OBLM EOD Configuration Guide

Oracle Banking Liquidity Management

Release 14.5.0.0.0

Part Number F41688-01

May 2021

OBLM EOD Configuration Guide

Oracle Financial Services Software Limited
Oracle Park
Off Western Express Highway
Gurgaon (East)
Mumbai, Maharashtra 400 063
India

Worldwide Inquiries:
Phone: +91 22 6718 3000
Fax: +91 22 6718 3001
https://www.oracle.com/industries/financial-services/index.html

Contents

1 Preface

1.1 Background

Oracle Banking Liquidity Management allows you to execute several functions every day on a routine basis as part of the End of Day (EOD) process. These functions can be run at various stages of the EOD process.

1.2 Introduction

The End of Day process is to tie up all the operations for a financial day and prepare the system for the next day. The EOD process should be defined for a branch and executed separately for each branch. When the process is running, you could choose to monitor it from Invoke EOD screen.

EOD uses OBMA Orchestrator and Batch service for orchestrating all the jobs required to complete End of Day processing. This document helps in the required set up to run EOD.

1.3 Document Accessibility

1. OBMA Orchestrator needs to be deployed as per installation guide.
2. OBMA Batch needs to be deployed as per installation guide.

1.4 Acronyms, Abbreviations and Definitions

Acronyms	**Definition**
OBLM	Oracle Banking Liquidity Management
OBMA	Oracle Banking Micro service Architecture
EOD	End of Day

1.5 Related Documents

The related documents are as follows:

- Oracle Banking Common Core User Guide
- Oracle Banking Liquidity Management User Guide

2 EOD configuration Steps

The following Functional Activities needs to be maintained in user’s role to perform EOD operations

CMC_FA_BRANCH_EOD_PROCESS

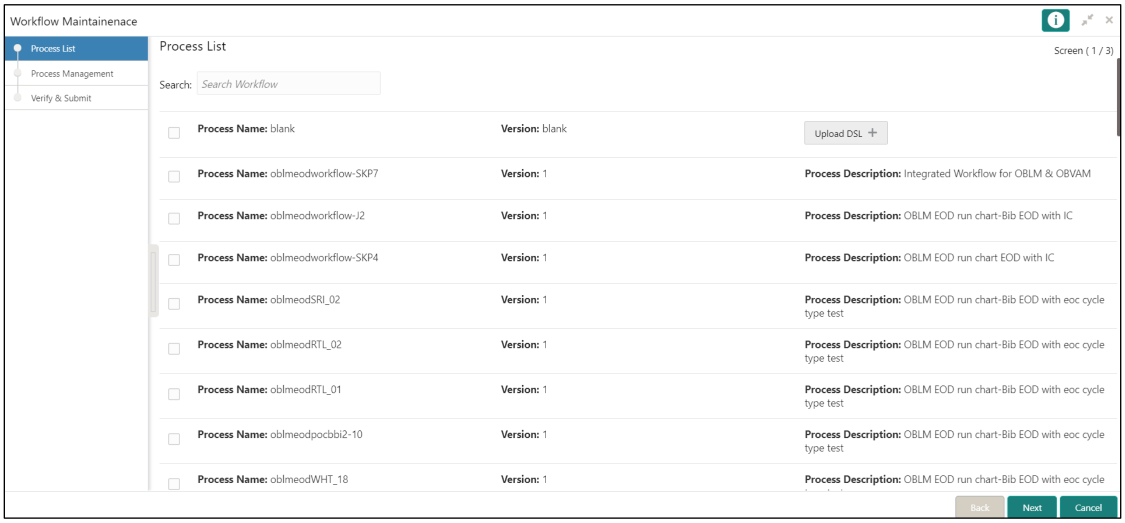
1. Save the below attachment to local folder. This is a standard batch process definition script for OBLM that includes the list of batch tasks to be automatically executed in a sequence.

   OBLMEod.json

2. On **Home Screen**, under **Tasks** menu, click **Business Process Maintenance** to import, create or modify batch process definition.

   → **Product List** screen is displayed.

