Expense Tracker Desktop Application

Built using Java Swing, iText, and JFreeChart

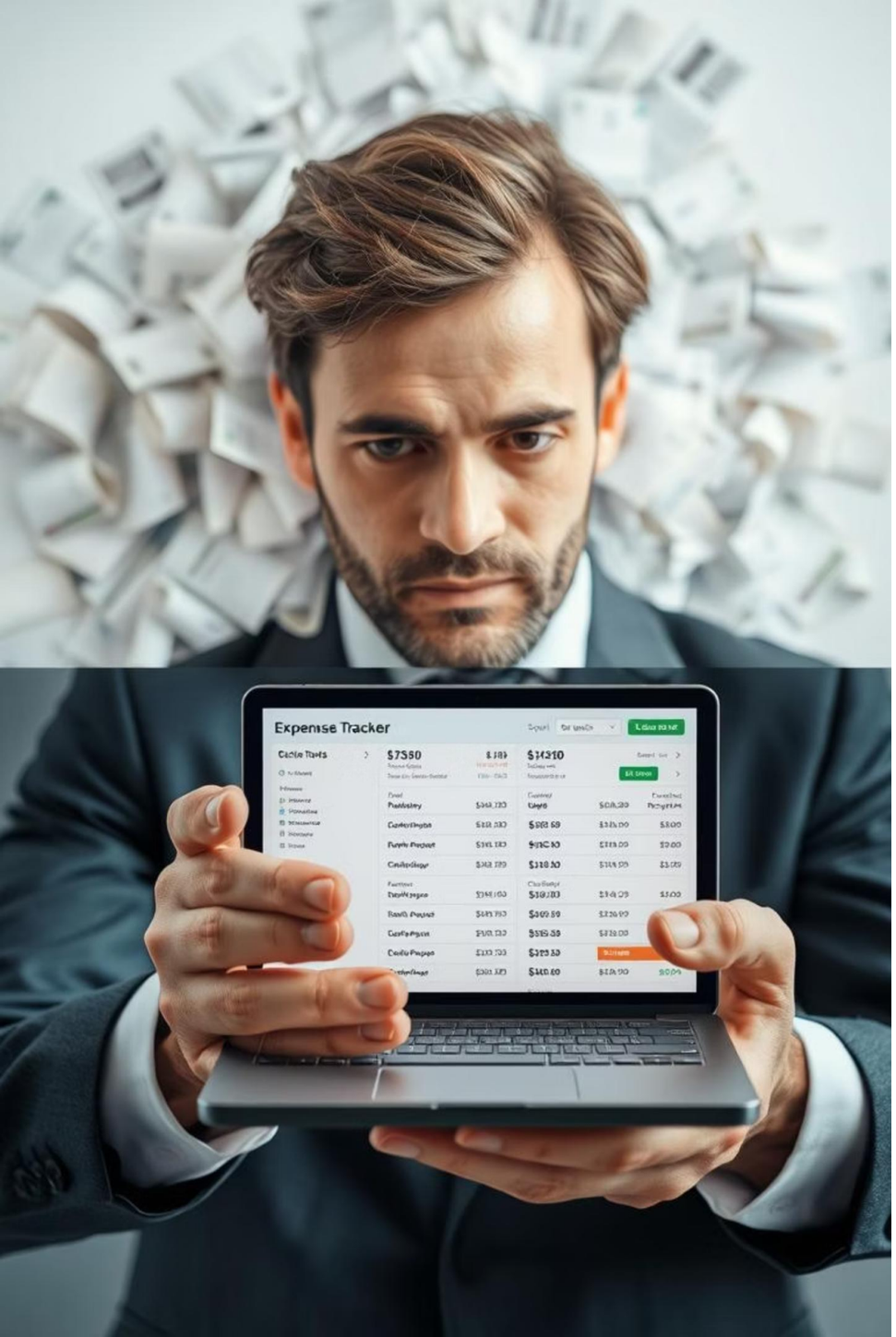
Chinmay Gawali – 22BTRAN006

Kaushal Raj – 22BTRAS019

Ayaan Quadeer – 22BTRAS051

Nupur Debnath – 22BTRAS029

AEROSPACE DEPARTMENT , JAIN UNIVERSITY



INTRODUCTION

This project addresses a common issue: people losing track of daily expenditures. It aims to provide a simple, offline desktop application for efficient expense management. Users can easily track, categorize, and visualize their spending. The application generates comprehensive reports and offers PDF export for record-keeping.

Tools & Technologies Used

Java SE

Core language for backend logic and file handling operations.

Java Swing

Used for building interactive Graphical User Interface (GUI) components.

iText Library

Enables the generation of dynamic PDF reports with structured tables.

JFreeChart

Integrates robust charting capabilities for data visualization.

File I/O

Manages data persistence through local file serialization.

Key Features

Add Expenses

Seamlessly enter amounts, categories, and descriptive notes.

Search & Filter

Quickly view expenses by week, month, or specific keywords.

