TCS INFRAMIND

CASE-STUDY: Technocrat Info Solutions_Cloud-Computing

By

D. Chanakya Srinivas (CT20162041536)

B. Sai Sowjanya (CT20172265661)

Youtube link:

https://youtu.be/xwb8C7SOqqI

Parameters defined in the script are:

- Environment name
- CIDR for VPC, public subnet and private subnet
- Keyname for the keypair use for the instances
- DB size(default 5GB)
- DB user, password, root password
- Instance type
- SSH location (default 0.0.0.0/0)
- Mappings used for the zones of AMI

Resources defined in the script are

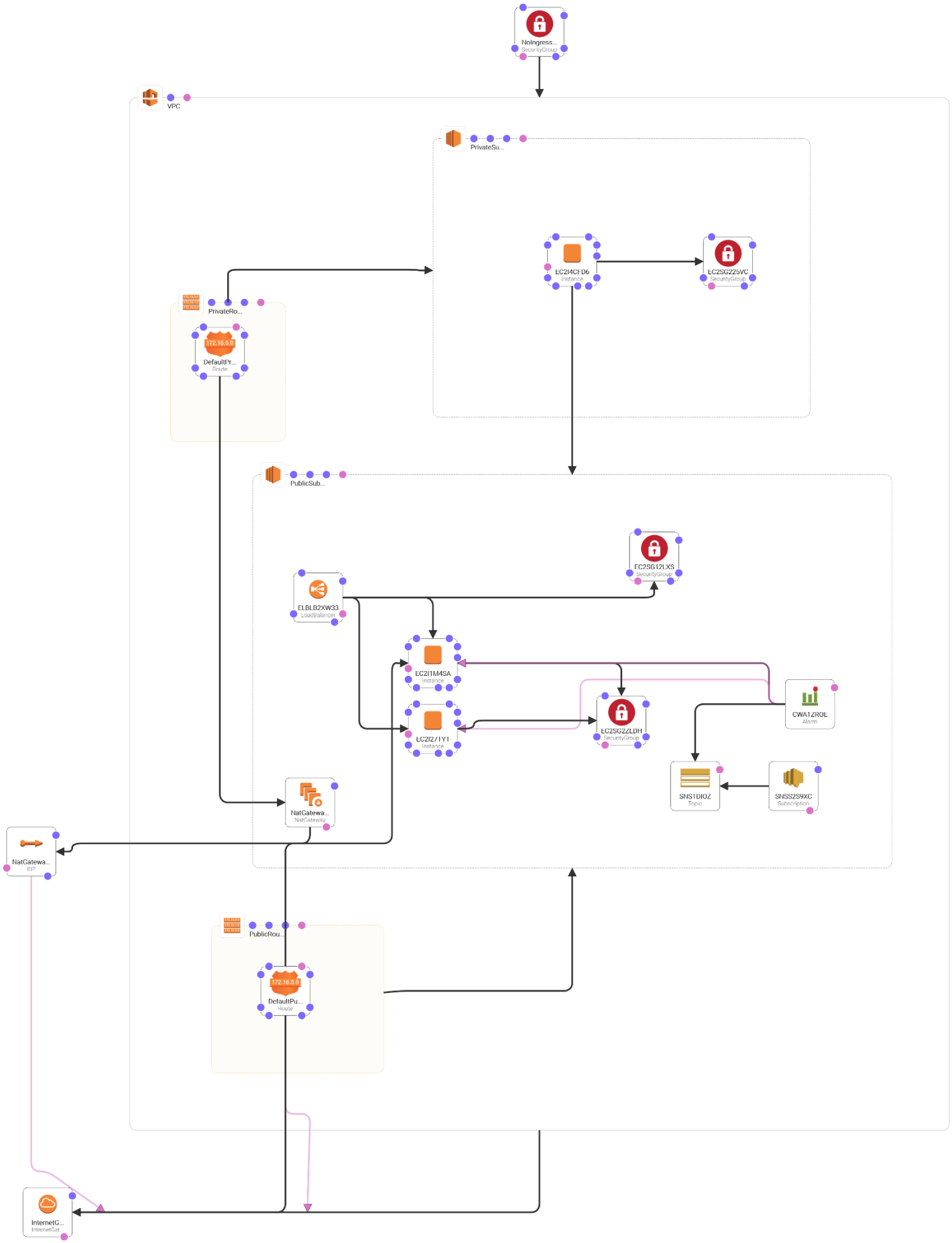
NETWORK RESOURCES:

- VPC (with the same name as environment name)
- Internet gateway
- Internet gateway attachment
- Public subnet
- Private subnet (differs from public subnet based on the value of MapPubliciponLaunch either True or False)
- Network Gateway
- Public and private route tables
- Public and private route table associations

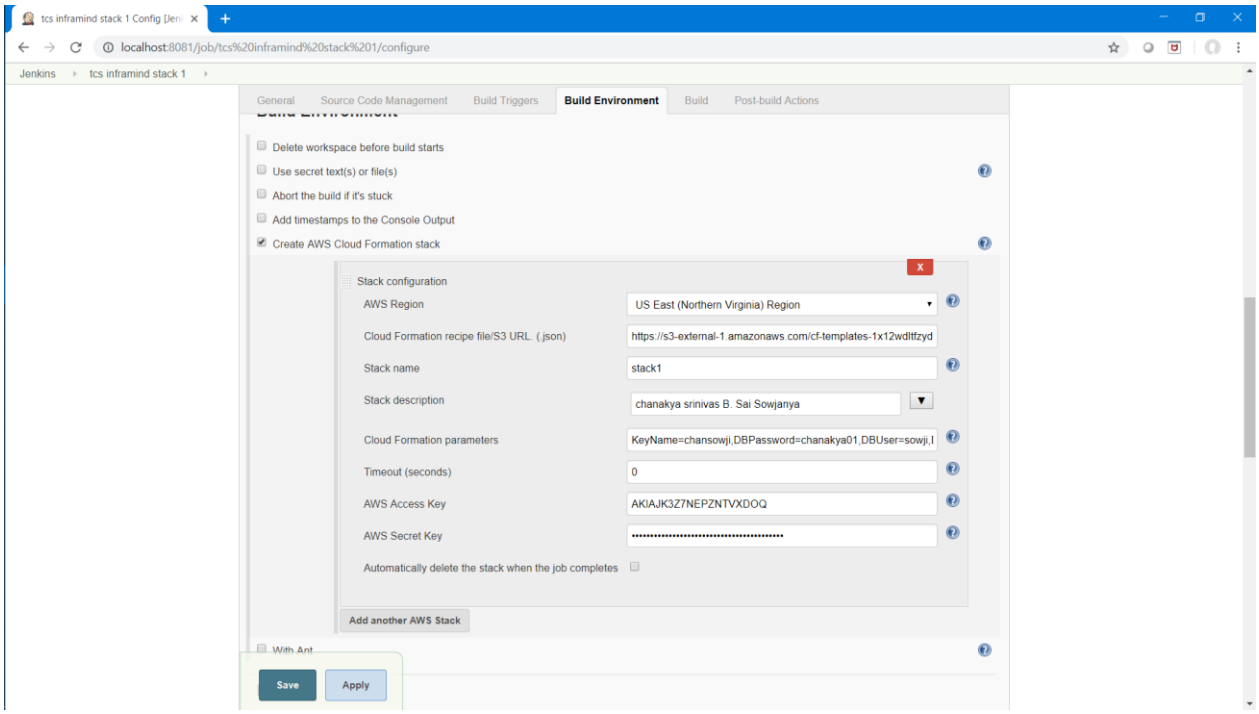
COMPUTING RESOURCES:

- EC2 Instances
- Security Groups
- CloudWatch alarms
- Block Storage for backup
- SNS topic and subscription endpoint
- Load Balancer
- Lambda and Dynamo DB for instance scheduler