Figure 1: Process List



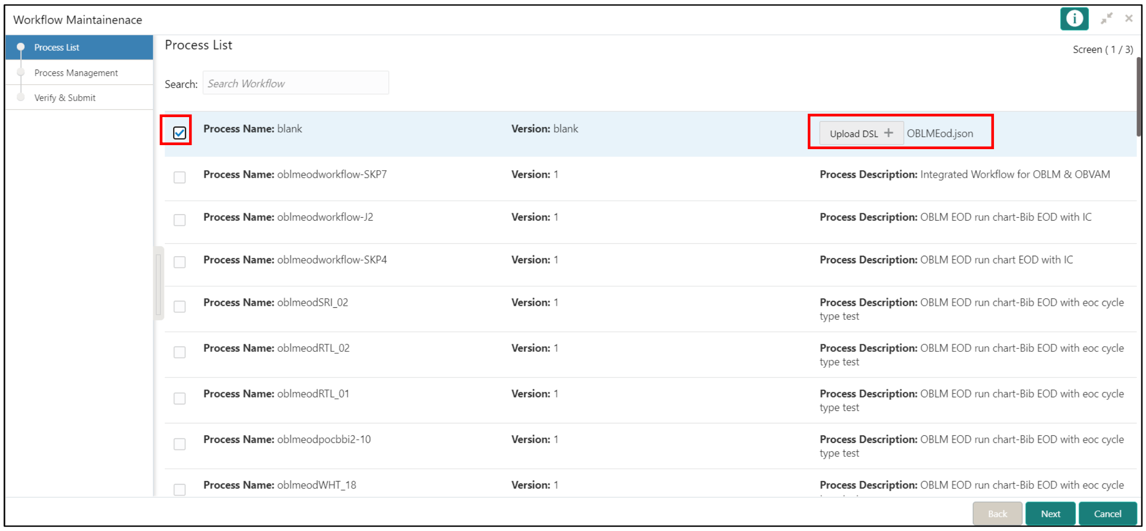
Process List

Search: Search Workflow

□	**Process Name:** blank	**Version:** blank	
□	**Process Name:** oblmeodworkflow-SKP7	**Version:** 1	**Process Description:** Integrated Workflow for OBLM & OBVAM
□	**Process Name:** oblmeodworkflow-J2	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with IC
□	**Process Name:** oblmeodworkflow-SKP4	**Version:** 1	**Process Description:** OBLM EOD run chart EOD with IC
□	**Process Name:** oblmeodSRI_02	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle type test
□	**Process Name:** oblmeodRTL_02	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle type test
□	**Process Name:** oblmeodRTL_01	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle type test
□	**Process Name:** oblmeodpocbbi2-10	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle type test
□	**Process Name:** oblmeodWHT_18	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle

3. Select the **Process Name: blank** checkbox. Click on **Upload DSL+** button and choose file **OBLMEod.json** from local folder.

Figure 2: Process List – Upload DSL



Workflow Maintainenace

Process List

Screen (1 / 3)

