Job Scheduler In PEGA

It is an internal Background Process that operates on the server/node

It is a replacement of Advanced Agents

They are instances of RULE-ASYNC-JOBSCHEDULER

We must have privileges of pzJobSchedulerAdministrator privilege to create an manage job scheduler rules.

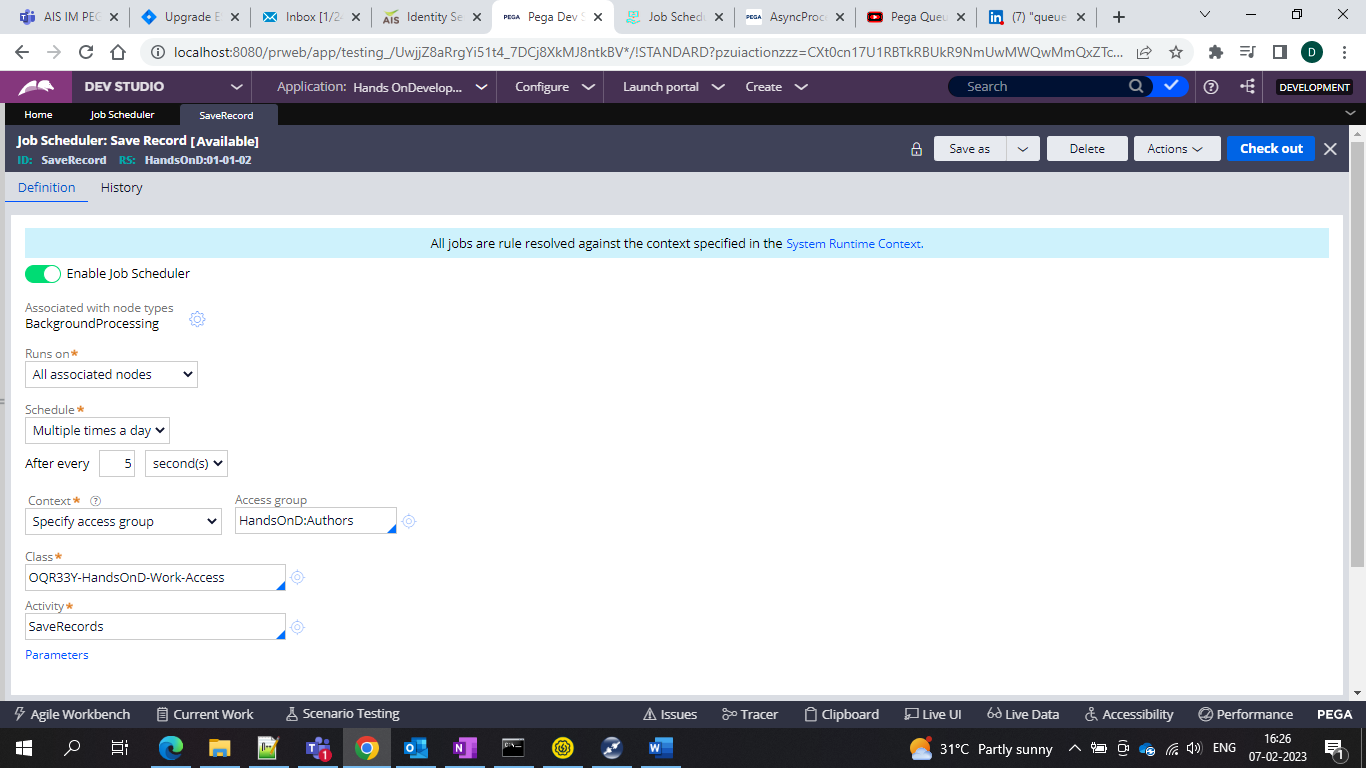
It is important that we have Access to the system runtime context or ASYNCPROCESSOR has access to the configured job scheduler.

