PEGA

* A low-code platform that provides an application development environment to create digital solutions for achieving business outcomes.

* Low-code platforms provide a visual, declarative technique like drag-and-drop to develop applications instead of coding.

Studios in Pega

* The Pega Platform provides four role based authoring workspaces called as studios.

→ **App Studio**

It is a low-code workspace for bussiness users & developers to create applications quickly.

example: A business analyst creates a simple customer service application.

→ **Dev Studio**

It is a advanced workspace for technical developers to create custom & complex applications.

Example: A technical developer integrates the simple customer service application with an existing CRM Tool.

→ **Prediction Studio**

A workspace used by data scientists for creating & managing predictive models & analytics.

Example: A data scientist provides analysis based on customer service application data.

→ **Admin Studio**

Workspace used by IT administrators to maintain and monitor the environment.

Example

An IT administrator monitors system performance & ensures the application runs smoothly.

Gen AI

Gen AI refers to the use of AI technologies to enhance application development & decision making processes.

Integration with External Applications

PEGA platform supports integration with external applications using REST & SOAP APIs.

It also supports RPA to integrate with external applications which doesn't support APIs

Design System

→ A design system in PEGA defines the design framework that provides a consistent user experience.

→ PEGA supports two design systems:

Constellation Design System

→ It is a react-based design system which provides reusable UI to enhance user excesience.

→ less customization

Cosmos Design system

→ It is a traditional design system which allows developers to create UI components by writing code.

* A design system is a builder kit that includes a shared library of
  - UI elements
  - styles
  - UX Patterns
  - Themes
  - Guideline

MicroJourney

A microjourney is a small part of overall customer journey and focuses on accomplishing a specific goal.

→ They are designed to deliver value within 60-90 days

Personas & Channels

→ personas determine who interacts with the applications

→ Channels determine how a personas interact with the application

Case Type

A case type is an abstract ~~transaction~~ model of a business Transaction

Case

A case is an specific instance of a case type.

Ex: In an online order system, order processing is case type and each order placed is a case.

case Life cycle

A case life cycle provides a clear visual representation of the workflow of a business process.

case life cycle consists of:

- Stages
- Process
- steps

Stages

Stages represent the case transfer from one caseworker to another or significant change in the case status.

→ When we design a case life cycle, we began by organizing work into stages.

Processes

Processes contain a series of steps or tasks that users complete as they work on the case.

Step

A step within a process is a user action or an automated action

Example

Employee onboarding process → case type

- Application Submission
- Interview
- Hiring

→ Stages

* Schedule Interview
* send offer letter

→ Processes

→ Review Application
→ send Email

→ Steps

It is a visual representation of all data elements within an organization & the connections between them.

It's main purpose is to define the data required by application to achieve business outcome.

It consists of:

* Fields: properties that store & format data
* Data object: categories of data that include fields, field mappings & connections to data source

Conceptual Data Model

The conceptual data model is a high-level representation of the data entities & their relationships.

It is used for initial discussions with stakeholders about data requirements.

Logical Data Model

It refines the conceptual data modeling by adding more detail.

It defines the data objects, attributes & their relationships.

Physical Data Model

It translates the logical data model into a specific database implementation.

It includes details about how data is stored, indexed & accessed.

Data Records

Data records are instances of data objects that contain specific values for each field.

Data Objects

Data objects group relevant fields together and we can reuse this fields in multiple places with different means

Address : Data object
- State → Address.State
- City → Address(1).City
- Pin

* Data Records are stored in three types:
  * No Store
  * Internal Page SOR
  * External SOR

Data Pages

Data pages are used to retrieve the data from records.

→ Temporary storage of fetched records is done using Data pages.

→ Single Data Page : Retrieves only one record

→ Data Page List : Retrieves multiple records.

→ Savable Data Page : It is used to Insert or Update data permanently.

→ Savable Data Page is deleted once data is persisted.

Integration Designer

Integration Designer in Pega Provides a centralized view of all data objects, data Pages, Object dependencies & external systems in an application

Field Groups

Cluster of individual fields that present related data inside a view.

→ Fields are individual pieces of data.

→ A field, when used to collect information becomes a property.

Simple Field Types

-> used to capture straight forward data

ex: Text, Integer, Boolean, Decimal, Date, Email, Phone

Fancy Field Types

-> More advanced data types that provide enhanced functionality

-> Picklist, Multi-select, Rich text, Attachment, Currency.

