🌟🌟🌟 Day 107 🌟🌟🌟
# of Our Daily UiPath Q&A Challenge!

Welcome to Day 107 of our exciting UiPath Q&A Challenge! Each day, we'll be answering one key question to help you master UiPath and revolutionize your automation journey. 🚀

⛔ Question for Today

How are exceptions handled in RE-Framework?

In UiPath REFramework (Robotic Enterprise Framework), exception handling is a core component designed to ensure robust and resilient automation. Here's how exceptions are handled within the frameworkL

🔄 1. Try-Catch Blocks

Each major part of the process (like Init, Get Transaction Data, Process, and End Process) is wrapped in Try-Catch blocks to catch and handle exceptions gracefully.

🧱 2. State Machine-Based Handling

RE-Framework uses a state machine with the following key states:

1. Init – Initializes applications and systems.
2. Get Transaction Data – Retrieves the next item to process.
3. Process Transaction – Processes the current item.
4. End Process – Closes applications and performs cleanup.

Each state transitions based on the outcome:

- Success
- System Exception
- Business Exception

⚠️ 3. System vs. Business Exceptions

- System Exceptions: Unexpected errors like application crashes, timeouts, etc. These typically cause the process to retry the transaction or reinitialize the system.
- Business Exceptions: Expected issues like invalid data formats or missing fields. These are logged and skipped without retrying.

🔁 4. Retry Mechanism

- The framework uses the Orchestrator Queue retry mechanism or a custom retry logic for System Exceptions.
- Business Exceptions are not retried by default.

📝 5. Logging and Screenshots

- All exceptions are logged using UiPath’s Logging mechanism.
- Optionally, screenshots can be taken on error for debugging.

🧰 6. SetTransactionStatus.xaml

This workflow is responsible for:

- Logging the outcome of each transaction.
- Setting the transaction status (Success, Business Exception, or System Exception).
- Handling retries and queue item status updates.