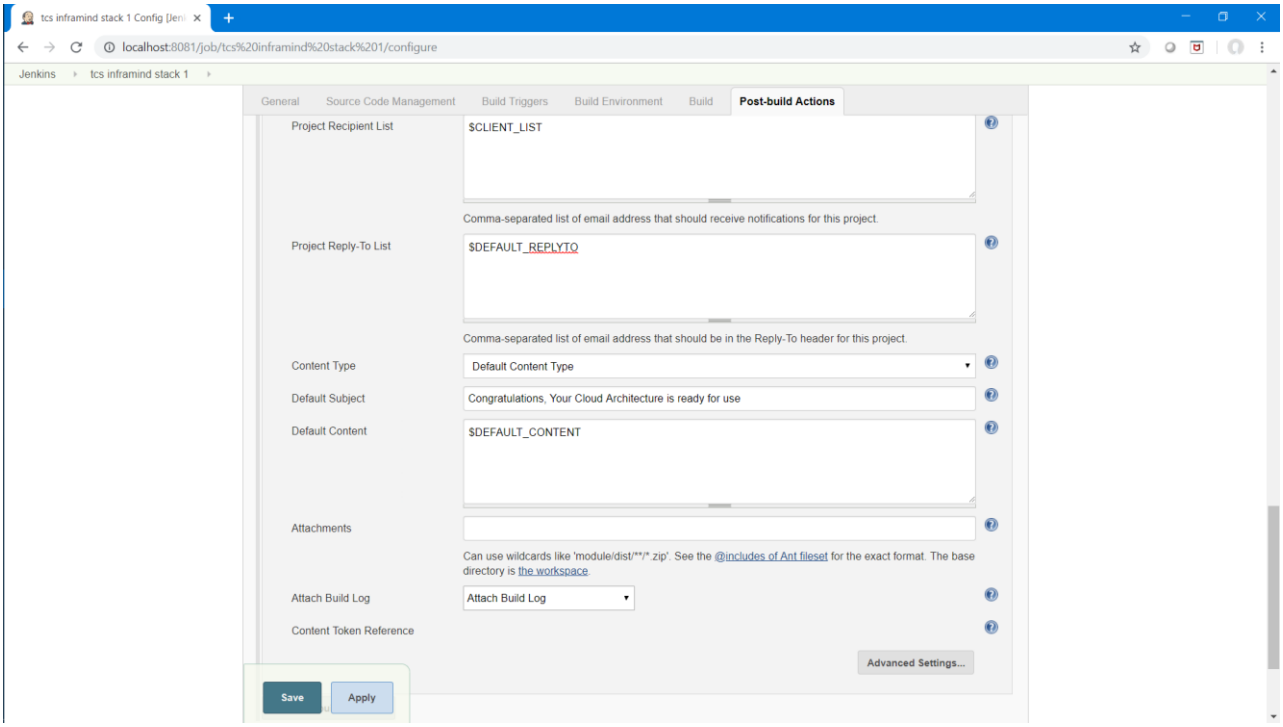
Architecture



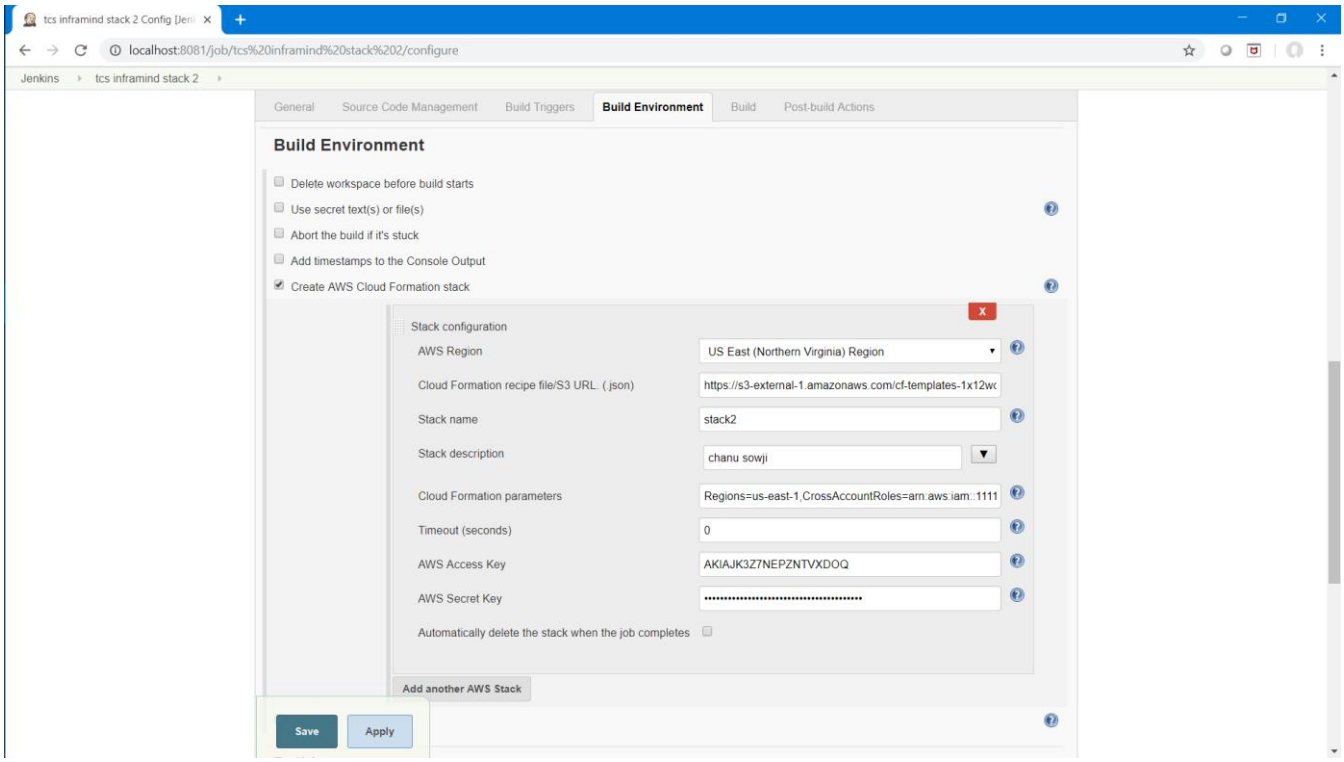
Jenkins Screenshots



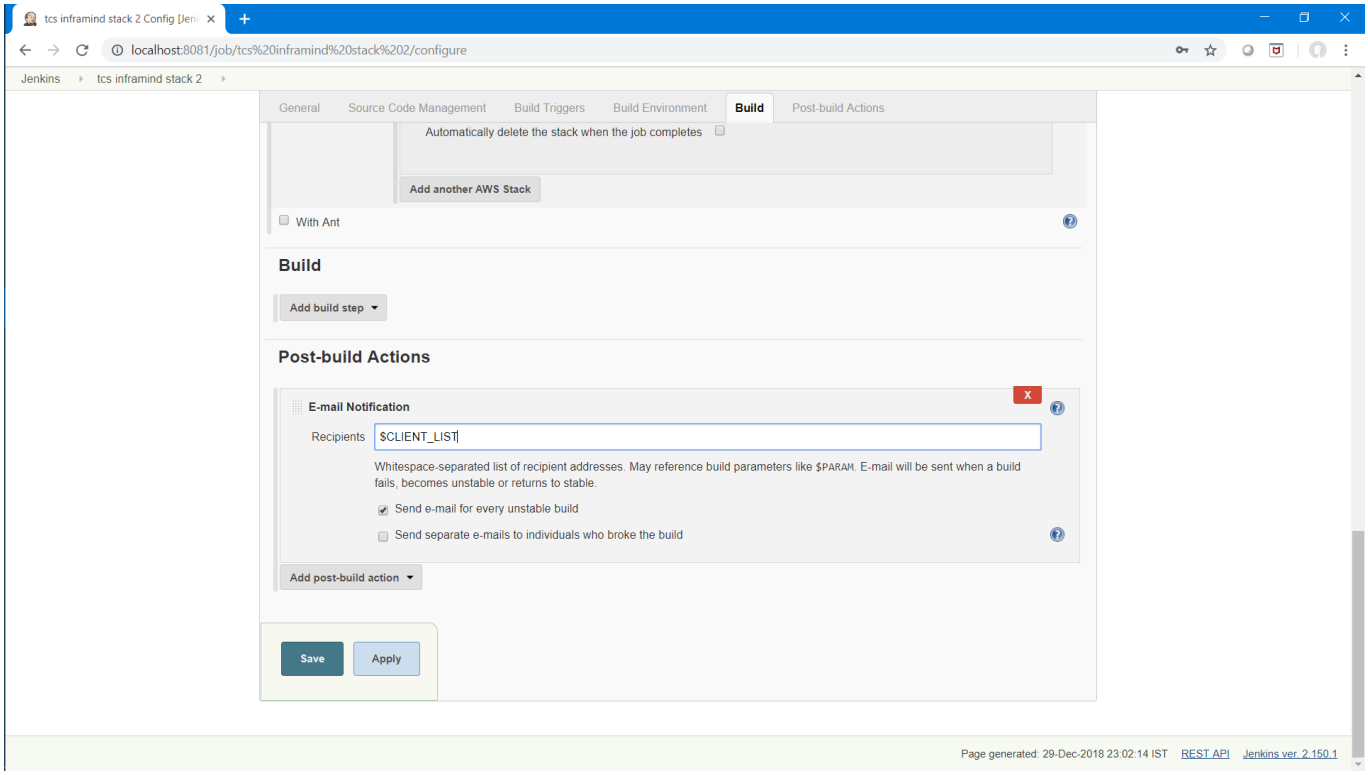
- Stack1 build environment



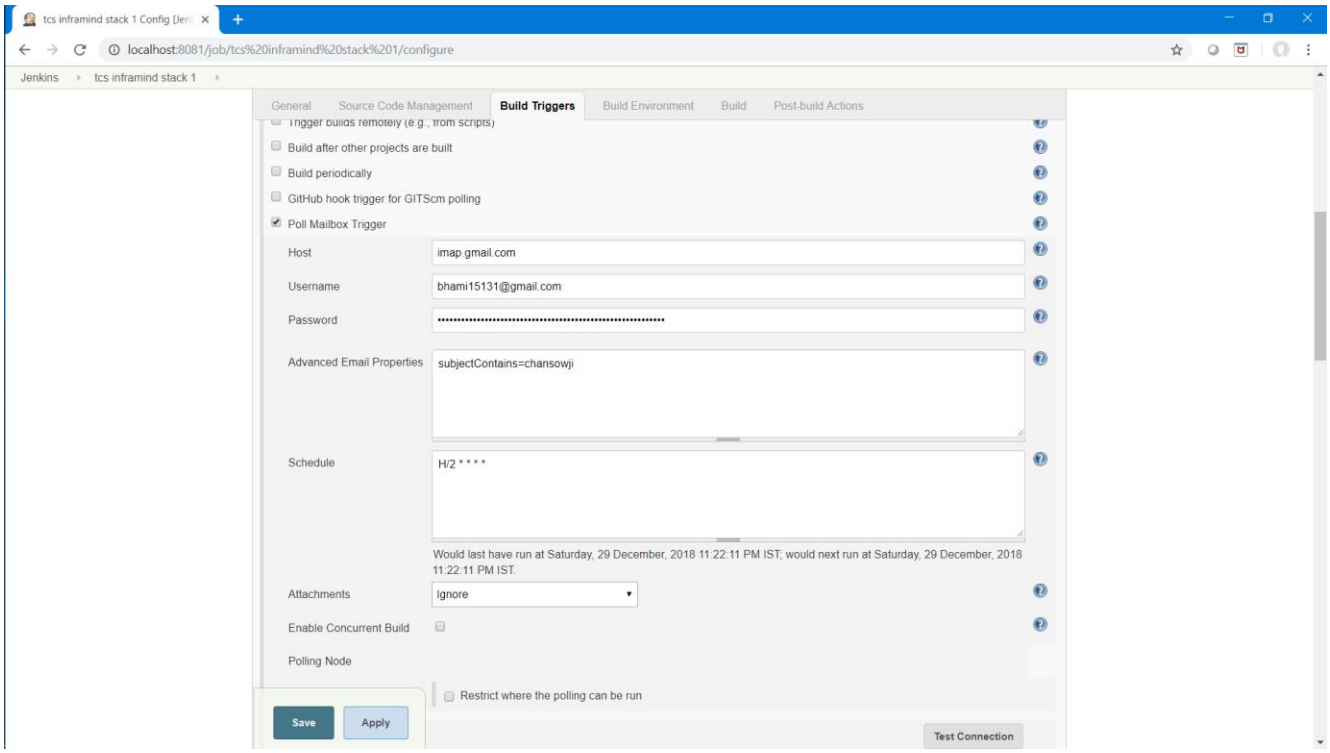
- Mail Notification
- Environment variable $CLIENT_LIST contains the client mail address



- Stack2 Build Environment



- Mail Notification



- Poll Mail Trigger setup

- We have Jenkins for automation as it is an open source, it is customizable and based on java, which enables us to design custom plugins.

- The Cloud-Formation script is saved with .json extension and uploaded into s3 with external private access so that only listed accounts can R/W

- For Mail notification and mail trigger admin mail is used

  Admin mail : bhami15131@gmail.com

  Allow less secure apps is enabled for this account.

Cloud-Formation JSON script:

