabel("Division");
button4.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        button4ActionPerformed(evt);
    }
});
add(button4);
```

```
        button4.setBounds(530, 260, 130, 40);

        button5.setFont(new java.awt.Font("DialogInput", 1, 14)); // NOI18N
        button5.setLabel("Clear");
        button5.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button5ActionPerformed(evt);
            }
        });
        add(button5);
        button5.setBounds(700, 260, 120, 40);

        pack();
    }
    private void exitForm(java.awt.event.WindowEvent evt) {
        System.exit(0);
    }

    private void button1ActionPerformed(java.awt.event.ActionEvent evt) {
        textField3.setText(Integer.parseInt(textField1.getText()) +
Integer.parseInt(textField2.getText()) + " ");
    }

    private void button2ActionPerformed(java.awt.event.ActionEvent evt) {
        textField3.setText(Integer.parseInt(textField1.getText()) -
Integer.parseInt(textField2.getText()) + " ");
    }

    private void button3ActionPerformed(java.awt.event.ActionEvent evt) {
        textField3.setText(Integer.parseInt(textField1.getText()) *
Integer.parseInt(textField2.getText()) + " ");
    }

    private void button4ActionPerformed(java.awt.event.ActionEvent evt) {
        textField3.setText(Integer.parseInt(textField1.getText()) /
Integer.parseInt(textField2.getText()) + " ");
    }

    private void button5ActionPerformed(java.awt.event.ActionEvent evt) {
        textField1.setText("");
        textField2.setText("");
        textField3.setText("");
    }
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new NewFrame1().setVisible(true);
            }
        });
    }



    // Variables declaration - do not modify
    private java.awt.Button button1;
    private java.awt.Button button2;
    private java.awt.Button button3;
    private java.awt.Button button4;
    private java.awt.Button button5;
    private java.awt.Label label1;
    private java.awt.Label label2;
    private java.awt.Label label3;
    private java.awt.TextField textField1;
    private java.awt.TextField textField2;
    private java.awt.TextField textField3;
    // End of variables declaration
}
```

Figure: Calculator GUI



Code:

```
package javayoutube;


/**
 *
 * @author ankit
 */
public class NewJFrame3 extends javax.swing.JFrame {


    private static final java.util.logging.Logger logger =
java.util.logging.Logger.getLogger(NewJFrame3.class.getName(

    public NewJFrame3() {

        initComponents();
```

```
}

@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jTextField1 = new javax.swing.JTextField();
    jButton1 = new javax.swing.JButton();
    jButton2 = new javax.swing.JButton();
    jButton3 = new javax.swing.JButton();
    jButton4 = new javax.swing.JButton();
    jButton5 = new javax.swing.JButton();
    jButton6 = new javax.swing.JButton();
    jButton7 = new javax.swing.JButton();
    jButton8 = new javax.swing.JButton();
    jButton9 = new javax.swing.JButton();
    jButton10 = new javax.swing.JButton();
    jButton11 = new javax.swing.JButton();
    jButton12 = new javax.swing.JButton();
    jButton13 = new javax.swing.JButton();
    jButton14 = new javax.swing.JButton();
    jButton15 = new javax.swing.JButton();
```

```
        jButton16 = new javax.swing.JButton();

        jButton17 = new javax.swing.JButton();



setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        getContentPane().setLayout(null);


        jTextField1.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); //
NOI18N

        jTextField1.setHorizontalAlignment(javax.swing.JTextField.RIGHT);

        jTextField1.setCursor(new
java.awt.Cursor(java.awt.Cursor.TEXT_CURSOR));

        getContentPane().add(jTextField1);

        jTextField1.setBounds(210, 30, 560, 50);


        jButton1.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N

        jButton1.setText("1");

        jButton1.addActionListener(new java.awt.event.ActionListener() {

            public void actionPerformed(java.awt.event.ActionEvent evt) {

                Digit(evt);

            }

        });

        getContentPane().add(jButton1);
```

```
jButton1.setBounds(220, 130, 100, 40);

jButton2.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton2.setText("2");
jButton2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        Digit(evt);
    }
});
getContentPane().add(jButton2);
jButton2.setBounds(360, 130, 100, 40);

jButton3.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton3.setText("3");
jButton3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        Digit(evt);
    }
});
getContentPane().add(jButton3);
jButton3.setBounds(500, 130, 100, 40);
```

```
jButton4.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton4.setText("4");
jButton4.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        Digit(evt);
    }
});
getContentPane().add(jButton4);
jButton4.setBounds(650, 130, 100, 40);

jButton5.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton5.setText("5");
jButton5.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        Digit(evt);
    }
});
getContentPane().add(jButton5);
jButton5.setBounds(220, 220, 100, 40);

jButton6.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton6.setText("6");
```