Undo Functionality

Ability to effortlessly undo the last added expense entry.

Expense Charts

Visualize spending patterns with intuitive bar and pie charts.

PDF Export

Generate professional PDF reports of all recorded expenses.

User Profiles

Support for multiple user profiles with optional password protection.

System Architecture / Workflow

1

User Input

Data entry through the intuitive GUI.

2

Validation

Ensuring data integrity and correctness.

3

Data Storage

Stored securely using file serialization.

4

Display

Presented clearly in a Java Swing JTable.

5

Visualization

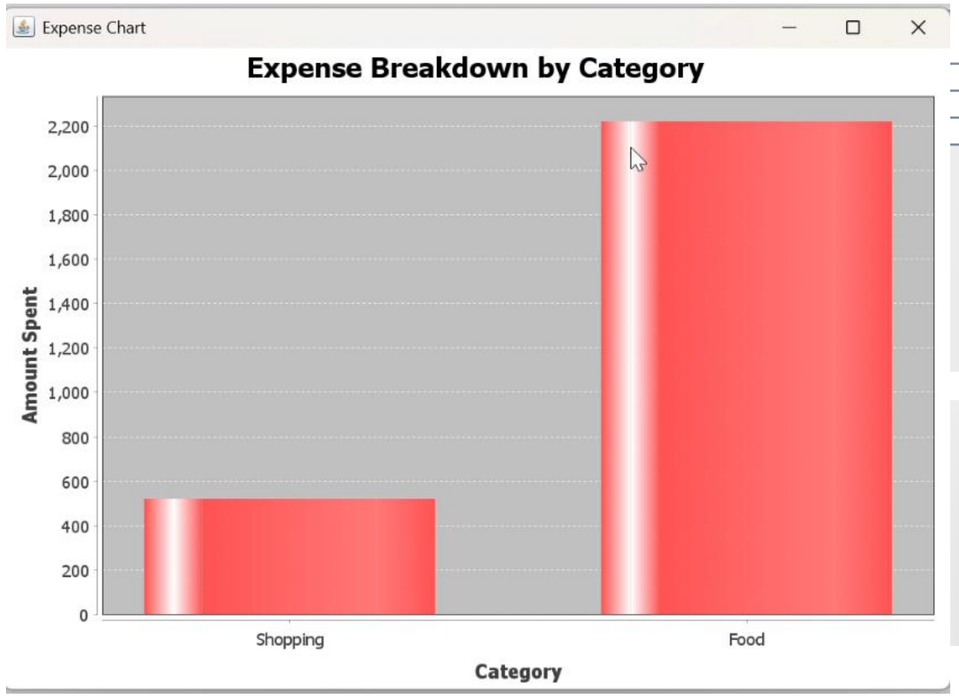
In-memory data used for charts and reports.

