While coding the project, I struggled to develop code that would display the file when the user pressed the button. I didn't know what to put in the try statement, but I did know how to do the catch. This is when a friend suggested I use the tool ChatGPT. I input my current code and provided a brief explanation of what I wanted the AI to accomplish.

The AI analyzed my code and suggested the try statement:

```
try {
                                        BufferedReader reader = new BufferedReader(new FileReader(FILE_PATH));
//opens file so program can read its text
                                        String line; //This creates a place to store 1 line of text from the file.
                                        String all = "";
                                        while ((line = reader.readLine()) != null) {
                                                all += line + "<br>";
                                        }
```

This solution worked, completing the code. After playing with the program, I researched what this code meant.

The first line, BufferedReader reader = new BufferedReader(new FileReader(FILE_PATH));, opens the file and allows the program to read its texts

The next two lines are related to storing the data in the file in a variable. The variable "line" stores 1 line of text from the file, and the variable "all" stores all of the data in the file (but starts empty).

Inside a while loop, the condition has "reader" which reads one line at a time in the file until no lines are left. Each time the loop reads a line, it saves that data to the variable "line." If the line is not null (meaning the file still has more text), the loop adds the line to the variable "all," which holds all of the values of the file. Once the file reaches the end, readLine() will return null and stop the loop.

reader.close(); let the file know that no more will be read

After the while loop is completed, the variable "all" (containing all of the file components) is outputted to the user.

```
displayLabel.setText("<html>" + all + "</html>");
```

Another error showed.
When I pressed the "View File Contents" button, nothing happened. I asked ChatGPT how to solve this issue, and it provided me with a couple of corrections

1) After finishing writing to the file, state that you are finished writing to fthe ile: writer.close();
2) After finishing reading the contents of the file, state that you are done reading the contents of the file: reader.close();

These suggestions corrected the code so the program worked!

Chat GPT convo:

What is the error with this code

ChatGPT said:

I don’t see any code in your message.
Please paste the code you want me to check, and I’ll tell you what the error is.