```
jButton6.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      Digit(evt);
   }
});
getContentPane().add(jButton6);
jButton6.setBounds(360, 220, 100, 40);

jButton7.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton7.setText("7");
jButton7.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      Digit(evt);
   }
});
getContentPane().add(jButton7);
jButton7.setBounds(500, 220, 100, 40);

jButton8.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton8.setText("8");
jButton8.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
        Digit(evt);
    }
});
getContentPane().add(jButton8);
jButton8.setBounds(650, 220, 100, 40);

jButton9.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton9.setText("9");
jButton9.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        Digit(evt);
    }
});
getContentPane().add(jButton9);
jButton9.setBounds(220, 310, 100, 40);

jButton10.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton10.setText("Clear");
jButton10.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton10ActionPerformed(evt);
    }
```

```
});
getContentPane().add(jButton10);
jButton10.setBounds(360, 310, 110, 40);

jButton11.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton11.setText(".");
jButton11.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        Digit(evt);
    }
});
getContentPane().add(jButton11);
jButton11.setBounds(500, 310, 110, 40);

jButton12.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton12.setText("0");
jButton12.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        Digit(evt);
    }
});
getContentPane().add(jButton12);
```

```
jButton12.setBounds(650, 310, 110, 40);


jButton13.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton13.setText("+");
jButton13.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton13ActionPerformed(evt);
    }
});
getContentPane().add(jButton13);
jButton13.setBounds(220, 400, 110, 40);


jButton14.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton14.setText("-");
jButton14.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton14ActionPerformed(evt);
    }
});
getContentPane().add(jButton14);
jButton14.setBounds(360, 400, 110, 40);
```

```
jButton15.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton15.setText("*");
jButton15.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      jButton15ActionPerformed(evt);
   }
});
getContentPane().add(jButton15);
jButton15.setBounds(500, 400, 110, 40);

jButton16.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton16.setText("/");
jButton16.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      jButton16ActionPerformed(evt);
   }
});
getContentPane().add(jButton16);
jButton16.setBounds(650, 400, 110, 40);

jButton17.setFont(new java.awt.Font("Segoe UI Black", 1, 24)); // NOI18N
jButton17.setText("=");
```

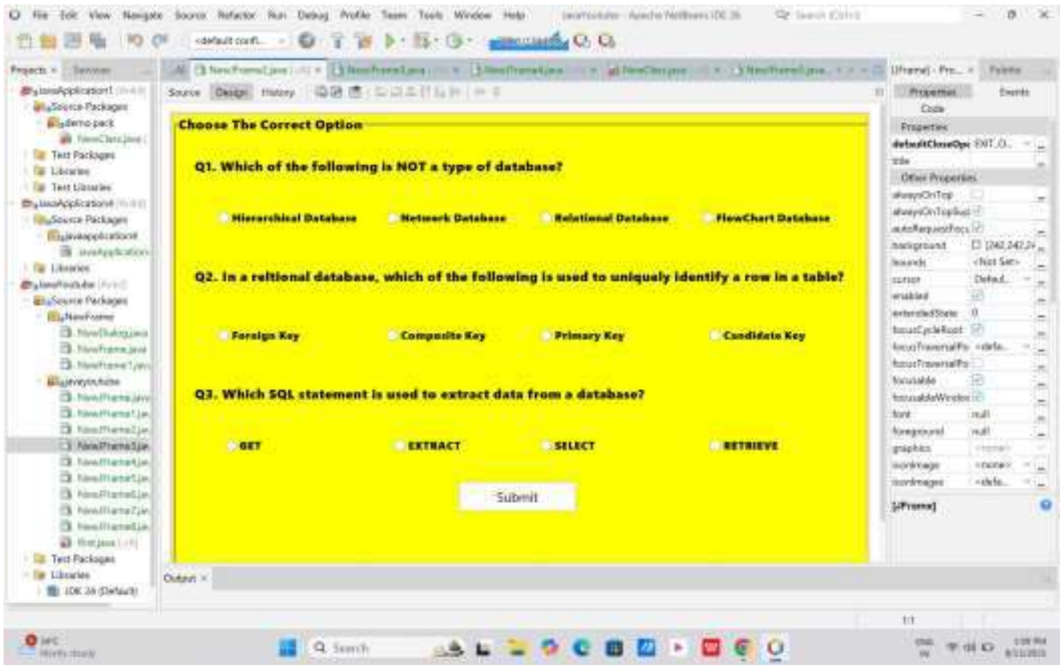
```
    jButton17.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton17ActionPerformed(evt);
        }
    });
    getContentPane().add(jButton17);
    jButton17.setBounds(330, 470, 340, 40);

    pack();
}// </editor-fold>

String op = "";
double no1, no2, ans;
private void Digit(java.awt.event.ActionEvent evt) {
    jTextField1.setText(jTextField1.getText() + evt.getActionCommand());
}

private void jButton13ActionPerformed(java.awt.event.ActionEvent evt) {
    op = "+";
    no1 = Double.parseDouble(jTextField1.getText());
    jTextField1.setText("");
}
```

