Water Service - Technical Document

Water Services

This is the egov application, which helps and gives flexibility to municipal and citizen to handle the water service like apply for water connection, search water connection. The application will go on different stages that state can verify by given roles, If that role can take the action then the application will go to the next state. Based on the state, citizen will get notification (SMS and in-app). Citizens also can pay for application fees or employees can collect the fee for the application.
The setup of the Application is as follows

Requirements

- Knowledge of Java/J2EE(preferably Java 8 version)
- Knowledge of Spring Boot and spring-boot microservices.
- Knowledge of Git or any version control system.
- Knowledge of RESTful Web services.
- Knowledge of the Lombok library will helpful.
- knowledge of eGov-mdms service, eGov-persister, eGov-idgen, eGov-sms, eGov-email,eGov-user, eGov-localization, eGov-workflow-service will be helpful.

Functionality

1. Apply for water connection.
2. Searching for water connections.
3. Apply for Property creation.
4. Can take different action based on state (Workflow)
5. Notification based on the application state.

Setup and usage

The **Application** is present among the ***municipal services*** group of applications available in the eGov-services git repository. The spring boot application needs the **Lombok*** extension added in your IDE to load it. Once the application is up and running API requests can be posted to the URL and ids can be generated.
in case of IntelliJ, the plugin can be installed directly, for eclipse the Lombok jar location has to be added in eclipse.ini file in this format javaagent:lombok.jar

API Information

Please refer to Swagger API for YAML file details. Link - https://github.com/egovernments/municipal-services/blob/RAIN-2317/docs/water-sewerage-services.yaml

Application.properties File Information

kafka topics persister configs for eGov persister
egov.waterservice.createwaterconnection = save-ws-connection
egov.waterservice.updatewaterconnection=update-ws-connection
egov.waterservice.updatewaterconnection.workflow.topic=update-ws-workflow

URLs for the external API references

eGvo mdms :-> egov.mdms.host = https://egov-micro-dev.egovernments.org/
eGov -idGen :-> egov.idgen.host = https://egov-micro-dev.egovernments.org/
localization service :-> egov.localization.host = https://egov-micro-dev.egovernments.org/
workflow-service:-> egov.workflow.host = https://egov-micro-dev.egovernments.org/

idGen Id formats :->
egov.idgen.wcid.name=waterservice.connection.id
egov.idgen.wcid.format=WS/[CITY.CODE]/[fy:yyyy-yy]/[SEQ_EGOV_COMMON]
egov.idgen.wcapid.name=waterservice.application.id
egov.idgen.wcapid.format=WS_AP/[CITY.CODE]/[fy:yyyy-yy]/[SEQ_EGOV_COMMON]

Configuration

Mdms configuration

https://github.com/egovernments/egov-mdms-data/tree/master/data - Connect to preview

```
ws-services-calculation
ws-services-masters
PropertyTax
```

master-config.json for water service

```
"ws-services-masters": {
  "connectionCategory": {
    "masterName": "connectionCategory",
    "isStateLevel": true,
    "uniqueKeys": [
      "$.code"
    ]
  },
  "connectionType": {
    "masterName": "connectionType",
    "isStateLevel": true,
    "uniqueKeys": [
      "$.code"
    ]
  },
  "waterSource": {
    "masterName": "waterSource",
    "isStateLevel": true,
    "uniqueKeys": [
      "$.code"
    ]
  },
  "billingPeriod": {
    "masterName": "billingPeriod",
    "isStateLevel": true,
    "uniqueKeys": [
      "$.billingCycle"
    ]
  },
  "waterSourceWithSubSource": {
    "masterName": "waterSourceWithSubSource",
    "isStateLevel": true,
    "uniqueKeys": []
  }
},
"ws-services-calculation": {
  "WaterCess": {
    "masterName": "WaterCess",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "Interest": {
    "masterName": "Interest",
    "isStateLevel": true,
    "uniqueKeys": [
      "$.fromFY"
    ]
  },
  "Rebate": {
    "masterName": "Rebate",
    "isStateLevel": true,
    "uniqueKeys": [
      "$.fromFY"
    ]
  },
  "Penalty": {
    "masterName": "Penalty",
    "isStateLevel": true,
    "uniqueKeys": [
      "$.fromFY"
    ]
  },
  "WCBillingSlab": {
    "masterName": "WCBillingSlab",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "WS_CHARGE": {
    "masterName": "WS_CHARGE",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "WS_TIME_PENALTY": {
    "masterName": "WS_TIME_PENALTY",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "WS_WATER_CESS": {
    "masterName": "WS_WATER_CESS",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "MeterStatus": {
    "masterName": "MeterStatus",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "WS_Round_Off": {
    "masterName": "WS_Round_Off",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "PlotSizeSlab": {
    "masterName": "PlotSizeSlab",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "PropertyUsageType": {
    "masterName": "PropertyUsageType",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "FeeSlab": {
    "masterName": "FeeSlab",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "RoadType": {
    "masterName": "RoadType",
    "isStateLevel": true,
    "uniqueKeys": []
  },
  "CalculationAttribute": {
    "masterName": "CalculationAttribute",
    "isStateLevel": true,
    "uniqueKeys": []
  }
}
```

Property creation through WNS module
https://github.com/egovernments/egov-mdms-data/blob/DEV/data/pb/PropertyTax/PTWorkflow.json

***Persister configuration*:**

water-persist.yml
https://github.com/egovernments/configs/blob/master/egov-persister/water-meter.yml

***Postman link*:**
https://www.getpostman.com/collections/b5b7248d1aeacc9431cb

Workflow business service config

```
{
 "BusinessServices": [
  {
    "tenantId": "pb",
    "businessService": "NewWS1",
    "business": "ws-services",
    "businessServiceSla": 259200000,
    "states": [
      {
        "sla": null,
        "state": null,
        "applicationStatus": null,
        "docUploadRequired": false,
        "isStartState": true,
        "isTerminateState": false,
        "isStateUpdatable": false,
        "actions": [
          {
            "action": "INITIATE",
            "nextState": "INITIATED",
            "roles": [
              "CITIZEN",
              "WS_CEMP"
            ]
          }
        ]
      },
      {
        "sla": null,
        "state": "INITIATED",
        "applicationStatus": "INITIATED",
        "docUploadRequired": false,
        "isStartState": false,
        "isTerminateState": false,
        "isStateUpdatable": true,
        "actions": [
          {
            "action": "SUBMIT_APPLICATION",
            "nextState": "PENDING_FOR_DOCUMENT_VERIFICATION",
            "roles": [
              "CITIZEN",
              "WS_CEMP"
            ]
          }
        ]
      },
      {
        "sla": null,
        "state": "PENDING_FOR_CITIZEN_ACTION",
        "applicationStatus": "PENDING_FOR_CITIZEN_ACTION",
        "docUploadRequired": false,
        "isStartState": false,
        "isTerminateState": false,
        "isStateUpdatable": true,
        "actions": [
          {
            "action": "RESUBMIT_APPLICATION",
            "nextState": "PENDING_FOR_DOCUMENT_VERIFICATION",
            "roles": [
              "CITIZEN",
              "WS_CEMP"
            ]
          }
        ]
      },
      {
        "sla": null,
        "state": "PENDING_FOR_DOCUMENT_VERIFICATION",
        "applicationStatus": "PENDING_FOR_DOCUMENT_VERIFICATION",
        "docUploadRequired": false,
        "isStartState": false,
        "isTerminateState": false,
        "isStateUpdatable": true,
        "actions": [
          {
            "action": "VERIFY_AND_FORWARD",
            "nextState": "PENDING_FOR_FIELD_INSPECTION",
            "roles": [
              "WS_DOC_VERIFIER"
            ]
          },
          {
            "action": "REJECT",
            "nextState": "REJECTED",
            "roles": [
              "WS_DOC_VERIFIER"
            ]
          },
          {
            "action": "SEND_BACK_TO_CITIZEN",
            "nextState": "PENDING_FOR_CITIZEN_ACTION",
            "roles": [
              "WS_DOC_VERIFIER"
            ]
          }
        ]
      },
      {
        "sla": null,
          "state": "REJECTED",
          "applicationStatus": "REJECTED",
          "isStateUpdatable": false,
          "docUploadRequired": false,
          "isStartState": false,
          "isTerminateState": true
        },
        {
          "sla": 86400000,
          "state": "PENDING_FOR_FIELD_INSPECTION",
          "applicationStatus": "PENDING_FOR_FIELD_INSPECTION",
          "docUploadRequired": false,
          "isStartState": false,
          "isStateUpdatable": true,
          "isTerminateState": false,
          "actions": [
            {
              "action": "VERIFY_AND_FORWARD",
              "nextState": "PENDING_APPROVAL_FOR_CONNECTION",
              "roles": [
                "WS_FIELD_INSPECTOR"
              ]
            },
            {
              "action": "REJECT",
              "nextState": "REJECTED",
              "roles": [
                "WS_FIELD_INSPECTOR"
              ]
            },
            {
              "action": "SEND_BACK_FOR_DOCUMENT_VERIFICATION",
              "nextState": "PENDING_FOR_DOCUMENT_VERIFICATION",
              "roles": [
                "WS_FIELD_INSPECTOR"
              ]
            }
          ]
        },
        {
          "sla": 43200000,
          "state": "PENDING_APPROVAL_FOR_CONNECTION",
          "applicationStatus": "PENDING_APPROVAL_FOR_CONNECTION",
          "docUploadRequired": false,
          "isStartState": false,
          "isStateUpdatable": true,
          "isTerminateState": false,
          "actions": [
            {
              "action": "APPROVE_FOR_CONNECTION",
              "nextState": "PENDING_FOR_PAYMENT",
              "roles": [
                "WS_APPROVER"
              ]
            },
            {
              "action": "REJECT",
              "nextState": "REJECTED",
              "roles": [
                "WS_APPROVER"
              ]
            },
            {
              "action": "SEND_BACK_FOR_FIELD_INSPECTION",
              "nextState": "PENDING_FOR_FIELD_INSPECTION",
              "roles": [
                "WS_APPROVER"
              ]
            }
          ]
        },
        {
          "sla": 43200000,
          "state": "PENDING_FOR_PAYMENT",
          "applicationStatus": "PENDING_FOR_PAYMENT",
          "docUploadRequired": false,
          "isStartState": false,
          "isTerminateState": false,
          "isStateUpdatable": false,
          "actions": [
            {
              "action": "PAY",
              "nextState": "PENDING_FOR_CONNECTION_ACTIVATION",
              "roles": [
                "CITIZEN",
                "WS_CEMP"
              ]
            }
          ]
        },
        {
          "sla": null,
          "state": "PENDING_FOR_CONNECTION_ACTIVATION",
          "applicationStatus": "PENDING_FOR_CONNECTION_ACTIVATION",
          "isStateUpdatable": true,
          "docUploadRequired": false,
          "isStartState": false,
          "isTerminateState": false,
          "actions": [
            {
              "action": "ACTIVATE_CONNECTION",
              "nextState": "CONNECTION_ACTIVATED",
              "roles": [
                "WS_CLERK"
              ]
            }
          ]
        },
        {
          "sla": null,
          "state": "CONNECTION_ACTIVATED",
          "applicationStatus": "CONNECTION_ACTIVATED",
          "isStateUpdatable": false,
          "docUploadRequired": false,
          "isStartState": false,
          "isTerminateState": true
        }
      ]
    }
  ]
}
```

