Project Report — Smart GUI-Based Calculator in Java Swing

# 1. Project Title

Smart GUI-Based Calculator Using Java Swing

# 2. Project Objective

This project aims to design and implement a feature-rich calculator using Java Swing. The primary objective is to deliver an interactive desktop tool that performs both basic and advanced mathematical operations while maintaining an intuitive and user-friendly interface. It allows users to compute values, view logs, and perform matrix-based operations—all within a structured GUI layout.

# 3. Tools & Technologies Used

|  |  |
| --- | --- |
| Tool/Technology | Purpose |
| Java | Core programming language for application logic |
| Java Swing | For building the graphical user interface (GUI) |
| AWT | Event handling for button actions |
| IntelliJ IDEA | IDE for coding, compiling, and debugging |
| JVM | Runtime environment for executing Java applications |

# 4. Project Working & Architecture

## User Interface (UI) Structure

The calculator window is divided into the following sections:  
- Top Header: A greeting label with instructions (e.g., “Click any operation button”).  
- Left Panel (Basic Operations): Add, Subtract, Multiply, Divide.  
- Right Panel (Advanced Operations): Square, Square Root, Power, Modulus, Logarithm, Natural Log, Factorial, Reciprocal, Matrix Addition, Multiplication, Transpose, and Game.  
- Center Panel:  
 - History Log: Stores a record of completed operations.  
 - Error Log: Displays error messages or invalid inputs.  
- Bottom Section: A result field and a “Clear All” button.

## Functional Logic

Each button uses an ActionListener that responds to user interactions. Mathematical computations are handled in real-time. Results and logs update dynamically. Matrix operations are handled through custom logic for handling arrays or list-based structures.

# 5. Key Features

- Basic + Scientific Functions  
- History and Error Logs for easy debugging  
- User Tips displayed on-screen  
- Clear All Option to reset fields  
- Matrix Support for advanced users  
- Game Button as an extension of creative ideas  
- Color-coded buttons for better UX

# 6. Advantages

- Enhances understanding of Java GUI programming  
- Event-driven approach ensures user interactivity  
- No terminal needed—completely mouse-driven  
- Platform-independent application  
- Real-time feedback and operation tracking

# 7. Conclusion

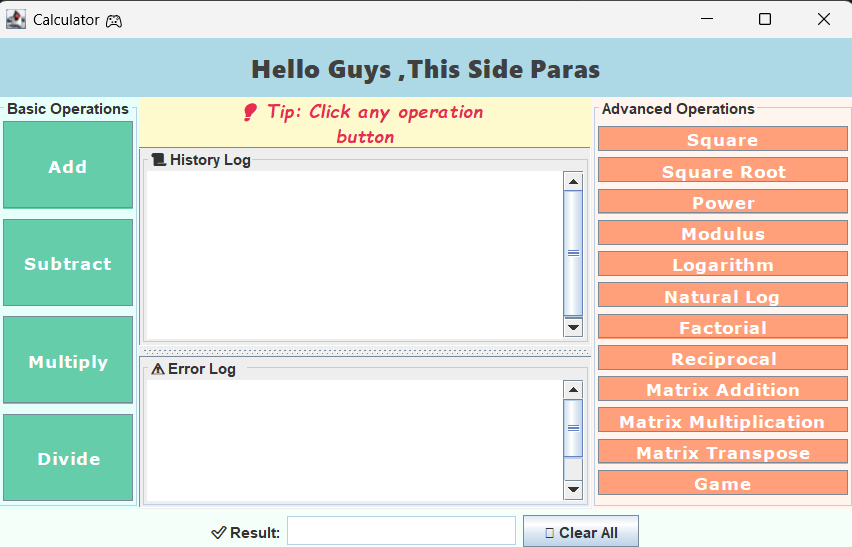
This project demonstrates how Java Swing can be utilized to build real-world, cross-functional GUI applications. By combining core arithmetic with advanced operations and a modern layout, it goes beyond a traditional calculator. It's ideal for both academic learning and practical desktop utility development.  
  
Created By: Paras Agrawal  
Built Using: Java Swing in IntelliJ IDEA

# 8. Future Scope

- Add graph plotting functionality  
- Introduce voice command support  
- Export history logs as PDF or text files  
- Add scientific constants and unit conversion  
- Include memory functions (M+, MR, MC, etc.)

# 9. Appendix / Screenshot

Below is the GUI screenshot of the final calculator:

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a game

AI-generated content may be incorrect.