Complex Field Types

* They are used to capture more intricate data structures & relationships.

-> Field Groups, Field Group Lists, Embedded data.

Case Statuses

-> case statuses indicate the progress of a case.

-> Pega provides default case statuses like New, Open, Pending-Approval, Resolved-Completed

-> We can also set a Status as our wish.

Routing

Routing determines how work assignments are directed to the appropriate users or work queue.

-> Worklist: A list of assignments for a specific user.

-> Work Queue: A list of assignments for a group of users.

* Push Routing: System automatically assigns work to worklist or work queue

* Pull Routing: Users select the assignments from a worklist or work queue.

- Route to current user
- Route to specific user (Reference, Participant, By Name, Reporting Manager)
- Route to worklist
- Route to workqueue
- Business Logic Routing

SLA

SLA defines the expected time frames for completing tasks or cases.

It includes 3 main intervals:

- Goal: Ideal time to complete
- Deadline: Maximum time allowed.
- Passed Deadline: Time after the deadline has passed

Escalation actions: send mails, increase urgency, reassign.

---

* Step level SLA
* Process level SLA
* Stage level SLA
* Case level SLA

Urgency: 0 to 100
Default urgency: 10

Transfers
Data Transform
Activity
Notify assign, manager.

Agile Workbench

It is a powerful tool within the PEGA platform that supports real-time collaboration, reporting and managing bugs during application development along with providing feedback.

It also allows us to connect with project management tools like Jira, agile workbench for enhanced tracking & reporting.

Sizing of a PEGA Platform

→ It involves estimating resources, efforts and time required to complete the project.

→ It is crucial for planning, budgeting & ensuring successful completion of project on time.

→ estimate time & add some time & inform delivery date.

Optional Actions

Optional actions can be defined as the actions performed by the users when case is not in their control without affecting the main flow.

Optional Uses Action

It is a single-step action that uses can perform at any time within a case or a stage.

Optional Processes

Optional processes are multi-step actions that users can perform at any time within a specific case or a stage.

Case-wide Actions

The optional actions that can be invoked at any time during the entire case life cycle.

Stage-only Actions

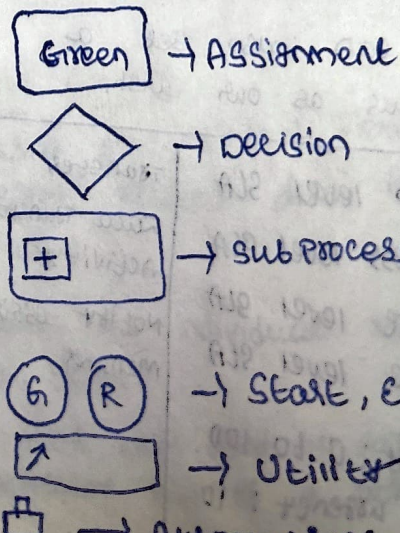
The optional actions that can be invoked only within a specific stage of a case life cycle.

Automating Decisions

Automating decisions in pega allows us to alter the flow of a case life cycle base on some business logic

→ It allows us to skip a step within a stage or to skip a stage.

→ Flow K → Confirmuse Process:



Child case

Child cases are used to model complex business processes that require multiple tasks to be handled separately but in parallel.

→ A child case must be completed to resolve the parent case.

→ A sub-case with no dependency is called as a spin-off case.

Duplicate case identification

→ It is a automation step in PEGA that allows us to identify a potential duplicate case that is very similar to another existing case.

→ It is done based on two parameters:

Base condition

* Basic rule that must be met for a case to be considered as potential duplicate.

Weighted conditions

* Additional rules that help identify duplicates by giving different importance levels to each rule.

Data validation ensures that correct data is passed & processed.

UI validation

It ensures that data entered by users is correct & meets the required format before submission.

It uses fields that are processed and validated in the client-side itself.

Validate Rule

It checks that the data entered by users meet specific conditions.

It is processed on server side using properties.

Edit validate Rule

Edit validate Rule is used to perform validations based on a pattern.

It is also processed on server side.

Data Transform

Run data Transform automation

Pre / Post Processing

It is used to manipulate data within an application.

It is used to:

→ copy data from one field to other

→ Move data between objects

→ Set default values

→ Convert data from one type to another

→ Calculate values

Insights & Reports

The Explore data landing page in Pega allows users to quickly explore and analyze data within their application.