Workflow for property creation through Water and Sewerage Module

```
{
  "BusinessServices": [
    {
```

```
    "tenantId": "pb",
    "businessService": "NewWS1",
    "business": "ws-services",
    "businessServiceSla": 259200000,
    "states": [
     {
       "sla": null,
       "state": null,
       "applicationStatus": "INWORKFLOW",
       "docUploadRequired": false,
       "isStartState": true,
       "isTerminateState": false,
       "isStateUpdatable": false,
       "actions": [
        {
         "action": "OPEN",
         "nextState": "INITIATED",
         "roles": [
          "CITIZEN",
          "WS_CEMP"
         ]
        }
       ]
      },
      {
       "sla": null,
       "state": "INITIATED",
       "applicationStatus": "INWORKFLOW",
       "docUploadRequired": false,
       "isStartState": true,
       "isTerminateState": false,
       "isStateUpdatable": true,
       "actions": [
        {
         "action": "SUBMIT",
         "nextState": "APPROVED",
         "roles": [
          "CITIZEN",
          "WS_CEMP"
         ]
        }
       ]
      },
      {
       "sla": null,
       "state": "APPROVED",
       "applicationStatus": "ACTIVE",
       "docUploadRequired": false,
       "isStartState": false,
       "isTerminateState": true,
       "isStateUpdatable": false,
       "actions": null
      }
```

```
   ]
  }
 ]
}
```

Indexer config for water-service

The indexer provides the facility for indexing the data to elastic search.

Setup

Write the configuration for water service. The structure of the config file is explained later in the same doc.
Provide the absolute path of the checked-in file to DevOps, to add it to the file-read path of egov-indexer. The file will be added to the egov-indexer's environment manifest file for it to be read at the start-up of the application.
Put indexer config file to the config repo under egov-indexer folder.(egovernments/configs)
Run the egov-indexer app, Since it is a consumer, it starts listening to the configured topics and indexes the data.

config Keys

The indexer uses a config file per module to store all the configurations pertaining to that module. Indexer reads multiple such files at start-up to support indexing for all the configured modules. The water service file contains the following keys:

a. **serviceName**: Name of the module to which this configuration belongs.
b. **summary**: Summary of the module.
c. **version**: The version of the configuration.
d. **mappings**: List of definitions within the module. Every definition corresponds to one index requirement. Which means, every object received onto the Kafka queue can be used to create multiple indexes, each of these indexes will need configuration, all such configurations belonging to one topic forms one entry in the mappings list. The keys listed henceforth together form one definition and multiple such definitions are part of this mappings key.
   i)topic: Topic on which the data is to be received to activate this particular configuration.
   ii)configKey: Key to identify to what type of job is this config for. values: INDEX, REINDEX, LEGACYINDEX. INDEX: LiveIndex, REINDEX: Reindex, LEGACYINDEX: LegacyIndex.

iii. **indexes**: Key to configure multiple index configurations for the data received on the particular topic. Multiple indexes based on different requirement can be created using the same object. This list of such configurations is a part of this key. uses the following keys:

1. **name**: Index name on the elasticsearch. (Index will be created if it doesn't exist with this name.)
2. **type**: Document type within that index to which the index json has to go. (Elasticsearch uses the structure of index/type/docId to locate any file within index/type with id = docId)
3. **id**: Takes comma separated JsonPaths. The JSONPath is applied on the record received on the queue, the values hence obtained are appended and used as id for the record.
4. **jsonPath**: Key to be used in case of indexing a part of the input JSON and in case of indexing a custom json where the values for custom json are to be fetched from this part of the input.
5. **timeStampField**: JSONPath of the field in the input which can be used to obtain the timestamp of the input.
6. i) **indexMapping**: A skeleton/mapping of the JSON that is to be indexed. Note that, this JSON must always contain a key called "Data" at the top-level and the custom mapping begins within this key. This is only a convention to smoothen dashboarding on Kibana when data from multiple indexes have to be fetched for a single dashboard.

iv) **fieldMapping**: Contains a list of configurations. Each configuration contains keys to identify the field of the input JSON that has to be mapped to the fields of the index json which is mentioned in the key 'indexMapping' in the config. Has the following keys:
inJsonPath: JSONPath of the field from the input.

outJsonPath: JSONPath of the field of the index json.

v) **externalUriMapping**: Contains a list of configurations. Each configuration contains keys to identify the field of the input JSON that are to be enriched using APIs from the external services. The confiugration for those APIs also is a part of this. Uses the following keys:

1. **path**: URI of the API to be used. (it should be POST/_search API.)
2. **queryParam**: Configruation of the query params to be used for the API call. It is a comma seperated key-value pair, where key is the parameter name as per the API contract and value is the JSONPath of the field to be equated against this paramter.
3. **apiRequest**: Request Body of the API. (Since we only use _search APIs, it should be only RequestInfo.)
4. **uriResponseMapping**: Contains a list of configuration. Each configuration contains two keys: One is a JSONPath to identify the field from response, Second is also a JSONPath to map the response field to a field of the index json mentioned in the key 'indexMapping'.
    i) **inJsonPath**: JSONPath to identify the field from response
    ii) **outJsonPath**: JSONPath to map the response field to a field of the index json