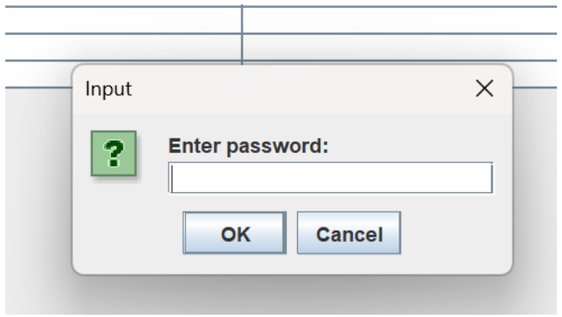
6

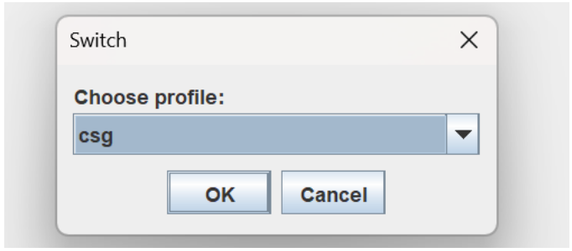
Export

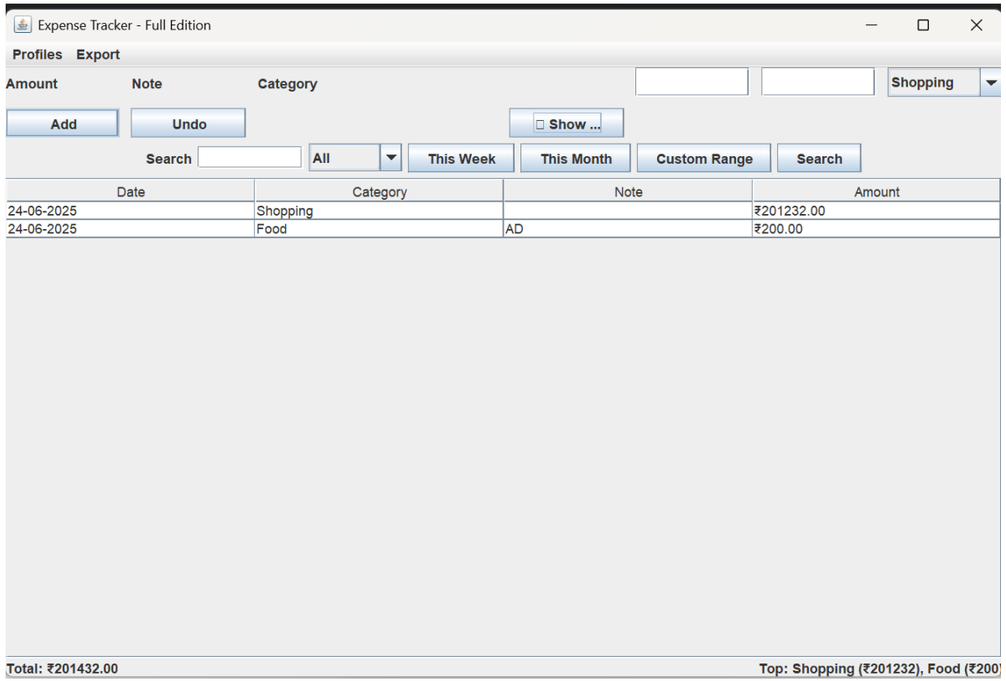
PDF generation via the iText library.

UI Screenshots









Visual examples of the application's user interface and its core functionalities.

Code Highlights

Adding an Expense

```
double amt = Double.parseDouble(amountField.getText());ExpenseEntry
entry = new ExpenseEntry(amt, note, cat,
LocalDate.now());profiles.get(currentProfile).computeIfAbsent(cat, k
-> new ArrayList<>()).add(entry);
```

This snippet captures user input, creates an **ExpenseEntry** object, and stores it under the current user's profile and category. It's crucial for recording financial transactions.

Generating a Chart

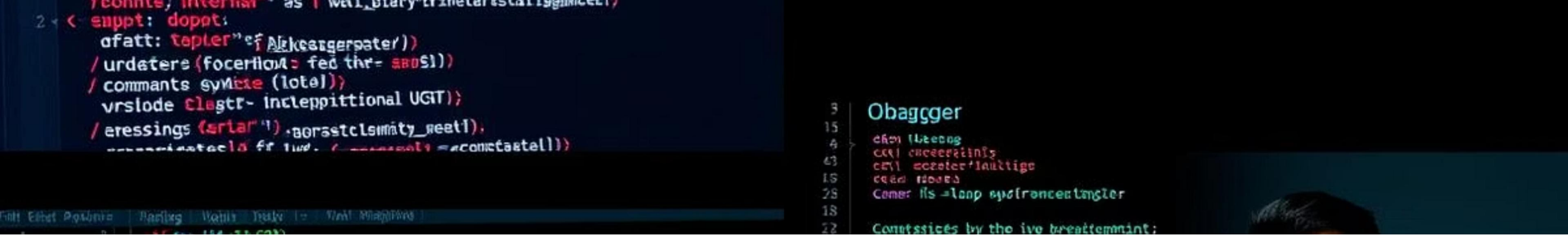
```
DefaultCategoryDataset dataset = new
DefaultCategoryDataset();dataset.addValue(500, "Amount",
"Food");JFreeChart chart = ChartFactory.createBarChart("Expenses",
"Category", "Amount", dataset);
```

This code demonstrates populating a **JFreeChart** dataset and generating a bar chart. It is vital for visual data analysis.

Exporting to PDF

```
PdfPTable table = new PdfPTable(4);table.addCell("Date");table.addCell("Category");table.addCell("Note");table.addCell("Amount");doc.add(table);
```

This section creates a PDF table, adds headers, and prepares it for data population before adding to the document. This is key for report generation.



Challenges Faced & Solutions

1 JAR File Detection

Issue: External library JARs were not detected.
Solution: Corrected classpath using the `-cp` flag in the terminal.

2 GUI Layout Breaking

Issue: Inconsistent or broken GUI element arrangements.
Solution: Implemented **GridLayout** and **BorderLayout** for structured design.

3 Chart Library Issues

Issue: JFreeChart libraries failed to integrate properly.
Solution: Ensured proper download and classpath addition of JFreeChart and JCommon.

4 File I/O Errors

Issue: Serialization errors during profile switching.
Solution: Utilized **ObjectOutputStream** for reliable data serialization.T

Future Scope

Enhanced Security

Implement login with encryption and user sessions.

Mobile Compatibility

Develop an Android version using Flutter or JavaFX.

Cloud Integration

Enable cloud backup and synchronization using Firebase or MySQL.

Advanced Reporting

Develop a graphical summary dashboard with monthly reports.

Diverse Export Options

Add export functionalities to Excel or CSV formats.

Conclusion

This project provided invaluable experience in building a full-fledged desktop application. I gained profound knowledge in GUI development, file handling, and integrating external libraries like **iText** and **JFreeChart**. My debugging, coding, and presentation skills significantly improved.

This project helped us apply Java concepts in a real-world application and gain confidence in full-stack Java development.