```
private void jButton14ActionPerformed(java.awt.event.ActionEvent evt) {
   op = "-";
   no1 = Double.parseDouble(jTextField1.getText());
   jTextField1.setText("");
}

private void jButton15ActionPerformed(java.awt.event.ActionEvent evt) {
   op = "*";
   no1 = Double.parseDouble(jTextField1.getText());
   jTextField1.setText("");
}

private void jButton16ActionPerformed(java.awt.event.ActionEvent evt) {
   op = "/";
   no1 = Double.parseDouble(jTextField1.getText());
   jTextField1.setText("");
}

private void jButton17ActionPerformed(java.awt.event.ActionEvent evt) {
   no2 = Double.parseDouble(jTextField1.getText());
```

```
    if (op.equals("+")) {

        ans = no1 + no2;

    }


    else if (op.equals("-")) {

        ans = no1 - no2;

    }


    else if (op.equals("*")) {

        ans = no1 * no2;

    }


    else if (op.equals("/")) {

        ans = no1 / no2;

    }


    jTextField1.setText(""+ans);

}


private void jButton10ActionPerformed(java.awt.event.ActionEvent evt) {

    jTextField1.setText("");

}
```

```
   /**
    * @param args the command line arguments
    */
   public static void main(String args[]) {
       /* Set the Nimbus look and feel */
       //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
       /* If Nimbus (introduced in Java SE 6) is not available, stay with the default
look and feel.
        * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
        */
       try {
           for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
               if ("Nimbus".equals(info.getName())) {
                   javax.swing.UIManager.setLookAndFeel(info.getClassName());
                   break;
               }
           }
       } catch (ReflectiveOperationException |
javax.swing.UnsupportedLookAndFeelException ex) {
           logger.log(java.util.logging.Level.SEVERE, null, ex);
       }

       //</editor-fold>


       /* Create and display the form */

       java.awt.EventQueue.invokeLater(() -> new
NewJFrame3().setVisible(true));

    }


   // Variables declaration - do not modify

   private javax.swing.JButton jButton1;

   private javax.swing.JButton jButton10;

   private javax.swing.JButton jButton11;

   private javax.swing.JButton jButton12;

   private javax.swing.JButton jButton13;

   private javax.swing.JButton jButton14;

   private javax.swing.JButton jButton15;

   private javax.swing.JButton jButton16;

   private javax.swing.JButton jButton17;

   private javax.swing.JButton jButton2;

   private javax.swing.JButton jButton3;

   private javax.swing.JButton jButton4;

   private javax.swing.JButton jButton5;
   private javax.swing.JButton jButton6;

   private javax.swing.JButton jButton7;

   private javax.swing.JButton jButton8;

   private javax.swing.JButton jButton9;

   private javax.swing.JTextField jTextField1;

   // End of variables declaration

}
```

Figure: Quiz Application GUI



Choose The Correct Option

Q1. Which of the following is NOT a type of database?

Hierarchical Database   Network Database   Relational Database   FlowChart Database

Q2. In a reltional database, which of the following is used to uniquely identify a row in a table?

Foreign Key   Composite Key   Primary Key   Candidate Key

Q3. Which SQL statement is used to extract data from a database?

GET   EXTRACT   SELECT   RETRIEVE

Submit

Code:

```
package javayoutube;


/**
```

```
 *
 * @author ankit
 */
public class NewJFrame2 extends javax.swing.JFrame {


    private static final java.util.logging.Logger logger =
java.util.logging.Logger.getLogger(NewJFrame2.class.getName());



    public NewJFrame2() {
        initComponents();
    }
    @SuppressWarnings("unchecked")