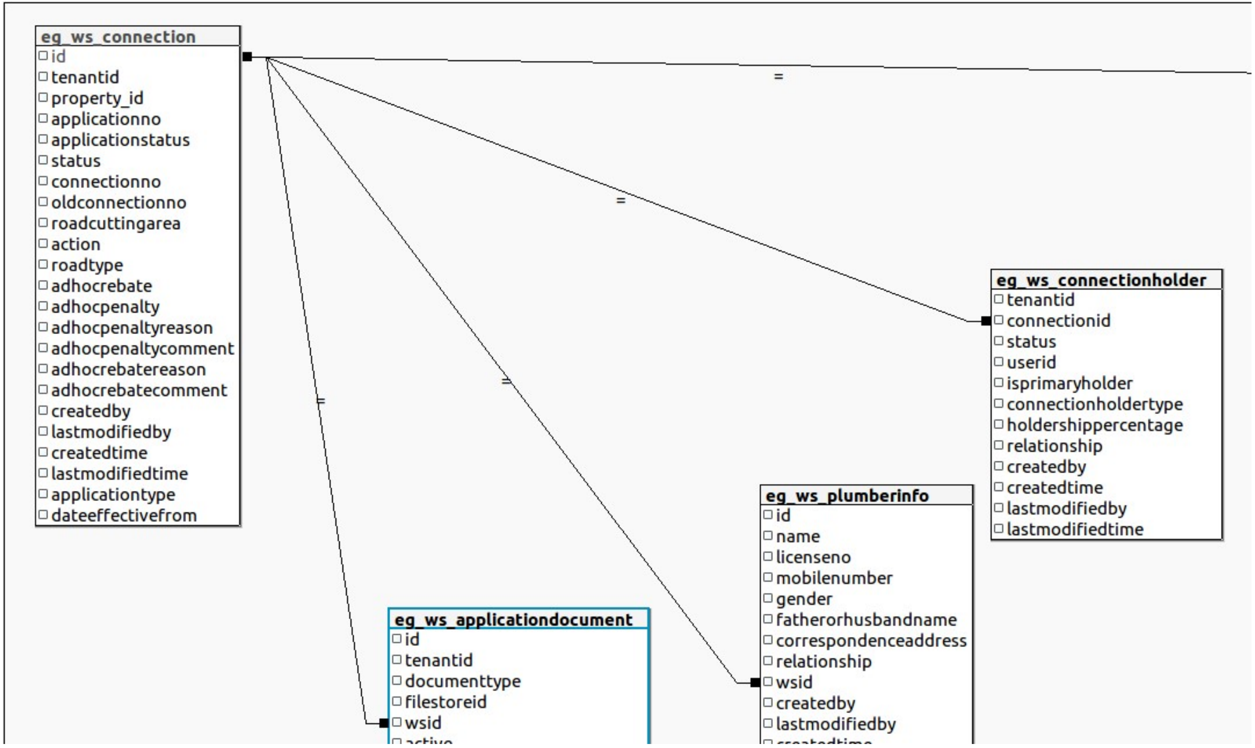
***water-service* indexer config**

```
ServiceMaps:
 serviceName: Water Service - rainmaker
 version: 1.0.0
 mappings:
  - topic: save-ws-connection
    configKey: INDEX
    indexes:
      - name: water-services
        type: general
          id: $.id,$.tenantId
          jsonPath: $.WaterConnection
          timeStampField: $.auditDetails.createdTime
          customJsonMapping:
            indexMapping: {"Data":{"workflow": {"state": {}, "action": "", "assignes": []},"applicationNo":"","applicationStatus":"","status":"","connectionNo":"","oldConnectionNo":"","plumberInfo":[],"roadCuttingInfo":[],"connectionHolders":[],"roadType":"","roadCuttingArea":"","connectionExecutionDate":"","connectionCategory":"","connectionType":"","additionalDetails":{},"id":"","propertyId":"","rainWaterHarvesting":"","waterSource":"","tenantId":"","meterId":"","meterInstallationDate":"","proposedPipeSize":"","proposedTaps":"","pipeSize":"","noOfTaps":"","applicationType":"","dateEffectiveFrom":"","history":{}}}
            fieldMapping:
            - inJsonPath: $.applicationStatus.state
              outJsonPath: $.Data.workflow.state
            - inJsonPath: $.processInstance.action
              outJsonPath: $.Data.workflow.action
            - inJsonPath: $.processInstance.assignes.*.uuid
              outJsonPath: $.Data.workflow.assignes
            - inJsonPath: $.applicationNo
              outJsonPath: $.Data.applicationNo
            - inJsonPath: $.applicationStatus
              outJsonPath: $.Data.applicationStatus
            - inJsonPath: $.status
              outJsonPath: $.Data.status
            - inJsonPath: $.connectionNo
              outJsonPath: $.Data.connectionNo
            - inJsonPath: $.oldConnectionNo
              outJsonPath: $.Data.oldConnectionNo
            - inJsonPath: $.plumberInfo
              outJsonPath: $.Data.plumberInfo
            - inJsonPath: $.roadCuttingInfo
              outJsonPath: $.Data.roadCuttingInfo
            - inJsonPath: $.connectionHolders
              outJsonPath: $.Data.connectionHolders
            - inJsonPath: $.roadType
              outJsonPath: $.Data.roadType
            - inJsonPath: $.roadCuttingArea
              outJsonPath: $.Data.roadCuttingArea
            - inJsonPath: $.connectionExecutionDate
              outJsonPath: $.Data.connectionExecutionDate
            - inJsonPath: $.connectionCategory
              outJsonPath: $.Data.connectionCategory
            - inJsonPath: $.connectionType
              outJsonPath: $.Data.connectionType
            - inJsonPath: $.additionalDetails
              outJsonPath: $.Data.additionalDetails
            - inJsonPath: $.id
              outJsonPath: $.Data.id
            - inJsonPath: $.propertyId
              outJsonPath: $.Data.propertyId
            - inJsonPath: $.rainWaterHarvesting
              outJsonPath: $.Data.rainWaterHarvesting
            - inJsonPath: $.waterSource
              outJsonPath: $.Data.waterSource
            - inJsonPath: $.tenantId
              outJsonPath: $.Data.tenantId
            - inJsonPath: $.meterId
              outJsonPath: $.Data.meterId
            - inJsonPath: $.meterInstallationDate
              outJsonPath: $.Data.meterInstallationDate
            - inJsonPath: $.proposedPipeSize
              outJsonPath: $.Data.proposedPipeSize
            - inJsonPath: $.proposedTaps
              outJsonPath: $.Data.proposedTaps
            - inJsonPath: $.pipeSize
              outJsonPath: $.Data.pipeSize
            - inJsonPath: $.noOfTaps
              outJsonPath: $.Data.noOfTaps
            - inJsonPath: $.applicationType
              outJsonPath: $.Data.applicationType
            - inJsonPath: $.dateEffectiveFrom
              outJsonPath: $.Data.dateEffectiveFrom
            externalUriMapping:
            - path: http://egov-workflow-v2.egov:8080/egov-workflow-v2/egov-wf/process/_search
              queryParam: businessIds=$.applicationNo,history=true,tenantId=$.tenantId
              apiRequest:
        {"RequestInfo":{"apiId":"org.egov.pt","ver":"1.0","ts":1502890899493,"action":"asd","did":"4354648646","key":
        "xyz","msgId":"654654","requesterId":"61","authToken":"d9994555-7656-4a67-ab3a-
        a952a0d4dfc8","userInfo":{"id":1,"uuid":"1fec8102-0e02-4d0a-b283-
cd80d5dab067","type":"EMPLOYEE","tenantId":"pb.amritsar","roles":[{"name":"Employee","code":"EMPLOYEE","tenantId":"pb.amritsar"}]}}}
          uriResponseMapping:
          - inJsonPath: $.ProcessInstances
            outJsonPath: $.Data.history


- topic: update-ws-connection
  configKey: INDEX
  indexes:
    - name: water-services
      type: general
      id: $.id,$.tenantId
      jsonPath: $.WaterConnection
      timeStampField: $.auditDetails.lastModifiedTime
      customJsonMapping:
        indexMapping: {"Data":{"workflow": {"state": {}, "action": "", "assignes": []},"applicationNo":"","applicationStatus":"","status":"","connectionNo":"","oldConnectionNo":"","plumberInfo":[],"roadCuttingInfo":[],"connectionHolders":[],"roadType":"","roadCuttingArea":"","connectionExecutionDate":"","connectionCategory":"","connectionType":"","additionalDetails":{},"id":"","propertyId":"","rainWaterHarvesting":"","waterSource":"","tenantId":"","meterId":"","meterInstallationDate":"","proposedPipeSize":"","proposedTaps":"","pipeSize":"","noOfTaps":"","applicationType":"","dateEffectiveFrom":"","history":{}}}
        fieldMapping:
        - inJsonPath: $.applicationStatus.state
          outJsonPath: $.Data.workflow.state
        - inJsonPath: $.processInstance.action
          outJsonPath: $.Data.workflow.action
        - inJsonPath: $.processInstance.assignes.*.uuid
          outJsonPath: $.Data.workflow.assignes
        - inJsonPath: $.applicationNo
          outJsonPath: $.Data.applicationNo
        - inJsonPath: $.applicationStatus
          outJsonPath: $.Data.applicationStatus
        - inJsonPath: $.status
          outJsonPath: $.Data.status
        - inJsonPath: $.connectionNo
          outJsonPath: $.Data.connectionNo
        - inJsonPath: $.oldConnectionNo
          outJsonPath: $.Data.oldConnectionNo
        - inJsonPath: $.plumberInfo
          outJsonPath: $.Data.plumberInfo
        - inJsonPath: $.roadCuttingInfo
          outJsonPath: $.Data.roadCuttingInfo
        - inJsonPath: $.connectionHolders
          outJsonPath: $.Data.connectionHolders
        - inJsonPath: $.roadType
          outJsonPath: $.Data.roadType
        - inJsonPath: $.roadCuttingArea
          outJsonPath: $.Data.roadCuttingArea
        - inJsonPath: $.connectionExecutionDate
          outJsonPath: $.Data.connectionExecutionDate
        - inJsonPath: $.connectionCategory
          outJsonPath: $.Data.connectionCategory
        - inJsonPath: $.connectionType
          outJsonPath: $.Data.connectionType
        - inJsonPath: $.additionalDetails
          outJsonPath: $.Data.additionalDetails
        - inJsonPath: $.id
          outJsonPath: $.Data.id
        - inJsonPath: $.propertyId
          outJsonPath: $.Data.propertyId
        - inJsonPath: $.rainWaterHarvesting
          outJsonPath: $.Data.rainWaterHarvesting
        - inJsonPath: $.waterSource
          outJsonPath: $.Data.waterSource
        - inJsonPath: $.tenantId
          outJsonPath: $.Data.tenantId
        - inJsonPath: $.meterId
          outJsonPath: $.Data.meterId
        - inJsonPath: $.meterInstallationDate
          outJsonPath: $.Data.meterInstallationDate
        - inJsonPath: $.proposedPipeSize
          outJsonPath: $.Data.proposedPipeSize
        - inJsonPath: $.proposedTaps
          outJsonPath: $.Data.proposedTaps
        - inJsonPath: $.pipeSize
          outJsonPath: $.Data.pipeSize
        - inJsonPath: $.noOfTaps
          outJsonPath: $.Data.noOfTaps
        - inJsonPath: $.applicationType
          outJsonPath: $.Data.applicationType
        - inJsonPath: $.dateEffectiveFrom
          outJsonPath: $.Data.dateEffectiveFrom
        externalUriMapping:
        - path: http://egov-workflow-v2.egov:8080/egov-workflow-v2/egov-wf/process/_search
          queryParam: businessIds=$.applicationNo,history=true,tenantId=$.tenantId
          apiRequest: {"RequestInfo":{"apiId":"org.egov.pt","ver":"1.0","ts":1502890899493,"action":"asd","did":"4354648646","key":"xyz","msgId":"654654","requesterId":"61","authToken":"d9994555-7656-4a67-ab3a-a952a0d4dfc8","userInfo":{"id":1,"uuid":"1fec8102-0e02-4d0a-b283-cd80d5dab067","type":"EMPLOYEE","tenantId":"pb.amritsar","roles":[{"name":"Employee","code":"EMPLOYEE","tenantId":"pb.amritsar"}]}}}
          uriResponseMapping:
          - inJsonPath: $.ProcessInstances
            outJsonPath: $.Data.history

- topic: update-ws-workflow
  configKey: INDEX
  indexes:
    - name: water-services
      type: general
      id: $.id,$.tenantId
      jsonPath: $.WaterConnection
      timeStampField: $.auditDetails.lastModifiedTime
      customJsonMapping:
        indexMapping: {"Data":{"workflow": {"state": {}, "action": "", "assignes": []},"applicationNo":"","applicationStatus":"","status":"","connectionNo":"","oldConnectionNo":"","plumberInfo":[],"roadCuttingInfo":[],"connectionHolders":[],"roadType":"","roadCuttingArea":"","connectionExecutionDate":"","connectionCategory":"","connectionType":"","additionalDetails":{},"id":"","propertyId":"","rainWaterHarvesting":"","waterSource":"","tenantId":"","meterId":"","meterInstallationDate":"","proposedPipeSize":"","proposedTaps":"","pipeSize":"","noOfTaps":"","applicationType":"","dateEffectiveFrom":"","history":{}}}
        fieldMapping:
        - inJsonPath: $.applicationStatus.state
          outJsonPath: $.Data.workflow.state
        - inJsonPath: $.processInstance.action
          outJsonPath: $.Data.workflow.action
        - inJsonPath: $.processInstance.assignes.*.uuid
          outJsonPath: $.Data.workflow.assignes
        - inJsonPath: $.applicationNo
          outJsonPath: $.Data.applicationNo
        - inJsonPath: $.applicationStatus
          outJsonPath: $.Data.applicationStatus
        - inJsonPath: $.status
          outJsonPath: $.Data.status
        - inJsonPath: $.connectionNo
          outJsonPath: $.Data.connectionNo
        - inJsonPath: $.oldConnectionNo
          outJsonPath: $.Data.oldConnectionNo
        - inJsonPath: $.plumberInfo
          outJsonPath: $.Data.plumberInfo
        - inJsonPath: $.roadCuttingInfo
          outJsonPath: $.Data.roadCuttingInfo
        - inJsonPath: $.connectionHolders
          outJsonPath: $.Data.connectionHolders
        - inJsonPath: $.roadType
          outJsonPath: $.Data.roadType
        - inJsonPath: $.roadCuttingArea
          outJsonPath: $.Data.roadCuttingArea
        - inJsonPath: $.connectionExecutionDate
          outJsonPath: $.Data.connectionExecutionDate
        - inJsonPath: $.connectionCategory
          outJsonPath: $.Data.connectionCategory
        - inJsonPath: $.connectionType
          outJsonPath: $.Data.connectionType
        - inJsonPath: $.additionalDetails
          outJsonPath: $.Data.additionalDetails
        - inJsonPath: $.id
          outJsonPath: $.Data.id
        - inJsonPath: $.propertyId
          outJsonPath: $.Data.propertyId
        - inJsonPath: $.rainWaterHarvesting
          outJsonPath: $.Data.rainWaterHarvesting
        - inJsonPath: $.waterSource
          outJsonPath: $.Data.waterSource
        - inJsonPath: $.tenantId
          outJsonPath: $.Data.tenantId
        - inJsonPath: $.meterId
          outJsonPath: $.Data.meterId
        - inJsonPath: $.meterInstallationDate
          outJsonPath: $.Data.meterInstallationDate
        - inJsonPath: $.proposedPipeSize
          outJsonPath: $.Data.proposedPipeSize
        - inJsonPath: $.proposedTaps
          outJsonPath: $.Data.proposedTaps
        - inJsonPath: $.pipeSize
          outJsonPath: $.Data.pipeSize
        - inJsonPath: $.noOfTaps
          outJsonPath: $.Data.noOfTaps
        - inJsonPath: $.applicationType
          outJsonPath: $.Data.applicationType
        - inJsonPath: $.dateEffectiveFrom
          outJsonPath: $.Data.dateEffectiveFrom
        externalUriMapping:
        - path: http://egov-workflow-v2.egov:8080/egov-workflow-v2/egov-wf/process/_search
          queryParam: businessIds=$.applicationNo,history=true,tenantId=$.tenantId
          apiRequest: {"RequestInfo":{"apiId":"org.egov.pt","ver":"1.0","ts":1502890899493,"action":"asd","did":"4354648646","key":"xyz","msgId":"654654","requesterId":"61","authToken":"d9994555-7656-4a67-ab3a-a952a0d4dfc8","userInfo":{"id":1,"uuid":"1fec8102-0e02-4d0a-b283-cd80d5dab067","type":"EMPLOYEE","tenantId":"pb.amritsar","roles":[{"name":"Employee","code":"EMPLOYEE","tenantId":"pb.amritsar"}]}}}
          uriResponseMapping:
          - inJsonPath: $.ProcessInstances
            outJsonPath: $.Data.history
```

