**D. Y. Patil College of College of Engineering and Technology, Kolhapur**

**Department of Computer Science & Engineering**

**Class: SY-A Subject: AOOC**

**Experiment no: 15**

**Group No. 10 Mini Project**

**Title of Mini-Project:- Multithreaded Download Simulator (with Swing GUI)**

**Problem Statement:**

In real-world applications, downloading multiple files simultaneously is a common need. Traditional single-threaded approaches often result in inefficient resource utilization and unresponsive user interfaces. This project aims to solve the problem by simulating and implementing a system that downloads multiple files in parallel using Java multithreading. The system also addresses user interactivity issues by providing a graphical user interface (GUI) with controls to pause, resume, and cancel downloads, while maintaining proper synchronization for safe thread management and real-time progress updates**.**

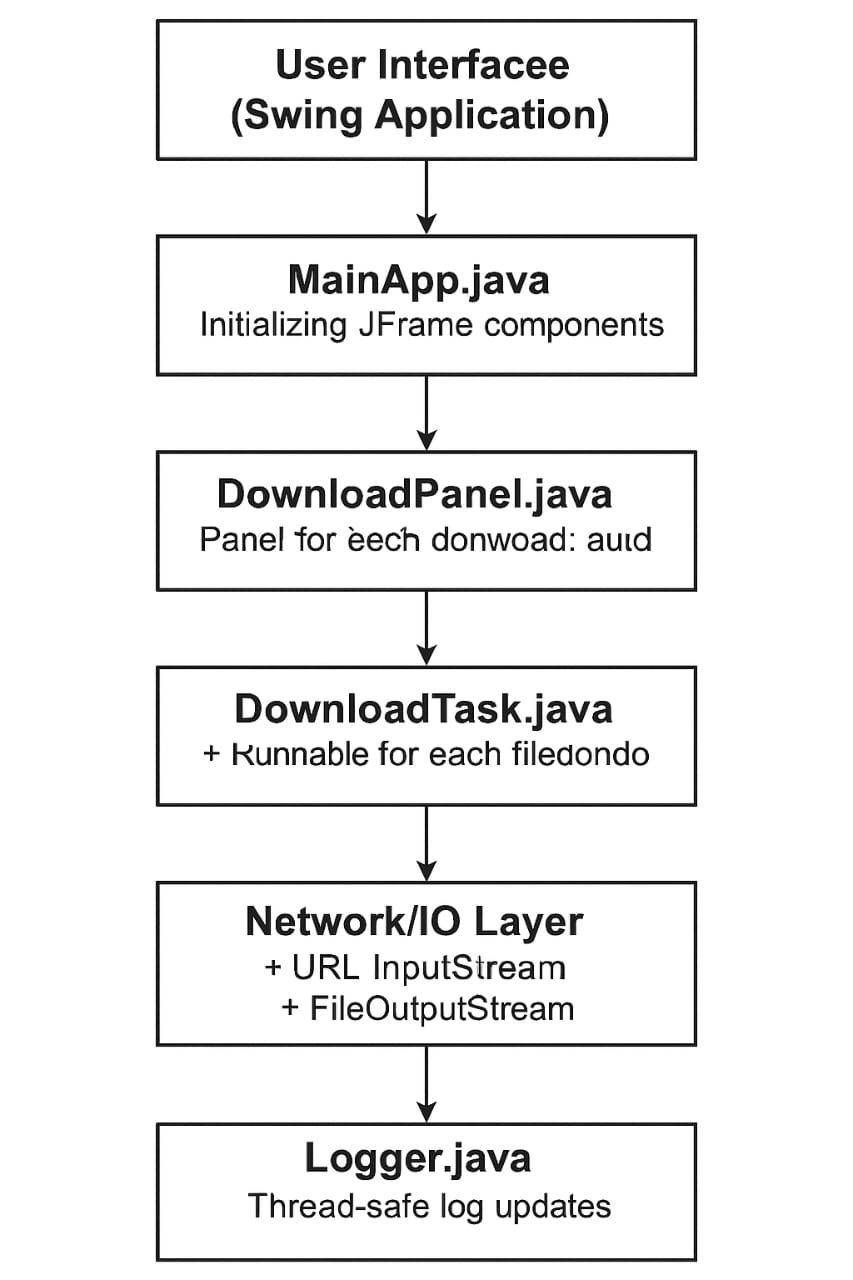
**Introduction:**

In the era of digital information, downloading multiple files efficiently is a common requirement. Traditional single-threaded applications download files sequentially, which leads to inefficient resource usage and longer download times. To solve this problem, our project introduces a Java-based Multithreaded Download Simulator that mimics real-world file downloads using concurrent threads. This simulator uses the Swing framework to create a user-friendly GUI that displays real-time progress for each file using progress bars.