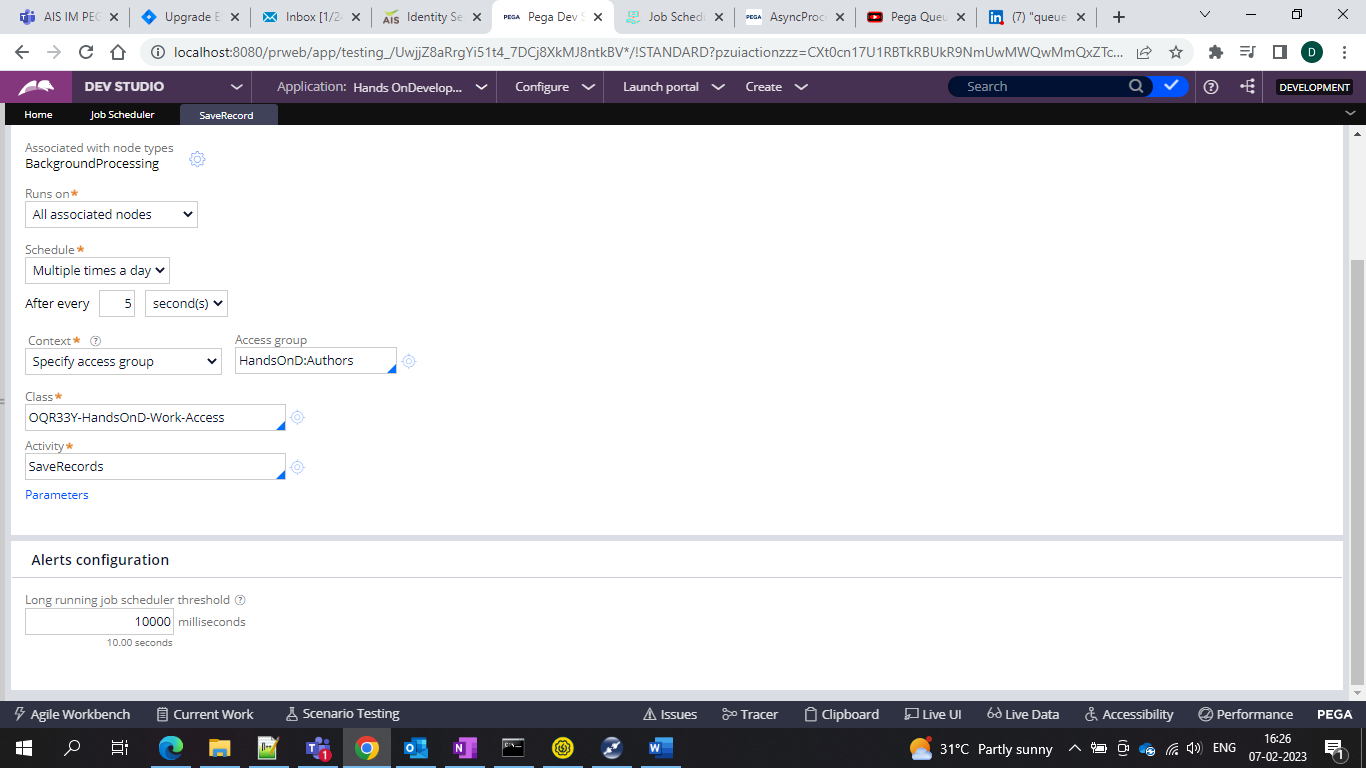
And one more thing is the Node and node means each environment thas server that will host PRPC rules engine. Each server is called a Node. So in our case it is Localhost.. we also can have multiple servers with different port numbers. There are many nodes like BIX ADM Web Users, Background Processing etc…