STACK 1


```
{
   "Description": "This template deploys a VPC, with public and private subnets. It deploys an Internet Gateway, with a default route on the public subnets. It deploys a pair of NAT Gateways and default routes for them in the private subnets. A pair of Webserver EC2 instance and a Database server ",
   "Parameters": {
      "EnvironmentName": {
         "Description": "An environment name that will be prefixed to resource names",
         "Type": "String",
         "Default": "chansowjiinfra"
      },
      "VpcCIDR": {
         "Description": "Please enter the IP range (CIDR notation) for this VPC",
         "Type": "String",
         "Default": "10.192.0.0/16"
      },
      "PublicSubnet1CIDR": {
         "Description": "cidr for public subnet",
         "Type": "String",
         "Default": "10.192.10.0/24"
      },
      "PrivateSubnet1CIDR": {
         "Description": "cidr for private subnet",
         "Type": "String",
         "Default": "10.192.20.0/24"
      },
      "KeyName": {
         "Description": "Name of an existing EC2 KeyPair to enable SSH access to the instance",
         "Type": "AWS::EC2::KeyPair::KeyName",
         "ConstraintDescription": "must be the name of an existing EC2 KeyPair."
      },
      "DBAllocatedStorage": {
         "Default": "5",
```

```
    "Description": "The size of the database (Gb)",
    "Type": "Number",
    "MinValue": "5",
    "MaxValue": "1024",
    "ConstraintDescription": "must be between 5 and 1024Gb."
},
"DBInstance": {
    "Description": "The database instance type",
    "Type": "String",
    "Default": "db.t2.micro",
    "AllowedValues": [
        "db.t1.micro",
        "db.m1.small",
        "db.m1.medium",
        "db.m1.large",
        "db.m1.xlarge",
        "db.m2.xlarge",
        "db.m2.2xlarge",
        "db.m2.4xlarge",
        "db.m3.medium",
        "db.m3.large",
        "db.m3.xlarge",
        "db.m3.2xlarge",
        "db.m4.large",
        "db.m4.xlarge",
        "db.m4.2xlarge",
        "db.m4.4xlarge",
        "db.m4.10xlarge",
        "db.r3.large",
        "db.r3.xlarge",
        "db.r3.2xlarge",
        "db.r3.4xlarge",
        "db.r3.8xlarge",
        "db.m2.xlarge",
        "db.m2.2xlarge",
```

```
        "db.m2.4xlarge",
        "db.cr1.8xlarge",
        "db.t2.micro",
        "db.t2.small",
        "db.t2.medium",
        "db.t2.large"
      ],
      "ConstraintDescription": "must select a valid database instance type."
    },
    "DBName": {
      "Default": "MyDatabase",
      "Description": "MySQL database name",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "64",
      "AllowedPattern": "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription": "must begin with a letter and contain only alphanumeric characters."
    },
    "DBUser": {
      "NoEcho": "true",
      "Description": "Username for MySQL database access",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "16",
      "AllowedPattern": "[a-zA-Z][a-zA-Z0-9]*",
      "ConstraintDescription": "must begin with a letter and contain only alphanumeric characters."
    },
    "DBPassword": {
      "NoEcho": "true",
      "Description": "Password for MySQL database access",
      "Type": "String",
      "MinLength": "1",
      "MaxLength": "41",
      "AllowedPattern": "[a-zA-Z0-9]*",
      "ConstraintDescription": "must contain only alphanumeric characters."
```

```
},
"DBRootPassword": {
   "NoEcho": "true",
   "Description": "Root password for MySQL",
   "Type": "String",
   "MinLength": "1",
   "MaxLength": "41",
   "AllowedPattern": "[a-zA-Z0-9]*",
   "ConstraintDescription": "must contain only alphanumeric characters."
},
"InstanceType": {
   "Description": "WebServer EC2 instance type",
   "Type": "String",
   "Default": "t2.micro",
   "AllowedValues": [
      "t1.micro",
      "t2.nano",
      "t2.micro",
      "t2.small",
      "t2.medium",
      "t2.large",
      "m1.small",
      "m1.medium",
      "m1.large",
      "m1.xlarge",
      "m2.xlarge",
      "m2.2xlarge",
      "m2.4xlarge",
      "m3.medium",
      "m3.large",
      "m3.xlarge",
      "m3.2xlarge",
      "m4.large",
      "m4.xlarge",
      "m4.2xlarge",
```

```
    "m4.4xlarge",
    "m4.10xlarge",
    "c1.medium",
    "c1.xlarge",
    "c3.large",
    "c3.xlarge",
    "c3.2xlarge",
    "c3.4xlarge",
    "c3.8xlarge",
    "c4.large",
    "c4.xlarge",
    "c4.2xlarge",
    "c4.4xlarge",
    "c4.8xlarge",
    "g2.2xlarge",
    "g2.8xlarge",
    "r3.large",
    "r3.xlarge",
    "r3.2xlarge",
    "r3.4xlarge",
    "r3.8xlarge",
    "i2.xlarge",
    "i2.2xlarge",
    "i2.4xlarge",
    "i2.8xlarge",
    "d2.xlarge",
    "d2.2xlarge",
    "d2.4xlarge",
    "d2.8xlarge",
    "hi1.4xlarge",
    "hs1.8xlarge",
    "cr1.8xlarge",
    "cc2.8xlarge",
    "cg1.4xlarge"
],
```

```
      "ConstraintDescription": "must be a valid EC2 instance type."
    },
    "SSHLocation": {
      "Description": " The IP address range that can be used to SSH to the EC2 instances",
      "Type": "String",
      "MinLength": "9",
      "MaxLength": "18",
      "Default": "0.0.0.0/0",
      "AllowedPattern": "(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})\\.(\\d{1,3})/(\\d{1,2})",
      "ConstraintDescription": "must be a valid IP CIDR range of the form x.x.x.x/x."
    }
  },
  "Mappings": {
    "AWSInstanceType2Arch": {
      "t1.micro": {
        "Arch": "HVM64"
      },
      "t2.nano": {
        "Arch": "HVM64"
      },
      "t2.micro": {
        "Arch": "HVM64"
      },
      "t2.small": {
        "Arch": "HVM64"
      },
      "t2.medium": {
        "Arch": "HVM64"
      },
      "t2.large": {
        "Arch": "HVM64"
      },
      "m1.small": {
        "Arch": "HVM64"
      },
```

```
"m1.medium": {
   "Arch": "HVM64"
},
"m1.large": {
   "Arch": "HVM64"
},
"m1.xlarge": {
   "Arch": "HVM64"
},
"m2.xlarge": {
   "Arch": "HVM64"
},
"m2.2xlarge": {
   "Arch": "HVM64"
},
"m2.4xlarge": {
   "Arch": "HVM64"
},
"m3.medium": {
   "Arch": "HVM64"
},
"m3.large": {
   "Arch": "HVM64"
},
"m3.xlarge": {
   "Arch": "HVM64"
},
"m3.2xlarge": {
   "Arch": "HVM64"
},
"m4.large": {
   "Arch": "HVM64"
},
"m4.xlarge": {
   "Arch": "HVM64"
```

```
},
"m4.2xlarge": {
   "Arch": "HVM64"
},
"m4.4xlarge": {
   "Arch": "HVM64"
},
"m4.10xlarge": {
   "Arch": "HVM64"
},
"c1.medium": {
   "Arch": "HVM64"
},
"c1.xlarge": {
   "Arch": "HVM64"
},
"c3.large": {
   "Arch": "HVM64"
},
"c3.xlarge": {
   "Arch": "HVM64"
},
"c3.2xlarge": {
   "Arch": "HVM64"
},
"c3.4xlarge": {
   "Arch": "HVM64"
},
"c3.8xlarge": {
   "Arch": "HVM64"
},
"c4.large": {
   "Arch": "HVM64"
},
"c4.xlarge": {
```

```
    "Arch": "HVM64"
  },
  "c4.2xlarge": {
    "Arch": "HVM64"
  },
  "c4.4xlarge": {
    "Arch": "HVM64"
  },
  "c4.8xlarge": {
    "Arch": "HVM64"
  },
  "g2.2xlarge": {
    "Arch": "HVMG2"
  },
  "g2.8xlarge": {
    "Arch": "HVMG2"
  },
  "r3.large": {
    "Arch": "HVM64"
  },
  "r3.xlarge": {
    "Arch": "HVM64"
  },
  "r3.2xlarge": {
    "Arch": "HVM64"
  },
  "r3.4xlarge": {
    "Arch": "HVM64"
  },
  "r3.8xlarge": {
    "Arch": "HVM64"
  },
  "i2.xlarge": {
    "Arch": "HVM64"
  },
```

```
    "i2.2xlarge": {
      "Arch": "HVM64"
    },
    "i2.4xlarge": {
      "Arch": "HVM64"
    },
    "i2.8xlarge": {
      "Arch": "HVM64"
    },
    "d2.xlarge": {
      "Arch": "HVM64"
    },
    "d2.2xlarge": {
      "Arch": "HVM64"
    },
    "d2.4xlarge": {
      "Arch": "HVM64"
    },
    "d2.8xlarge": {
      "Arch": "HVM64"
    },
    "hi1.4xlarge": {
      "Arch": "HVM64"
    },
    "hs1.8xlarge": {
      "Arch": "HVM64"
    },
    "cr1.8xlarge": {
      "Arch": "HVM64"
    },
    "cc2.8xlarge": {
      "Arch": "HVM64"
    }
  },
  "AWSInstanceType2NATArch": {
```

```
"t1.micro": {
   "Arch": "NATHVM64"
},
"t2.nano": {
   "Arch": "NATHVM64"
},
"t2.micro": {
   "Arch": "NATHVM64"
},
"t2.small": {
   "Arch": "NATHVM64"
},
"t2.medium": {
   "Arch": "NATHVM64"
},
"t2.large": {
   "Arch": "NATHVM64"
},
"m1.small": {
   "Arch": "NATHVM64"
},
"m1.medium": {
   "Arch": "NATHVM64"
},
"m1.large": {
   "Arch": "NATHVM64"
},
"m1.xlarge": {
   "Arch": "NATHVM64"
},
"m2.xlarge": {
   "Arch": "NATHVM64"
},
"m2.2xlarge": {
   "Arch": "NATHVM64"
```

```
},
"m2.4xlarge": {
   "Arch": "NATHVM64"
},
"m3.medium": {
   "Arch": "NATHVM64"
},
"m3.large": {
   "Arch": "NATHVM64"
},
"m3.xlarge": {
   "Arch": "NATHVM64"
},
"m3.2xlarge": {
   "Arch": "NATHVM64"
},
"m4.large": {
   "Arch": "NATHVM64"
},
"m4.xlarge": {
   "Arch": "NATHVM64"
},
"m4.2xlarge": {
   "Arch": "NATHVM64"
},
"m4.4xlarge": {
   "Arch": "NATHVM64"
},
"m4.10xlarge": {
   "Arch": "NATHVM64"
},
"c1.medium": {
   "Arch": "NATHVM64"
},
"c1.xlarge": {
```

```
    "Arch": "NATHVM64"
},
"c3.large": {
    "Arch": "NATHVM64"
},
"c3.xlarge": {
    "Arch": "NATHVM64"
},
"c3.2xlarge": {
    "Arch": "NATHVM64"
},
"c3.4xlarge": {
    "Arch": "NATHVM64"
},
"c3.8xlarge": {
    "Arch": "NATHVM64"
},
"c4.large": {
    "Arch": "NATHVM64"
},
"c4.xlarge": {
    "Arch": "NATHVM64"
},
"c4.2xlarge": {
    "Arch": "NATHVM64"
},
"c4.4xlarge": {
    "Arch": "NATHVM64"
},
"c4.8xlarge": {
    "Arch": "NATHVM64"
},
"g2.2xlarge": {
    "Arch": "NATHVMG2"
},
```

```
"g2.8xlarge": {
   "Arch": "NATHVMG2"
},
"r3.large": {
   "Arch": "NATHVM64"
},
"r3.xlarge": {
   "Arch": "NATHVM64"
},
"r3.2xlarge": {
   "Arch": "NATHVM64"
},
"r3.4xlarge": {
   "Arch": "NATHVM64"
},
"r3.8xlarge": {
   "Arch": "NATHVM64"
},
"i2.xlarge": {
   "Arch": "NATHVM64"
},
"i2.2xlarge": {
   "Arch": "NATHVM64"
},
"i2.4xlarge": {
   "Arch": "NATHVM64"
},
"i2.8xlarge": {
   "Arch": "NATHVM64"
},
"d2.xlarge": {
   "Arch": "NATHVM64"
},
"d2.2xlarge": {
   "Arch": "NATHVM64"
```

```
    },
    "d2.4xlarge": {
      "Arch": "NATHVM64"
    },
    "d2.8xlarge": {
      "Arch": "NATHVM64"
    },
    "hi1.4xlarge": {
      "Arch": "NATHVM64"
    },
    "hs1.8xlarge": {
      "Arch": "NATHVM64"
    },
    "cr1.8xlarge": {
      "Arch": "NATHVM64"
    },
    "cc2.8xlarge": {
      "Arch": "NATHVM64"
    }
  },
  "AWSRegionArch2AMI": {
    "us-east-1": {
      "HVM64": "ami-0ff8a91507f77f867",
      "HVMG2": "ami-0a584ac55a7631c0c"
    },
    "us-west-2": {
      "HVM64": "ami-a0cfeed8",
      "HVMG2": "ami-0e09505bc235aa82d"
    },
    "us-west-1": {
      "HVM64": "ami-0bdb828fd58c52235",
      "HVMG2": "ami-066ee5fd4a9ef77f1"
    },
    "eu-west-1": {
      "HVM64": "ami-047bb4163c506cd98",
```

```
    "HVMG2": "ami-0a7c483d527806435"
  },
  "eu-west-2": {
    "HVM64": "ami-f976839e",
    "HVMG2": "NOT_SUPPORTED"
  },
  "eu-west-3": {
    "HVM64": "ami-0ebc281c20e89ba4b",
    "HVMG2": "NOT_SUPPORTED"
  },
  "eu-central-1": {
    "HVM64": "ami-0233214e13e500f77",
    "HVMG2": "ami-06223d46a6d0661c7"
  },
  "ap-northeast-1": {
    "HVM64": "ami-06cd52961ce9f0d85",
    "HVMG2": "ami-053cdd503598e4a9d"
  },
  "ap-northeast-2": {
    "HVM64": "ami-0a10b2721688ce9d2",
    "HVMG2": "NOT_SUPPORTED"
  },
  "ap-northeast-3": {
    "HVM64": "ami-0d98120a9fb693f07",
    "HVMG2": "NOT_SUPPORTED"
  },
  "ap-southeast-1": {
    "HVM64": "ami-08569b978cc4dfa10",
    "HVMG2": "ami-0be9df32ae9f92309"
  },
  "ap-southeast-2": {
    "HVM64": "ami-09b42976632b27e9b",
    "HVMG2": "ami-0a9ce9fecc3d1daf8"
  },
  "ap-south-1": {
```

```
        "HVM64": "ami-0912f71e06545ad88",
        "HVMG2": "ami-097b15e89dbdcfcf4"
      },
      "us-east-2": {
        "HVM64": "ami-0b59bfac6be064b78",
        "HVMG2": "NOT_SUPPORTED"
      },
      "ca-central-1": {
        "HVM64": "ami-0b18956f",
        "HVMG2": "NOT_SUPPORTED"
      },
      "sa-east-1": {
        "HVM64": "ami-07b14488da8ea02a0",
        "HVMG2": "NOT_SUPPORTED"
      },
      "cn-north-1": {
        "HVM64": "ami-0a4eaf6c4454eda75",
        "HVMG2": "NOT_SUPPORTED"
      },
      "cn-northwest-1": {
        "HVM64": "ami-6b6a7d09",
        "HVMG2": "NOT_SUPPORTED"
      }
    }
  },
  "Resources": {
    "VPC": {
      "Type": "AWS::EC2::VPC",
      "Properties": {
        "CidrBlock": {
          "Ref": "VpcCIDR"
        },
        "EnableDnsSupport": true,
        "EnableDnsHostnames": true,
        "Tags": [
```

```
        {
          "Key": "Name",
          "Value": {
            "Ref": "EnvironmentName"
          }
        }
      ]
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "9af6bbd2-3079-47c0-9948-fe7929d70a3c"
      }
    }
  },
  "InternetGateway": {
    "Type": "AWS::EC2::InternetGateway",
    "Properties": {
      "Tags": [
        {
          "Key": "Name",
          "Value": {
            "Ref": "EnvironmentName"
          }
        }
      ]
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "dcf669e1-aa34-4dbd-bf85-176826b84a6e"
      }
    }
  },
  "InternetGatewayAttachment": {
    "Type": "AWS::EC2::VPCGatewayAttachment",
    "Properties": {
```

```
      "InternetGatewayId": {
        "Ref": "InternetGateway"
      },
      "VpcId": {
        "Ref": "VPC"
      }
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "c2217419-b20e-4240-9e85-ae95498e91a1"
      }
    }
  },
  "PublicSubnet1": {
    "Type": "AWS::EC2::Subnet",
    "Properties": {
      "VpcId": {
        "Ref": "VPC"
      },
      "AvailabilityZone": {
        "Fn::Select": [
          "0",
          {
            "Fn::GetAZs": {
              "Ref": "AWS::Region"
            }
          }
        ]
      },
      "CidrBlock": {
        "Ref": "PublicSubnet1CIDR"
      },