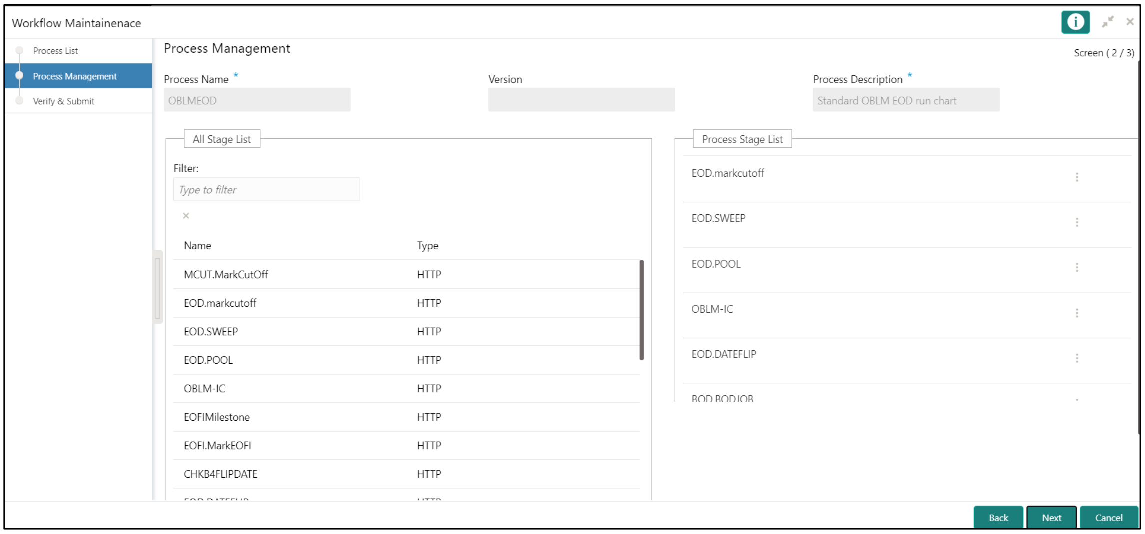
Search: Search Workflow

☑	**Process Name:** blank	**Version:** blank	Upload DSL + OBLMEod.json
□	**Process Name:** oblmeodworkflow-SKP7	**Version:** 1	**Process Description:** Integrated Workflow for OBLM & OBVAM
□	**Process Name:** oblmeodworkflow-J2	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with IC
□	**Process Name:** oblmeodworkflow-SKP4	**Version:** 1	**Process Description:** OBLM EOD run chart EOD with IC
□	**Process Name:** oblmeodSRI_02	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle type test
□	**Process Name:** oblmeodRTL_02	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle type test
□	**Process Name:** oblmeodRTL_01	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle type test
□	**Process Name:** oblmeodpocbbi2-10	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle type test
□	**Process Name:** oblmeodWHT_18	**Version:** 1	**Process Description:** OBLM EOD run chart-Bib EOD with eoc cycle

4. Click **Next** button

→ **Product Management** screen is displayed.

Figure 3: Process Management



Workflow Maintainenace

Process Management

Screen (2 / 3)

Process Name *: OBLMEOD

Version:

Process Description *: Standard OBLM EOD run chart

All Stage List

Filter: Type to filter

Name	Type
MCUT.MarkCutOff	HTTP
EOD.markcutoff	HTTP
EOD.SWEEP	HTTP
EOD.POOL	HTTP
OBLM-IC	HTTP
EOFIMilestone	HTTP
EOFI.MarkEOFI	HTTP
CHKB4FLIPDATE	HTTP

Process Stage List

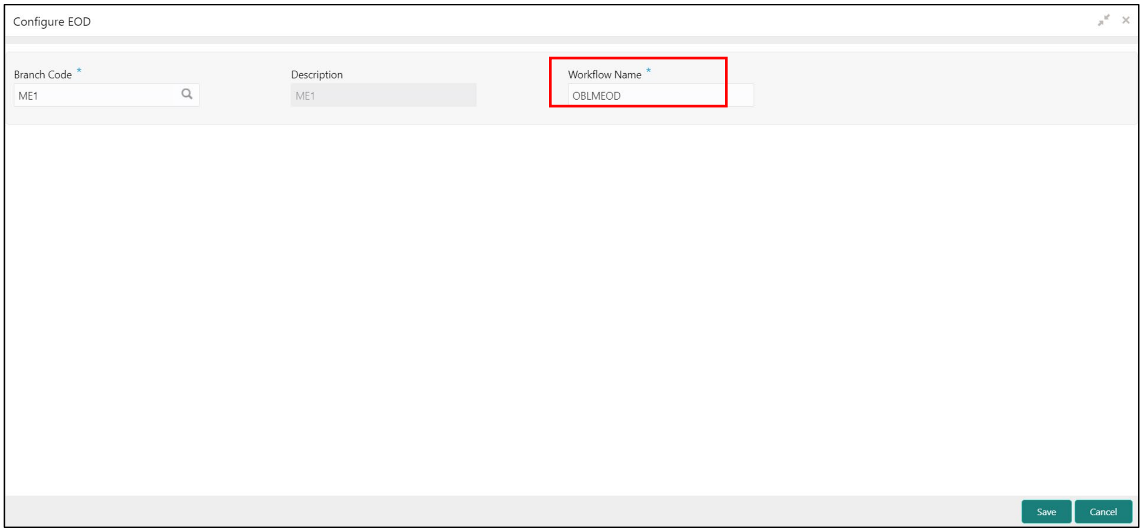
- EOD.markcutoff
- EOD.SWEEP
- EOD.POOL
- OBLM-IC
- EOD.DATEFLIP
- BOD.BODJOB

5. Click **Next** button and Click **Review** or **Create Process** in **Verify & Submit** screen to register the batch.

6. On **Core Maintenance** menu, under **Branch EOD**, click **Configure EOD** to configure batch for a branch. Refer **Section 2.5** in **Oracle Banking Common Core User Guide**.