meter reading **indexer config**

```
ServiceMaps:
 serviceName: Meter Connection - rainmaker
 version: 1.0.0
 mappings:
  - topic: save-ws-meter
    configKey: INDEX
    indexes:
      - name: meter-services
        type: general
        id: $.id,$.connectionNo
        jsonPath: $.meterReadings
        customJsonMapping:
          indexMapping: {"Data":{"connectionNo":"","meterStatus":"","currentReadingDate":"","currentReading":"","lastReading":"","billingPeriod":"","lastReadingDate":""}}
          fieldMapping:
          - inJsonPath: $.connectionNo
            outJsonPath: $.Data.connectionNo
          - inJsonPath: $.meterStatus
            outJsonPath: $.Data.meterStatus
          - inJsonPath: $.currentReadingDate
            outJsonPath: $.Data.currentReadingDate
          - inJsonPath: $.currentReading
            outJsonPath: $.Data.currentReading
          - inJsonPath: $.lastReading
            outJsonPath: $.Data.lastReading
          - inJsonPath: $.billingPeriod
            outJsonPath: $.Data.billingPeriod
          - inJsonPath: $.lastReadingDate
            outJsonPath: $.Data.lastReadingDate

```

table UML diagram



Modify connection

After connection activation or legacy connection, we can edit the connection. This process based on defined workflow. Any action is based on defined roles on the action level. For edit connection, we need to upload some supporting documents and mandatory info.

Workflow config for edit connection

```
{
 "BusinessServices": [
  {
   "tenantId": "pb",
   "businessService": "ModifyWSConnection",
   "business": "ws-services",
   "businessServiceSla": 259200000,
   "states": [
    {
      "sla": null,
      "state": null,
      "applicationStatus": null,
      "docUploadRequired": false,
      "isStartState": true,
      "isTerminateState": false,
      "isStateUpdatable": false,
      "actions": [
        {
          "action": "INITIATE",
          "nextState": "INITIATED",
          "roles": [
            "WS_CEMP"
          ]
        }
      ]
    },
    {
      "sla": null,
      "state": "INITIATED",
      "applicationStatus": "INITIATED",
      "docUploadRequired": false,
      "isStartState": false,
      "isTerminateState": false,
      "isStateUpdatable": true,
      "actions": [
        {
          "action": "SUBMIT_APPLICATION",
          "nextState": "PENDING_FOR_APPROVAL",
          "roles": [
            "WS_CEMP"
          ]
        }
      ]
    },
    {
      "sla": 86400000,
      "state": "PENDING_FOR_APPROVAL",
      "applicationStatus": "PENDING_FOR_APPROVAL",
      "docUploadRequired": false,
      "isStartState": false,
      "isStateUpdatable": true,
      "isTerminateState": false,
      "actions": [
        {
          "action": "APPROVE_CONNECTION",
          "nextState": "APPROVED",
          "roles": [
            "WS_APPROVER"
          ]
        },
        {
          "action": "REJECT",
          "nextState": "REJECTED",
          "roles": [
            "WS_APPROVER"
          ]
        }
```

```
          ]
        },
        {
          "sla": null,
          "state": "REJECTED",
          "applicationStatus": "REJECTED",
          "isStateUpdatable": false,
          "docUploadRequired": false,
          "isStartState": false,
          "isTerminateState": true
        },
        {
          "sla": null,
          "state": "APPROVED",
          "applicationStatus": "APPROVED",
          "isStateUpdatable": false,
          "docUploadRequired": false,
          "isStartState": false,
          "isTerminateState": true
        }
      ]
    }
  ]
}
```

Notification

Notification will be sent to the property owners and connection holders based on different application states.

Capturing connection holders

We can add connection holders to the water connection which will be the owner of the connection. We can fill the connection holders' details or we can just make the property owner to the connection holder.

The connection holder will get notification based on a different state of the application. We are pushing the data of the connection holders in the user service too.

Multiple Road Type Support

We can add road cutting details of multiple roads to the water connection. For each road which goes under cutting process we have to fill their road type details and road cutting area.

Based on this information, application one time fee estimate is calculated.

Water Calculator Service - Technical Document

This application is used for creating meter reading, searching meter reading, updating existing meter reading, calculation of water charge, demand generation, SMS & email notification to ULB officials on-demand generation and estimation of water charge(one-time cost) which involves cost like road-cutting charge, form fee, scrutiny fee, etc.

Billing Slabs

Criteria:

- connection type
- building type
- calculation attribute
- property usage type

The combination of the above can be used to define the billing slab. Billing Slab is defined in MDMS under ws-services-calculation folder with the WCBillingSlab. The following is the sample slab.

```
{
    "id": "1",
    "buildingType": "RESIDENTIAL",
    "connectionType": "Metered",
    "calculationAttribute": "Water consumption",
    "minimumCharge": 100,
    "slabs": [
      {
        "from": 0,
          "to": 10,
          "charge": 2,
          "meterCharge": 50
        },
        {
          "from": 10,
          "to": 20,
          "charge": 2.5,
          "meterCharge": 50
        },
        {
          "from": 20,
          "to": 30,
          "charge": 8,
          "meterCharge": 150
        },
        {
          "from": 30,
          "to": 40,
          "charge": 12,
          "meterCharge": 150
        },
        {
          "from": 40,
          "to": 1000000000,
          "charge": 15,
          "meterCharge": 150
        }
      ]
    }
```

If all criteria will match for that water connection this slab will use for calculation.

Estimation
For application one-time fee, the estimation will return all the related tax head based on criteria. For estimation, all configurations are present in ws-services-calculation.

- FeeSlab.json
- PlotSizeSlab.json
- RoadType.json

All the above master configuration is used for estimation.

Following are the exemptions and taxes that are calculated:

- Form fee
- Scrutiny fee
- Meter charge (For metered connection)
- Other charges
- Road cutting charges
- One time fee
- Security charges
- Tax and cess

Water Charge and Tax

Water charge is based on billing slab, for water application charge will be based on slab and tax based on master configuration.

Interest

Below is a sample of master data JSON for interest:

```
{
 "tenantId": "pb",
 "moduleName": "ws-services-calculation",
 "Interest": [
  {
    "rate": 5,
    "minAmount": null,
    "applicableAfterDays":0,
    "flatAmount": null,
    "maxAmount": null,
    "fromFY": "2019-20",
    "startingDay": "1/01/2019"
  }
 ]
}
```

Penalty

Below is a sample of master data JSON for penalty:

```
{
 "tenantId": "pb",
 "moduleName": "ws-services-calculation",
 "Penalty": [
  {
    "rate": 10,
    "minAmount": null,
    "applicableAfterDays": 0,
    "flatAmount": null,
    "fromFY": "2019-20",
    "startingDay": "1/01/2019"
  }
 ]
}
```

Round Off

If the fraction is greater than equal to 0.5 the number is round up else it’s round down. eg: 100.4 will be rounded to 100 while 100.6 will be rounded to 101.

Adding Adhoc penalty or rebate

The only employee can apply for a penalty or rebate for an existing connection. As an employee, I can update or add the penalty and rebate of a connection. This applied penalty or rebate will be added or updated in existing demand as tax heads. For configuration, we have to add the tax head in TaxHeadMaster.json file.

```
{
    "category": "TAX",
    "service": "WS",
    "name": "Water adhoc rebate",
    "code": "WS_TIME_ADHOC_REBATE",
    "isDebit": false,
    "isActualDemand": true,
    "order": "5",
    "isRequired": false
    },
    {
      "category": "TAX",
      "service": "WS",
      "name": "Water adhoc penalty",
      "code": "WS_TIME_ADHOC_PENALTY",
      "isDebit": false,
      "isActualDemand": true,
      "order": "6",
      "isRequired": false
    },
```

Demand Generation

Once water is sent to calculator it’s tax estimates are calculated. Using this tax head estimates demand details are created. For every tax head, estimate demand generates function will create a corresponding demand detail.

Whenever _calculate API is called demand is first searched based on the connection no or application no and the demand from and to period. If demand already exists the same demand is updated else new demand is generated with consumer code as connection no or application no and demand from and to a period equal to financial year start and end period.

In case of the update if the tax head estimates change, the difference in amount for that tax head is added as new demand detail. For example, if the initial demand has one demand detail with WATER_CHARGE equal to 120.

```
"demandDetails": [
          {
            "id": "77ba1e93-a535-409c-b9d1-a312c409bd45",
            "demandId": "687c3176-305b-461d-9cec-2fa26a30c88f",
            "taxHeadMasterCode": "WATER_CHARGE",
            "taxAmount": 120,
            "collectionAmount": 120,
            "additionalDetails": null,
            "auditDetails": {
              "createdBy": "04956309-87cd-4526-b4e6-48123abd4f3d",
              "lastModifiedBy": "04956309-87cd-4526-b4e6-48123abd4f3d",
              "createdTime": 1583675275873,
              "lastModifiedTime": 1583675298705
            },
            "tenantId": "pb.amritsar"
          }
        ],
```

After updating, if the WATER_CHARGE increases to 150 we add one more demand detail to account for the increased amount. The demand detail will be updated to:

```
"demandDetails": [
         {
           "id": "77ba1e93-a535-409c-b9d1-a312c409bd45",
           "demandId": "687c3176-305b-461d-9cec-2fa26a30c88f",
           "taxHeadMasterCode": "WATER_CHARGE",
           "taxAmount": 120,
           "collectionAmount": 0,
           "additionalDetails": null,
           "auditDetails": {
              "createdBy": "04956309-87cd-4526-b4e6-48123abd4f3d",
              "lastModifiedBy": "04956309-87cd-4526-b4e6-48123abd4f3d",
              "createdTime": 1583675275873,
              "lastModifiedTime": 1583675298705
            },
            "tenantId": "pb.amritsar"
          },
          {
            "id": "0d83f4b0-6442-11ea-bc55-0242ac130003 ",
            "demandId": "687c3176-305b-461d-9cec-2fa26a30c88f",
            "taxHeadMasterCode": "WATER_CHARGE",
            "taxAmount": 30,
            "collectionAmount": 0,
            "additionalDetails": null,
            "auditDetails": {
              "createdBy": "04956309-87cd-4526-b4e6-48123abd4f3d",
              "lastModifiedBy": "04956309-87cd-4526-b4e6-48123abd4f3d",
              "createdTime": 1583675275873,
              "lastModifiedTime": 1583675298705
            },
            "tenantId": "pb.amritsar"
          }
        ],
```

RoundOff is bill based i.e every time bill is generated round off is adjusted so that payable amount is the whole number. Individual WS_ROUNDOFF in demand detail can be greater than 0.5 but the sum of all WS_ROUNDOFF will always be less than 0.5.