* Insights can be generated by anyone who is having access to and are dynamic that are used to handle unplanned situations using KPI (Key Performance Indicators).

* Reports are generated by developers using a static query and here we have more control over the data.

Views → Conversation, Cosmos: Design Templates

→ Views are used to collect and display information.

→ Views can be configured in two ways:
- Workflow configuration
- UX Tab

→ Types of views:

* List view: Display data in a table format
* Partial view: Various layouts (1, 2, 3 columns, sub-tabs, narrow)
* Form view: Displays a form for data entry.
* Full-case view: case info
* Read-only: Displays fields as non-editable.

Rule

→ Rule is the building block of pega platform that define its behaviour.

→ Each Rule is an instance of Rule Type.

→ The Rule type determines the type of application behaviour modeled by rule.

→ The use of Rule makes our application modular which provides us with three benefits:

* versioning: Developers create a new version of rule whenever new changes are required.
* Delegation: Developer delegate rules to allow business users to update case behaviour.
* Reuse: Rules can be reused whenever the same behaviour is expected again.

Ruleset

Ruleset is a container that holds a set of rules

→ Every Ruleset has Ruleset versioning which involves creating new versions of a ruleset to manage changes.

→ Each version is identified by a three part number (04-01-01) indicating minor, patch and major versions updates.

→ Whenever updates are needed, a new version is created and the older version is locked.

Ruleset Stack

It is the sequence of rules that an application uses.

→ It determines the order in which Pega searches for a rule.

→ Each application has a unique ruleset stack, which include all the versions of a ruleset from the highest to the oldest.

Instance Key

PEGA Platform creates a unique instance key to identify each rule on the system that is stored by using the property .pzInsKey.

It consists of the following:

→ The internal name for the rule type.

→ The applies to class for the rule.

→ The identifier of the rule.

→ The timestamp from when the rule was created.

Classes

→ Classes in Pega can be defined as group of rules based on their scope of reusability.

→ Each application consist of three class types.

Work Class

It contains the rules that describe behaviour about case life cycle.

Integration Class

It contains the rules that describe the behaviour about how applications interact with other systems.

Data Class

It contains the rules that describes the data types in the application such as a customer data.

Pattern Inheritance

It is automatic and uses class name structure to inherit the rule. It searches for rules in class that share a common prefix.

Directed Inheritance

It is explicit and allows rules to be inherited from a specified parent class which may be outside the business class hierarchy.

→ @baseClass: searches here at last, if not found: error

Class Hierarchy

The classes that comprise an application are arranged into a multi-level class hierarchy to organize application assets.

The class hierarchy determines how developers can reuse the rules within an application.

Enterprise class scope (ECS)
Case Type → Work → Application
→ DIV/UNIT → Organization
→ Other Apps → PEGA Platform

Ways to Create a Rule

Using Create Menu

Create Menu → Data Model
→ Edit validate → Label
→ Create & Open

Using APP Explorer

→ APP Explorer → Right Click
→ Create → Data Model
→ Edit validate → Label
→ Create & Open

Using Records EXPLORER

→ Records EXPLORER → Data Model
→ Edit validate

Save an Existing Rule

→ Open a Rule → Save AS
→ Create & Open.

Rule Types

Flow Rules

Define the sequence of tasks or processes in a case.

Data Transform Rule

Transform & map data from one format to another.

Validate Rule

Ensures the data meets specific criteria.

Section Rule

Define the layout & content of UI forms.

Declare Expression Rule

Automatically calculate values for properties based on change in their properties.

Activity Rule

→ It is a Rule type that defines a sequence of steps to automate Process

Difference Between Rule & Activity

* Rules can be declarative i.e. they can automatically respond to changes but activities are procedural i.e they execute a series of steps in a defined order.

* Activities are used for more complex, step-by-step processing tasks, but rules are used for a wide range of purposes (like UI design, data validation.

Properties

-> In Pesta, properties are used to store data that can have different modes based on their structure and usage.

Single Value Property

-> It stores a single piece of data such as string, number or date.

Value List

-> It stores an ordered list of single values.

-> Each value in the list is indexed by a number.

Value Group

-> It stores an unordered collection of single values, each identified by a unique key.

Page

-> It is used to store a single complex data structure, which can contain multiple properties.

-> Customer Details Page stores details like name, email, Phone, etc.

Page List

Stores an ordered list of pages that is indexed by a number.

Page Group

Stores an unordered collection of Pages that is identified by a unique key.

