**Organization Name: ALFA**

**Framework Application Name/Division Name: ALFA Insurance Application (ALFAInsu)**

**Implementation Application Name/Division Name: ALFA Auto Insurance Application (ALFAAUTO)**

**Implementation Application Name/Division Name: ALFA Life Insurance Application (ALFALife)**

* **Frame Work Application ID/User ID/Operator ID: Admin@ALFA, Password: rules**
* **Implementation Application ID/User ID/Operator ID: Admin@ALFA, Password: rules**
* **In Framework App No Need to create any Rules. In Framework we add only Implementation Applications**
* **Login with Implementation Application ID/ User ID/ Operator ID: Admin@ALFA, Password: rules**

**Understanding about Classes:**

**Purpose of Class:  Any Pega Rules we create & execute/Compile can be done only under a class. i.e only under classes we can create rules & execute the rules.**

**Class:**

**{**

**PEGA rules**

**}**

**If Class is not available, then we cannot create Pega Rules & We cannot compile the rules. PRPC Will automatically creates all the classes in PEGA.**

**PEGA Contains diff classes:**

1. **Predefined**
2. **User defined**

**Predefined Classes: 14**

**@baseclass (Super Class)**

**Work-, Data-, Int-, Code-, Index-, Rule-, CMIS-, Embed-,History-, Assign-, Log-, System,-, PEGA-**

**All Predefined Classes are Abstract Classes.**

**User Defined Classes:**

**Organization Names, Application Name/Division , Unit Name ( Work, Data, Int)  it will generate user defined Classes**

**Class Layers in PEGA:**

1. **Work Layer: (ALFA-ALFAAUTO-Work)**
2. **Data Layer: (ALFA-ALFAAUTO-Data)**
3. **Int Layer: (ALFA-ALFAAUTO-Int)**

**Implementation Class/Work Class Group/Work Pool/ Work Layer/ Class Structure: (ALFA-ALFAAUTO -Work)**

**Any Business rule we create, it will be executed only under ur Work class group. Class group is the main instance of the application.**

**Class group is a combination of Organization, Division & Unit Classes. Data Layer & Int Layer is also combination of Organization, Division & Unit Classes.**

* **ALFA-ALFAAUTO-Work**
* **ALFA: Organization Class ( Abstract)**
* **ALFA-ALFAAUTO: Division Class/Application name  ( Abstract)**
* **Work: Unit Class ( Concrete)**

**---------------------**

**Types of Classes in PEGA:**

**There are 2 diff types of classes in PEGA:**

**Abstract:**

* **These classes are meant for Reusability above the unit level (Work)**
* **In Abstract Classes we can add Pega Rules, We cannot compile/Execute the Rules in abstract classes (Non - initialized Classes). To Compile the Rules use them in the Concrete Classes ( Work Class)**
* **We cannot create Work Objects under Abstract Classes**
* **Before 7 Version abstract class names should end with - (AXIS-), from 7 version it is not required.**

**Ex:**

* **Data Class Group/Data Layer  (ALFA-ALFAAUTO -Data)**
* **Int Class Group/Int Layer (ALFA-ALFAAUTO -Int)**
* **Organization Class (ALFA)**
* **Division Class   ALFA-ALFAAUTO**
* **All Predefined Classes**

**Concrete: 3 types**

1. **Concrete-Is a Class group**
2. **Concrete-Belongs to Class group**
3. **Concrete-Doesn’t belongs to class group**

**Are Initialized classes.**

**We can add the Rule, Reuse Rules from abstract classes & Compile the rules**

**We can create Work Objects under Concrete Classes(  Only Under Work Class Group & Work Classes)**

**Ex:**

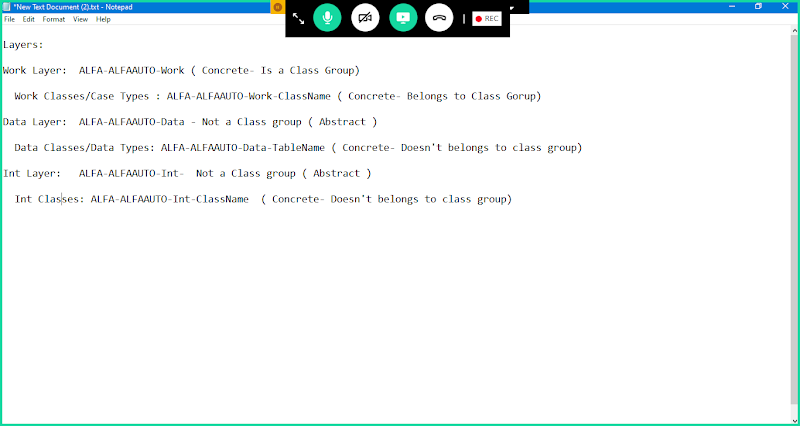
**Work Class group (ALFA-ALFAAUTO -Work)   - Concrete - Is a Class group**