Frontend

Configurations:

ws-services-masters MDMS folder:

https://github.com/egovernments/egov-mdms-data/tree/master/data/pb/ws-services-masters - Connect to preview

Documents.json

Used to display the order of the files in the Documents section which needs to be uploaded from the Citizen or Employee while creating the Water Service application.

WaterSource.json - Provides details of the different types of water source and their sub types.

ws-services-calculation MDMS folder:

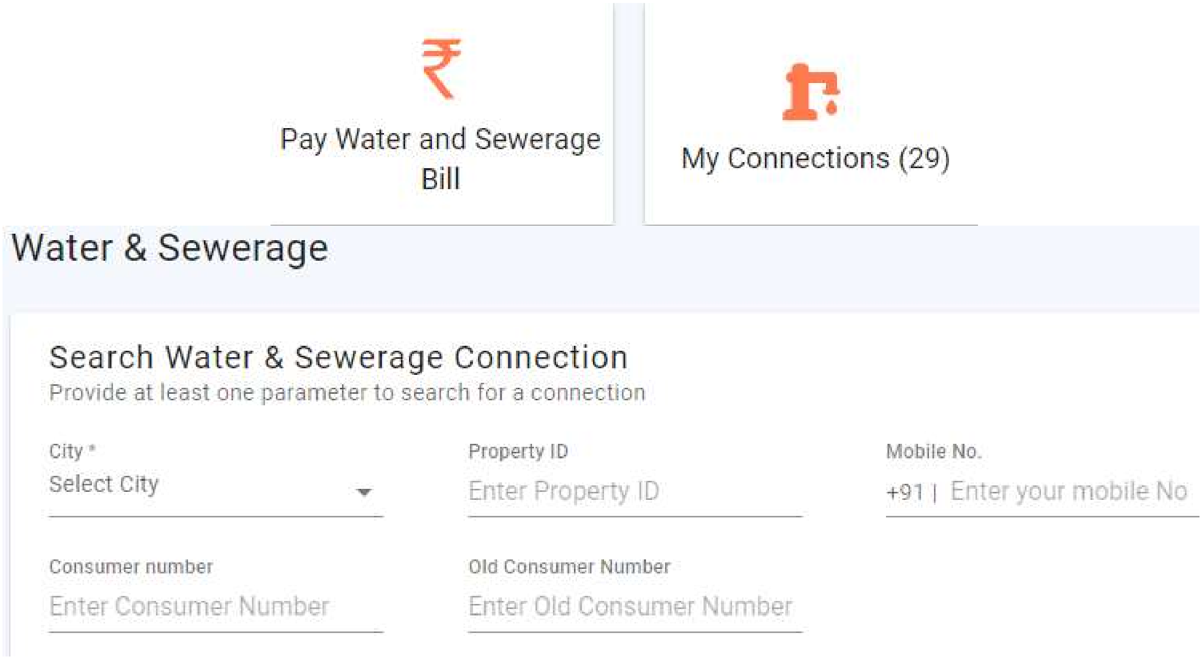
https://github.com/egovernments/egov-mdms-data/tree/master/data/pb/ws-services-calculation - Connect to preview

Pipesize.json - Provides the details of the pipe sizes.
Roadtype.json - Provides the details of the different types of the Roads and their cutting charges.

PDF Configurations:
The UI and the PDFService retrieves the Data and Format configurations from the following path:
https://github.com/egovernments/configs/tree/master/pdf-service - Connect to preview

Citizen UI Guide
List of features available in the W&S service for Citizen role.

a) Search Bills & Pay:
Citizen, by using different search criteria to find the particular connection and also he/she can able to pay the water and sewerage bill for the particular connection.



₹ Pay Water and Sewerage Bill

My Connections (29)

Water & Sewerage

Search Water & Sewerage Connection
Provide at least one parameter to search for a connection

City * — Select City

Property ID — Enter Property ID

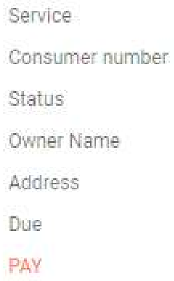
Mobile No. — +91 | Enter your mobile No

Consumer number — Enter Consumer Number

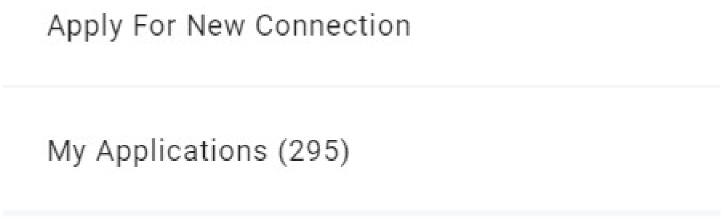
Old Consumer Number — Enter Old Consumer Number

My Connections & Connection Details:

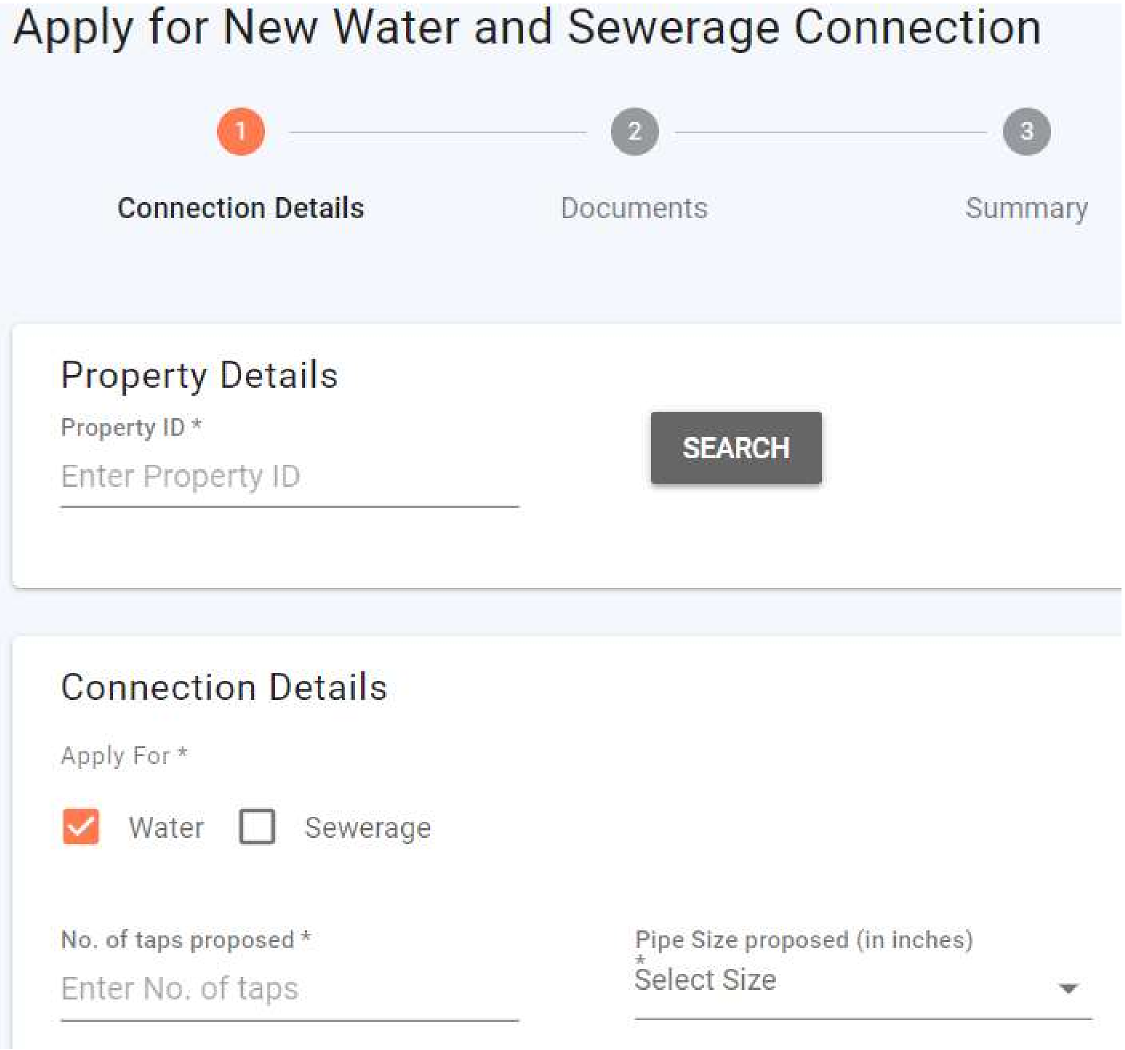
All the consumer numbers are clickable in connections list. citizen can see the all connection details and also able to download.



b) Create new application



citizen can able to create new application on click of “Apply for new connection”



Apply for New Water and Sewerage Connection

1 Connection Details — 2 Documents — 3 Summary

Property Details

Property ID *

Enter Property ID

SEARCH

Connection Details

Apply For *

☑ Water ☐ Sewerage

No. of taps proposed *

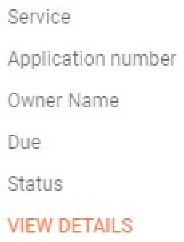
Enter No. of taps

Pipe Size proposed (in inches) *

Select Size

In this page citizen need to fill all relevant details for creating the application this is the first page of application, second page is documents upload, Third page is the summary page which includes all the provided details.

c) View Application



Service

Application number

Owner Name

Due

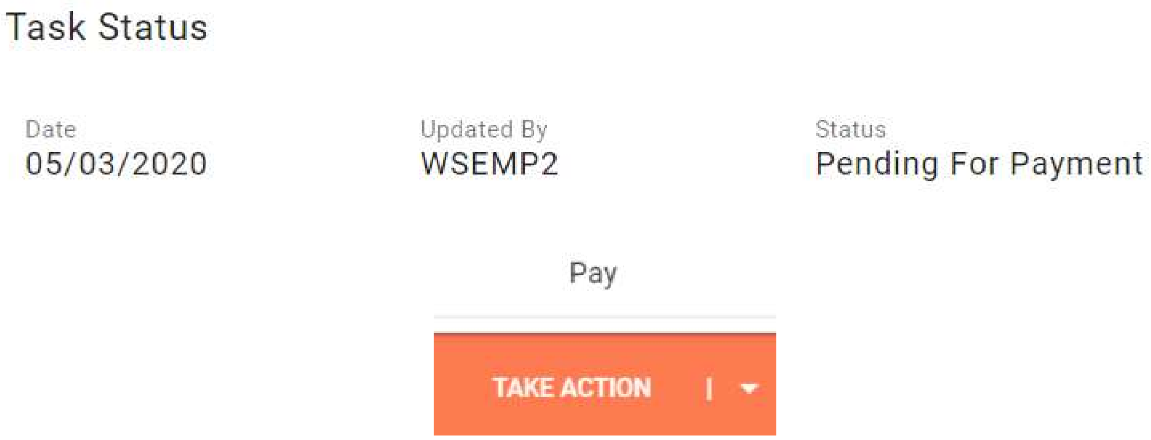
Status

VIEW DETAILS

In My Applications, Citizen can see the list of applications he/she have. For every application Citizen can see the above fields. on click of ViewDetails button citizen can see the workflow page,There Citizen can perform the actions like(Edit and Resubmit)the application.

d) Pay

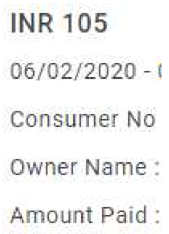
Citizen can also Pay the Due amount by using VIEW DETAILS link based on status (Pending for payment).