Each file is handled by a separate thread, allowing downloads to occur in parallel, thereby maximizing efficiency. The system also includes advanced features like pause, resume, and cancel for individual downloads, enhancing user control. Logging is implemented through a JTextArea, providing a clear and synchronized display of progress updates and events. Additionally, the program supports actual file downloads from the internet using Java’s networking and I/O classes.

This project serves as an educational example of how multithreading, synchronization, GUI development, and networking can be integrated in Java to create responsive and interactive desktop applications.

**System Architecture:**

****

**Main Components:**

* MainApp.java: Initializes GUI and starts download threads.
* DownloadPanel.java: Contains progress bar and buttons.
* DownloadTask.java: Handles downloading logic in separate threads.
* Logger.java: Synchronizes log messages to GUI.

**Architecture Flow:**

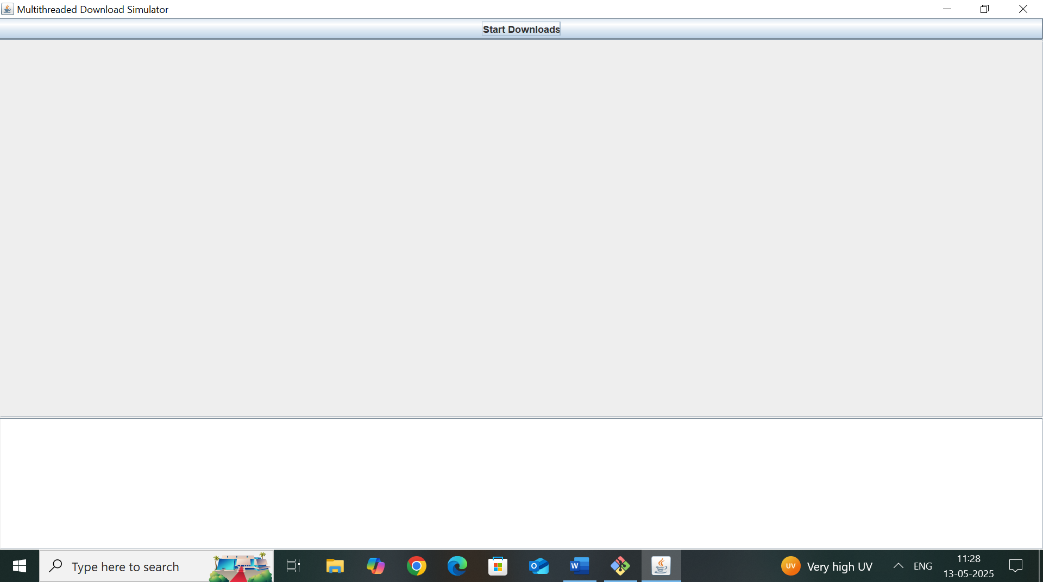
1. User clicks "Start Downloads"
2. Threads start downloading files
3. Each thread updates its progress bar
4. Logger logs progress or errors

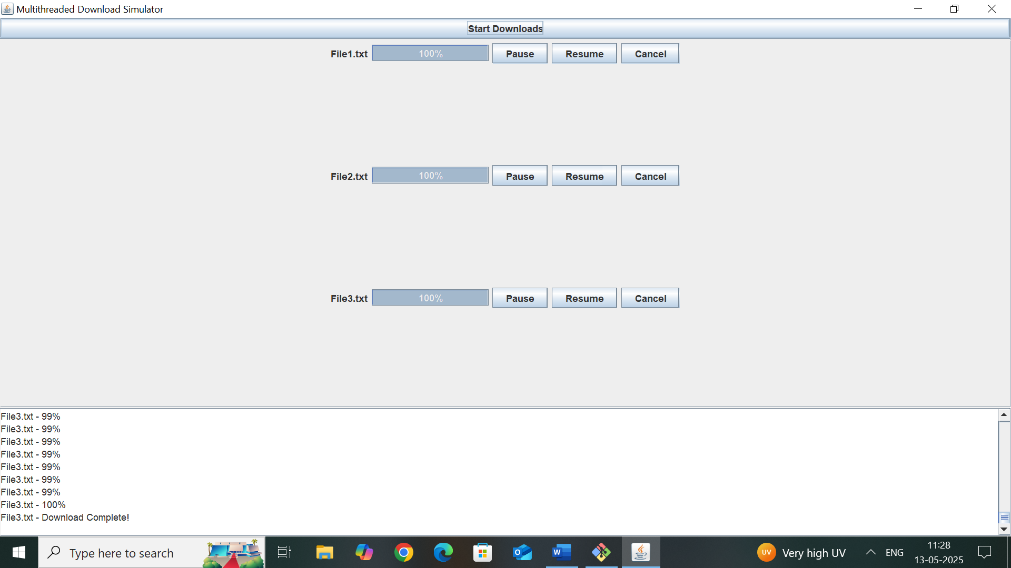
**Module description or working of system:**

