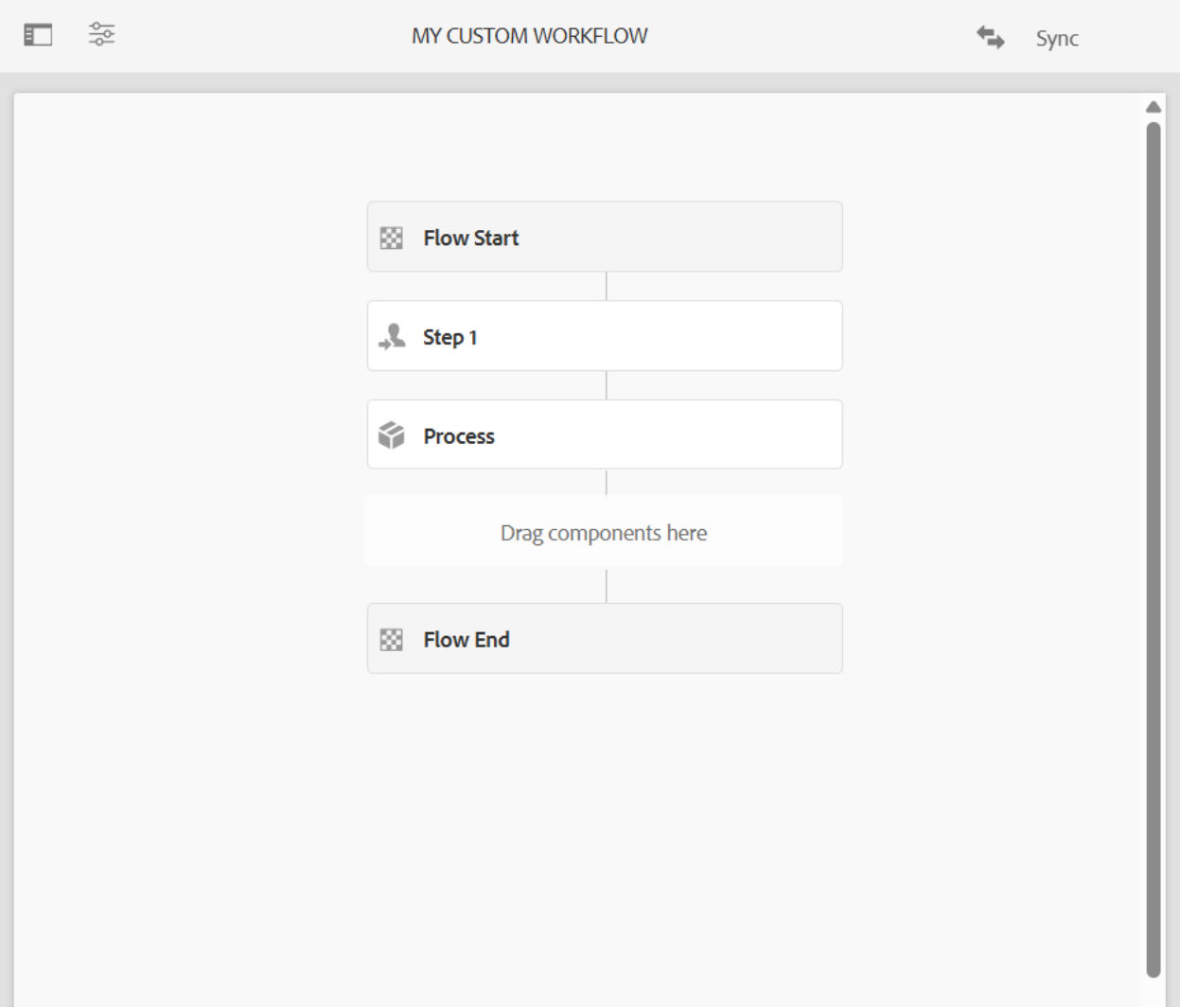
Assignment:

1.Create Custom Workflow (my custom workflow)