Task Status

Date	Updated By	Status
05/03/2020	WSEMP2	Pending For Payment

e) Past payments
Citizen can see his past payment records like which month he paid how much money and basic details are shown in this.



INR 105

06/02/2020 - (

Consumer No

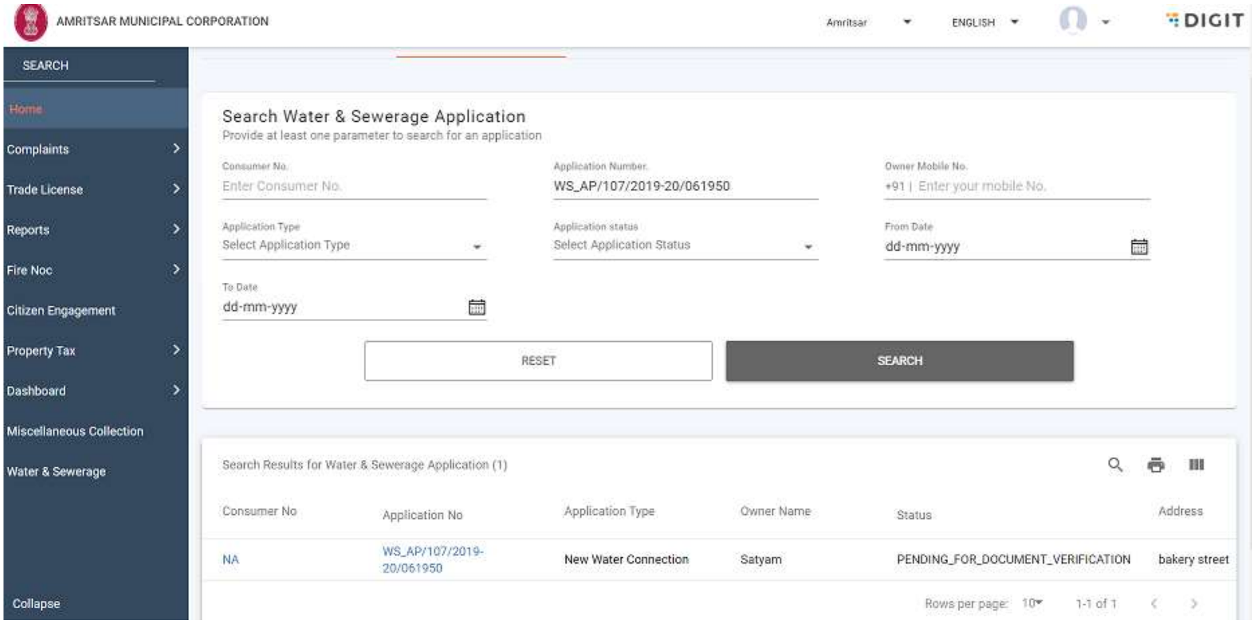
Owner Name :

Amount Paid :

Employee UI Guide

SEARCH APPLICATION / CONNECTION

Once an Application is created (INITIATED state in Workflow), the application number can be used to search the application. There are several other criteria’s that can be used to search the application.



Search Water & Sewerage Application

Provide at least one parameter to search for an application

Consumer No. Enter Consumer No.	Application Number WS_AP/107/2019-20/061950	Owner Mobile No. +91 | Enter your mobile No.
Application Type Select Application Type	Application status Select Application Status	From Date dd-mm-yyyy
To Date dd-mm-yyyy		

Search Results for Water & Sewerage Application (1)

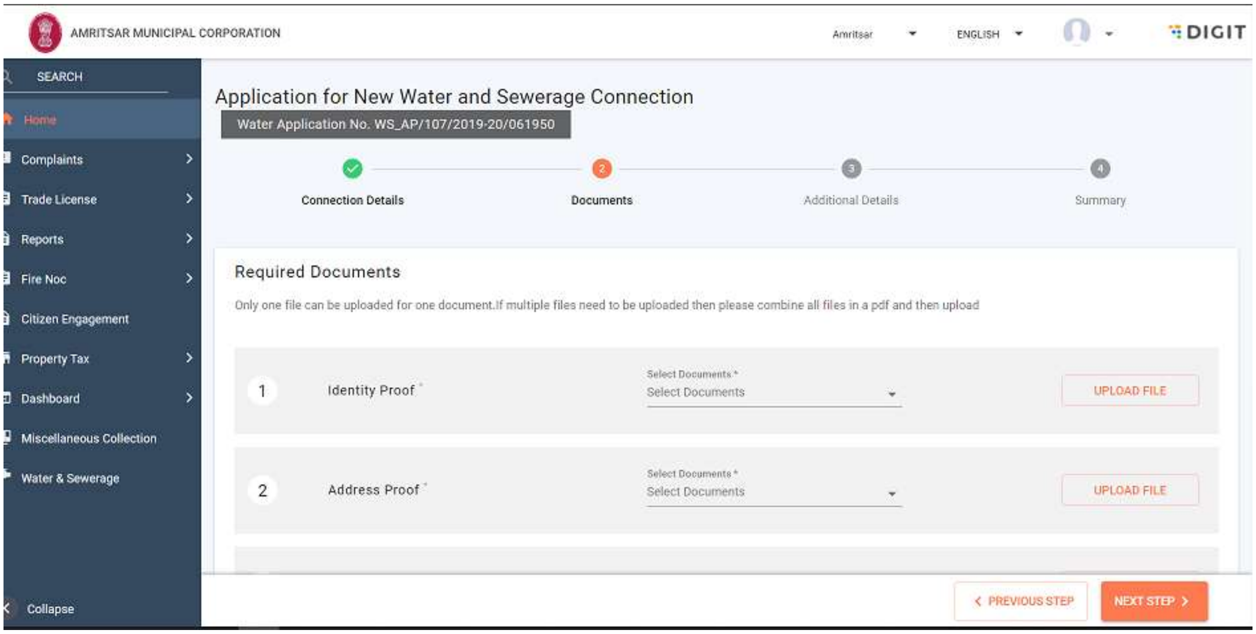
Consumer No	Application No	Application Type	Owner Name	Status	Address
NA	WS_AP/107/2019-20/061950	New Water Connection	Satyam	PENDING_FOR_DOCUMENT_VERIFICATION	bakery street

Rows per page: 10 1-1 of 1

APPLY FOR NEW WATER AND SEWERAGE CONNECTION

All the application once INITIATED can have until CONNECTION ACTIVATION can have multiple actions buttons. And, selected roles can forward the application with a specific action to the next stage or can make corrections to it using EDIT.

A water application has been INITIATED (The First Stage in Workflow)



Application for New Water and Sewerage Connection

Water Application No. WS_AP/107/2019-20/061950

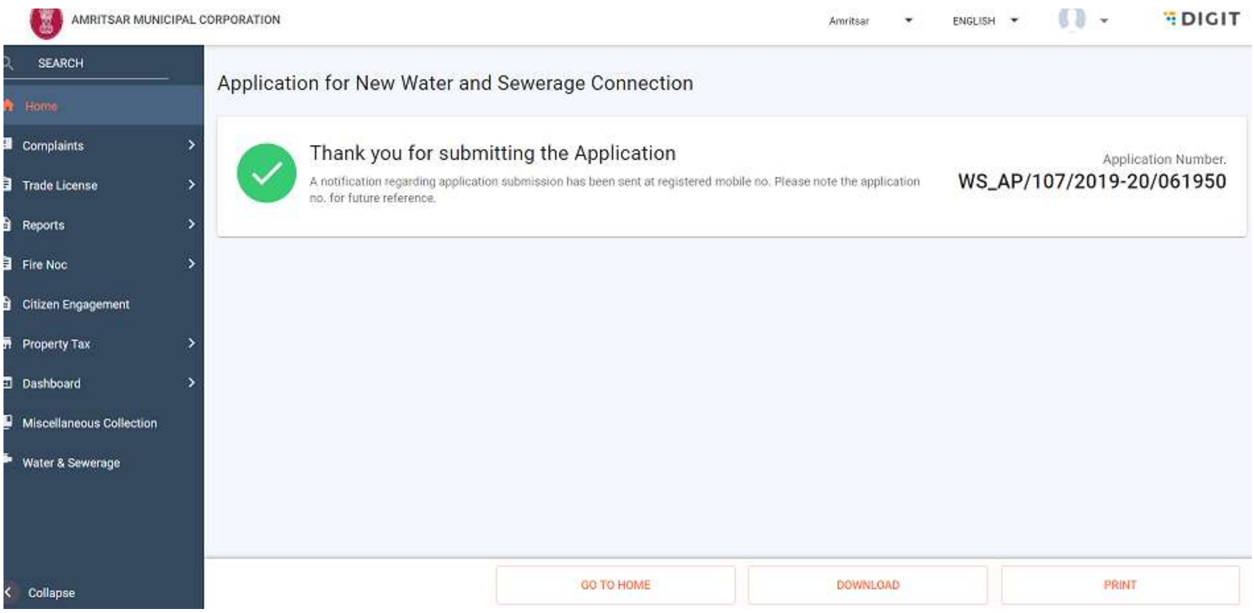
Required Documents

Only one file can be uploaded for one document.If multiple files need to be uploaded then please combine all files in a pdf and then upload

1	Identity Proof *	Select Documents * Select Documents
2	Address Proof *	Select Documents * Select Documents

All the applications that has been created once can be submitted (SUBMIT_APPLICATION state in Workflow), by either searching them in using their application number or employee can continue to the next stages and add all the fields required and they will be able to submit the application. Once an application is submitted, they will be redirected to a screen depicted in the below image.