Prefix in Properties

Properties contain prefixes to indicate their type of purpose.

Px

Indicates that the property is Read-only and are used for system level information (PxCreateDateTime)
↓
instance.

Py

Indicates that the property can be modified and are often used for application level information (PyID) (unique ID for case

Pz

Indicates that the properties are read-only and should not be modified, used for internal processing
PzInskey (unique ID for an instance in database

Decision Tables

Decision Tables are used to derive a value based on multiple conditions.

Decision Tables evaluate the same set of properties to return a result.

Decision Tables contain rows for condition & columns for properties & rules.

Decision Trees

Decision Trees are used to handle logic that returns a result from a set of test conditions.

They evaluate different set of properties and can lead to additional comparisons based on true evaluations.

Decision Trees contains Branches, Hierarchical, Nested conditions.

Branch

→ Each branch represents a condition and an action. The action can be to return a result, continue or stop evaluation.

Hierarchical

organized in a tree structure starting from the root & branching out to more specific conditions.

Nested Conditions

can have nested branches for more complex decision-making.

NOTE

* Both Decision Tables & Decision tree evaluates properties or conditions that return results when a comparison evaluates to true.

* Decision Tables evaluate against the same set of properties or conditions whereas Decision Trees evaluate against different properties.

Decision Table: returning loan roi based on loan type.

Decision Tree: Deciding whether to select or reject a job application based on various properties like skills, experience, etc.

Cascading Approvals

Cascading approvals in Pega are used to configure a series of approvals for a case.

This processes ensure that multiple levels of approvals are obtained before a case can proceed.

-> There are two main modes:

Report Structure

It is used when approvals move up the reporting hierarchy of the submitter.

Ex: An expense report might need approval from the manager, senior manager, director.

Authority Matrix

It is a flexible model for cascading approvals that uses a decision table or other data structures to determine the approval chain.

It allows for approvals from various entities, both within & outside the submitter's organization.

Ex: An expense report that includes billable time might require approvals from accounts payable, the manager of the consultant, etc.

Note

* In Reporting structure approvals follow a hierarchical path within the organization.

* In Authority Matrix, approvals can come from various entities ~~within~~ both inside & outside the organization

Operators

In Pega, an operator is a specific user who interacts with the application.

Each operator has a unique identifier typically an email address called as Operator ID.

Each operator is associated with one or more work group, which includes a set of operators and a work queue that consists of pending assignments.

Each operator is associated with an Access Group that defines their permissi & access to different parts of an applicatio

Work Groups

→ A work group is a cross-functional team that contains a manager, a set of operators, and a work queue.

→ Work groups allow resources to be shared among units, divisions, or the entire organization.

Access Groups

→ Access Groups define the permissions & access levels for users.

→ Access groups determines what parts of the application a user can access and what actions they can perform.

Rule Resolution

Rule Resolution is a search algorithm that a system uses to find the most appropriate instance of a rule to execute in any situation.

The rule resolution algorithm uses inputs from key parts of a rule to identify the most appropriate rule instance.

Swimming: Swim rule from top to bottom.

Inputs to Rule Resolution

* Predefined rule keys that are used as unique identifiers, such as Apply to: Class, Rule name, etc.

* User's ruleset list

* Class hierarchy of the rule.

* Availability of Rule

→ The output of the resolution process is first rule found that matches all the input criteria.

Rule Availability

Rule availability determines whether a rule can be used during the case resolution process & whether it can be viewed, copied or edited.

We can update key parts of a rule instance such as rule availability.

Rule Availability Settings

Available

Can be used during rule resolution and also be executed, copied & viewed.

Final

Can be used during rule resolution but cannot be copied, edited.

Not Available

Cannot be used during rule resolution.

Blocked

Rule is considered during rule resolution but will not be executed, halts and throws error.

Withdrawn

The rule and all other lower versions are not considered during rule resolution.

Rule Locking

Rule locking is used to controll access to application rules & prevent conflicts when multiple developers are working on the same rule simultaneously.

Pessimistic Locking

It is a type of locking which ensures that only one user can edit the rule and it prevents others to edit the rule until the lock is released.

Lock Acquisition → Recorded in PR_SYS_Locks → Table → Other user tries to access → error until lock is released (default 30 mins)

Optimistic Locking

It is a strategy where no exclusive lock is applied when a rule is opened & multiple users can open & edit the rule simultaneously.

The system checks for changes before committing any updates.