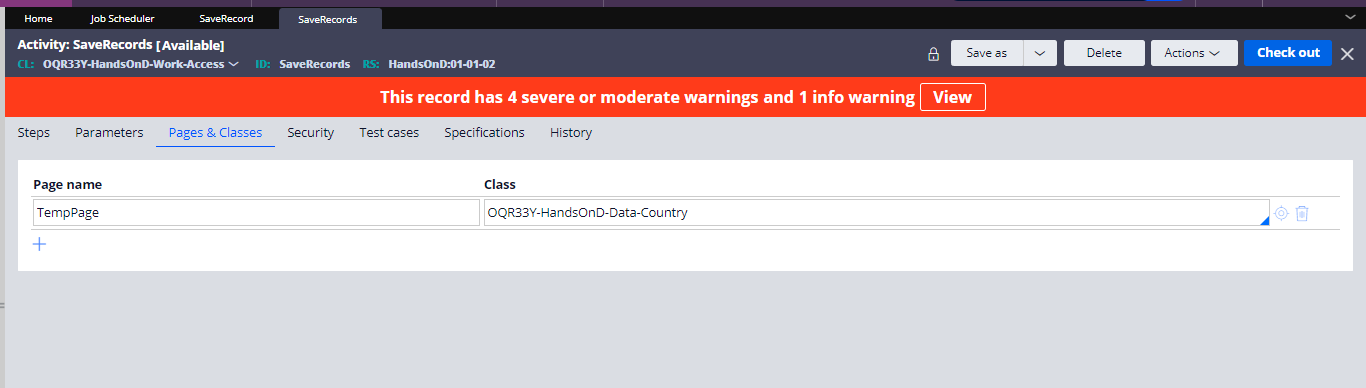
And also we have periodic schedule and define when and what time it is important to do it.

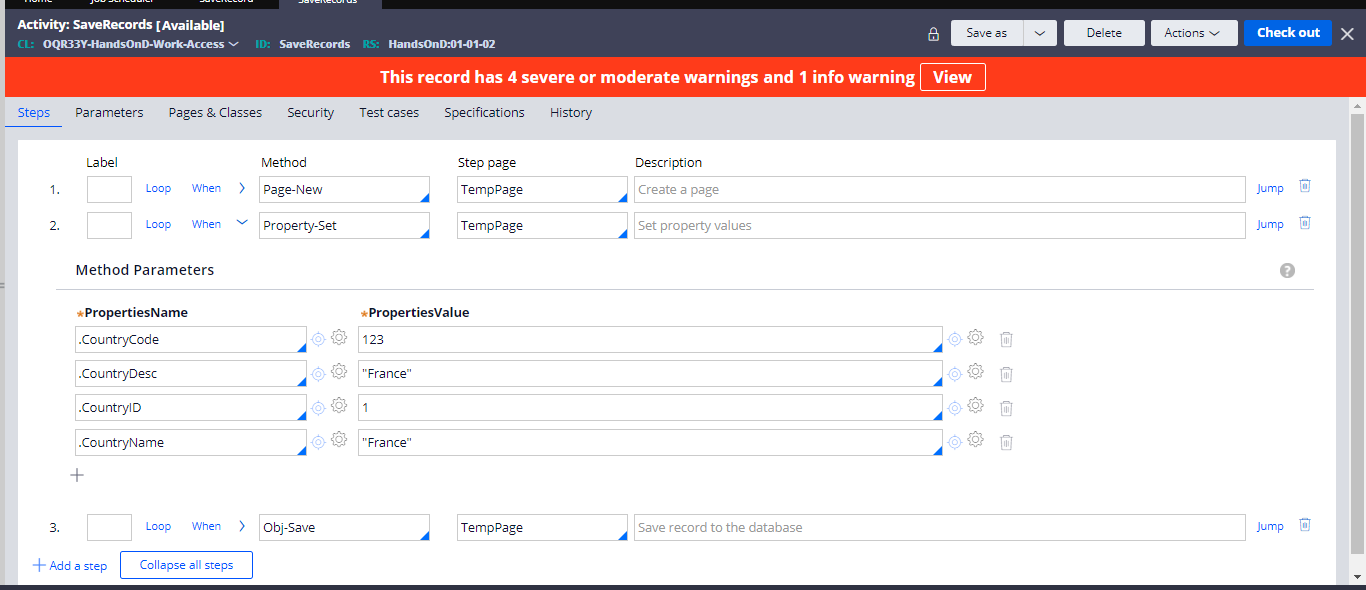
and in the Class and the Activity which will tell the job scheduler what it should do and then we can go for the tracing in the Admin Studio on a single or multiple nodes to trace and check the error and then we can also go for the checking of the failure..