    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        buttonGroup1 = new javax.swing.ButtonGroup();
        buttonGroup2 = new javax.swing.ButtonGroup();
        buttonGroup3 = new javax.swing.ButtonGroup();
        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jRadioButton1 = new javax.swing.JRadioButton();
        jRadioButton2 = new javax.swing.JRadioButton();
        jRadioButton3 = new javax.swing.JRadioButton();
        jRadioButton4 = new javax.swing.JRadioButton();
        jLabel2 = new javax.swing.JLabel();
        jRadioButton5 = new javax.swing.JRadioButton();
        jRadioButton6 = new javax.swing.JRadioButton();
        jRadioButton7 = new javax.swing.JRadioButton();
        jRadioButton8 = new javax.swing.JRadioButton();
        jLabel3 = new javax.swing.JLabel();
        jButton1 = new javax.swing.JButton();
        jLabel4 = new javax.swing.JLabel();
        jRadioButton9 = new javax.swing.JRadioButton();
        jRadioButton10 = new javax.swing.JRadioButton();
        jRadioButton11 = new javax.swing.JRadioButton();
        jRadioButton12 = new javax.swing.JRadioButton();
        jLabel5 = new javax.swing.JLabel();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);


        jPanel1.setBackground(new java.awt.Color(255, 255, 0));
        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.MatteBorder(null), "Choose The Correct Option",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new
java.awt.Font("Segoe UI Black", 1, 18))); // NOI18N
        jPanel1.setLayout(null);
        jLabel1.setFont(new java.awt.Font("Segoe UI Black", 1, 18)); // NOI18N
        jLabel1.setText("Q1. Which of the following is NOT a type of database?");
        jPanel1.add(jLabel1);
        jLabel1.setBounds(30, 50, 510, 40);

        buttonGroup1.add(jRadioButton1);
        jRadioButton1.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N
        jRadioButton1.setText("Hierarchical Database");
        jPanel1.add(jRadioButton1);
        jRadioButton1.setBounds(60, 120, 200, 40);

        buttonGroup1.add(jRadioButton2);
        jRadioButton2.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N
        jRadioButton2.setText("Network Database");
        jPanel1.add(jRadioButton2);
        jRadioButton2.setBounds(290, 120, 170, 40);

        buttonGroup1.add(jRadioButton3);
        jRadioButton3.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N
        jRadioButton3.setText("Relational Database");
        jPanel1.add(jRadioButton3);
        jRadioButton3.setBounds(500, 120, 180, 40);


        buttonGroup1.add(jRadioButton4);
        jRadioButton4.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N
        jRadioButton4.setText("FlowChart Database");
        jPanel1.add(jRadioButton4);
        jRadioButton4.setBounds(720, 120, 180, 40);


        jLabel2.setFont(new java.awt.Font("Segoe UI Black", 1, 18)); // NOI18N
        jLabel2.setText("Q2. In a reltional database, which of the following is used
to uniquely identify a row in a table?");
        jPanel1.add(jLabel2);
        jLabel2.setBounds(30, 200, 900, 40);


        buttonGroup2.add(jRadioButton5);
        jRadioButton5.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N
        jRadioButton5.setText("Foreign Key");
        jPanel1.add(jRadioButton5);
        jRadioButton5.setBounds(60, 280, 120, 40);


        buttonGroup2.add(jRadioButton6);
        jRadioButton6.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N
        jRadioButton6.setText("Composite Key");
        jPanel1.add(jRadioButton6);

        jRadioButton6.setBounds(290, 280, 140, 40);


        buttonGroup2.add(jRadioButton7);

        jRadioButton7.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N

        jRadioButton7.setText("Primary Key");

        jPanel1.add(jRadioButton7);

        jRadioButton7.setBounds(500, 280, 120, 40);


        buttonGroup2.add(jRadioButton8);

        jRadioButton8.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N

        jRadioButton8.setText("Candidate Key");

        jPanel1.add(jRadioButton8);

        jRadioButton8.setBounds(730, 280, 140, 40);


        jLabel3.setFont(new java.awt.Font("Segoe UI Black", 1, 18)); // NOI18N

        jLabel3.setText("Q3. Which SQL statement is used to extract data from a
database?");

        jPanel1.add(jLabel3);

        jLabel3.setBounds(30, 360, 630, 40);


        jButton1.setFont(new java.awt.Font("Segoe UI", 0, 18)); // NOI18N

        jButton1.setText("Submit");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });
        jPanel1.add(jButton1);
        jButton1.setBounds(390, 500, 160, 40);

        jLabel4.setFont(new java.awt.Font("Segoe UI Black", 1, 18)); // NOI18N
        jPanel1.add(jLabel4);
        jLabel4.setBounds(70, 580, 410, 40);

        buttonGroup3.add(jRadioButton9);
        jRadioButton9.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N
        jRadioButton9.setText("GET");
        jPanel1.add(jRadioButton9);
        jRadioButton9.setBounds(70, 430, 80, 40);

        buttonGroup3.add(jRadioButton10);
        jRadioButton10.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); //
NOI18N
        jRadioButton10.setText("EXTRACT");
        jPanel1.add(jRadioButton10);
        jRadioButton10.setBounds(300, 430, 100, 40);
        buttonGroup3.add(jRadioButton11);
        jRadioButton11.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); // 
NOI18N
        jRadioButton11.setText("SELECT");
        jPanel1.add(jRadioButton11);
        jRadioButton11.setBounds(500, 430, 90, 40);

