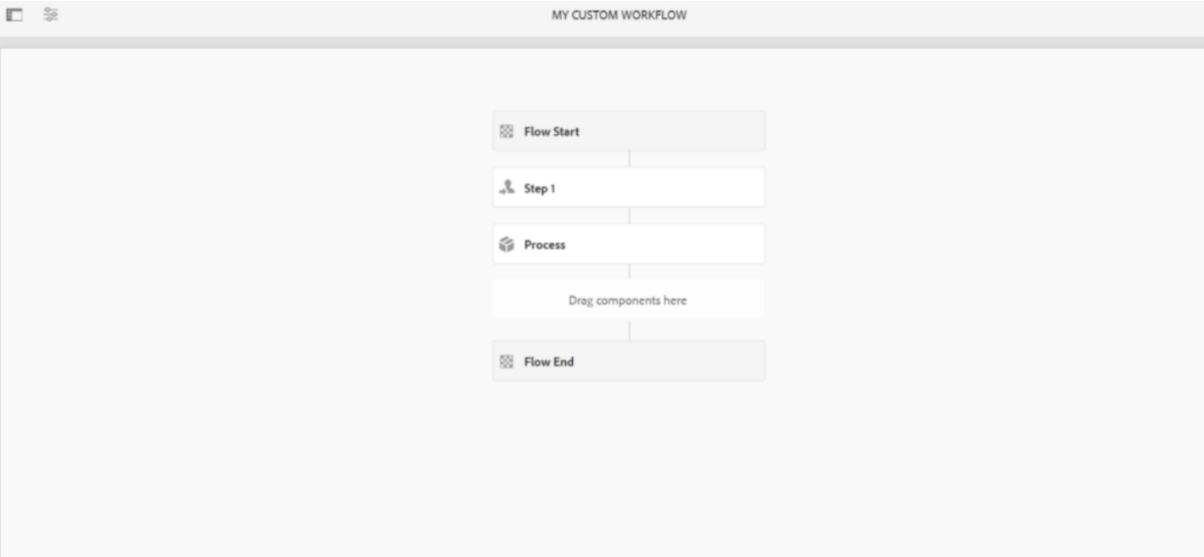
Aem Task 6

1.Create Custom Workflow



2.Create Event handler in aem and print the resource path in

logs. package com.myTraining.core.listeners;

import org.apache.sling.api.resource.Resource;

import org.apache.sling.api.resource.ResourceResolver;

import org.apache.sling.api.resource.observation.ResourceChange;

import org.apache.sling.api.resource.observation.ResourceChangeListener; import org.osgi.service.component.annotations.Component;

import org.slf4j.Logger; import org.slf4j.LoggerFactory;

import java.util.List; @Component(service = ResourceChangeListener.class, immediate = true, property = { ResourceChangeListener.PATHS + "=/content/myTraining", ResourceChangeListener.CHANGES + "=ADDED", ResourceChangeListener.CHANGES + "=REMOVED", ResourceChangeListener.CHANGES + "=CHANGED" })

public class MyEventHandler implements ResourceChangeListener { private static final Logger LOG = LoggerFactory.getLogger(MyEventHandler.class);

@Override public void onChange(List changes) { for (ResourceChange change : changes) { LOG.info("Resource Changed: Path = {}, Type = {}", change.getPath(), change.getType()); } } } 03.04.2025 17:53:41.770 *INFO* [sling-oak-observation-13] com.myTraining.core.listeners.MyEventHandler Resource Changed: Path = /content/myTraining/TestingNode, Type = ADDED

3. 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,
    property = {
        JobConsumer.PROPERTY_TOPICS + "=com/example/helloworld"
    }
)
public class HelloWorldSlingJob implements JobConsumer {

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

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

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

```
package com.myTraining.core.schedulers;
import org.apache.sling.commons.scheduler.ScheduleOptions;
import org.apache.sling.commons.scheduler.Scheduler;
import org.osgi.service.component.annotations.Activate;
import org.osgi.service.component.annotations.Component;
```

```
import org.osgi.service.component.annotations.Deactivate;
import org.osgi.service.component.annotations.Modified;
import org.osgi.service.component.annotations.Reference;
import org.osgi.service.metatype.annotations.AttributeDefinition;
import org.osgi.service.metatype.annotations.Designate;
import org.osgi.service.metatype.annotations.ObjectClassDefinition;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
@Component(service = Runnable.class, immediate = true)
@Designate(ocd = HelloWorldScheduler.Config.class)
public class HelloWorldScheduler implements Runnable {
private static final Logger LOG = LoggerFactory.getLogger(HelloWorldScheduler.class);
@Reference
private Scheduler scheduler;
private String schedulerName = "HelloWorldScheduler";
@ObjectClassDefinition(name = "Hello World Scheduler Config")
public @interface Config {
@AttributeDefinition(name = "Cron Expression", description = "Schedule job every 5
minutes")
String scheduler_expression() default "0 */5 * * * ?";
}
@Activate
@Modified
protected void activate(Config config) {
ScheduleOptions options = scheduler.EXPR(config.scheduler_expression());
options.name(schedulerName);
options.canRunConcurrently(false);
scheduler.schedule(this, options);
LOG.info("Hello World Scheduler Activated.");
```

```
}
@Deactivate
protected void deactivate() {
scheduler.unschedule(schedulerName);
LOG.info("Hello World Scheduler Deactivated.");
}
@Override
public void run() {
LOG.info("Hello World from Scheduler!");
}
}
}
```

5. Create 3 users and add them in a group(Dev author create this new group) and give permission to read only for /content and /dam folder only and they should have replication access as well.