      "MapPublicIpOnLaunch": true,
      "Tags": [
        {
```

```
            "Key": "Name",
            "Value": {
              "Fn::Sub": "${EnvironmentName} Public Subnet"
            }
          }
        ]
      },
      "Metadata": {
        "AWS::CloudFormation::Designer": {
          "id": "5350852a-6e99-47bd-a848-44f609af5ab1"
        }
      }
    },
    "PrivateSubnet1": {
      "Type": "AWS::EC2::Subnet",
      "Properties": {
        "VpcId": {
          "Ref": "VPC"
        },
        "AvailabilityZone": {
          "Fn::Select": [
            "0",
            {
              "Fn::GetAZs": {
                "Ref": "AWS::Region"
              }
            }
          ]
        },
        "CidrBlock": {
          "Ref": "PrivateSubnet1CIDR"
        },
        "MapPublicIpOnLaunch": false,
        "Tags": [
          {
```

```
            "Key": "Name",
            "Value": {
              "Fn::Sub": "${EnvironmentName} Private Subnet"
            }
          }
        ]
      },
      "Metadata": {
        "AWS::CloudFormation::Designer": {
          "id": "bff7432e-8d88-451e-b3d2-f8710e825c1e"
        }
      }
    },
    "NatGateway1EIP": {
      "Type": "AWS::EC2::EIP",
      "DependsOn": "InternetGatewayAttachment",
      "Properties": {
        "Domain": "vpc"
      },
      "Metadata": {
        "AWS::CloudFormation::Designer": {
          "id": "45f7b637-8d7d-4a6d-a88b-839e69beb37f"
        }
      }
    },
    "NatGateway1": {
      "Type": "AWS::EC2::NatGateway",
      "Properties": {
        "AllocationId": {
          "Fn::GetAtt": [
            "NatGateway1EIP",
            "AllocationId"
          ]
        },
        "SubnetId": {
```

```
            "Ref": "PublicSubnet1"
          }
        },
        "Metadata": {
          "AWS::CloudFormation::Designer": {
            "id": "b709b2b6-efdf-4817-a045-49b08a98847a"
          }
        }
      },
      "PublicRouteTable": {
        "Type": "AWS::EC2::RouteTable",
        "Properties": {
          "VpcId": {
            "Ref": "VPC"
          },
          "Tags": [
            {
              "Key": "Name",
              "Value": {
                "Fn::Sub": "${EnvironmentName} Public Routes"
              }
            }
          ]
        },
        "Metadata": {
          "AWS::CloudFormation::Designer": {
            "id": "2e5a7ad7-b14e-4c27-bdf3-c49d0259b590"
          }
        }
      },

      "PublicSubnet1RouteTableAssociation": {
        "Type": "AWS::EC2::SubnetRouteTableAssociation",
        "Properties": {
          "RouteTableId": {
```

```
        "Ref": "PublicRouteTable"
      },
      "SubnetId": {
        "Ref": "PublicSubnet1"
      }
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "6a07f4ea-2806-49ed-83ed-7b2ef592c486"
      }
    }
  },
  "PrivateRouteTable1": {
    "Type": "AWS::EC2::RouteTable",
    "Properties": {
      "VpcId": {
        "Ref": "VPC"
      },
      "Tags": [
        {
          "Key": "Name",
          "Value": {
            "Fn::Sub": "${EnvironmentName} Private Routes (AZ1)"
          }
        }
      ]
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "6024b332-5ba1-4af2-a579-0dc861c29198"
      }
    }
  },

  "PrivateSubnet1RouteTableAssociation": {
```

```
    "Type": "AWS::EC2::SubnetRouteTableAssociation",
    "Properties": {
      "RouteTableId": {
        "Ref": "PrivateRouteTable1"
      },
      "SubnetId": {
        "Ref": "PrivateSubnet1"
      }
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "6cfc7568-6e8b-42f2-a5b2-a95dc0c978eb"
      }
    }
  },
  "NoIngressSecurityGroup": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupName": "no-ingress-sg",
      "GroupDescription": "Security group with no ingress rule",
      "VpcId": {
        "Ref": "VPC"
      }
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "5f4e3a05-c33b-43ea-af6b-3e3e6f1653aa"
      }
    }
  },
  "EC2I27TYT": {
    "Type": "AWS::EC2::Instance",
    "Properties": {
      "ImageId": {
        "Fn::FindInMap": [
```

```
      "AWSRegionArch2AMI",
      {
        "Ref": "AWS::Region"
      },
      {
        "Fn::FindInMap": [
          "AWSInstanceType2Arch",
          {
            "Ref": "InstanceType"
          },
          "Arch"
        ]
      }
    ]
  },
  "InstanceType": {
    "Ref": "InstanceType"
  },
  "SecurityGroups": [
    {
      "Ref": "EC2SG2ZLDH"
    }
  ],
  "KeyName": {
    "Ref": "KeyName"
  },
  "SubnetId": {
    "Ref": "PublicSubnet1"
  },
  "UserData": {
    "Fn::Base64": {
      "Fn::Join": [
        "",
        [
          "#!/bin/bash -xe\n",
```

```
                "yum update -y aws-cfn-bootstrap\n",
                "# Install the files and packages from the metadata\n",
                "/opt/aws/bin/cfn-init -v ",
                "       --stack ",
                {
                  "Ref": "AWS::StackName"
                },
                "       --resource EC2I27TYT ",
                "       --configsets InstallAndRun ",
                "       --region ",
                {
                  "Ref": "AWS::Region"
                },
                "\n",
                "# Signal the status from cfn-init\n",
                "/opt/aws/bin/cfn-signal -e $? ",
                "       --stack ",
                {
                  "Ref": "AWS::StackName"
                },
                "       --resource EC2I27TYT ",
                "       --region ",
                {
                  "Ref": "AWS::Region"
                },
                "\n"
              ]
            ]
          }
        }
      },
      "Metadata": {
        "AWS::CloudFormation::Init": {
          "configSets": {
            "InstallAndRun": [
```

```
      "Install",
      "Configure"
    ]
  },
  "Install": {
    "packages": {
      "yum": {
        "httpd": [],
        "php": [],
        "php-mysql": []
      }
    },
    "files": {
      "/var/www/html/index.php": {
        "content": {
          "Fn::Join": [
            "",
            [
              "<html>\n",
              "  <head>\n",
              "    <title>Chanakya Sowjanya **TCS Inframind**</title>\n",
              "    <meta http-equiv=\"Content-Type\" content=\"text/html; charset=ISO-8859-1\">\n",
              "  </head>\n",
              "  <body>\n",
              "    <h1>chanakya srinivas, Sai Sowjanya  GVPCE(A) Visakhapatnam</h1>\n",
              "    <p/>\n",
              "    <?php\n",
              "      // Print out the current data and time\n",
              "      print \"The Current Date and Time is: <br/>\";\n",
              "      print date(\"g:i A l, F j Y.\");\n",
              "    ?>\n",
              "    <p/>\n",
              "    <?php\n",
              "      // Setup a handle for CURL\n",
              "      $curl_handle=curl_init();\n",
```

```
"     curl_setopt($curl_handle,CURLOPT_CONNECTTIMEOUT,2);\n",
"     curl_setopt($curl_handle,CURLOPT_RETURNTRANSFER,1);\n",
"     // Get the hostname of the intance from the instance metadata\n",
"     curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/latest/meta-
data/public-hostname');\n",
"     $hostname = curl_exec($curl_handle);\n",
"     if (empty($hostname))\n",
"     {\n",
"       print \"Sorry, for some reason, we got no hostname back <br />\";\n",
"     }\n",
"     else\n",
"     {\n",
"       print \"Server = \" . $hostname . \"<br />\";\n",
"     }\n",
"     // Get the instance-id of the intance from the instance metadata\n",
"     curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/latest/meta-
data/instance-id');\n",
"     $instanceid = curl_exec($curl_handle);\n",
"     if (empty($instanceid))\n",
"     {\n",
"       print \"Sorry, for some reason, we got no instance id back <br />\";\n",
"     }\n",
"     else\n",
"     {\n",
"       print \"EC2 instance-id = \" . $instanceid . \"<br />\";\n",
"     }\n",
"     $Database   = \"localhost\";\n",
"     $DBUser     = \"",
{
   "Ref": "DBUser"
},
"\";\n",
"     $DBPassword = \"",
{
   "Ref": "DBPassword"
},
```

```
            "\";\n",
            "     print \"Database = \" . $Database . \"<br />\";\n",
            "     $dbconnection = mysql_connect($Database, $DBUser, $DBPassword)\n",
            "                  or die(\"Could not connect: \" . mysql_error());\n",
            "     print (\"Connected to $Database successfully\");\n",
            "     mysql_close($dbconnection);\n",
            "   ?>\n",
            "   <h2>PHP Information</h2>\n",
            "   <p/>\n",
            "   <?php\n",
            "     phpinfo();\n",
            "   ?>\n",
            " </body>\n",
            "</html>\n"
          ]
        ]
      },
      "mode": "000600",
      "owner": "apache",
      "group": "apache"
    },
  "/etc/cfn/cfn-hup.conf": {
      "content": {
        "Fn::Join": [
          "",
          [
            "[main]\n",
            "stack=",
            {
              "Ref": "AWS::StackId"
            },
            "\n",
            "region=",
            {
              "Ref": "AWS::Region"
```

```
            },
            "\n"
          ]
        ]
      },
      "mode": "000400",
      "owner": "root",
      "group": "root"
    },
    "/etc/cfn/hooks.d/cfn-auto-reloader.conf": {
      "content": {
        "Fn::Join": [
          "",
          [
            "[cfn-auto-reloader-hook]\n",
            "triggers=post.update\n",
            "path=Resources.EC2I27TYT.Metadata.AWS::CloudFormation::Init\n",
            "action=/opt/aws/bin/cfn-init -v ",
            "        --stack ",
            {
              "Ref": "AWS::StackName"
            },
            "        --resource EC2I27TYT ",
            "        --configsets InstallAndRun ",
            "        --region ",
            {
              "Ref": "AWS::Region"
            },
            "\n",
            "runas=root\n"
          ]
        ]
      },
      "mode": "000400",
      "owner": "root",
```

```
        "group": "root"
      }
    },
    "services": {
        "httpd": {
          "enabled": "true",
          "ensureRunning": "true"
        },
        "cfn-hup": {
          "enabled": "true",
          "ensureRunning": "true",
          "files": [
            "/etc/cfn/cfn-hup.conf",
            "/etc/cfn/hooks.d/cfn-auto-reloader.conf"
          ]
        }
      }
    }
  },
  "Configure": {
    "commands": {
      "01_set_mysql_root_password": {
        "command": {
          "Fn::Join": [
            "",
            [
              "mysqladmin -u root password '",
              {
                "Ref": "DBRootPassword"
              },
              "'"
            ]
          ]
        },
        "test": {
```

```
      "Fn::Join": [
        "",
        [
          "$(mysql ",
          {
            "Ref": "DBName"
          },
          " -u root --password='",
          {
            "Ref": "DBRootPassword"
          },
          "' >/dev/null 2>&1 </dev/null); (( $? != 0 ))"
        ]
      ]
    }
  },
  "02_create_database": {
    "command": {
      "Fn::Join": [
        "",
        [
          "mysql -u root --password='",
          {
            "Ref": "DBRootPassword"
          },
          "' < /tmp/setup.mysql"
        ]
      ]
    },
    "test": {
      "Fn::Join": [
        "",
        [
          "$(mysql ",
          {
```

```
                        "Ref": "DBName"
                      },
                      " -u root --password='",
                      {
                        "Ref": "DBRootPassword"
                      },
                      "' >/dev/null 2>&1 </dev/null); (( $? != 0 ))"
                    ]
                  ]
                }
              }
            }
          }
        },
        "AWS::CloudFormation::Designer": {
          "id": "b3b7eb8f-3f91-4368-a911-386a211f859c"
        }
      }
    },
    "EC2I1M4SA": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
        "ImageId": {
          "Fn::FindInMap": [
            "AWSRegionArch2AMI",
            {
              "Ref": "AWS::Region"
            },
            {
              "Fn::FindInMap": [
                "AWSInstanceType2Arch",
                {
                  "Ref": "InstanceType"
                },
                "Arch"
```

```
        ]
      }
    ]
},
"InstanceType": {
   "Ref": "InstanceType"
},
"SecurityGroups": [
   {
      "Ref": "EC2SG2ZLDH"
   }
],
"KeyName": {
   "Ref": "KeyName"
},
"SubnetId": {
   "Ref": "PublicSubnet1"
},
"UserData": {
   "Fn::Base64": {
      "Fn::Join": [
         "",
         [
            "#!/bin/bash -xe\n",
            "yum update -y aws-cfn-bootstrap\n",
            "# Install the files and packages from the metadata\n",
            "/opt/aws/bin/cfn-init -v ",
            "        --stack ",
            {
               "Ref": "AWS::StackName"
            },
            "        --resource EC2I1M4SA ",
            "        --configsets InstallAndRun ",
            "        --region ",
            {
```

```
                "Ref": "AWS::Region"
              },
              "\n",
              "# Signal the status from cfn-init\n",
              "/opt/aws/bin/cfn-signal -e $? ",
              "        --stack ",
              {
                "Ref": "AWS::StackName"
              },
              "        --resource EC2I1M4SA ",
              "        --region ",
              {