        buttonGroup3.add(jRadioButton12);
        jRadioButton12.setFont(new java.awt.Font("Segoe UI Black", 1, 15)); // 
NOI18N
        jRadioButton12.setText("RETRIEVE");
        jRadioButton12.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jRadioButton12ActionPerformed(evt);
            }
        });
        jPanel1.add(jRadioButton12);
        jRadioButton12.setBounds(730, 430, 110, 40);

        jLabel5.setFont(new java.awt.Font("Segoe UI Black", 1, 18)); // NOI18N
        jPanel1.add(jLabel5);
        jLabel5.setBounds(690, 20, 250, 40);

        getContentPane().add(jPanel1);
        jPanel1.setBounds(0, 0, 950, 650);


        pack();
    }// </editor-fold>


    private void jRadioButton12ActionPerformed(java.awt.event.ActionEvent 
evt) {


    }


    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {


        int marks = 0;


        if(jRadioButton4.isSelected()) {
            marks++;
        }
        if(jRadioButton7.isSelected()) {
            marks++;
        }
        if(jRadioButton11.isSelected()) {
            marks++;
        }
        jLabel4.setText("Total Marks Obtained : " + marks);
        jLabel5.setText("Test Completed");
        jButton1.setEnabled(false);


    }


    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {


        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code 
(optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default 
look and feel.
         * For details see 
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : 
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ReflectiveOperationException |
javax.swing.UnsupportedLookAndFeelException ex) {
            logger.log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(() -> new
NewJFrame2().setVisible(true));
    }