A water application has been SUBMIT_APPLICATION (The Submit Application in Workflow)



Application for New Water and Sewerage Connection

Thank you for submitting the Application

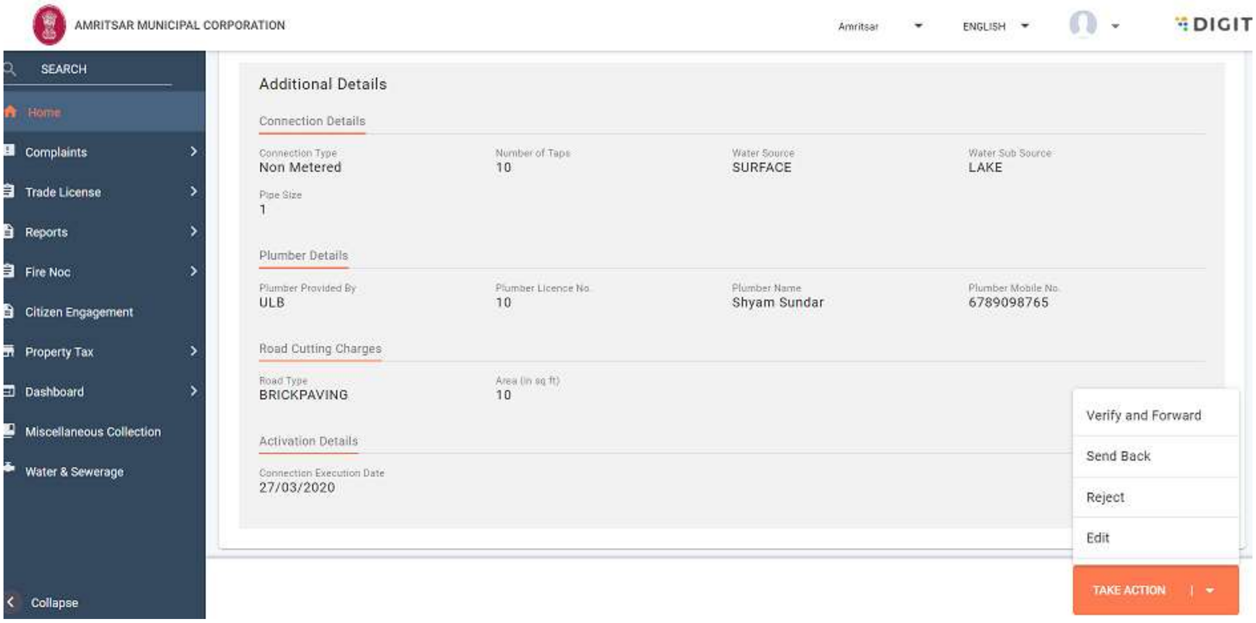
A notification regarding application submission has been sent at registered mobile no. Please note the application no. for future reference.

Application Number.
WS_AP/107/2019-20/061950

Here they will get an option to Download and Print the application, which they have just submitted. **An employee can create both Water and Sewerage application at once.**

VIEW APPLICATION & CONNECTION DETAILS

Here an Employee can edit the application, VERIFY AND FORWARD the application to the next stages, REJECT, SEND BACK TO CITIZEN who has applied for this connection. These actions that are seen in the below image, appear only for employees having a specific role which allows the employee to take the below actions.



Additional Details

Connection Details

Connection Type	Number of Taps	Water Source	Water Sub Source
Non Metered	10	SURFACE	LAKE

Pipe Size
1

Plumber Details

Plumber Provided By	Plumber Licence No.	Plumber Name	Plumber Mobile No.
ULB	10	Shyam Sundar	6789098765

Road Cutting Charges

Road Type	Area (in sq ft)
BRICKPAVING	10

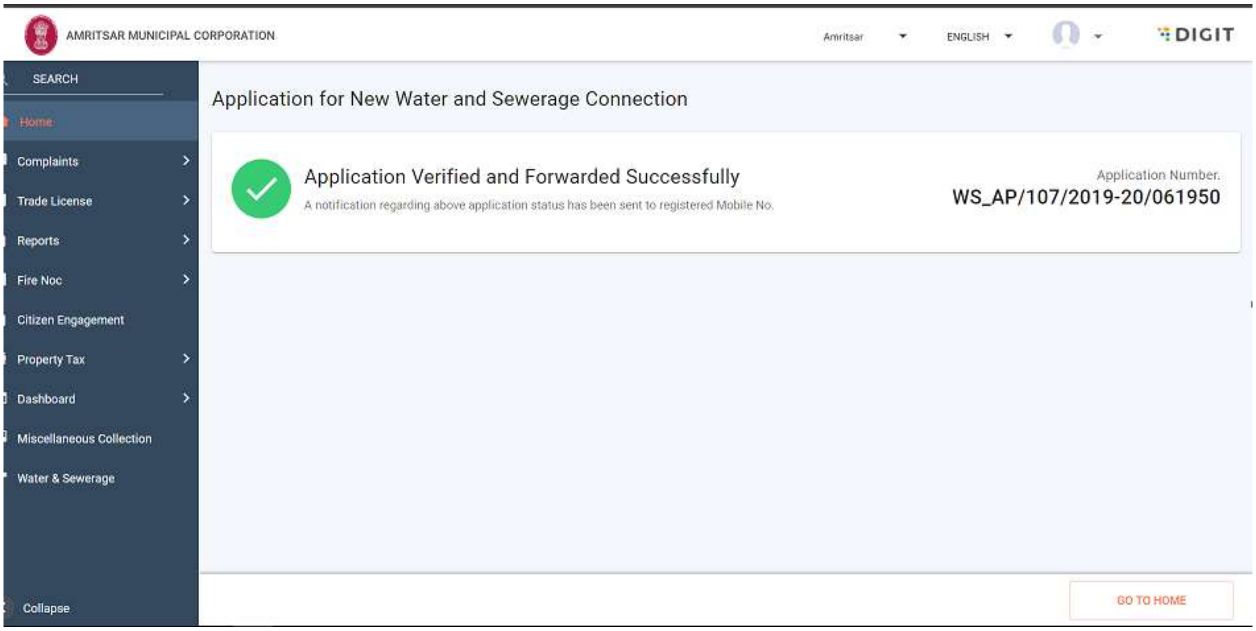
Activation Details

Connection Execution Date
27/03/2020

From the options provided in the above image, if the employee clicks on EDIT, it will be redirected to the apply screen where the employee will find the details of the application when it was last updated. The employee can click other option as well. It can also click on one of these options after editing the application and then it can do whatever that the employee deems right for the application.

Note: The employee will only get these options if it is authorized to take any of such actions provided in the above image.

On successful completion of any of the above processes, the employee will be redirected to the below screen. The messages may change based on the actions clicked. Here the action that I have taken is VERIFY_AND_FORWARD. It can be any of the action provided in the above image.



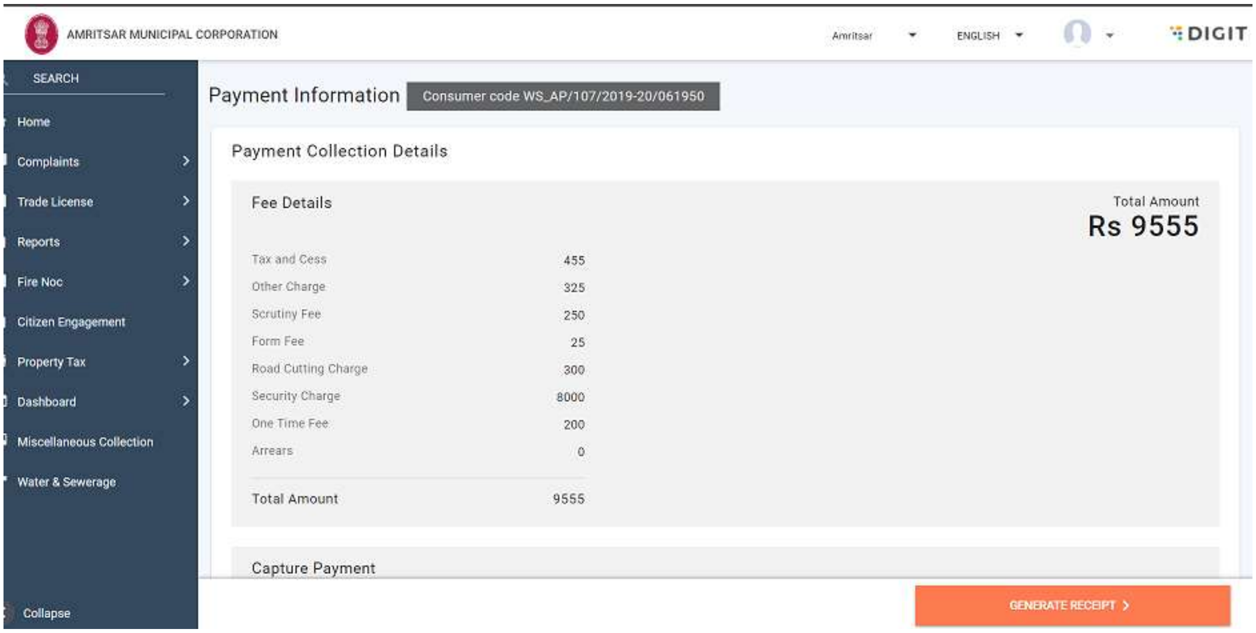
Application for New Water and Sewerage Connection

Application Verified and Forwarded Successfully

A notification regarding above application status has been sent to registered Mobile No.

Application Number.
WS_AP/107/2019-20/061950

On successful completion of all the states triggered from the actions taken, the employee will reach to the action as PAY. On click of Pay, the employee will be redirected to the below screen. Here the employee can generate receipt of the amount collected from citizen. Employees will be able to see the PAY option, if they are authorized to Collect Payment from citizen.



Payment Information Consumer code WS_AP/107/2019-20/061950

Payment Collection Details

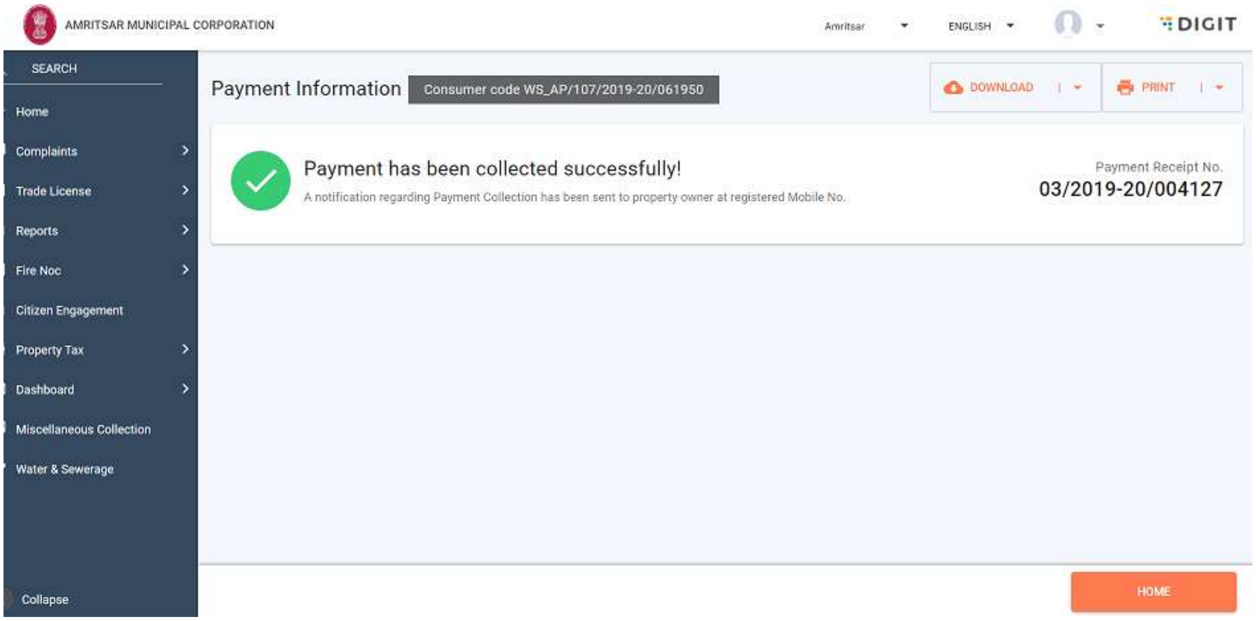
Fee Details

Total Amount
Rs 9555

Tax and Cess	455
Other Charge	325
Scrutiny Fee	250
Form Fee	25
Road Cutting Charge	300
Security Charge	8000
One Time Fee	200
Arrears	0
Total Amount	9555

Capture Payment

Once the payment is collected, the employee will be redirected to the below screen. Here employee can download and print receipt.