                "Ref": "AWS::Region"
              },
              "\n"
            ]
          ]
        }
      }
    },
    "Metadata": {
      "AWS::CloudFormation::Init": {
        "configSets": {
          "InstallAndRun": [
            "Install",
            "Configure"
          ]
        },
        "Install": {
          "packages": {
            "yum": {
              "httpd": [],
              "php": [],
              "php-mysql": []
            }
```

```
        },
        "files": {
          "/var/www/html/index.php": {
            "content": {
              "Fn::Join": [
                "",
                [
                  "<html>\n",
                  "  <head>\n",
                  "    <title>Chanakya Sowjanya TCS Inframind</title>\n",
                  "    <meta http-equiv=\"Content-Type\" content=\"text/html; charset=ISO-8859-1\">\n",
                  "  </head>\n",
                  "  <body>\n",
                  "    <h1>chanakya srinivas, Sai Sowjanya  GVPCE(A) Visakhapatnam</h1>\n",
                  "    <p/>\n",
                  "    <?php\n",
                  "      // Print out the current data and time\n",
                  "      print \"The Current Date and Time is: <br/>\";\n",
                  "      print date(\"g:i A l, F j Y.\");\n",
                  "    ?>\n",
                  "    <p/>\n",
                  "    <?php\n",
                  "      // Setup a handle for CURL\n",
                  "      $curl_handle=curl_init();\n",
                  "      curl_setopt($curl_handle,CURLOPT_CONNECTTIMEOUT,2);\n",
                  "      curl_setopt($curl_handle,CURLOPT_RETURNTRANSFER,1);\n",
                  "      // Get the hostname of the intance from the instance metadata\n",
                  "      curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/latest/meta-
data/public-hostname');\n",
                  "      $hostname = curl_exec($curl_handle);\n",
                  "      if (empty($hostname))\n",
                  "      {\n",
                  "        print \"Sorry, for some reason, we got no hostname back <br />\";\n",
                  "      }\n",
                  "      else\n",
```

```
"     {\n",
"       print \"Server = \" . $hostname . \"<br />\";\n",
"     }\n",
"     // Get the instance-id of the intance from the instance metadata\n",
"     curl_setopt($curl_handle,CURLOPT_URL,'http://169.254.169.254/latest/meta-
data/instance-id');\n",
"     $instanceid = curl_exec($curl_handle);\n",
"     if (empty($instanceid))\n",
"     {\n",
"       print \"Sorry, for some reason, we got no instance id back <br />\";\n",
"     }\n",
"     else\n",
"     {\n",
"       print \"EC2 instance-id = \" . $instanceid . \"<br />\";\n",
"     }\n",
"     $Database   = \"localhost\";\n",
"     $DBUser     = \"",
{
   "Ref": "DBUser"
},
"\";\n",
"     $DBPassword = \"",
{
   "Ref": "DBPassword"
},
"\";\n",
"     print \"Database = \" . $Database . \"<br />\";\n",
"     $dbconnection = mysql_connect($Database, $DBUser, $DBPassword)\n",
"                     or die(\"Could not connect: \" . mysql_error());\n",
"     print (\"Connected to $Database successfully\");\n",
"     mysql_close($dbconnection);\n",
"   ?>\n",
"   <h2>PHP Information</h2>\n",
"   <p/>\n",
"   <?php\n",
```

```
            "     phpinfo();\n",
            "    ?>\n",
            "  </body>\n",
            "</html>\n"
          ]
        ]
    },
    "mode": "000600",
    "owner": "apache",
    "group": "apache"
},

"/etc/cfn/cfn-hup.conf": {
    "content": {
      "Fn::Join": [
        "",
        [
          "[main]\n",
          "stack=",
          {
            "Ref": "AWS::StackId"
          },
          "\n",
          "region=",
          {
            "Ref": "AWS::Region"
          },
          "\n"
        ]
      ]
    },
    "mode": "000400",
    "owner": "root",
    "group": "root"
},
```

```
    "/etc/cfn/hooks.d/cfn-auto-reloader.conf": {
      "content": {
        "Fn::Join": [
          "",
          [
            "[cfn-auto-reloader-hook]\n",
            "triggers=post.update\n",
            "path=Resources.EC2I1M4SA.Metadata.AWS::CloudFormation::Init\n",
            "action=/opt/aws/bin/cfn-init -v ",
            "       --stack ",
            {
              "Ref": "AWS::StackName"
            },
            "       --resource EC2I1M4SA ",
            "       --configsets InstallAndRun ",
            "       --region ",
            {
              "Ref": "AWS::Region"
            },
            "\n",
            "runas=root\n"
          ]
        ]
      },
      "mode": "000400",
      "owner": "root",
      "group": "root"
    }
  },
  "services": {
    "sysvinit": {
      "mysqld": {
        "enabled": "true",
        "ensureRunning": "true"
      },
```

```
        "httpd": {
          "enabled": "true",
          "ensureRunning": "true"
        },
        "cfn-hup": {
          "enabled": "true",
          "ensureRunning": "true",
          "files": [
            "/etc/cfn/cfn-hup.conf",
            "/etc/cfn/hooks.d/cfn-auto-reloader.conf"
          ]
        }
      }
    }
  },
  "Configure": {
    "commands": {
      "01_set_mysql_root_password": {
        "command": {
          "Fn::Join": [
            "",
            [
              "mysqladmin -u root password '",
              {
                "Ref": "DBRootPassword"
              },
              "'"
            ]
          ]
        },
        "test": {
          "Fn::Join": [
            "",
            [
              "$(mysql ",
```

```
            {
               "Ref": "DBName"
            },
            " -u root --password='",
            {
               "Ref": "DBRootPassword"
            },
            "' >/dev/null 2>&1 </dev/null); (( $? != 0 ))"
         ]
      ]
   }
},
"02_create_database": {
   "command": {
      "Fn::Join": [
         "",
         [
            "mysql -u root --password='",
            {
               "Ref": "DBRootPassword"
            },
            "' < /tmp/setup.mysql"
         ]
      ]
   },
   "test": {
      "Fn::Join": [
         "",
         [
            "$(mysql ",
            {
               "Ref": "DBName"
            },
            " -u root --password='",
            {
```

```
                        "Ref": "DBRootPassword"
                      },
                      "' >/dev/null 2>&1 </dev/null); (( $? != 0 ))"
                    ]
                  ]
                }
              }
            }
          }
        },
        "AWS::CloudFormation::Designer": {
          "id": "d1ec9baa-b935-42b7-8389-3c05dac84939"
        }
      }
    },
    "EC2SG12LXS": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Enable HTTP access via port 443",
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": "443",
            "ToPort": "443",
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": "80",
            "ToPort": "80",
            "CidrIp": {
              "Ref": "SSHLocation"
            }
          }
        ],
```

```
      "VpcId": {
        "Ref": "VPC"
      }
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "fe7c1ef6-a76d-40aa-ba14-44a1862c4789"
      }
    }
  },

  "EC2SG225VC": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupDescription": "Enable HTTP access via port 3306",
      "SecurityGroupIngress": [
        {
          "IpProtocol": "tcp",
          "FromPort": "3306",
          "ToPort": "3306",
          "CidrIp": "0.0.0.0/0"
        },
        {
          "IpProtocol": "tcp",
          "FromPort": "1403",
          "ToPort": "1403",
          "CidrIp": {
            "Ref": "SSHLocation"
          }
        },
        {
          "IpProtocol": "tcp",
          "FromPort": "22",
          "ToPort": "22",
          "CidrIp": {
```

```
            "Ref": "SSHLocation"
          }
        },
        {
          "IpProtocol": "tcp",
          "FromPort": "3389",
          "ToPort": "3389",
          "CidrIp": {
            "Ref": "SSHLocation"
          }
        }
      ]
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "45f23ab4-8995-43f3-a72e-649595e7797e"
      }
    }
  },
  "EC2SG2ZLDH": {
    "Type": "AWS::EC2::SecurityGroup",
    "Properties": {
      "GroupDescription": "Enable HTTP access via port 443",
      "SecurityGroupIngress": [
        {
          "IpProtocol": "tcp",
          "FromPort": "443",
          "ToPort": "443",
          "CidrIp": "0.0.0.0/0"
        },
        {
          "IpProtocol": "tcp",
          "FromPort": "22",
          "ToPort": "22",
          "CidrIp": {
```

```
            "Ref": "SSHLocation"
          }
        },
        {
          "IpProtocol": "tcp",
          "FromPort": "3389",
          "ToPort": "3389",
          "CidrIp": {
            "Ref": "SSHLocation"
          }
        },
        {
          "IpProtocol": "tcp",
          "FromPort": "80",
          "ToPort": "80",
          "CidrIp": {
            "Ref": "SSHLocation"
          }
        }
      ],
      "VpcId": {
        "Ref": "VPC"
      }
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "884c6e90-ea93-4839-a868-2dd766fbb52c"
      }
    }
  },
  "ELBLB2XW33": {
    "Type": "AWS::ElasticLoadBalancing::LoadBalancer",
    "Properties": {
      "Subnets": [
        {
```

```
            "Ref": "PublicSubnet1"
          }
        ],
        "Instances": [
          {
            "Ref": "EC2I1M4SA"
          },
          {
            "Ref": "EC2I27TYT"
          }
        ],
        "SecurityGroups": [
          {
            "Ref": "EC2SG12LXS"
          }
        ]
      },
      "Metadata": {
        "AWS::CloudFormation::Designer": {
          "id": "0279fb20-3290-4aa6-9dc9-deefa6a1a722"
        }
      }
    },
    "CWA1ZROE": {
      "Type": "AWS::CloudWatch::Alarm",
      "Properties": {
        "AlarmDescription": "CPU alarm for my instance",
        "AlarmActions": [
          {
            "Ref": "SNSTDIOZ"
          }
        ],
        "MetricName": "CPUUtilization",
        "Namespace": "AWS/EC2",
        "Statistic": "Average",
```

```
      "Period": "60",
      "EvaluationPeriods": "3",
      "Threshold": "80",
      "ComparisonOperator": "GreaterThanThreshold",
      "Dimensions": [
        {
          "Name": "EC2I1M4SA",
          "Value": {
            "Ref": "EC2I1M4SA"
          }
        }
      ]
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "39bee738-c820-4475-bae5-61bbb9b592eb"
      }
    },
    "DependsOn": [
      "EC2I1M4SA",
      "EC2I27TYT"
    ]
  },
  "SNSTDIOZ": {
    "Type": "AWS::SNS::Topic",
    "Properties": {},
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "10ea76c3-044f-4a44-9959-e3ddfc1c4f42"
      }
    }
  },
  "SNSS2S9XC": {
    "Type": "AWS::SNS::Subscription",
    "Properties": {
```

```
      "Endpoint": "chanakyasrinu01@gmail.com",
      "Protocol": "email",
      "TopicArn": {
        "Ref": "SNSTDIOZ"
      }
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "0cf8d224-8c33-4062-9c66-cd286b813f30"
      }
    }
  },
  "EC2I4CFD6": {
    "Type": "AWS::EC2::Instance",
    "Properties": {
      "ImageId": {
        "Fn::FindInMap": [
          "AWSRegionArch2AMI",
          {
            "Ref": "AWS::Region"
          },
          {
            "Fn::FindInMap": [
              "AWSInstanceType2Arch",
              {
                "Ref": "InstanceType"
              },
              "Arch"
            ]
          }
        ]
      },
      "InstanceType": {
        "Ref": "InstanceType"
      },
```

```
"SecurityGroups": [
   {
      "Ref": "EC2SG225VC"
   }
],
"KeyName": {
   "Ref": "KeyName"
},
"SubnetId": {
   "Fn::Base64": {
      "Fn::Join": [
         "",
         [
            "#!/bin/bash -xe\n",
            "yum update -y aws-cfn-bootstrap\n",
            "# Install the files and packages from the metadata\n",
            "/opt/aws/bin/cfn-init -v ",
            "        --stack ",
            {
               "Ref": "AWS::StackName"
            },
            "        --resource EC2I4CFD6 ",
            "        --configsets InstallAndRun ",
            "        --region ",
            {
               "Ref": "AWS::Region"
            },
            "\n",
            "# Signal the status from cfn-init\n",
            "/opt/aws/bin/cfn-signal -e $? ",
            "        --stack ",
            {
```

```
                "Ref": "AWS::StackName"
              },
              "       --resource EC2I4CFD6 ",
              "       --region ",
              {
                "Ref": "AWS::Region"
              },
              "\n"
            ]
          ]
        }
      }
    },
    "Metadata": {
      "AWS::CloudFormation::Init": {
        "configSets": {
          "InstallAndRun": [
            "Install",
            "Configure"
          ]
        },
        "Install": {
          "packages": {
            "yum": {
              "mysql": [],
              "mysql-server": [],
              "mysql-libs": []
            }
          },
          "files": {
            "/tmp/setup.mysql": {
              "content": {
                "Fn::Join": [
                  "",
                  [
```

```
            "CREATE DATABASE ",
            {
              "Ref": "DBName"
            },
            ";\n",
            "GRANT ALL ON ",
            {
              "Ref": "DBName"