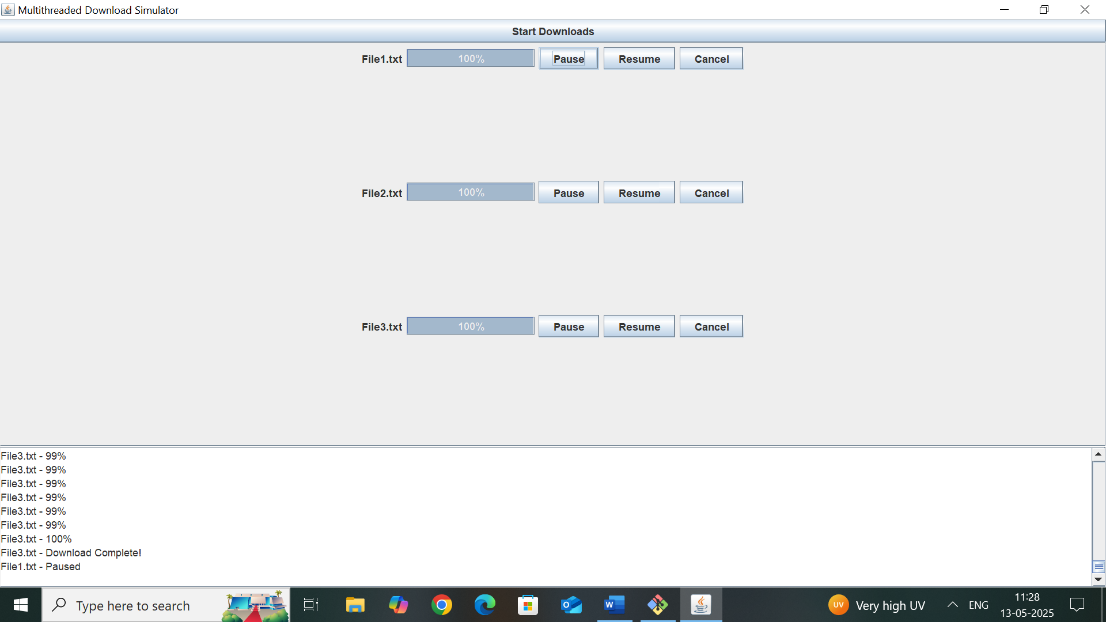
Working of the System

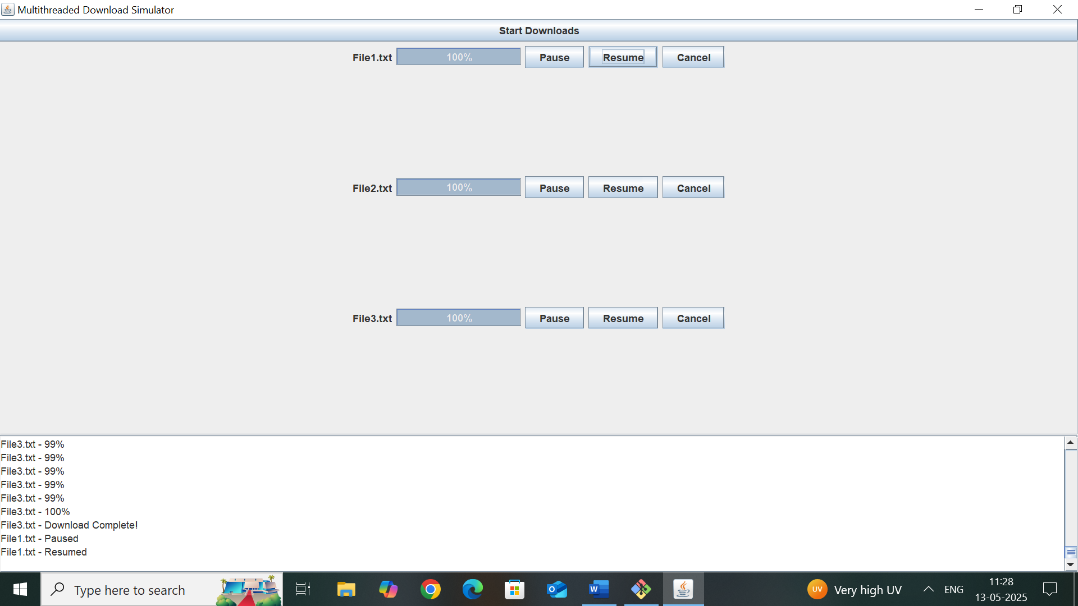
* On clicking "Start Downloads", threads are created for each file.
* Each thread downloads a real file using URL and InputStream.
* Users can pause/resume/cancel downloads via GUI buttons.
* Progress is updated on JProgressBars.
* Logs are displayed in a JTextArea panel.