→ **Configure EOD** screen is displayed.

Figure 4: Configure EOD



Configure EOD

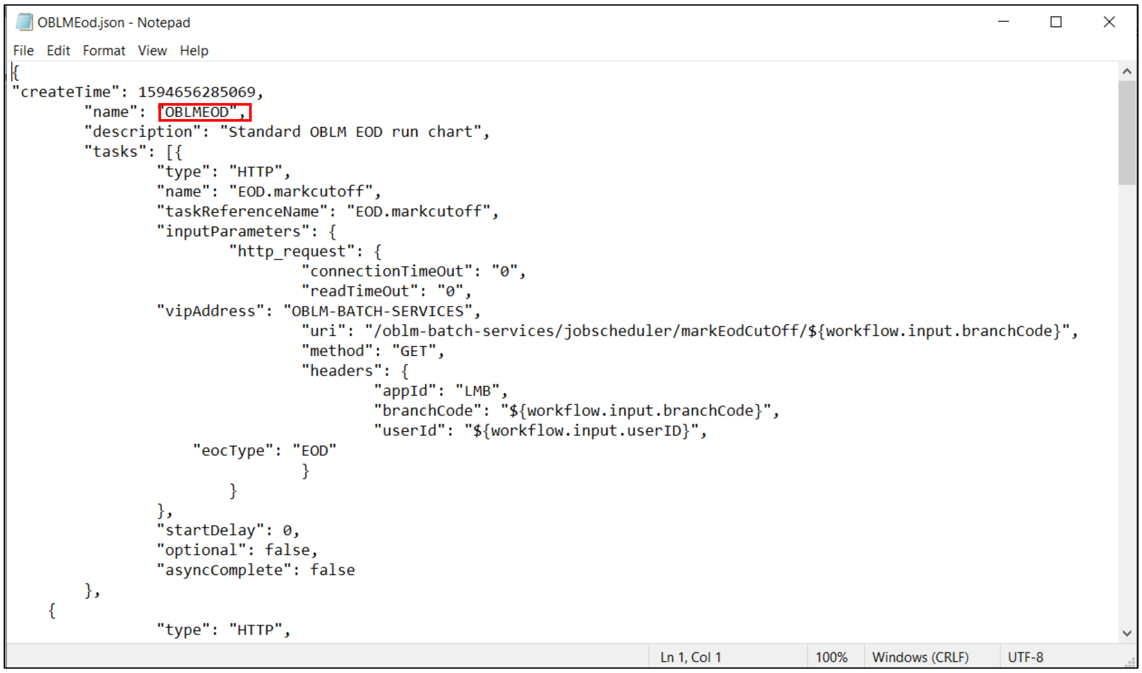
Branch Code * ME1

Description ME1

Workflow Name * OBLMEOD

7. Select branch code to configure the batch.

Note: The value specified in Workflow name field must be same as the workflow **name** attribute specified in 3rd line of batch script **OBLMEod.json** file