No Initial lock → Multiple users can Edit → Before Committing Checks for conflicts → If conflicts arise → Ask to resolve → Else → Commit.

Clipboard

The clipboard is a temporary memory area on the server associated with every connected Pega Platform requestor, including browser-based users and unauthenticated guest users.

It is crucial for managing & processing data within a Pega application.

The clipboard consists of four many Pages:

* User Page * Data Page
* System Page
* Linked Property Page

-> The clipboard is the portion of memory on the servers reserved by Pega Platform for the data generated by applications.

-> When a value is assigned to a data element, the data element & its value are on the clipboard.

-> Your application uses the clipboard to populate fields on UI forms, perform calculations & evaluate decisions.

-> The clipboard is organized into various pages, each serving a specific purpose.

Users Page

The users page category contains pages created directly or indirectly as a result of user action.

Ex: Whereever we enter details in form they are stored in user page.

Data Pages

These category retrieves the data from specified source & cache this data in memory.

Example: Your app converts US dollars to euros. Exchange rates that determine conversion are cached in data page.

Linked Property Pages

Linked property pages list read-only pages created by related properties which contain information about the data types referenced by an associated property.

System Pages

The system pages category includes pages that describe the current user session such as the active user and the active application.

Example: When a user is signed in to pega platform, the platform maintains a clipboard page that contains information about the user. Such as their time zone.

Refresh strategy

Reload once per interaction

Do not reload when condition

~~When~~ reload is older

Escalated actions

* Apply datatransform
* Notify, Assign, Manager, Party
* Transfer
* Run activity.

Data Page

A Data page retrieves data from a specified data source & caches that data in memory.

Data Page Scope

The scope of a data page determines its visibility & how it is shared across different parts of a application.

Thread-level Scope

It is used when data is specific to a single user interaction or transaction.

Ex

Imagine you are filling out a form to book a flight. The data page that retrieves available flight options is specific to your session.

Requestor-level Scope

Suitable for data that needs to be available across multiple interactions within a single user session.

Ex

When you login to KL-ERP, a data page may load your details like Id, name. This data is ~~stored~~ available throughout your session, but not shared with anyone.

Node-level Scope

It is ideal for data that is frequently accessed and can be shared with multiple users & sessions.

Ex

A retail website stores the list of available products in a data page and this data is shared across all users and sessions on the same server node.

Data Sause Configurables

* Data Transform
* Activity
* Connector

Internal
→ obj open
obj save
obj handle

* Report definition
* Database lookup
* Robotic automation
* Robotic Desktop automation

Alternate Stage

* To handle exceptions which occurs during case processing.

Specific users

- By Name
- By Reference ID
- Reporting Manager
- Participant

Configure → UI
→ Application readiness
→ Localization → Translate to new language.

SLA
App Explorer → Process
→ Create → SLA → label
→ Create & Open.

Rule
* Rule Type
* RuleSet
* Apply to
* label

Data Transform Superclass
Superclass feature in data transform allows us to combine several data transforms to set values at multiple levels of the class hierarchy.

Process
→ Identify the parent of current class.
→ Continues until the highest parent is located
→ Finds the data transform at each level.
→ Invokes the data transform in sequence.

Steps
* Create same data transform with same name at each class level.
* Ensure the call superclass datatransform is selected.
* If properties are specified in both parent & sub class, sub class overrides the parent class.

Methods!
case Flow → connector
→ set properties → APPLY
Data Transform.

Pages & class
Page Name : PgWorkPage
Class : Name

Definition : source, target, relation
set
when
for each
update page
append, append & map to

Method 2
Click on step → Configure view
→ Pre/Post Processing
→ Define new data transform

Data Pages : Data Types
Data object → right click - create new
- Data page
- Page List
- Savable data page

Structure : Page , Page - list
Object type* ; Name
Mode : Savable , Editable , Read-only
Scope : Thread , Requestor , Node

Sources : 8

*Connector : External Data Source (SOAP & REST connectors)

*Data Transform

*Report Definition : Query to fetch multiple records from 1 or more sources.

*Activity

*Lookup : Pass a key, Fetch 0 - 1 record

*Robotic Automation

*Robotic Desktop Automation

*Aggregate source.

Data Page , Savable Data Page : Lookup
Data Page List : Report Definition

Routing
Step -> Contextual Property
Pane -> Route to (General)

SLA
Step -> Contextual Property
Pane -> Goal & deadline