            },
            ".* TO '",
            {
              "Ref": "DBUser"
            },
            "'@localhost IDENTIFIED BY '",
            {
              "Ref": "DBPassword"
            },
            "';\n"
          ]
        ]
      },
      "mode": "000400",
      "owner": "root",
      "group": "root"
    },
    "/etc/cfn/cfn-hup.conf": {
      "content": {
        "Fn::Join": [
          "",
          [
            "[main]\n",
            "stack=",
            {
              "Ref": "AWS::StackId"
            },
```

```
            "\n",
            "region=",
            {
              "Ref": "AWS::Region"
            },
            "\n"
          ]
        ]
      },
      "mode": "000400",
      "owner": "root",
      "group": "root"
    },
    "/etc/cfn/hooks.d/cfn-auto-reloader.conf": {
      "content": {
        "Fn::Join": [
          "",
          [
            "[cfn-auto-reloader-hook]\n",
            "triggers=post.update\n",
            "path=Resources.WebServerInstance.Metadata.AWS::CloudFormation::Init\n",
            "action=/opt/aws/bin/cfn-init -v ",
            "        --stack ",
            {
              "Ref": "AWS::StackName"
            },
            "        --resource WebServerInstance ",
            "        --configsets InstallAndRun ",
            "        --region ",
            {
              "Ref": "AWS::Region"
            },
            "\n",
            "runas=root\n"
          ]
```

```
            ]
          },
          "mode": "000400",
          "owner": "root",
          "group": "root"
        }
      },
      "services": {
        "sysvinit": {
          "mysqld": {
            "enabled": "true",
            "ensureRunning": "true"
          },
          "cfn-hup": {
            "enabled": "true",
            "ensureRunning": "true",
            "files": [
              "/etc/cfn/cfn-hup.conf",
              "/etc/cfn/hooks.d/cfn-auto-reloader.conf"
            ]
          }
        }
      }
    },
    "Configure": {
      "commands": {
        "01_set_mysql_root_password": {
          "command": {
            "Fn::Join": [
              "",
              [
                "mysqladmin -u root password '",
                {
                  "Ref": "DBRootPassword"
                },
```

```
                        """
                      ]
                    ]

        "AWS::CloudFormation::Designer": {
          "id": "e86b84d8-bdc2-4cc3-9447-50f698255428"
        }
      }
    }
  },
  "Outputs": {
    "VPC": {
      "Description": "A reference to the created VPC",
      "Value": {
        "Ref": "VPC"
      }
    },
    "WebsiteURL": {
      "Description": "URL for the ec2 instance",
      "Value": {
        "Fn::Join": [
          "",
          [
            "http://",
            {
              "Fn::GetAtt": [
                "EC2I27TYT",
                "PublicDnsName"
              ]
            }
          ]
        ]
      }
    },
    "PublicSubnets": {
```

```
    "Description": "A list of the public subnets",
    "Value": {
      "Fn::Join": [
        ",",
        [
          {
            "Ref": "PublicSubnet1"
          }
        ]
      ]
    }
  },
  "PrivateSubnets": {
    "Description": "A list of the private subnets",
    "Value": {
      "Fn::Join": [
        ",",
        [
          {
            "Ref": "PrivateSubnet1"
          }
        ]
      ]
    }
  },
  "PublicSubnet1": {
    "Description": "A reference to the public subnet in the 1st Availability Zone",
    "Value": {
      "Ref": "PublicSubnet1"
    }
  },
  "PrivateSubnet1": {
    "Description": "A reference to the private subnet in the 1st Availability Zone",
    "Value": {
      "Ref": "PrivateSubnet1"
```

```
      }
    },
    "NoIngressSecurityGroup": {
      "Description": "Security group with no ingress rule",
      "Value": {
        "Ref": "NoIngressSecurityGroup"
      }
    }
  },
  "Metadata": {
    "AWS::CloudFormation::Designer": {
      "9af6bbd2-3079-47c0-9948-fe7929d70a3c": {
        "size": {
          "width": 1350,
          "height": 1350
        },
        "position": {
          "x": 550,
          "y": 610
        },
        "z": 1,
        "embeds": [
          "6024b332-5ba1-4af2-a579-0dc861c29198",
          "2e5a7ad7-b14e-4c27-bdf3-c49d0259b590",
          "bff7432e-8d88-451e-b3d2-f8710e825c1e",
          "5350852a-6e99-47bd-a848-44f609af5ab1"
        ]
      },
      "5f4e3a05-c33b-43ea-af6b-3e3e6f1653aa": {
        "size": {
          "width": 60,
          "height": 60
        },
        "position": {
          "x": 1720,
```

```
      "y": 2130
    },
    "z": 0,
    "embeds": [],
    "iscontainedinside": [
      "9af6bbd2-3079-47c0-9948-fe7929d70a3c"
    ]
  },
  "6024b332-5ba1-4af2-a579-0dc861c29198": {
    "size": {
      "width": 140,
      "height": 170
    },
    "position": {
      "x": 600,
      "y": 950
    },
    "z": 2,
    "parent": "9af6bbd2-3079-47c0-9948-fe7929d70a3c",
    "embeds": [
      "685caec3-b30c-4267-88a5-fa97d3f34cdd"
    ],
    "iscontainedinside": [
      "9af6bbd2-3079-47c0-9948-fe7929d70a3c",
    ]
  },
  "2e5a7ad7-b14e-4c27-bdf3-c49d0259b590": {
    "size": {
      "width": 210,
      "height": 180
    },
    "position": {
      "x": 760,
      "y": 1730
    },
```

```
    "z": 2,
    "parent": "9af6bbd2-3079-47c0-9948-fe7929d70a3c",
    "embeds": [
      "bf1938b8-ac7d-44b3-8d39-b1fc5ce7210e"
    ],
    "iscontainedinside": [
      "9af6bbd2-3079-47c0-9948-fe7929d70a3c",
    ]
  },
  "bf1938b8-ac7d-44b3-8d39-b1fc5ce7210e": {
    "size": {
      "width": 60,
      "height": 60
    },
    "position": {
      "x": 820,
      "y": 1780
    },
    "z": 3,
    "parent": "2e5a7ad7-b14e-4c27-bdf3-c49d0259b590",
    "embeds": [],
    "isassociatedwith": [
      "dcf669e1-aa34-4dbd-bf85-176826b84a6e",
      "b3b7eb8f-3f91-4368-a911-386a211f859c",
      "d1ec9baa-b935-42b7-8389-3c05dac84939"
    ],
    "iscontainedinside": [
      "2e5a7ad7-b14e-4c27-bdf3-c49d0259b590",
    ],
    "dependson": [
      "d13e9044-3811-41d1-bb24-193a69de22ea",
      "c2217419-b20e-4240-9e85-ae95498e91a1"
    ]
  },
  "bff7432e-8d88-451e-b3d2-f8710e825c1e": {
```

```
      "size": {
        "width": 940,
        "height": 340
      },
      "position": {
        "x": 920,
        "y": 750
      },
      "z": 2,
      "parent": "9af6bbd2-3079-47c0-9948-fe7929d70a3c",
      "embeds": [
        "45f23ab4-8995-43f3-a72e-649595e7797e",
        "e86b84d8-bdc2-4cc3-9447-50f698255428"
      ],
      "iscontainedinside": [
"9af6bbd2-3079-47c0-9948-fe7929d70a3c"
      ]
    },
    "5350852a-6e99-47bd-a848-44f609af5ab1": {
      "size": {
        "width": 780,
        "height": 480
      },
      "position": {
        "x": 700,
        "y": 1160
      },
      "z": 2,
      "parent": "9af6bbd2-3079-47c0-9948-fe7929d70a3c",
      "embeds": [
        "10ea76c3-044f-4a44-9959-e3ddfc1c4f42",
        "0cf8d224-8c33-4062-9c66-cd286b813f30",
        "884c6e90-ea93-4839-a868-2dd766fbb52c",
        "fe7c1ef6-a76d-40aa-ba14-44a1862c4789",
        "d1ec9baa-b935-42b7-8389-3c05dac84939",
```

```
        "b3b7eb8f-3f91-4368-a911-386a211f859c",
        "39bee738-c820-4475-bae5-61bbb9b592eb",
        "0279fb20-3290-4aa6-9dc9-deefa6a1a722",
        "b709b2b6-efdf-4817-a045-49b08a98847a"
    ],
    "iscontainedinside": [
        "9af6bbd2-3079-47c0-9948-fe7929d70a3c",
        "9af6bbd2-3079-47c0-9948-fe7929d70a3c"
    ]
},
"b709b2b6-efdf-4817-a045-49b08a98847a": {
    "size": {
        "width": 60,
        "height": 60
    },
    "position": {
        "x": 740,
        "y": 1530
    },
    "z": 3,
    "parent": "5350852a-6e99-47bd-a848-44f609af5ab1",
    "embeds": [],
    "iscontainedinside": [
        "5350852a-6e99-47bd-a848-44f609af5ab1",
        "5350852a-6e99-47bd-a848-44f609af5ab1"
    ]
},
"45f7b637-8d7d-4a6d-a88b-839e69beb37f": {
    "size": {
        "width": 60,
        "height": 60
    },
    "position": {
        "x": 400,
        "y": 2180
```

```
    },
    "z": 1,
    "embeds": [],
    "dependson": [
      "d13e9044-3811-41d1-bb24-193a69de22ea",
      "c2217419-b20e-4240-9e85-ae95498e91a1"
    ]
  },
  "dcf669e1-aa34-4dbd-bf85-176826b84a6e": {
    "size": {
      "width": 60,
      "height": 60
    },
    "position": {
      "x": 740,
      "y": 2350
    },
    "z": 1,
    "embeds": []
  },
  "6cfc7568-6e8b-42f2-a5b2-a95dc0c978eb": {
    "source": {
      "id": "6024b332-5ba1-4af2-a579-0dc861c29198"
    },
    "target": {
      "id": "bff7432e-8d88-451e-b3d2-f8710e825c1e"
    },
    "z": 2
  },
  "6a07f4ea-2806-49ed-83ed-7b2ef592c486": {
    "source": {
      "id": "2e5a7ad7-b14e-4c27-bdf3-c49d0259b590"
    },
    "target": {
      "id": "5350852a-6e99-47bd-a848-44f609af5ab1"
```

```
    },
    "z": 2
},
"c2217419-b20e-4240-9e85-ae95498e91a1": {
    "source": {
        "id": "9af6bbd2-3079-47c0-9948-fe7929d70a3c"
    },
    "target": {
        "id": "dcf669e1-aa34-4dbd-bf85-176826b84a6e"
    },
    "z": 1
},
"685caec3-b30c-4267-88a5-fa97d3f34cdd": {
    "size": {
        "width": 60,
        "height": 60
    },
    "position": {
        "x": 630,
        "y": 980
    },
    "z": 3,
    "parent": "6024b332-5ba1-4af2-a579-0dc861c29198",
    "embeds": [],
    "isassociatedwith": [
        "b709b2b6-efdf-4817-a045-49b08a98847a",
        "d1ec9baa-b935-42b7-8389-3c05dac84939"
    ],
    "iscontainedinside": [
        "6024b332-5ba1-4af2-a579-0dc861c29198",
        "6024b332-5ba1-4af2-a579-0dc861c29198",
    ]
},
"b3b7eb8f-3f91-4368-a911-386a211f859c": {
    "size": {
```

```
      "width": 60,
      "height": 60
    },
    "position": {
      "x": 890,
      "y": 1430
    },
    "z": 3,
    "parent": "5350852a-6e99-47bd-a848-44f609af5ab1",
    "embeds": [],
    "isassociatedwith": [
      "884c6e90-ea93-4839-a868-2dd766fbb52c"
    ],
    "iscontainedinside": [
      "5350852a-6e99-47bd-a848-44f609af5ab1",
      "5350852a-6e99-47bd-a848-44f609af5ab1",

    ]
  },
  "d1ec9baa-b935-42b7-8389-3c05dac84939": {
    "size": {
      "width": 60,
      "height": 60
    },
    "position": {
      "x": 890,
      "y": 1360
    },
    "z": 3,
    "parent": "5350852a-6e99-47bd-a848-44f609af5ab1",
    "embeds": [],
    "isassociatedwith": [
      "884c6e90-ea93-4839-a868-2dd766fbb52c"
    ],
    "iscontainedinside": [
```

```
        "5350852a-6e99-47bd-a848-44f609af5ab1"
      ]
    },
    "fe7c1ef6-a76d-40aa-ba14-44a1862c4789": {
      "size": {
        "width": 60,
        "height": 60
      },
      "position": {
        "x": 1190,
        "y": 1230
      },
      "z": 3,
      "parent": "5350852a-6e99-47bd-a848-44f609af5ab1",
      "embeds": [],
      "iscontainedinside": [
        "9af6bbd2-3079-47c0-9948-fe7929d70a3c"
      ]
    },
    "45f23ab4-8995-43f3-a72e-649595e7797e": {
      "size": {
        "width": 60,
        "height": 60
      },
      "position": {
        "x": 1250,
        "y": 870
      },
      "z": 3,
      "parent": "bff7432e-8d88-451e-b3d2-f8710e825c1e",
      "embeds": []
    },
    "884c6e90-ea93-4839-a868-2dd766fbb52c": {
      "size": {
        "width": 60,
```

```
      "height": 60
    },
    "position": {
      "x": 1120,
      "y": 1430
    },
    "z": 3,
    "parent": "5350852a-6e99-47bd-a848-44f609af5ab1",
    "embeds": [],
    "iscontainedinside": [
      "9af6bbd2-3079-47c0-9948-fe7929d70a3c"
    ]
  },
  "0279fb20-3290-4aa6-9dc9-deefa6a1a722": {
    "size": {
      "width": 60,
      "height": 60
    },
    "position": {
      "x": 750,
      "y": 1280
    },
    "z": 3,
    "parent": "5350852a-6e99-47bd-a848-44f609af5ab1",
    "embeds": [],
    "isassociatedwith": [
      "d1ec9baa-b935-42b7-8389-3c05dac84939",
      "b3b7eb8f-3f91-4368-a911-386a211f859c",
      "fe7c1ef6-a76d-40aa-ba14-44a1862c4789"
    ],
    "iscontainedinside": [
      "5350852a-6e99-47bd-a848-44f609af5ab1"
    ]
  },
  "39bee738-c820-4475-bae5-61bbb9b592eb": {
```

```
    "size": {
      "width": 60,
      "height": 60
    },
    "position": {
      "x": 1350,
      "y": 1410
    },
    "z": 3,
    "parent": "5350852a-6e99-47bd-a848-44f609af5ab1",
    "embeds": [],
    "dependson": [
      "b3b7eb8f-3f91-4368-a911-386a211f859c"
    ]
  },
  "10ea76c3-044f-4a44-9959-e3ddfc1c4f42": {
    "size": {
      "width": 60,
      "height": 60
    },
    "position": {
      "x": 1210,
      "y": 1510
    },
    "z": 3,
    "parent": "5350852a-6e99-47bd-a848-44f609af5ab1",
    "embeds": []
  },
  "0cf8d224-8c33-4062-9c66-cd286b813f30": {
    "size": {
      "width": 60,
      "height": 60
    },
    "position": {
      "x": 1330,
```

```
            "y": 1510
          },
          "z": 3,
          "parent": "5350852a-6e99-47bd-a848-44f609af5ab1",
          "embeds": [],
          "isassociatedwith": [
            "10ea76c3-044f-4a44-9959-e3ddfc1c4f42"
          ]
        },
        "e86b84d8-bdc2-4cc3-9447-50f698255428": {
          "size": {
            "width": 60,
            "height": 60
          },
          "position": {
            "x": 1050,
            "y": 870
          },
          "z": 3,
          "parent": "bff7432e-8d88-451e-b3d2-f8710e825c1e",
          "embeds": [],
          "isassociatedwith": [
            "45f23ab4-8995-43f3-a72e-649595e7797e"
          ],
          "iscontainedinside": [
            "bff7432e-8d88-451e-b3d2-f8710e825c1e"
          ]
        }
      }
    }
}
```