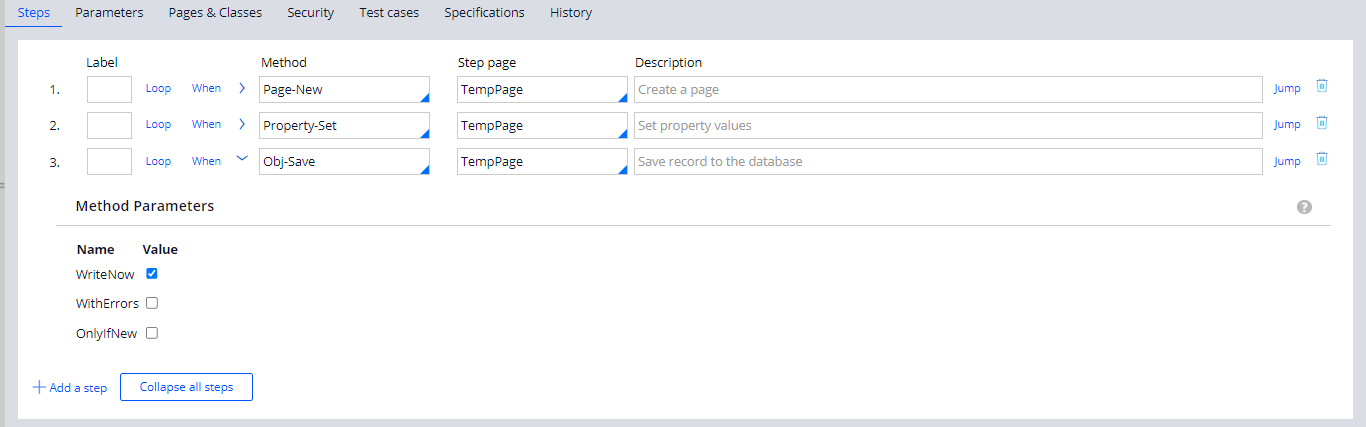
and now let us do a small practical on it I want to save a record to a database and I am doing it by using the Job Scheduler and it is pinned in the Activity



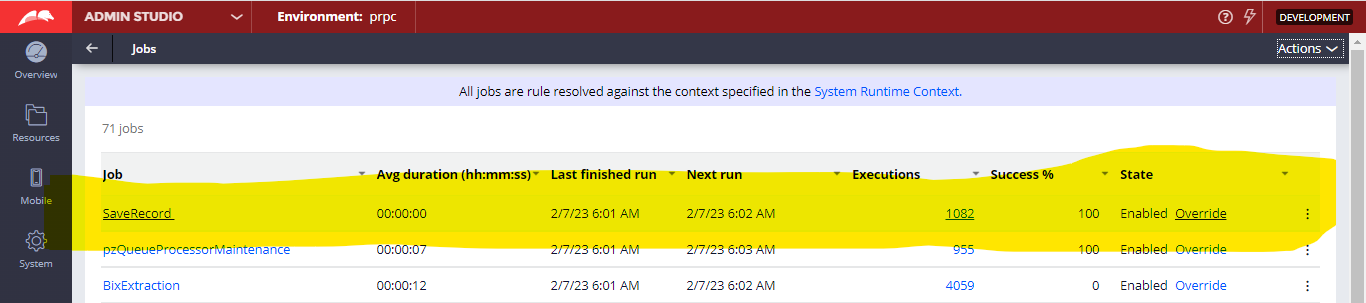


And in this we have the activity which is having method to save the records 





And this activity will run through job scheduler and then we can see the job scheduler how many times it ran



And also the success or failure and what is the failure and under the exception we can see the error

If any failure occurs

And then before 8.4 we used to have the ASYNCProcessor context also now it is using the System Runtime Context now and here you can see the error.

We learnt that job scheduler we understood that we can use job scheduler to run a defined task in a defined time interval as internal background process. So if we want to execute the same/different process as an internal background process at different time intervals…

And now let’s get into the Queue Processors

Queue Processors are now the replacement of Standard Agents

**And the most recommended agent processing is Standard agents and NOT advance agents.**

But still we go for the advance agents because there are downstream agents systems where we need info from

As the name suggests each queue processor will be mapped to Queue9Topic) and processes the messages published into the respective topic.

Each queue processor will have real time data flow associated with it. Associated data flow will be used to receive the messages from topic and process it.