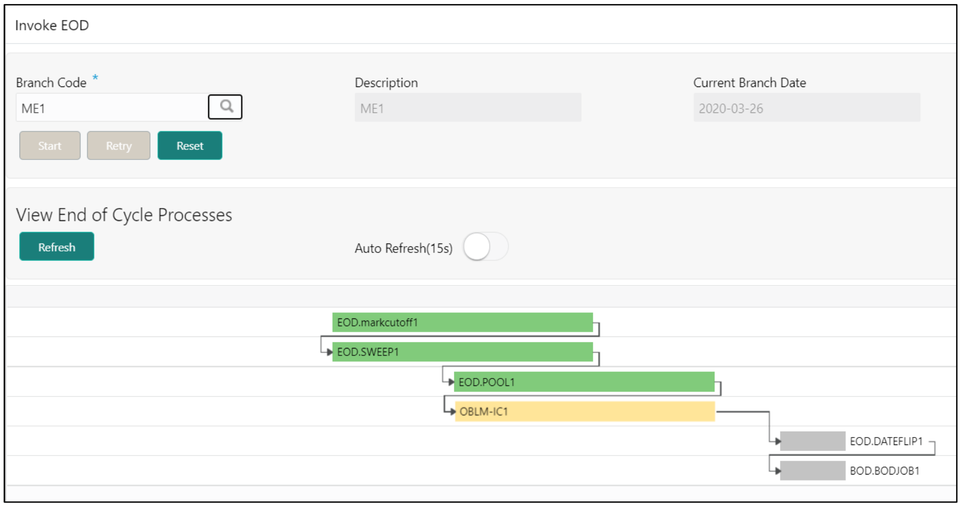
```
{
"createTime": 1594656285069,
        "name": "OBLMEOD",
        "description": "Standard OBLM EOD run chart",
        "tasks": [{
                "type": "HTTP",
                "name": "EOD.markcutoff",
                "taskReferenceName": "EOD.markcutoff",
                "inputParameters": {
                        "http_request": {
                                "connectionTimeOut": "0",
                                "readTimeOut": "0",
                "vipAddress": "OBLM-BATCH-SERVICES",
                                "uri": "/oblm-batch-services/jobscheduler/markEodCutOff/${workflow.input.branchCode}",
                                "method": "GET",
                                "headers": {
                                        "appId": "LMB",
                                        "branchCode": "${workflow.input.branchCode}",
                                        "userId": "${workflow.input.userID}",
                    "eocType": "EOD"
                                }
                        }
                },
                "startDelay": 0,
                "optional": false,
                "asyncComplete": false
        },
    {
                "type": "HTTP",
```

2.1 Steps to run EOD for a branch

1. On **Core Maintenance** menu, under **Branch EOD**, click **Invoke EOD**.

   → **Invoke EOD** screen is displayed.

   Figure 5: Invoke EOD



Invoke EOD

Branch Code * ME1

Description ME1

Current Branch Date 2020-03-26

View End of Cycle Processes

2. Select the branch to run EOD.

3. Click **Refresh** to view the current status of branch.

3 Job definition Naming Convention

Naming convention to be followed when a custom job is introduced as a task into EOD process

1. Milestone task name must be prefixed with “MS-“. Ex: MS-BranchCutOff

   Milestone stage

   Milestone stage will pause the batch execution till it is manually resumed.

   Sample template for milestone stage

```
{
    "name": "MS-CHKAFTEREOTI",
    "taskReferenceName": "MS-CHKAFTEREOTI",
    "inputParameters": {
        "http_request": {
            "connectionTimeOut": "0",
            "readTimeOut": "0",
            "vipAddress": "CMC-BRANCH-SERVICES",
            "uri": "/cmc-branch-services/brancheod/milestone",
            "method": "POST",
            "headers": {
                "appId": "CMNCORE",
                "branchCode": "${workflow.input.branchCode}",
                "userId": "${workflow.input.userID}"
            },
            "body": {
                "data": [
                    {
                        "workflowId": "${workflow.workflowId}",
                        "taskId": "${CPEWF_TASK_ID}",
                        "waitTime": "5000"
                    }
                ]
            }
        }
    },
    "type": "HTTP",
    "startDelay": 0,
    "optional": false,
    "asyncComplete": true
}
```

Steps to integrate Custom Jobs

1. If the custom job uses OBMA Batch service, then use the below template to include the job as a task in EOD Flow definition.

```
{
  "type":"HTTP",
  "name":"<MilestoneCode.JobName>",
  "taskReferenceName":"<MilestoneCode.JobName>",
  "inputParameters":{
    "http_request":{
      "connectionTimeOut":"0",
      "readTimeOut":"0",
      "vipAddress":"PLATO-BATCH-SERVER",
      "uri":"/plato-batch-server/jobLauncher/launch/",
      "method":"POST",
      "headers":{
        "appId":"${workflow.input.appId}",
        "branchCode":"${workflow.input.branchCode}",
        "userId":"${workflow.input.userId}"
      },
      "body":{
        "jobName":"<JobName>",
        "jobParameters":[
          {
            "key":"appId",
            "value":"<Application ID of microservice>"
          },
          {
            "key":"microServiceName",
   "value":"<Microservice name>"
},
{
   "key":"contextRoot",
   "value":"<Context root of microservice>"
},
{
   "key":"workflowId",
   "value":"${workflow.workflowId}"
},
{
   "key":"referenceTaskName",
   "value":"<MilestoneCode.JobName>"
},
{
   "key":"userId",
   "value":"${workflow.input.userId}"
},
{
   "key":"branchCode",
   "value":"${workflow.input.branchCode}"
},
{
   "key":"isCallback",
   "value":"Y"
},
{
   "key":"callbackType",
            "value":"PLATOORCH"
          }
        ]
      }
    },
    "asyncComplete":true
  },
  "startDelay":0,
  "optional":false,
  "asyncComplete":true
}
```