- Stack1 is uploaded into S3 bucket and the link is given to Jenkins cloudformation plugin along aws access key and secret key for chansowji IAM user. After stack creation, an email is sent to the client along with the build log

STACK 2

Cloud-Formation JSON Script:


```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Instance scheduler, version 2.2.2.0",
  "Parameters": {
    "SchedulingActive": {
      "Type": "String",
      "AllowedValues": [
        "Yes",
        "No"
      ],
      "Default": "Yes",
      "Description": "Activate or deactivate scheduling."
    },
    "ScheduledServices": {
      "Type": "String",
      "AllowedValues": [
        "EC2",
        "RDS",
        "Both"
      ],
      "Default": "EC2",
      "Description": "Scheduled Services"
    },
    "MemorySize": {
      "Type": "Number",
      "AllowedValues": [
        128,
        384,
        512,
        640,
        768,
        896,
```

```
            1024,
            1152,
            1280,
            1408,
            1536
        ],
        "Default": 128,
        "Description": "Size of the Lambda function running the scheduler, increase size when processing large numbers of instances"
    },
    "UseCloudWatchMetrics": {
        "Type": "String",
        "AllowedValues": [
            "Yes",
            "No"
        ],
        "Default": "No",
        "Description": "Collect instance scheduling data using CloudWatch metrics."
    },
    "LogRetentionDays": {
        "Type": "Number",
        "Default": 30,
        "AllowedValues": [
            1,
            3,
            5,
            7,
            14,
            30,
            60,
            90,
            120,
            150,
            180,
            365,
            400,
            545,
```

```
        731,
        1827,
        3653
    ],
    "Description": "Retention days for scheduler logs."
},
"Trace": {
    "Type": "String",
    "AllowedValues": [
        "Yes",
        "No"
    ],
    "Default": "No",
    "Description": "Enable logging of detailed informtion in CloudWatch logs."
},
"TagName": {
    "Type": "String",
    "Default": "Schedule",
    "MinLength": 1,
    "MaxLength": 127,
    "Description": "Name of tag to use for associating instance schedule schemas with service instances."
},
"DefaultTimezone": {
    "Type": "String",
    "Default": "UTC",
    "AllowedValues": [
        "Africa/Abidjan",
        "Africa/Accra",
        "Africa/Addis_Ababa",
        "Africa/Algiers",
        "Africa/Asmara",
        "Africa/Bamako",
        "Africa/Bangui",
        "Africa/Banjul",
        "Africa/Bissau",
        "Africa/Blantyre",
```

