**Event Ticketing System**

This Python application is a graphical user interface (GUI)-based event ticketing system built using tkinter. It allows users to register for tickets (VIP or Regular), process queued tickets and cancel previously registered tickets. The system tracks ticket transactions and updates a log of all actions.

**Features**

* **Register Tickets**: Users can register for VIP or Regular tickets with custom quantities. The system validates input to ensure correct data entry.
* **Process Next Ticket**: The system processes tickets in the order they were registered (VIP tickets first).
* **Cancel Tickets**: Registered tickets can be canceled, and the system restores the available ticket count.
* **Transaction Log**: All ticket transactions are logged with details such as name, ticket type, status, and timestamp. The log can be viewed within the application and is saved to a text file.
* **Dynamic Ticket Availability**: The system dynamically updates the number of available tickets as users register and cancel tickets.

**Requirements**

* Python 3.x
* tkinter library (usually comes with Python by default)

**Installation**

1. Ensure Python 3.x is installed on your machine.
2. Download or clone this repository.
3. Run the application using:

bash

CopyEdit

python ticketing\_system.py

**Usage**

**Register Ticket**

1. Click the **Register Ticket** button.
2. Fill in the form with your first name, last name, ticket type (VIP or Regular), and the number of tickets.
3. The system will validate the input and check ticket availability.
4. If registration is successful, the transaction is logged, and the available tickets count is updated.

**Process Next Ticket**

1. Click the **Process Next Ticket** button.
2. The system will process the next ticket in the queue (VIP tickets are processed first).

**Cancel Ticket**

1. Click the **Cancel Registered Ticket** button.
2. Select a ticket from the list of registered tickets to cancel.
3. Confirm the cancellation. The ticket will be removed from the queue, and the ticket count will be updated.

**Exit Application**

1. Click the **Exit** button to close the application after confirming your choice.

**Code Structure**

* **TicketSystemValidator**: Contains validation logic for user inputs, such as names.
* **TicketSystemApp**: The main class for the ticketing system GUI, which handles user interactions, ticket processing, and transaction logging.
* **Helper Methods**: Various helper methods handle ticket status updates, dynamic color changes, and saving the transaction log to a text file.

**Key Components**

1. **Ticket Registration**:
   * Users input their first and last names.
   * Ticket type (VIP or Regular) and quantity are selected.
   * Transaction details are stored and logged.
2. **Transaction Log**:
   * A table displays the transaction history with columns for Transaction ID, Name, Ticket Type, Status, and Timestamp.
   * The log is saved to a text file (transaction\_log.txt).
3. **Ticket Queues**:
   * Two queues track VIP and Regular ticket registrations.
   * VIP tickets are processed first, followed by Regular tickets.
4. **Exit Confirmation**:
   * The application prompts the user to confirm before exiting.

**Screenshot**

Here's an example of what the application looks like:

**Author**

Created by Kwaku Boateng. Contact via [boatengkwaku1965@gmail.com](mailto:boatengkwaku1965@gmail.com).