RE Framework is used for:

* Proper Exception Handling
* Recovery Abilities
* Effective Logging
* Reporting Functionalities
* High Maintainability
* Extensibility
* Reusability
* Ease of Development

RE Framework uses **State Machine Diagram** and has **4 states**:

* **Init**
* **Get Transaction Data**
* **Process Transaction**
* **End Process**

**Init:** In Init state, Robot reads the configuration, settings and initializes(Start) the applications.

* If this step is **completed successfully**, robot moves on the next step, i.e. **Get Transaction Data**.
* If a system **error** is encountered, the next state that is executed is **End Process.**

**Get Transaction Data:** In Get Transaction Data State, there are two possible courses of action

* Either a **New Transaction** comes next and is processed by the robot. i.e. **Process Transaction**
* Or all transaction has been operated and **No Data** is left, so the process comes to an end i.e. **End Process**

**Process Transaction:** After retrieving a new transaction, the **Process Transaction** state is executed.

The **Process Transaction state** can have 3 outcomes:

1. **Success:** The first one is **Success**, in which case a loop is performed and the next transaction data is processed.
2. **Rule Exception:** The second possible outcome is **Business Rule Exception**, in which some specific action needs to be taken. Afterwards, the execution loop back to the **Get Transaction Data** state.
3. **Error:** Finally, the System Error transition, which requires us to take the necessary steps towards the recovery from the error. That means **all the applications are closed** and the execution loops back to the **Init** **State** in which the applications are restarted.

**End Process:**

**Steps:**

* The overall exception handling is already in place throughout the framework, which is why there are Try Catch activities in every state.
* To check whether there's an exception or not, we set the Exception object to **SystemError** and initialize it with “**Nothing**.”
* The next steps are to read the Config file, and after that, to start all the applications.
* We catch any exception that might come up, and then set the SystemError object to that specific exception.
* Depending on the value of SystemError. Either Get Transaction Data or the End Process state are performed.
  + If SystemError is noting, the Robot moves on to Get Transaction Data.
  + If a system exception is thrown, we log a Fatal error message stating that initialization failed, and the Robot moves on to End Process.

Init State: Read Config, Init Applications

Get Transaction Data: with two important variables, TransactionItem and TransactionNumber

Process Transaction: with the three possible transactions: Success, BusinessRuleException and SystemError

Set Transaction Status

GetAppCredentials