    // Variables declaration - do not modify
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.ButtonGroup buttonGroup2;
    private javax.swing.ButtonGroup buttonGroup3;
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JRadioButton jRadioButton1;
    private javax.swing.JRadioButton jRadioButton10;
    private javax.swing.JRadioButton jRadioButton11;
    private javax.swing.JRadioButton jRadioButton12;
    private javax.swing.JRadioButton jRadioButton2;
    private javax.swing.JRadioButton jRadioButton3;
    private javax.swing.JRadioButton jRadioButton4;
    private javax.swing.JRadioButton jRadioButton5;
    private javax.swing.JRadioButton jRadioButton6;
    private javax.swing.JRadioButton jRadioButton7;
    private javax.swing.JRadioButton jRadioButton8;
    private javax.swing.JRadioButton  jRadioButton9;
    // End of variables declaration
}
```

•

List of Abbreviations

Abbreviation	**Full Form**
GUI	Graphical User Interface
IDE	Integrated Development Environment
JDK	Java Development Kit
JVM	Java Virtual Machine
API	Application Programming Interface
AWT	Abstract Window Toolkit
OOP	Object-Oriented Programming
SQL	Structured Query Language
DBMS	Database Management System
JDBC	Java Database Connectivi

Brief Description of the Project

This internship report presents the work carried out during the **Advanced Java Programming with GUI** training at **CSE Pathshala**. The focus of the internship was to gain hands-on experience in building interactive desktop applications using **Java AWT** and **Swing** frameworks. The training involved developing three core projects: an **Arithmetic Operations GUI**, a **Calculator GUI**, and a **Quiz Application GUI**. These projects demonstrated the use of graphical components, layout management, and event-driven programming to create user-friendly and functional interfaces. Through this internship, I acquired practical skills in GUI design, Java programming, and application development, preparing me for real-world software development tasks.

- **Project 1: Arithmetic Operations GUI (Individual Project):**

This application was a hands-on exercise in implementing basic arithmetic functionality within a graphical user interface. Developed using Java AWT, the program allows users to perform addition, subtraction, multiplication, and division by entering two numbers and selecting the desired operation. The project focused on understanding AWT components such as Frame, Label, TextField, and Button, along with event handling through the ActionListener interface. This project strengthened my skills in building simple, interactive desktop applications while reinforcing the principles.

- **Project 2: Calculator GUI (Individual Project):**

This project was a practical implementation of a fully functional calculator using **Java Swing**. It allowed users to perform basic arithmetic operations such as **addition, subtraction, multiplication, and division** through an intuitive interface with digit buttons, operation buttons, and a display panel. The application leveraged `JFrame`, `JButton`, and `JTextField` components, along with **lambda expressions** for concise event handling. This project enhanced my understanding of **Swing layout management**, component styling, and event-driven programming, while focusing on creating a responsive and user-friendly GUI.

- **Project 3: Quiz Application GUI (Individual Project):**

  This project involved designing and developing a **multiple-choice quiz application** using **Java Swing**. The program presents a series of questions with four possible answers, allows users to select an option, and calculates the final score at the end. Key components used include JFrame, JLabel, JRadioButton, ButtonGroup, and JButton. The project emphasized implementing **event handling** to manage answer selection, navigate between questions, and display results. This application improved my skills in **state management**, **user interaction design**, and handling dynamic data within a GUI environment.

- **Position of Internship and Roles**

- During my internship in **Advanced Java Programming with GUI** at **CSE Pathshala**, I was engaged in end-to-end application development, from conceptualizing the project idea to delivering a fully functional desktop application. My work primarily revolved around designing intuitive interfaces and implementing the underlying logic using **Java AWT** and **Swing**.

- **Key Roles:**

- **Java Developer:**
  I was responsible for developing the functional core of each application. This included creating arithmetic operations calculators, a fully functional Swing-based calculator, and an interactive quiz application. I worked extensively with **Java classes**, **interfaces**, and **AWT/Swing components**, implementing features such as input handling, result computation, and dynamic updates.

- **UI Designer:**
  I designed interfaces that were not only functional but also visually balanced. By working with different **Layout Managers** (FlowLayout, BorderLayout, GridLayout), I ensured that each component — buttons, labels, text fields — was positioned in an organized and user-friendly manner.

- **Event Handler:**
  Implementing **event-driven programming** was central to my role. I used ActionListener, ItemListener, and ActionCommand to respond to user interactions such as button clicks, text field inputs, and option selections. For example, in the quiz application, the system had to detect which answer was selected, verify correctness, and track the score in real time.

- **Project Manager (Individual Projects):**
  Since the projects were individual, I also handled planning. I created a structured coding plan, dividing the project into UI creation, logic implementation, validation, and testing. This approach helped in avoiding last-minute errors and maintaining clear, reusable code.

- **Activities/Equipment Handled**

- While the internship was software-focused and did not require specialized hardware, I utilized a range of **software tools and programming environments**:
- **IDEs:** NetBeans IDE and Eclipse IDE for coding, debugging, and running Java applications.
- **Programming Language:** Java SE for core logic and GUI development.
- **Frameworks/Libraries:** Java AWT and Swing for GUI components and event handling.
- **Version Control:** Git for tracking code changes and creating version backups.
- **JDK:** Java Development Kit (JDK 17) to compile and run applications.
- **Operating System:** Windows 10 for the entire development and testing cycle.
- These tools collectively streamlined the development process, making coding, debugging, and UI designing efficient.

- **Challenges Faced and How Those Were Tackled**

- **Challenge 1 – Component Alignment and Layout Management**

- **Problem:** Early in development, my interfaces suffered from poor alignment — buttons would shift positions when resizing the window, and components sometimes overlapped.