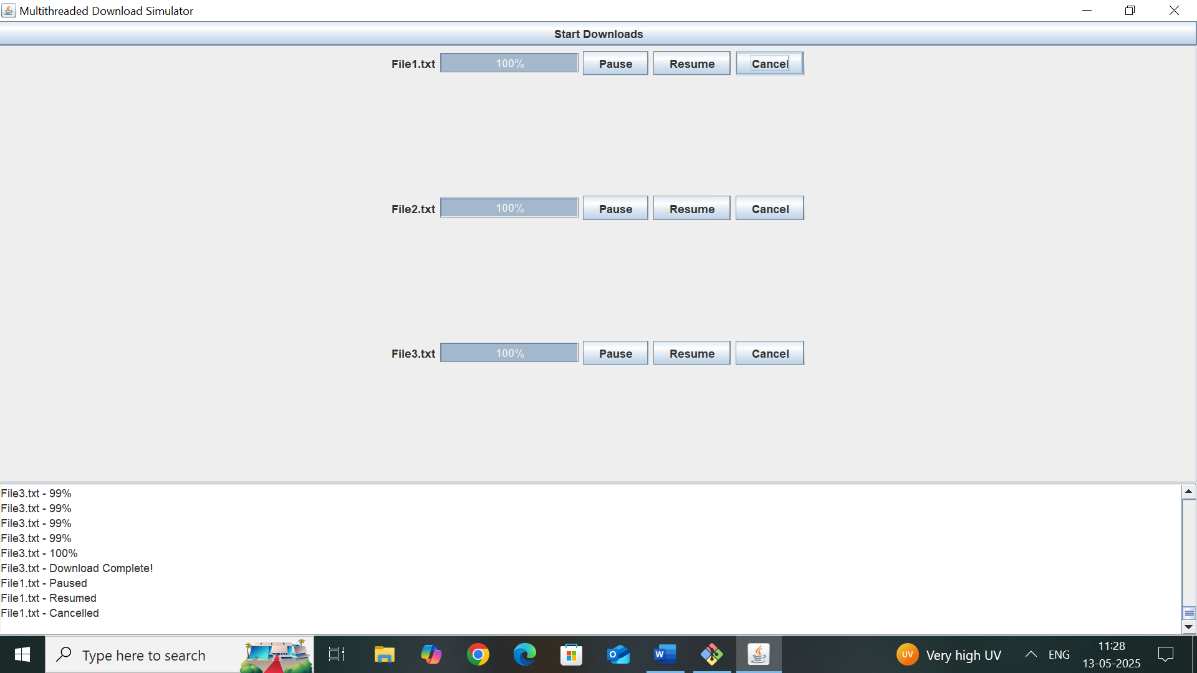
**Screenshots:**

****

****

****

****

****

**Group Members:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Unique id** | **Roll No** | **Name of Student** | **Sign** |
| EN23173547 | 44 | Shreyash Sanjay Lavande |  |
| EN23262495 | 47 | Kiran Shrikant Mulik |  |
| EN23170960 | 48 | Bhavika Naresh Lataye |  |
| EN23104659 | 49 | Sandhyarani Mahipati Torase |  |