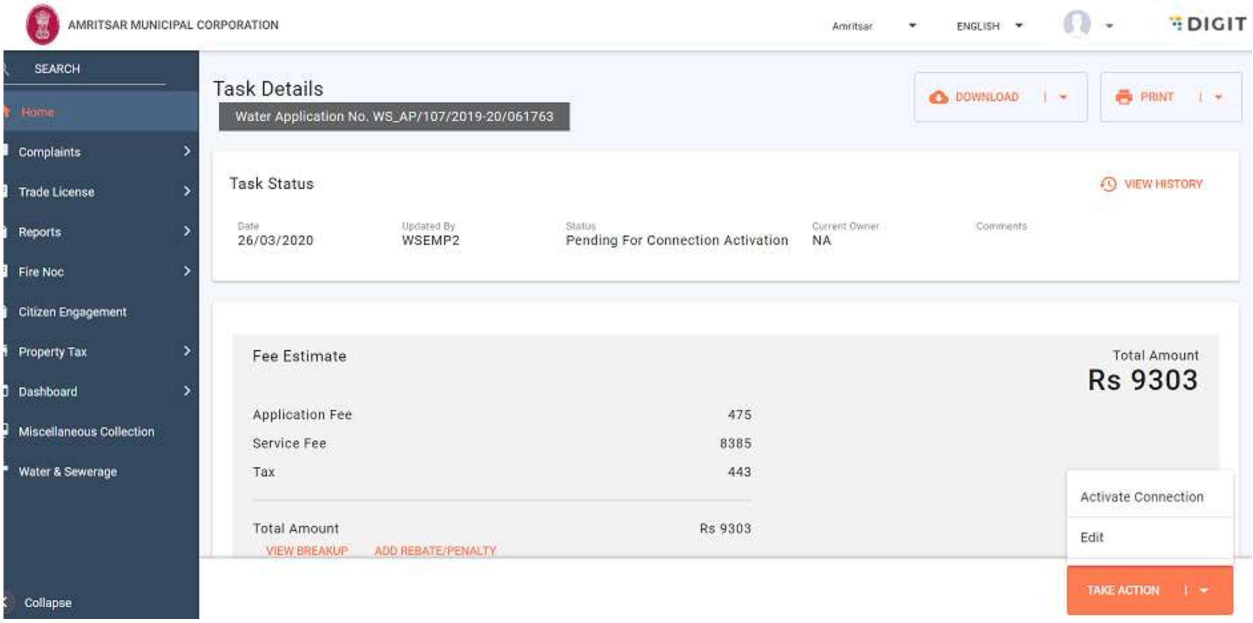
Payment Information Consumer code WS_AP/107/2019-20/061950

Payment has been collected successfully!

A notification regarding Payment Collection has been sent to property owner at registered Mobile No.

Payment Receipt No.
03/2019-20/004127

Once payment has been accepted and the receipt has been generated, the employee can go to search and search the application based on application number. The employee will find the application in Pending Connection Activation state, similar to what is show in the image below. If the employee is authorized to activate the connection, it can activate the connection.



Task Details
Water Application No. WS_AP/107/2019-20/061763

Task Status

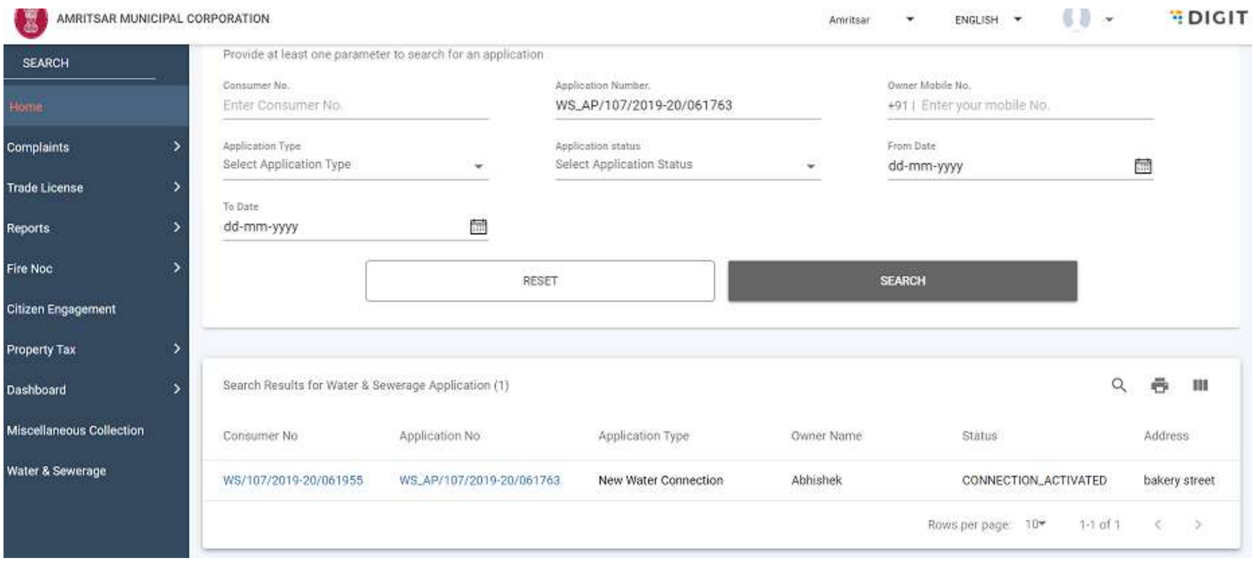
Date	Updated By	Status	Current Owner	Comments
26/03/2020	WSEMP2	Pending For Connection Activation	NA	

Fee Estimate

Application Fee	475
Service Fee	8385
Tax	443
Total Amount	Rs 9303

Total Amount
Rs 9303

After the connection has been activated, employee can go to the search and search the connection based on connection number, as shown in the image below.



Provide at least one parameter to search for an application

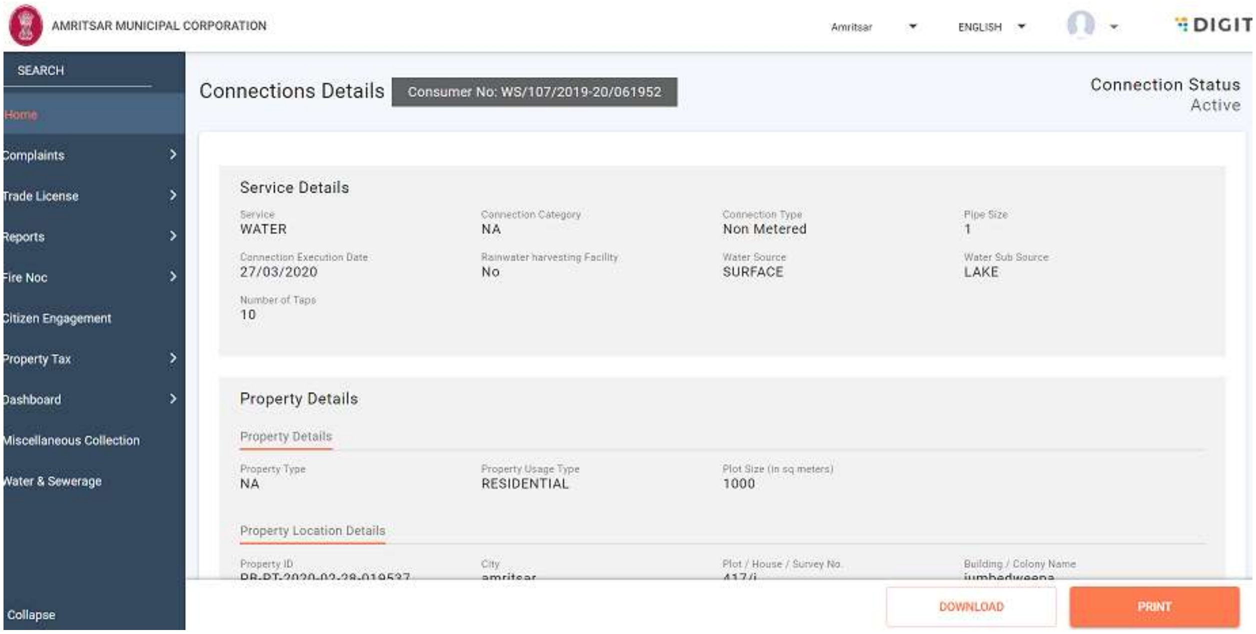
Consumer No.	Application Number.	Owner Mobile No.
Enter Consumer No.	WS_AP/107/2019-20/061763	+91 | Enter your mobile No.
Application Type	**Application status**	**From Date**
Select Application Type	Select Application Status	dd-mm-yyyy
To Date		
dd-mm-yyyy		

Search Results for Water & Sewerage Application (1)

Consumer No	Application No	Application Type	Owner Name	Status	Address
WS/107/2019-20/061955	WS_AP/107/2019-20/061763	New Water Connection	Abhishek	CONNECTION_ACTIVATED	bakery street

Rows per page: 10 1-1 of 1

After getting the search result as in the above image for the activate connection, the employee can see the details of the connection (Water or Sewerage) after clicking on the connection number in the Consumer No. column. The connection details looks like below, where the employee can find all the details related to the connection created.



Connections Details Consumer No: WS/107/2019-20/061952

Connection Status
Active

Service Details

Service	Connection Category	Connection Type	Pipe Size
WATER	NA	Non Metered	1
Connection Execution Date	**Rainwater harvesting Facility**	**Water Source**	**Water Sub Source**
27/03/2020	No	SURFACE	LAKE
Number of Taps			
10			

Property Details

Property Details

Property Type	Property Usage Type	Plot Size (in sq meters)
NA	RESIDENTIAL	1000

Property Location Details

Property ID	City	Plot / House / Survey No.	Building / Colony Name
PB-PT-2020-02-28-019537	amritsar	417/i	jumbedweena

Employee can also download and print the connection details if required.