- **Solution:** I studied Java's **Layout Managers** in depth. For instance, in the calculator, I used GridLayout to ensure buttons maintained uniform size and position. For forms, I used FlowLayout for a cleaner, flowing arrangement. I also learned to **nest panels** with different layouts to achieve complex, well-organized designs.

- **Challenge 2 – Event Handling for Multiple Components**
- **Problem:** Initially, each button had a separate ActionListener, resulting in repetitive code that was hard to maintain.
- **Solution:** I implemented **ActionCommands** and assigned each button a unique command string. A single ActionListener was then able to handle all actions based on the command received, significantly reducing code duplication.

- **Challenge 3 – Input Validation**
- **Problem:** Users could enter invalid data (letters instead of numbers) which caused runtime errors.
- **Solution:** Added try-catch blocks to handle exceptions like NumberFormatException, showing error dialogs using JOptionPane to guide users towards correct inputs.

- **Challenge 4 – Application Responsiveness**
- **Problem:** In the quiz application, loading a large set of questions caused the UI to freeze temporarily.
- **Solution:** I used **multithreading** with SwingWorker to load data in the background while keeping the UI responsive. This was my first practical experience with concurrency in GUI applications.

- **Learning Outcomes**

- This internship transformed theoretical Java knowledge into real-world development skills. My key takeaways were:
- **GUI Development Skills:** I became proficient in building desktop applications using Java AWT and Swing, designing clean, responsive interfaces.
- **Event-Driven Programming:** I learned to manage user interactions systematically, improving maintainability and performance.
- **Layout Management Expertise:** I mastered combining layout managers to create professional, adaptable designs.
- **Error Handling and Validation:** I developed robust validation mechanisms to handle incorrect or unexpected user inputs gracefully.
- **Independent Project Execution:** I learned to plan, code, test, and refine applications entirely on my own, simulating real-world work environments.

- **Data Analysis**

- Although the internship wasn’t about statistical analysis, I worked extensively with **data modeling, processing, and validation** in each project:
- **Data Modeling:** In the arithmetic and calculator applications, the primary data consisted of user-entered numbers. In the quiz application, I modeled questions and answers in arrays and lists, assigning indexes to track correct responses.
- **Data Processing:** I implemented logic to process inputs in real time — performing calculations in the calculator and score evaluations in the quiz application.
- **Validation and Feedback Loops:** I analyzed how users might misuse input fields (e.g., entering special characters) and implemented validation logic to prevent errors and guide correct usage.
- This approach ensured that each application handled data efficiently, with minimal errors and maximum usability

INTRODUCTION OF THE PROJECT UNDERTAKEN

1.1 Overview of the Summer Training Program

This report details the work and learning accomplished during the **Summer Internship Program on "Advanced Java Programming with GUI"**, conducted by **CSE Pathshala**.

- The program, conducted from **June 10, 2025, to July 28, 2025**, was an intensive, hands-on training designed to provide **practical experience in creating interactive desktop applications** using **Java Standard Edition (Java SE)**, specifically the **Abstract Window Toolkit (AWT)** and **Swing** libraries.

- The course followed a **project-based learning approach** where each project progressively built upon the skills learned in the previous one. The training was organized around **three core projects**:

  - **Arithmetic Operations Application** – introducing GUI basics, event handling, and layout managers.

  - **Calculator Application** – focusing on intermediate design, multi-button event handling, and input validation.

  - **Quiz Application** – integrating multiple components, user input tracking, and scoring logic.

- Through these projects, I gained **end-to-end development experience**, from **planning the UI layout** to **implementing backend logic** and **testing application performance**.

- **1.2 Objectives of the Work Undertaken**

- The primary objectives of this internship were:

- To gain hands-on knowledge of **Java GUI application development**.
- To **design and implement** interactive and user-friendly interfaces using AWT and Swing.
- To **understand and apply layout managers** for proper component arrangement.
- To develop **event-driven applications** using listeners such as ActionListener and ItemListener.
- To implement **input validation** and **error handling** using try-catch blocks and JOptionPane.
- To explore **code modularization** for better maintainability.

- **1.3 Scope and Applicability**
- The skills learned during this internship have **broad applicability** in the field of desktop application development.
  - **Educational Software** – Developing applications like quizzes, interactive exercises, and learning tools.
  - **Business Tools** – Building standalone applications for calculations, record-keeping, and reporting.
  - **Productivity Applications** – Creating utilities like task planners, calculators, and converters.
- The techniques mastered here are **fundamental to Java development** and form a strong foundation for **transitioning into advanced topics** like database integration (Java JDBC), network programming, or mobile app development with Java-based frameworks.

- **1.4 Introduction to the Organization: CSE Pathshala**

- CSE Pathshala is a dedicated IT training institute specializing in **practical programming education** for aspiring software developers. The organization focuses on **bridging the gap between academic theory and real-world application** by delivering hands-on, project-oriented training.

- Its **Advanced Java Programming with GUI** program is designed by industry experts, emphasizing **clean code practices, modular design, and interactive application development** to ensure participants can directly apply their skills in professional settings.

- **Chapter-2: TECHNICAL FOUNDATIONS**

- **2.1 Technology Stack Overview**

- All applications developed during this internship were built with **Java SE** using the following core technologies:

  • **Java AWT** – For creating basic GUI components and handling low-level events.

  • **Java Swing** – For building rich, customizable, and platform-independent interfaces.

  • **Event Listeners** – Implemented ActionListener, ItemListener, and WindowListener to capture and respond to user actions.

  • **Layout Managers** – FlowLayout, GridLayout, and BorderLayout for flexible and responsive component arrangement.

- **2.2 Development Environment**