Queue processor rules produce maximum throughput than the standard agents because it mostly follows data flows and horizontal and vertical scaling.

If we increase the number of nodes without changing the size of the node is horizontal scaling

And if we increase the size of the nodes without adding new nodes is called as Vertical scaling.

And then to configure we should have the pzQueueProcessorAdministartion and pzQueueProcessorObserver privilege

Unlike the job scheduler queue processor can only be run on a single node.

And here we have the class and activity and class to which it belongs that defines the logic to process the received messages from the topic(queue(.

When to process-

Immediate and delayed

Immediate means it will instruct the system to process the messages as sson as it gets published to the queue.

Delayed- selecting this will give an additional option to mention a date time property when we invoke the queue processor rule.

For the maximum records which the queue processor can process is 10. And it has 20 partitions in each topic and our queue processor can record 10 partitions since max count is 10.

Max Attempts- So this can process the number of attempts the messages in a topic should be reprocessed in case of failure. If more times it has failed then it will go to the broken item queue.

If we want to process immediately we will with immediate option and if we want delayed we may choose delayed

Delay factor is nothing but the period between next retry attempts by multiplying the previous delay value by the factor of 2. The delay sequence is 1 min and delay factor is 2. So the next attempts would be like the 2 min 4 min 6 mins 8 mins 10 mins etc.. and max wait time is 24 hours.

Suppose 1 st attempt is failed with delay factor 2 then 2 nd attempt delay factor would be 4 then again for the 3 rd time 4 \* 2= 8 would be the 3 rd time this is how delay factor will work.

Now what we can do is we can call this flow by using the run in background shape in the flow.

Previously we used to have system-queue-default entry for standard agents

And we also have the System-Message-QueueProcessor and inside this we have the BrokenItem and DelayedItem.

And the difference between standard and the dedicated queue processor is that

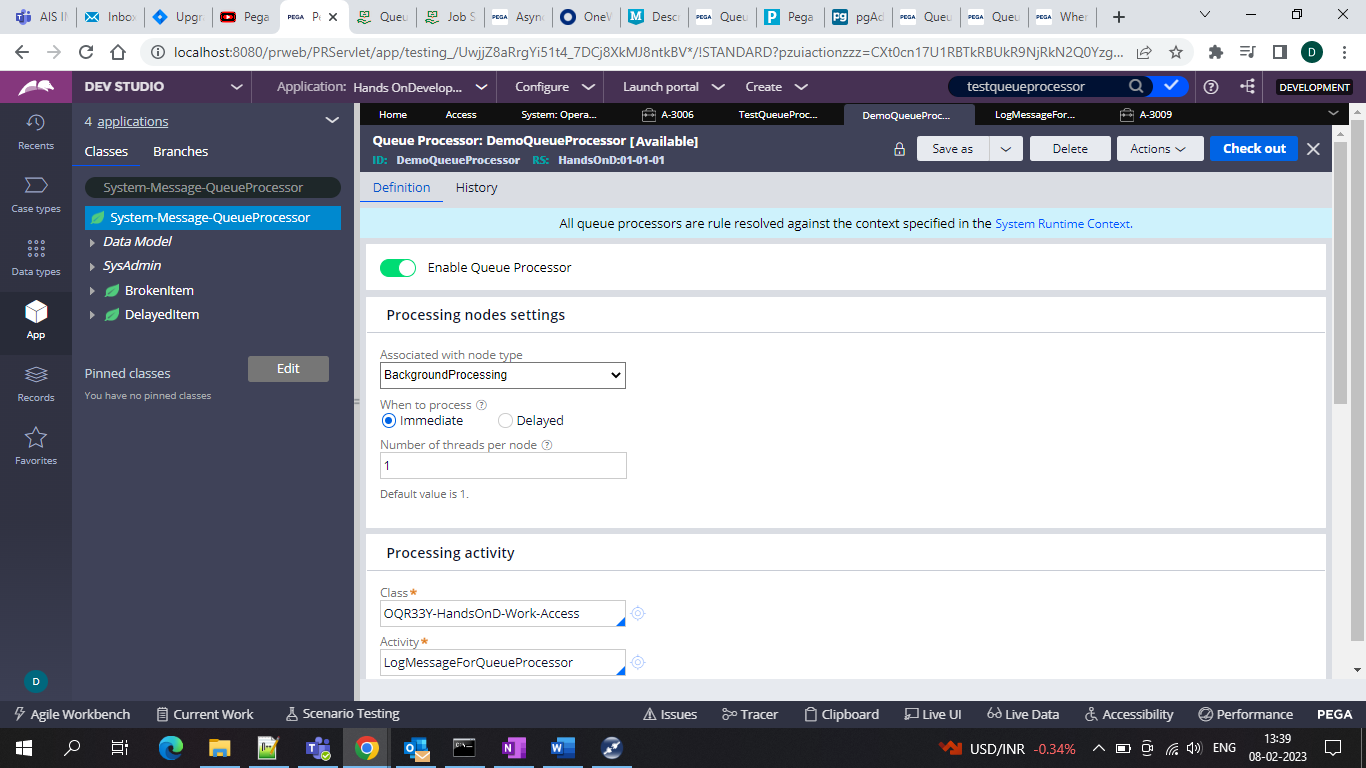
Standard if we select we would be having the pzStandardQueueProcessing which would handle all and we would have the default activity which would be overwritten if we specify your activity.