2. If the custom job doesn’t uses OBMA Batch service and the Batch API is implemented as a synchronous call, then use the below template to include the job as a task in EOD Flow definition

```
{
  "type":"HTTP",
  "name":"<MilestoneCode.JobName>",
  "taskReferenceName":"<MilestoneCode.JobName>",
  "inputParameters":{
    "http_request":{
      "connectionTimeOut":"0",
      "readTimeOut":"0",
      "vipAddress":"<Microservice name registered in eureka>",
      "uri":"<relative URL>",
      "method":"<HTTP Method>",
      "headers":{
        "appId":"${workflow.input.appId}",
        "branchCode":"${workflow.input.branchCode}",
        "userId":"${workflow.input.userId}"
      }
    },
    "asyncComplete":false
  },
  "startDelay":0,
  "optional":false,
  "asyncComplete":true
}
```

NOTE: HTTP Method - One of the GET, PUT, POST, DELETE, OPTIONS, HEAD

3. If the custom job doesn't uses OBMA Batch service and if the Batch API is implemented as an asynchronous call, then call back needs to be implemented in the respective API. Please use the below template to include the job as a task in EOD Flow Definition.

```
{
  "type":"HTTP",
  "name":"<MilestoneCode.JobName>",
  "taskReferenceName":"<MilestoneCode.JobName>",
  "inputParameters":{
    "http_request":{
      "connectionTimeOut":"0",
      "readTimeOut":"0",
      "vipAddress":"<Microservice name registered in eureka>",
      "uri":"<relative URL>",
      "method":"<HTTP Method>",
      "headers":{
        "appId":"${workflow.input.appId}",
        "branchCode":"${workflow.input.branchCode}",
        "userId":"${workflow.input.userId}"
      }
    },
    "asyncComplete":true
  },
  "startDelay":0,
  "optional":false,
  "asyncComplete":true
}
```

The following API should be used as a call back to update the status of a task.

Method	**POST**	
Url	http://<hostname>:<port>/plato-orch-service/api/tasks	
Headers	userId : <Logged in user id> branchCode : <Logged in branch code> appId : platoorch Content-Type : application/json Accept : application/json	userId – User who updates the task branchCode – Branch where the update is performed

Body	{
	"workflowInstanceId": "<EOD_Workflow_ID",
	"taskId": "<Task_ID>",
	"status": "<Status>"
	}
		EOD_Workflow_ID – A Workflow ID gets generated when EOD is invoked
		Task_ID – Unique task ID gets generated for each task once it starts
		Status – COMPLETED / FAILED_WITH_TERMINAL_ERROR / FAILED / IN_PROGRESS

NOTE: asyncComplete – field in EOD workflow definition should be set to true if the Http task makes an asynchronous call and the task has to be updated explicitly by calling above update APIs. Only after successful update, next task will get executed.

4 OBLM Job

S No	EOD stage	Job Name	Description	Input Parameters
1	MCUT	markcutoff	Job will check for pending tasks and any existing running process before starting EOD	BranchCode
	MS-EOD	MS-EOD	Milestone stage for running EOD	
2	EOD	SWEEP	Job will execute sweep process scheduled to run during EOD	BranchCode
3	EOD	POOL	Job will execute all pool structures.	BranchCode
4	EOD	IC	Job will execute Interest batch	BranchCode
5	EOD	DATEFLIP	Job will change system date to next working date	BranchCode
6	MS-BOD	MS-BOD	Milestone stage for running BOD	BranchCode
6	BOD	BOD	Job will execute Reallocation, Reverse sweep and bod sweeps in sequence	BranchCode