You said:

```
package Mastery; import java.awt.EventQueue; import javax.swing.JFrame; import
javax.swing.JLabel; import javax.swing.JTextField; import javax.swing.JButton; import
java.io.File; import java.io.FileWriter; import java.io.FileReader; import java.io.BufferedReader;
import java.io.IOException; import java.awt.event.ActionListener; import
java.awt.event.ActionEvent; public class studentSemesterAverage {          // This is the file that
will store the saved data          private static final String FILE_PATH = "studentData.txt";     private
JFrame frame;          private JTextField studentName;       private JTextField gradeLevel;
private JTextField semesterNumber; private JTextField gradeOne; private JTextField gradeTwo;
private JTextField gradeThree;         private JTextField gradeFour;          private JLabel
avgLabel;       private JLabel displayLabel;  public static void main(String[] args) {
EventQueue.invokeLater(new Runnable() {                    public void run() {
          try {                                   studentSemesterAverage window = new
studentSemesterAverage();                             window.frame.setVisible(true);
                    } catch (Exception e) {
e.printStackTrace();                          }                   }                 });      }
public studentSemesterAverage() {             initialize();      }        private void initialize() {
          // Create or check the file              File textFile = new File(FILE_PATH);
try {                         if (textFile.createNewFile()) {
System.out.println("File created.");                   }                 } catch (IOException e) {
               System.out.println("Error creating file.");              }
//Formating display and making elements                                 frame = new JFrame();
          frame.setBounds(100, 100, 450, 500);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.getContentPane().setLayout(null);                                 JLabel lbl1 = new
JLabel("Student Name:");             lbl1.setBounds(10, 10, 100, 20);
frame.getContentPane().add(lbl1);            studentName = new JTextField();
studentName.setBounds(120, 10, 200, 20);           frame.getContentPane().add(studentName);
               JLabel lbl2 = new JLabel("Grade Level:");            lbl2.setBounds(10, 40, 100,
20);           frame.getContentPane().add(lbl2);            gradeLevel = new JTextField();
          gradeLevel.setBounds(120, 40, 200, 20);
frame.getContentPane().add(gradeLevel);          JLabel lbl3 = new JLabel("Semester:");
          lbl3.setBounds(10, 70, 100, 20);            frame.getContentPane().add(lbl3);
semesterNumber = new JTextField();               semesterNumber.setBounds(120, 70, 200,
20);            frame.getContentPane().add(semesterNumber);            JLabel lbl4 = new
JLabel("Grade 1:");            lbl4.setBounds(10, 100, 100, 20);
frame.getContentPane().add(lbl4);            gradeOne = new JTextField();
gradeOne.setBounds(120, 100, 200, 20);            frame.getContentPane().add(gradeOne);
        JLabel lbl5 = new JLabel("Grade 2:");            lbl5.setBounds(10, 130, 100, 20);
        frame.getContentPane().add(lbl5);            gradeTwo = new JTextField();
gradeTwo.setBounds(120, 130, 200, 20);            frame.getContentPane().add(gradeTwo);
        JLabel lbl6 = new JLabel("Grade 3:");            lbl6.setBounds(10, 160, 100, 20);
        frame.getContentPane().add(lbl6);            gradeThree = new JTextField();
gradeThree.setBounds(120, 160, 200, 20);            frame.getContentPane().add(gradeThree);
        JLabel lbl7 = new JLabel("Grade 4:");            lbl7.setBounds(10, 190, 100, 20);
        frame.getContentPane().add(lbl7);            gradeFour = new JTextField();
gradeFour.setBounds(120, 190, 200, 20);            frame.getContentPane().add(gradeFour);
        JLabel lblAvg = new JLabel("Average:");            lblAvg.setBounds(10, 220, 100, 20);
        frame.getContentPane().add(lblAvg);            avgLabel = new JLabel("");
avgLabel.setBounds(120, 220, 200, 20);            frame.getContentPane().add(avgLabel);
                        displayLabel = new JLabel(""); //Label that will show saved data
displayLabel.setBounds(10, 280, 400, 150);            frame.getContentPane().add(displayLabel);
        JButton viewButton = new JButton("View File Contents"); //View file button
viewButton.setBounds(170, 250, 150, 25);            frame.getContentPane().add(viewButton);
                                JButton saveButton = new JButton("Save to File");// SAVE
BUTTON            saveButton.setBounds(10, 250, 150, 25);
frame.getContentPane().add(saveButton);            saveButton.addActionListener(new
ActionListener() { //When the save button is pressed            public void
actionPerformed(ActionEvent e) {                        try {
// Convert grades to numbers                                    double g1 =
Double.parseDouble(gradeOne.getText()); //Changing information from string to double for
grade 1                        double g2 =
Double.parseDouble(gradeTwo.getText()); // for grade 2                        double
g3 = Double.parseDouble(gradeThree.getText());//grade 3                        double
g4 = Double.parseDouble(gradeFour.getText());// grade 4
                        double avg = (g1 + g2 + g3 + g4) / 4; //Calculate average
                                        avgLabel.setText(Double.toString(avg));       //
display average                                    // Write to file
FileWriter writer = new FileWriter(FILE_PATH, true);
//Information that we are writing to file for:                        writer.write("Student
Name: " + studentName.getText() + "\n"); //Student name3
writer.write("Grade Level: " + gradeLevel.getText() + "\n"); //grade level
        writer.write("Semester: " + semesterNumber.getText() + "\n"); //semester number
                        writer.write("Grades: " + g1 + ", " + g2 + ", " + g3 + ", " + g4 + "\n");
//grades                        writer.write("Average: " + avg + "\n");//average
                                writer.write("----------------------------\n"); //This writes a line to
separate each students data
displayLabel.setText("Saved to file!"); //Confirming to the user that the data inputed has been
sent and stored in the file                              } catch (Exception ex) { //If there is a
problem with sending input to file                                      displayLabel.setText("Error
saving to file."); //What is said if there is a problem                              }
}                 });                              viewButton.addActionListener(new ActionListener()
{ //if view file button is pressed                       public void actionPerformed(ActionEvent e)
{                                  try {                                       BufferedReader reader =
new BufferedReader(new FileReader(FILE_PATH)); //opens file so program can read its text
                                   String line; //This creates a place to store 1 line of text from the
file.                                       String all = "";                                   while ((line =
reader.readLine()) != null) { //Reading one line in the file at a time until no lines are left
                                   all += line + "<br>";                                        }
                                                         displayLabel.setText("<html>" + all +
"</html>");      // Display info in file                          } catch (IOException ex) { //if for
some reason the file cannot be accessed
displayLabel.setText("Error reading file."); //What is outputed to the user if the file cannot be
accessed                                    }                        }                 });       } }
```

ChatGPT said:

Here is the **main error** in your code:

❌ You NEVER close the FileWriter or BufferedReader

That means:

- The file may not actually save correctly
- The file may stay locked
- Reading might fail randomly
- Some systems will throw an I/O error

This is a real problem because your code does this:

```
FileWriter writer = new FileWriter(FILE_PATH, true);
// write...
// BUT NEVER writer.close();
```

and:

```
BufferedReader reader = new BufferedReader(new FileReader(FILE_PATH));
// read...
// BUT NEVER reader.close();
```

Both **must be closed**.

✅ How to Fix It

Add:

```
writer.close();
```

and:

```
reader.close();
```

📌 Corrected Save Button Code

```
FileWriter writer = new FileWriter(FILE_PATH, true);
writer.write("Student Name: " + studentName.getText() + "\n");
writer.write("Grade Level: " + gradeLevel.getText() + "\n");
writer.write("Semester: " + semesterNumber.getText() + "\n");
writer.write("Grades: " + g1 + ", " + g2 + ", " + g3 + ", " + g4 + "\n");
writer.write("Average: " + avg + "\n");
writer.write("-----------------------------\n");
writer.close();   // <<< FIXED
```

📌 Corrected View Button Code

```
BufferedReader reader = new BufferedReader(new FileReader(FILE_PATH));
String line;
String all = "";

while ((line = reader.readLine()) != null) {
    all += line + "<br>";
}

reader.close();   // <<< FIXED
```