When we selected the delayed we need to give date and time

Now first I am creating a Queue Processor from the designer studio

and then give the following when to process is immediate or delayed

and number of threads per node means on one node how many work items can be processed if we give one node then the usage for the one node would be one work item



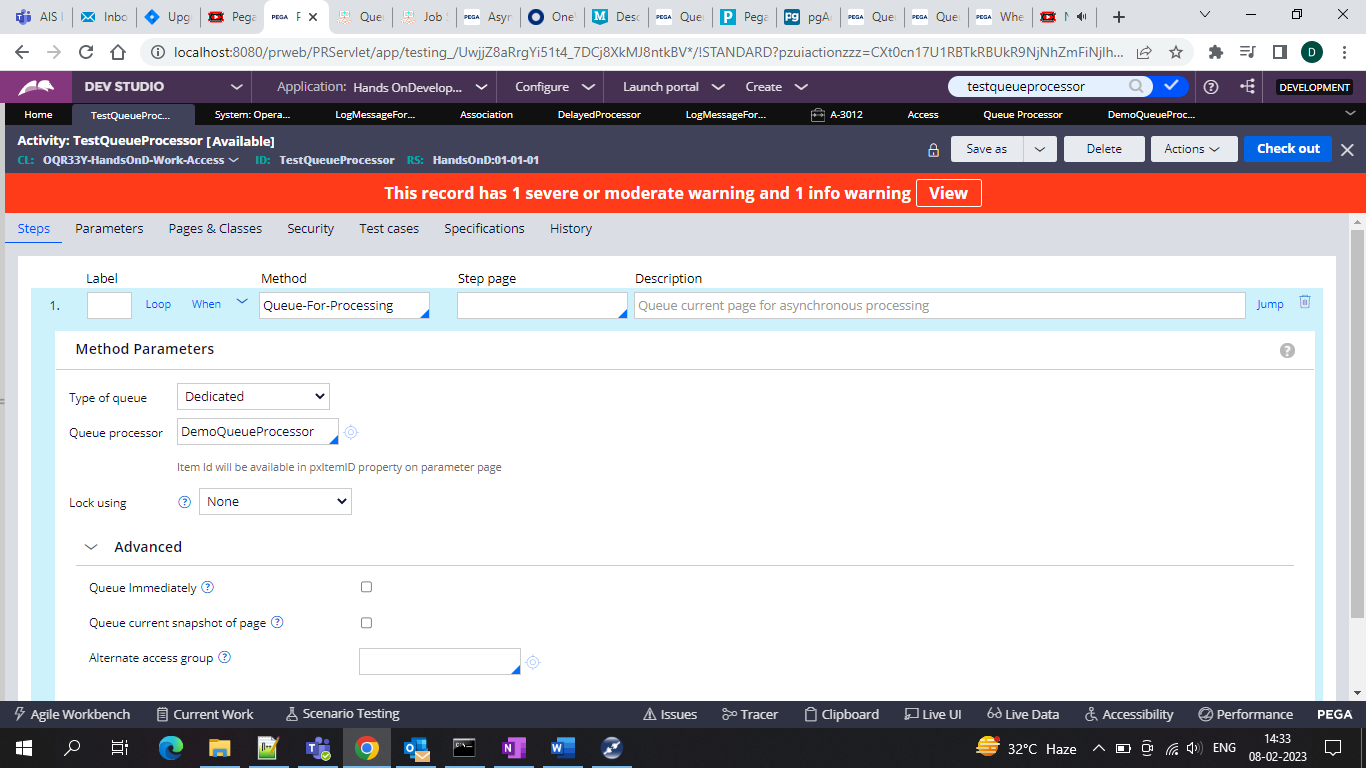
And specify the class and the activity same as job scheduler and in the max attempts if we have them as 3 that means after 3 attempsts they will go into the broken queue