2.Create custom workflow process and print the page title in logs and run this workflow in page so that it can give some metadata in logs

```
package com.example.core.workflow;


import com.adobe.granite.workflow.exec.*;

import com.adobe.granite.workflow.metadata.MetaDataMap;

import com.day.cq.wcm.api.Page;
```

```
import org.apache.sling.api.resource.*;
import org.osgi.service.component.annotations.Component;
import org.osgi.framework.Constants;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

@Component(service = WorkflowProcess.class,
        property = {
           Constants.SERVICE_DESCRIPTION + "=Print Page Title Workflow Process",
           "process.label=Print Page Title"
        })
public class PrintPageTitleProcess implements WorkflowProcess {

   private static final Logger log = LoggerFactory.getLogger(PrintPageTitleProcess.class);

   @Override
   public void execute(WorkItem item, WorkflowSession session, MetaDataMap args) {
      String payloadPath = item.getWorkflowData().getPayload().toString();
      log.info("Payload Path: {}", payloadPath);

      try (ResourceResolver resolver = session.adaptTo(ResourceResolver.class)) {
         Resource resource = resolver.getResource(payloadPath);
         if (resource != null) {
            Page page = resource.adaptTo(Page.class);
            if (page != null) {
               log.info("Page Title: {}", page.getTitle());
```

```
            }
        }
    } catch (Exception e) {
        log.error("Error fetching page title", e);
    }
  }
}
```

3.Create Event handler in aem and print the resource path in logs.

```
package com.example.core.listeners;

import org.apache.sling.api.SlingConstants;
import org.apache.sling.api.resource.Resource;
import org.osgi.service.component.annotations.Component;
import org.osgi.service.event.EventConstants;
import org.osgi.service.event.EventHandler;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.osgi.service.event.Event;

@Component(
  service = EventHandler.class,
  immediate = true,
  property = {
    EventConstants.EVENT_TOPIC + "=" + SlingConstants.TOPIC_RESOURCE_ADDED
  }
```

```
)
public class ResourceAddedEventHandler implements EventHandler {

   private static final Logger log =
LoggerFactory.getLogger(ResourceAddedEventHandler.class);

   @Override
   public void handleEvent(Event event) {
      String path = (String) event.getProperty(SlingConstants.PROPERTY_PATH);
      log.info("Resource added at: {}", path);
   }
}
```

4.create sling job to print hello world message in logs

```
package com.example.core.jobs;

import org.apache.sling.event.jobs.Job;
import org.apache.sling.event.jobs.consumer.JobConsumer;
import org.osgi.service.component.annotations.Component;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

@Component(
   service = JobConsumer.class,
   immediate = true,
   property = {
```

```
        JobConsumer.PROPERTY_TOPICS + "=com/example/jobs/helloworld"
    }
)
public class HelloWorldJob implements JobConsumer {

    private static final Logger log = LoggerFactory.getLogger(HelloWorldJob.class);

    @Override
    public JobResult process(Job job) {
        log.info("Hello World from Sling Job!");
        return JobResult.OK;
    }
}
```

5. Create one schedular to print the yellow world in logs in every 5 mins through custom configuration using cron expression.

```
package com.example.core.schedulers;

import org.osgi.service.component.annotations.*;
import org.osgi.service.metatype.annotations.*;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

@Component(service = Runnable.class,
        immediate = true,
        configurationPolicy = ConfigurationPolicy.REQUIRE,
        configurationPid = "com.example.core.schedulers.YellowWorldScheduler")
```

```
@Designate(ocd = YellowWorldScheduler.Config.class)
public class YellowWorldScheduler implements Runnable {

  private static final Logger log = LoggerFactory.getLogger(YellowWorldScheduler.class);

  @ObjectClassDefinition(name = "Yellow World Scheduler Config")
  public @interface Config {
    @AttributeDefinition(name = "Cron Expression", defaultValue = "0 0/5 * * * ?")
    String scheduler_expression();
  }
  private String cronExpression;
  @Activate
  @Modified
  protected void activate(Config config) {
    this.cronExpression = config.scheduler_expression();
  }
  @Override
  public void run() {
    log.info("Yellow World from Scheduler!");
  }
}
```

```
2025-04-09 20:20:00.123  INFO 12345 --- [   scheduling-1] YellowWorldSchedul
2025-04-09 20:25:00.456  INFO 12345 --- [   scheduling-1] YellowWorldSchedul
2025-04-09 20:30:00.789  INFO 12345 --- [   scheduling-1] YellowWorldSchedul
```