  • **IDE:** NetBeans 17 and Eclipse IDE for Java Developers.

  • **JDK:** Java Development Kit 17 for compiling and running applications.

- **Operating System:** Windows 10, ensuring compatibility with common user environments.
- **Version Control:** Git for tracking project iterations.

- **2.3 Key Development Tools and Practices**
  - **Error Handling:** Implemented structured exception handling using try-catch blocks.
  - **User Feedback:** Integrated JOptionPane dialogs for alerts, confirmations, and error messages.
  - **Reusable Code:** Encapsulated repetitive logic into reusable methods to promote modularity.

- **Chapter-3: PROJECT 1 – Arithmetic Operations GUI (Individual Project)**
- **3.1 Project Overview**
- This introductory project focused on building a **two-number arithmetic calculator** that can perform addition, subtraction, multiplication, and division.
- **Learning Goals:**
  - Understanding **JFrame** as the main application window.
  - Arranging GUI elements logically.
  - Handling **button click events** to perform calculations.

- **3.2 Implementation Details**
  - **UI Components:**

- Two JTextField inputs for numbers.
- Four JButton components for each arithmetic operation.
- A JLabel to display the result.
- **Event Handling:**
- Attached an ActionListener to each button.
- Retrieved input values, parsed them to numeric types, performed the calculation, and displayed results.
- **Validation:**
- Added try-catch to handle NumberFormatException.
- Displayed error dialogs if input was invalid.

- **Chapter-4: PROJECT 2 – Calculator Application**
- **4.1 Project Overview**
- The second project expanded on the first by designing a **fully functional calculator** with **GridLayout** and enhanced input handling.
- **Key Additions:**
  - Numeric keypad (0–9) buttons.
  - Operator buttons (+, −, ×, ÷).
  - A clear (C) button to reset fields.
-

- **4.2 Implementation Highlights**
  - **Single Event Handler:** Used ActionCommand values to manage all button clicks from one ActionListener.

- **Edge Case Handling:** Prevented division by zero with error messages.
- **UI Layout:** Used nested JPanels to separate display and keypad areas for cleaner design.

Chapter-5: PROJECT 3 – Quiz Application GUI

5.1 Project Overview

The final project was an **interactive quiz application** that displayed multiple-choice questions, allowed navigation, and calculated the final score.

5.2 Implementation Details

- **Question Management:** Stored questions and options in arrays.
- **UI Components:**
- JLabel for question text.
- JRadioButton options grouped in ButtonGroup.
- Next and Submit buttons.
- **Scoring Logic:** Increased score counter for correct answers.
- **Result Display:** Used JOptionPane to display the final score after submission.

Final Chapter: CONCLUSION AND FUTURE PERSPECTIVE

This Summer Internship in Advanced Java Programming with GUI has been a pivotal experience, bridging the gap between theoretical academic concepts and their practical application in real-world software development. The program's project-driven structure allowed me to build multiple interactive desktop applications using Java AWT and Swing, while deepening my understanding of event-driven programming, layout management, and user interface design. By completing three distinct projects — an *Arithmetic Operations App*, a *Calculator GUI*, and a *Quiz Application* — I was able to experience the complete software development cycle, from planning and design to coding, debugging, and testing.

The internship successfully achieved its objectives. The Arithmetic Operations App reinforced the fundamentals of GUI component creation, event handling, and validation. The Calculator GUI pushed me to manage more complex logic, optimize layout designs, and handle exceptions gracefully. The Quiz Application introduced concepts like state management, data storage in arrays, and enforcing controlled user interaction through radio buttons and validation prompts. These projects collectively strengthened my ability to design intuitive, responsive, and reliable applications.

This experience also sharpened my problem-solving skills. Challenges such as managing layout consistency across different resolutions, handling invalid user inputs, and avoiding code duplication through modular design taught me to think systematically and apply best practices in software development. Additionally, the internship cultivated my ability to write clean, maintainable code and to present technical work in a professional format.

Future Scope of the Projects

The applications developed during this internship serve as functional prototypes that can be further enhanced with advanced features:

- Arithmetic Operations App: Can be extended to support complex mathematical operations, history tracking, and integration with Java's Math library for scientific functions.

- Calculator GUI: Could evolve into a scientific or financial calculator with additional operations, memory functions, and an expression parser for multi-step calculations.
- Quiz Application: Could include a database backend using JDBC to store questions, track scores across sessions, implement timers, and provide randomized question sets for replayability.

Personal and Professional Growth

This internship has significantly shaped my professional outlook. It has solidified my interest in Java-based desktop application development and inspired me to explore integrating GUI-based applications with backend databases, APIs, and potentially even AI-based features in the future. I am now confident in my ability to independently design and deliver functional applications, a skill set that will be invaluable in my final academic projects and professional career.

The technical expertise gained — from mastering Java's GUI frameworks to managing event-driven architectures — has prepared me to handle more advanced software engineering challenges. Furthermore, the experience has improved my ability to plan projects, troubleshoot efficiently, and present solutions effectively.

In conclusion, this internship has been a journey of growth, skill development, and discovery. I am grateful for the mentorship and resources provided during this program. I leave this experience not only with a stronger technical portfolio but also with the confidence and clarity needed to pursue a successful career in the dynamic field of software development.

References

- **Oracle Java Documentation** – Official reference for Java programming and GUI components.
https://docs.oracle.com/javase/

- **Oracle Swing Tutorial** – Java Swing components and event handling guide.
https://docs.oracle.com/javase/tutorial/uiswing/

- **Oracle AWT Documentation** – Abstract Window Toolkit (AWT) for GUI design in Java.
https://docs.oracle.com/javase/8/docs/api/java/awt/package-summary.html

- **Java Platform SE API Specification** – Complete reference for Java SE.
https://docs.oracle.com/javase/8/docs/api/