And in the initial delay in minutes means how much delayed it should be queued and processed after the specified time.

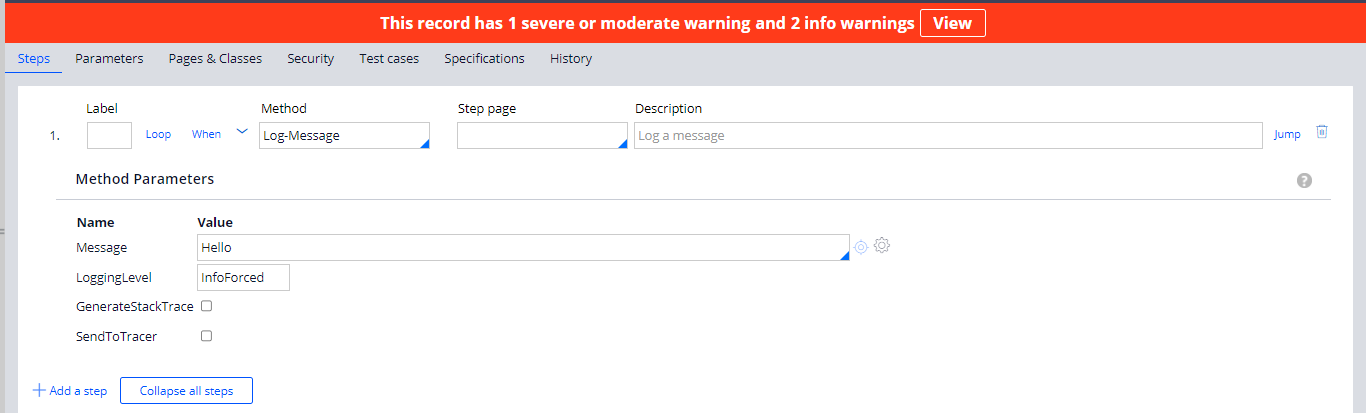
And the delay factor is coming into picture based on the attempts and then the delay factor Suppose 1 st attempt is failed with delay factor 2 then 2 nd attempt delay factor would be 4 then again for the 3 rd time 4 \* 2= 8 would be the 3 rd time this is how delay factor will work.

And that’s it we can now use the activity to publish Log-Messages through queue processor

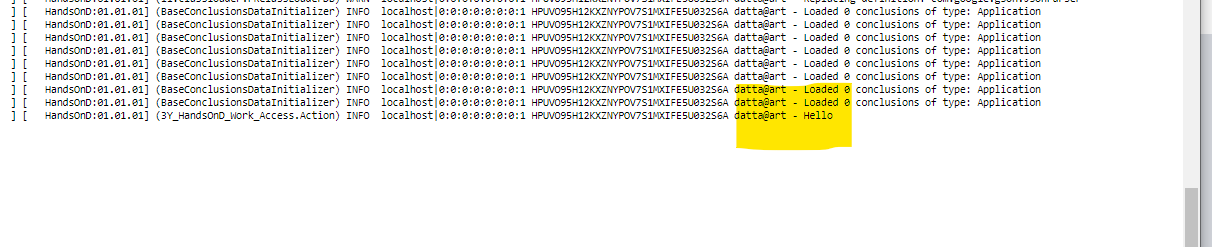
So I am using queue-for-processing method and when we define it right as the type of queue as dedicated and queue processor and lock I did not use the anything and save the activity and we can run the activity