**==**

**Work Layer--> Work Classes ( Case Types)**

**Data Layer--> Data Classes ( Tables/Data Types)**

**Int Layer--> Int Classes ( Integration Classes)**



**Concrete : 3 types:**

**Is a Class group: Work Class group/Work Layer  (ALFA-ALFAAUTO -Work)**

**belongs to Class group: Work Classes/Case Type ( If we are New Case Types) - (ALFA-ALFAAUTO -Work-Insurance)**

**Doesn't belongs to Class group: Data Classes, Int Classes  (ALFA-ALFAAUTO -Data -TableName ALFA-ALFAAUTO -Int -ServiceName)**

**-----------------------------**

**Parent Class for Work Class Group: Work-/Work-Cover-**

**Parent Class for Data Layer: Data-**

**Parent Class for Int Layer: Int-**

**======**

**Inheritance:**

**Inheritance: Makes Parent & Child relationship b/w classes**

**Class A ( Parent)**

**Class B ( Child)**

**Class A inherits Class B( Not Valid)**

**Class B inherits Class A ( Valid)**

**2 types:**

**Pattern Inheritance:**

* **Pattern is Created based on User Defined Classes, Pattern is represented by a CheckBox( It will be Enable by default)**
* **In Pattern Inheritance a Child Class will have one or more parent classes**
* **Pattern Inheritance is optional, but it is required for every class to reuse the rules in child classes( By Default for every class Pattern Inheritance is Defined)**

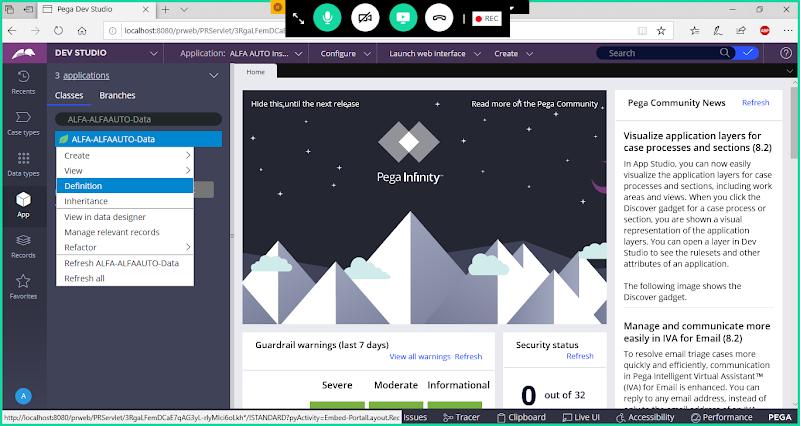
**Directed Inheritance:**

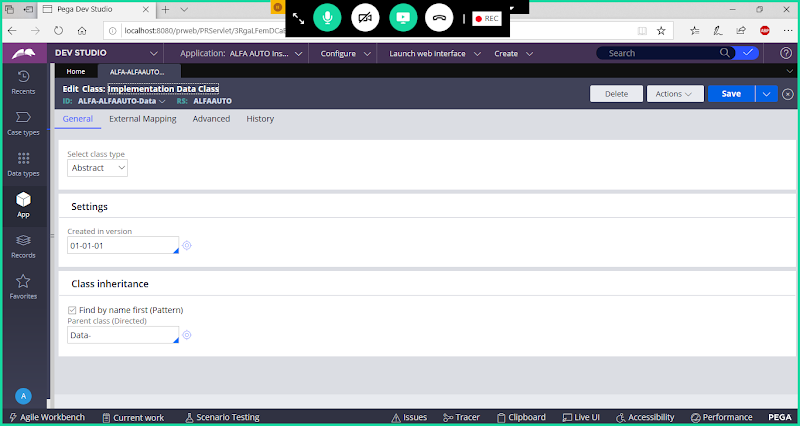
* **In Directed Inheritance Parent class will be a Predefined Class ( Layers)**
* **Child Class will have only one directed parent**
* **Direct Inheritance is Mandatory**
* **In Direct Inheritance, for Work Class group the parent class must be either Work-/Work-Cover-**
* **Data Layer Direct Parent is Data-**
* **Int Layer Direct Parent is Int-**

**==========**

**How to Open Class Definition:**

**Right Click on Class-> Click on definition**





**To Check Inheritance Tree for Any Class:**