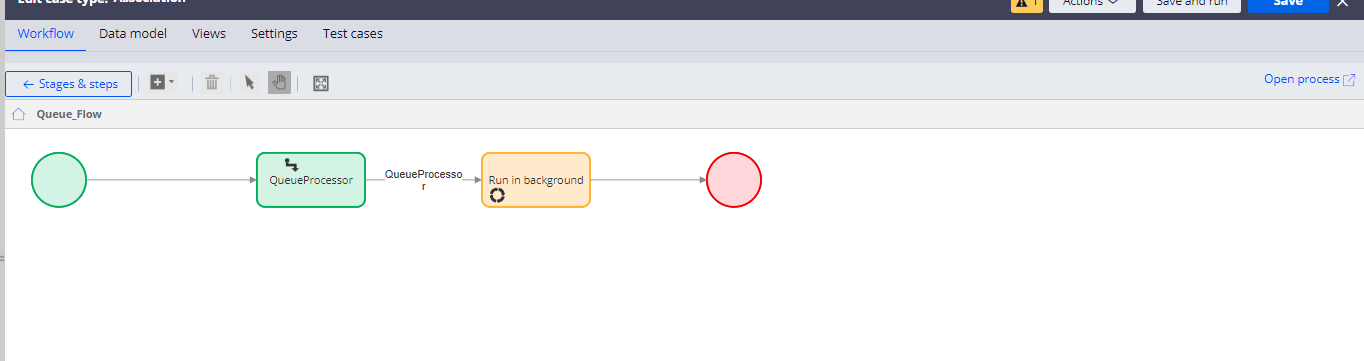
And in that we can now run the activity and see the log message In the logs since in the queue processor I gave the activity as the log message



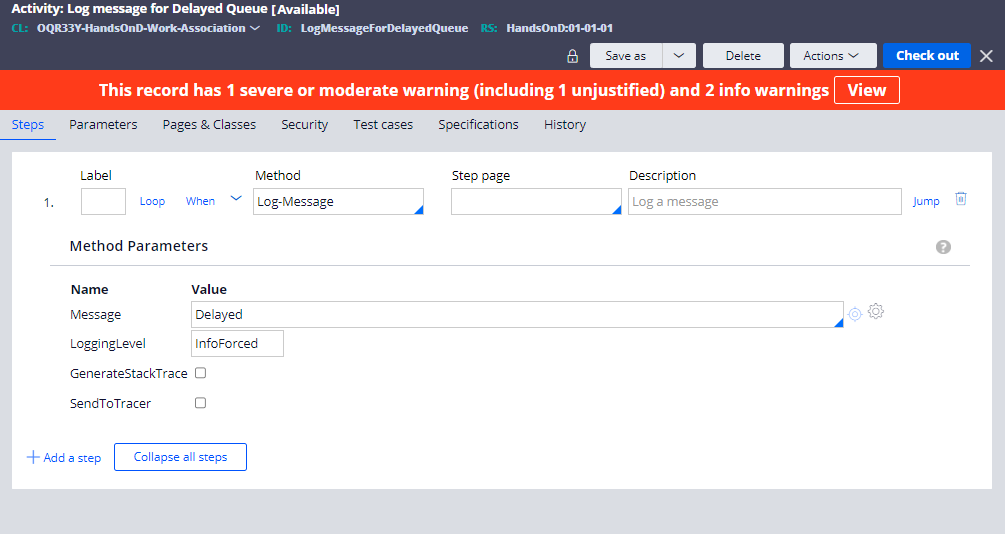
And then we can get the log message when we run the Activity and then check in the system operations > logs and then we can see it



And now for the main things we would be doing the delayed queue and then we would be having the flow shape as run in background and then



In that give your queue processor because it comes to the delayed queue and then we can process it so in the queue processor we would get the delayed queue we would see the instance and for this we did the activity is same as the log-message and then we can use the activity in the log message and then I changed message and then we can give the activity in the flow



And then we will define the activity in the queue processor in the flow in the background processing