"Africa/Brazzaville",

"Africa/Bujumbura",

"Africa/Cairo",

"Africa/Casablanca",

"Africa/Ceuta",

"Africa/Conakry",

"Africa/Dakar",

"Africa/Dar_es_Salaam",

"Africa/Djibouti",

"Africa/Douala",

"Africa/El_Aaiun",

"Africa/Freetown",

"Africa/Gaborone",

"Africa/Harare",

"Africa/Johannesburg",

"Africa/Juba",

"Africa/Kampala",

"Africa/Khartoum",

"Africa/Kigali",

"Africa/Kinshasa",

"Africa/Lagos",

"Africa/Libreville",

"Africa/Lome",

"Africa/Luanda",

"Africa/Lubumbashi",

"Africa/Lusaka",

"Africa/Malabo",

"Africa/Maputo",

"Africa/Maseru",

"Africa/Mbabane",

"Africa/Mogadishu",

"Africa/Monrovia",

"Africa/Nairobi",

"Africa/Ndjamena",

"Africa/Niamey",

"Africa/Nouakchott",

"Africa/Ouagadougou",

"Africa/Porto-Novo",

"Africa/Sao_Tome",

"Africa/Tripoli",

"Africa/Tunis",

"Africa/Windhoek",

"America/Adak",

"America/Anchorage",

"America/Anguilla",

"America/Antigua",

"America/Araguaina",

"America/Argentina/Buenos_Aires",

"America/Argentina/Catamarca",

"America/Argentina/Cordoba",

"America/Argentina/Jujuy",

"America/Argentina/La_Rioja",

"America/Argentina/Mendoza",

"America/Argentina/Rio_Gallegos",

"America/Argentina/Salta",

"America/Argentina/San_Juan",

"America/Argentina/San_Luis",

"America/Argentina/Tucuman",

"America/Argentina/Ushuaia",

"America/Aruba",

"America/Asuncion",

"America/Atikokan",

"America/Bahia",

"America/Bahia_Banderas",

"America/Barbados",

"America/Belem",

"America/Belize",

"America/Blanc-Sablon",

"America/Boa_Vista",

"America/Bogota",

"America/Boise",

"America/Cambridge_Bay",

"America/Campo_Grande",

"America/Cancun",

"America/Caracas",

"America/Cayenne",

"America/Cayman",

"America/Chicago",

"America/Chihuahua",

"America/Costa_Rica",

"America/Creston",

"America/Cuiaba",

"America/Curacao",

"America/Danmarkshavn",

"America/Dawson",

"America/Dawson_Creek",

"America/Denver",

"America/Detroit",

"America/Dominica",

"America/Edmonton",

"America/Eirunepe",

"America/El_Salvador",

"America/Fortaleza",

"America/Glace_Bay",

"America/Godthab",

"America/Goose_Bay",

"America/Grand_Turk",

"America/Grenada",

"America/Guadeloupe",

"America/Guatemala",

"America/Guayaquil",

"America/Guyana",

"America/Halifax",

"America/Havana",

"America/Hermosillo",

"America/Indiana/Indianapolis",

"America/Indiana/Knox",

"America/Indiana/Marengo",

"America/Indiana/Petersburg",

"America/Indiana/Tell_City",

"America/Indiana/Vevay",

"America/Indiana/Vincennes",

"America/Indiana/Winamac",

"America/Inuvik",

"America/Iqaluit",

"America/Jamaica",

"America/Juneau",

"America/Kentucky/Louisville",

"America/Kentucky/Monticello",

"America/Kralendijk",

"America/La_Paz",

"America/Lima",

"America/Los_Angeles",

"America/Lower_Princes",

"America/Maceio",

"America/Managua",

"America/Manaus",

"America/Marigot",

"America/Martinique",

"America/Matamoros",

"America/Mazatlan",

"America/Menominee",

"America/Merida",

"America/Metlakatla",

"America/Mexico_City",

"America/Miquelon",

"America/Moncton",

"America/Monterrey",

"America/Montevideo",

"America/Montreal",

"America/Montserrat",

"America/Nassau",

"America/New_York",

"America/Nipigon",

"America/Nome",

"America/Noronha",

"America/North_Dakota/Beulah",

"America/North_Dakota/Center",

"America/North_Dakota/New_Salem",

"America/Ojinaga",

"America/Panama",

"America/Pangnirtung",

"America/Paramaribo",

"America/Phoenix",

"America/Port-au-Prince",

"America/Port_of_Spain",

"America/Porto_Velho",

"America/Puerto_Rico",

"America/Rainy_River",

"America/Rankin_Inlet",

"America/Recife",

"America/Regina",

"America/Resolute",

"America/Rio_Branco",

"America/Santa_Isabel",

"America/Santarem",

"America/Santiago",

"America/Santo_Domingo",

"America/Sao_Paulo",

"America/Scoresbysund",

"America/Sitka",

"America/St_Barthelemy",

"America/St_Johns",

"America/St_Kitts",

"America/St_Lucia",

"America/St_Thomas",

"America/St_Vincent",

"America/Swift_Current",

"America/Tegucigalpa",

"America/Thule",

"America/Thunder_Bay",

"America/Tijuana",

"America/Toronto",

"America/Tortola",

"America/Vancouver",

"America/Whitehorse",

"America/Winnipeg",

"America/Yakutat",

"America/Yellowknife",

"Antarctica/Casey",

"Antarctica/Davis",

"Antarctica/DumontDUrville",

"Antarctica/Macquarie",

"Antarctica/Mawson",

"Antarctica/McMurdo",

"Antarctica/Palmer",

"Antarctica/Rothera",

"Antarctica/Syowa",

"Antarctica/Vostok",

"Arctic/Longyearbyen",

"Asia/Aden",

"Asia/Almaty",

"Asia/Amman",

"Asia/Anadyr",

"Asia/Aqtau",

"Asia/Aqtobe",

"Asia/Ashgabat",

"Asia/Baghdad",

"Asia/Bahrain",

"Asia/Baku",

"Asia/Bangkok",

"Asia/Beirut",

"Asia/Bishkek",

"Asia/Brunei",

"Asia/Choibalsan",

"Asia/Chongqing",

"Asia/Colombo",

"Asia/Damascus",

"Asia/Dhaka",

"Asia/Dili",

"Asia/Dubai",

"Asia/Dushanbe",

"Asia/Gaza",

"Asia/Harbin",

"Asia/Hebron",

"Asia/Ho_Chi_Minh",

"Asia/Hong_Kong",

"Asia/Hovd",

"Asia/Irkutsk",

"Asia/Jakarta",

"Asia/Jayapura",

"Asia/Jerusalem",

"Asia/Kabul",

"Asia/Kamchatka",

"Asia/Karachi",

"Asia/Kashgar",

"Asia/Kathmandu",

"Asia/Khandyga",

"Asia/Kolkata",

"Asia/Krasnoyarsk",

"Asia/Kuala_Lumpur",

"Asia/Kuching",

"Asia/Kuwait",

"Asia/Macau",

"Asia/Magadan",

"Asia/Makassar",

"Asia/Manila",

"Asia/Muscat",

"Asia/Nicosia",

"Asia/Novokuznetsk",

"Asia/Novosibirsk",

"Asia/Omsk",

"Asia/Oral",

"Asia/Phnom_Penh",

"Asia/Pontianak",

"Asia/Pyongyang",

"Asia/Qatar",

"Asia/Qyzylorda",

"Asia/Rangoon",

"Asia/Riyadh",

"Asia/Sakhalin",

"Asia/Samarkand",

"Asia/Seoul",

"Asia/Shanghai",

"Asia/Singapore",

"Asia/Taipei",

"Asia/Tashkent",

"Asia/Tbilisi",

"Asia/Tehran",

"Asia/Thimphu",

"Asia/Tokyo",

"Asia/Ulaanbaatar",

"Asia/Urumqi",

"Asia/Ust-Nera",

"Asia/Vientiane",

"Asia/Vladivostok",

"Asia/Yakutsk",

"Asia/Yekaterinburg",

"Asia/Yerevan",

"Atlantic/Azores",

"Atlantic/Bermuda",

"Atlantic/Canary",

"Atlantic/Cape_Verde",

"Atlantic/Faroe",

"Atlantic/Madeira",

"Atlantic/Reykjavik",

"Atlantic/South_Georgia",

"Atlantic/St_Helena",

"Atlantic/Stanley",

"Australia/Adelaide",

"Australia/Brisbane",

"Australia/Broken_Hill",

"Australia/Currie",

"Australia/Darwin",

"Australia/Eucla",

"Australia/Hobart",

"Australia/Lindeman",

"Australia/Lord_Howe",

"Australia/Melbourne",

"Australia/Perth",

"Australia/Sydney",

"Canada/Atlantic",

"Canada/Central",

"Canada/Eastern",

"Canada/Mountain",

"Canada/Newfoundland",

"Canada/Pacific",

"Europe/Amsterdam",

"Europe/Andorra",

"Europe/Athens",

"Europe/Belgrade",

"Europe/Berlin",

"Europe/Bratislava",

"Europe/Brussels",

"Europe/Bucharest",

"Europe/Budapest",

"Europe/Busingen",

"Europe/Chisinau",

"Europe/Copenhagen",

"Europe/Dublin",

"Europe/Gibraltar",

"Europe/Guernsey",

"Europe/Helsinki",

"Europe/Isle_of_Man",

"Europe/Istanbul",

"Europe/Jersey",

"Europe/Kaliningrad",

"Europe/Kiev",

"Europe/Lisbon",

"Europe/Ljubljana",

"Europe/London",

"Europe/Luxembourg",

"Europe/Madrid",

"Europe/Malta",

"Europe/Mariehamn",

"Europe/Minsk",

"Europe/Monaco",

"Europe/Moscow",

"Europe/Oslo",

"Europe/Paris",

"Europe/Podgorica",

"Europe/Prague",

"Europe/Riga",

"Europe/Rome",

"Europe/Samara",

"Europe/San_Marino",

"Europe/Sarajevo",

"Europe/Simferopol",

"Europe/Skopje",

"Europe/Sofia",

"Europe/Stockholm",

"Europe/Tallinn",

"Europe/Tirane",

"Europe/Uzhgorod",

"Europe/Vaduz",

"Europe/Vatican",

"Europe/Vienna",

"Europe/Vilnius",

"Europe/Volgograd",

"Europe/Warsaw",

"Europe/Zagreb",

"Europe/Zaporozhye",

"Europe/Zurich",

"GMT",

"Indian/Antananarivo",

"Indian/Chagos",

"Indian/Christmas",

"Indian/Cocos",

"Indian/Comoro",

"Indian/Kerguelen",

"Indian/Mahe",

"Indian/Maldives",

"Indian/Mauritius",

"Indian/Mayotte",

"Indian/Reunion",

"Pacific/Apia",

"Pacific/Auckland",

"Pacific/Chatham",

"Pacific/Chuuk",

"Pacific/Easter",

"Pacific/Efate",

"Pacific/Enderbury",

"Pacific/Fakaofo",

"Pacific/Fiji",

"Pacific/Funafuti",

"Pacific/Galapagos",

"Pacific/Gambier",

"Pacific/Guadalcanal",

"Pacific/Guam",

"Pacific/Honolulu",

"Pacific/Johnston",

"Pacific/Kiritimati",

"Pacific/Kosrae",

"Pacific/Kwajalein",

"Pacific/Majuro",

"Pacific/Marquesas",

```
            "Pacific/Midway",
            "Pacific/Nauru",
            "Pacific/Niue",
            "Pacific/Norfolk",
            "Pacific/Noumea",
            "Pacific/Pago_Pago",
            "Pacific/Palau",
            "Pacific/Pitcairn",
            "Pacific/Pohnpei",
            "Pacific/Port_Moresby",
            "Pacific/Rarotonga",
            "Pacific/Saipan",
            "Pacific/Tahiti",
            "Pacific/Tarawa",
            "Pacific/Tongatapu",
            "Pacific/Wake",
            "Pacific/Wallis",
            "US/Alaska",
            "US/Arizona",
            "US/Central",
            "US/Eastern",
            "US/Hawaii",
            "US/Mountain",
            "US/Pacific",
            "UTC"
        ],
        "Description": "Choose the default Time Zone. Default is 'UTC'"
    },
    "Regions": {
        "Type": "CommaDelimitedList",
        "Description": "List of regions in which instances are scheduled, leave blank for current region only."
    },
    "CrossAccountRoles": {
        "Type": "CommaDelimitedList",
        "Description": "Comma separated list of ARN's for cross account access roles. These roles must be created in all checked accounts the scheduler to start and stop instances."
    },
```

```
    "StartedTags": {
        "Type": "String",
        "Description": "Comma separated list of tagname and values on the formt name=value,name=value,.. that are set on started instances"
    },
    "StoppedTags": {
        "Type": "String",
        "Description": "Comma separated list of tagname and values on the formt name=value,name=value,.. that are set on stopped instances"
    },
    "SchedulerFrequency": {
        "Type": "String",
        "AllowedValues": [
            "1",
            "2",
            "5",
            "10",
            "15",
            "30",
            "60"
        ],
        "Default": "5",
        "Description": "Scheduler running frequency in minutes."
    },
    "ScheduleLambdaAccount": {
        "Type": "String",
        "AllowedValues": [
            "Yes",
            "No"
        ],
        "Default": "Yes",
        "Description": "Schedule instances in this account."
    },
    "SendAnonymousData": {
        "Type": "String",
        "AllowedValues": [
            "Yes",
```

```
        "No"
      ],
      "Default": "Yes",
      "Description": "Send Anonymous Metrics Data."
    }
  },
  "Metadata": {
    "AWS::CloudFormation::Interface": {
      "ParameterGroups": [
        {
          "Label": {
            "default": "Scheduler (version 2.2.2.0)"
          },
          "Parameters": [
            "TagName",
            "ScheduledServices",
            "SchedulingActive",
            "Regions",
            "DefaultTimezone",
            "CrossAccountRoles",
            "ScheduleLambdaAccount",
            "SchedulerFrequency",
            "MemorySize"
          ]
        },
        {
          "Label": {
            "default": "Options"
          },
          "Parameters": [
            "UseCloudWatchMetrics",
            "SendAnonymousData",
            "Trace"
          ]
        },
        {
```

```
        "Label": {
            "default": "Other parameters"
        },
        "Parameters": [
            "LogRetentionDays",
            "StartedTags",
            "StoppedTags"
        ]
    }
],
"ParameterLabels": {
    "LogRetentionDays": {
        "default": "Log retention days"
    },
    "StartedTags": {
        "default": "Started tags"
    },
    "StoppedTags": {
        "default": "Stopped tags"
    },
    "SchedulingActive": {
        "default": "Scheduling enabled"
    },
    "CrossAccountRoles": {
        "default": "Cross-account roles"
    },
    "ScheduleLambdaAccount": {
        "default": "This account"
    },
    "UseCloudWatchMetrics": {
        "default": "Enable CloudWatch Metrics"
    },
    "Trace": {
        "default": "Enable CloudWatch Logs"
    },
    "TagName": {
```

```
            "default": "Instance Scheduler tag name"
        },
        "ScheduledServices": {
            "default": "Service(s) to schedule"
        },
        "DefaultTimezone": {
            "default": "Default time zone"
        },
        "SchedulerFrequency": {
            "default": "Frequency"
        },
        "Regions": {
            "default": "Region(s)"
        },
        "MemorySize": {
            "default": "Memory size"
        },
        "SendAnonymousData": {
            "default": "Send anonymous usage data"
        }
      }
    }
},
"Mappings": {
    "TrueFalse": {
      "Yes": {
        "Value": "True"
      },
      "No": {
        "Value": "False"
      }
    },
    "EnabledDisabled": {
      "Yes": {
        "Value": "ENABLED"
      },
```

```
    "No": {
        "Value": "DISABLED"
    }
},
"Services": {
    "EC2": {
        "Value": "ec2"
    },
    "RDS": {
        "Value": "rds"
    },
    "Both": {
        "Value": "ec2,rds"
    }
},
"Timeouts": {
    "1": {
        "Value": "cron(0/1 * * * ? *)"
    },
    "2": {
        "Value": "cron(0/2 * * * ? *)"
    },
    "5": {
        "Value": "cron(0/5 * * * ? *)"
    },
    "10": {
        "Value": "cron(0/10 * * * ? *)"
    },
    "15": {
        "Value": "cron(0/15 * * * ? *)"
    },
    "30": {
        "Value": "cron(0/30 * * * ? *)"
    },
    "60": {
        "Value": "cron(0 0/1 * * ? *)"
```

```
        }
    },
    "Settings": {
        "Metrics": {
            "Url": "https://metrics.awssolutionsbuilder.com/generic",
            "SolutionId": "S00030"
        },
        "DynamoDbTableCapacities": {
            "ConfigTableMinRead": 2,
            "ConfigTableMaxRead": 10,
            "ConfigTableMinWrite": 2,
            "ConfigTableMaxWrite": 5,
            "StateTableMinRead": 5,
            "StateTableMaxRead": 50,
            "StateTableMinWrite": 5,
            "StateTableMaxWrite": 50,
            "RdsTagCacheTableMinRead": 5,
            "RdsTagCacheTableMaxRead": 50,
            "RdsTagCacheTableMinWrite": 5,
            "RdsTagCacheTableMaxWrite": 50
        },
        "RdsTagCaching": {
            "Enabled": "True"
        }
    }
},
"Conditions": {
    "NotUsingCrossAccountRoles": {
        "Fn::Equals": [
            {
                "Fn::Join": [
                    "",
                    {
                        "Ref": "CrossAccountRoles"
                    }
                ]
```

```
        },
        ""
      ]
    }
  },
  "Resources": {
    "SchedulerPolicy": {
      "Type": "AWS::IAM::Policy",
      "Properties": {
        "PolicyName": "SchedulerPolicy",
        "Roles": [
          {
            "Ref": "SchedulerRole"
          }
        ],
        "PolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
              ],
              "Resource": "*"
            },
            {
              "Effect": "Allow",
              "Action": [
                "ec2:CreateTags",
                "ec2:DeleteTags",
                "ec2:DescribeInstances",
                "ec2:DescribeRegions",
                "ec2:ModifyInstanceAttribute",
                "ec2:StartInstances",
```

```
            "ec2:StopInstances"
        ],
        "Resource": [
            "*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "rds:AddTagsToResource",
            "rds:DeleteDBSnapshot",
            "rds:DescribeDBInstances",
            "rds:DescribeDBSnapshots",
            "rds:RemoveTagsFromResource",
            "rds:StartDBInstance",
            "rds:StopDBInstance"
        ],
        "Resource": [
            "*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "dynamodb:DeleteItem",
            "dynamodb:GetItem",
            "dynamodb:PutItem",
            "dynamodb:Query",
            "dynamodb:Scan",
            "dynamodb:BatchWriteItem"
        ],
        "Resource": [
            {
                "Fn::Join": [
                    "",
                    [
```

```
            {
              "Fn::Join": [
                ":",
                [
                  "arn:aws:dynamodb",
                  {
                    "Ref": "AWS::Region"
                  },
                  {
                    "Ref": "AWS::AccountId"
                  },
                  "table/"
                ]
              ]
            },
            {
              "Ref": "StateTable"
            }
          ]
        ]
      },
      {
        "Fn::Join": [
          "",
          [
            {
              "Fn::Join": [
                ":",
                [
                  "arn:aws:dynamodb",
                  {
                    "Ref": "AWS::Region"
                  },
                  {
                    "Ref": "AWS::AccountId"
                  },
```

```
                        "table/"
                    ]
                ]
            },
            {
                "Ref": "ConfigTable"
            }
        ]
    }
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "logs:DescribeLogStreams",
    "logs:PutRetentionPolicy"
  ],
  "Resource": [
    "*"
  ]
},
{
  "Effect": "Allow",
  "Action": "sns:Publish",
  "Resource": [
    {
      "Ref": "InstanceSchedulerSnsTopic"
    }
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "lambda:InvokeFunction"
  ],
```

```
    "Resource": [
      {
        "Fn::Join": [
          ":",
          [
            "arn:aws:lambda",
            {
              "Ref": "AWS::Region"
            },
            {
              "Ref": "AWS::AccountId"
            },
            "function",
            {
              "Fn::Join": [
                "-",
                [
                  {
                    "Ref": "AWS::StackName"
                  },
                  "InstanceSchedulerMain"
                ]
              ]
            }
          ]
        ]
      }
    ]
  },
  {
    "Effect": "Allow",
    "Action": [
      "tag:GetResources"
    ],
    "Resource": [
      "*"
```

```
            ]
          },
          {
            "Effect": "Allow",
            "Action": [
              "cloudwatch:PutMetricData"
            ],
            "Resource": [
              "*"
            ]
          },
          {
            "Effect": "Allow",
            "Action": [
              "sts:AssumeRole"
            ],
            "Resource": {
              "Fn::If": [
                "NotUsingCrossAccountRoles",
                {
                  "Fn::Sub": "arn:aws:iam::${AWS::AccountId}:role/None"
                },
                {
                  "Ref": "CrossAccountRoles"
                }
              ]
            }
          }
        ]
      }
    }
  },
  "SchedulerRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
```

```
            "Version": "2012-10-17",
            "Statement": [
              {
                "Effect": "Allow",
                "Principal": {
                  "Service": "lambda.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
              },
              {
                "Effect": "Allow",
                "Principal": {
                  "Service": "events.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
              }
            ]
          },
          "Path": "/"
        }
      },
      "SchedulerDynamoDBScalingRole": {
        "Type": "AWS::IAM::Role",
        "Properties": {
          "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
              {
                "Effect": "Allow",
                "Principal": {
                  "Service": "application-autoscaling.amazonaws.com",
                  "AWS": {
                    "Fn::GetAtt": [
                      "SchedulerRole",
                      "Arn"
                    ]
```

```
                }
            },
            "Action": "sts:AssumeRole"
        }
      ]
    },
    "Policies": [
      {
        "PolicyName": "OpsAutomatorScalingRolePolicy",
        "PolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Action": [
                "dynamodb:DescribeTable",
                "dynamodb:UpdateTable",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:SetAlarmState",
                "cloudwatch:DeleteAlarms"
              ],
              "Resource": "*"
            }
          ]
        }
      }
    ],
    "ManagedPolicyArns": [
      "arn:aws:iam::aws:policy/service-role/AWSLambdaDynamoDBExecutionRole",
      "arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole"
    ],
    "Path": "/"
  },
  "DependsOn": [
```

```
        "SchedulerRole"
      ]
    },
    "StateTable": {
      "Type": "AWS::DynamoDB::Table",
      "Properties": {
        "AttributeDefinitions": [
          {
            "AttributeName": "service",
            "AttributeType": "S"
          },
          {
            "AttributeName": "account-region",
            "AttributeType": "S"
          }
        ],
        "KeySchema": [
          {
            "AttributeName": "service",
            "KeyType": "HASH"
          },
          {
            "AttributeName": "account-region",
            "KeyType": "RANGE"
          }
        ],
        "ProvisionedThroughput": {
          "ReadCapacityUnits": {
            "Fn::FindInMap": [
              "Settings",
              "DynamoDbTableCapacities",
              "StateTableMinRead"
            ]
          },
          "WriteCapacityUnits": {
            "Fn::FindInMap": [
```

```
                "Settings",
                "DynamoDbTableCapacities",
                "StateTableMinWrite"
            ]
          }
        }
    },
"StateTableAutoScalingReadTarget": {
    "Type": "AWS::ApplicationAutoScaling::ScalableTarget",
    "Properties": {
        "MaxCapacity": {
            "Fn::FindInMap": [
                "Settings",
                "DynamoDbTableCapacities",
                "StateTableMaxRead"
            ]
        },
        "MinCapacity": {
            "Fn::FindInMap": [
                "Settings",
                "DynamoDbTableCapacities",
                "StateTableMinRead"
            ]
        },
        "ResourceId": {
            "Fn::Join": [
                "/",
                [
                    "table",
                    {
                        "Ref": "StateTable"
                    }
                ]
            ]
        },
```

```
      "RoleARN": {
        "Fn::GetAtt": [
          "SchedulerDynamoDBScalingRole",
          "Arn"
        ]
      },
      "ScalableDimension": "dynamodb:table:ReadCapacityUnits",
      "ServiceNamespace": "dynamodb"
    }
  },
  "StateTableAutoScalingReadPolicy": {
    "Type": "AWS::ApplicationAutoScaling::ScalingPolicy",
    "Properties": {
      "PolicyName": {
        "Fn::Join": [
          "-",
          [
            {
              "Ref": "AWS::StackName"
            },
            "StateTableAutoScalingReadPolicy"
          ]
        ]
      },
      "PolicyType": "TargetTrackingScaling",
      "ScalingTargetId": {
        "Ref": "StateTableAutoScalingReadTarget"
      },
      "TargetTrackingScalingPolicyConfiguration": {
        "TargetValue": 75.0,
        "ScaleInCooldown": 60,
        "ScaleOutCooldown": 60,
        "PredefinedMetricSpecification": {
          "PredefinedMetricType": "DynamoDBReadCapacityUtilization"
        }
      }
```

```
    }
  },
  "StateTableAutoScalingWriteTarget": {
    "Type": "AWS::ApplicationAutoScaling::ScalableTarget",
    "Properties": {
      "MaxCapacity": {
        "Fn::FindInMap": [
          "Settings",
          "DynamoDbTableCapacities",
          "StateTableMaxWrite"
        ]
      },
      "MinCapacity": {
        "Fn::FindInMap": [
          "Settings",
          "DynamoDbTableCapacities",
          "StateTableMinWrite"
        ]
      },
      "ResourceId": {
        "Fn::Join": [
          "/",
          [
            "table",
            {
              "Ref": "StateTable"
            }
          ]
        ]
      },
      "RoleARN": {
        "Fn::GetAtt": [
          "SchedulerDynamoDBScalingRole",
          "Arn"
        ]
      },
```

```
        "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
        "ServiceNamespace": "dynamodb"
      }
    },
    "StateTableAutoScalingWritePolicy": {
      "Type": "AWS::ApplicationAutoScaling::ScalingPolicy",
      "Properties": {
        "PolicyName": {
          "Fn::Join": [
            "-",
            [
              {
                "Ref": "AWS::StackName"
              },
              "StateTableAutoScalingWritePolicy"
            ]
          ]
        },
        "PolicyType": "TargetTrackingScaling",
        "ScalingTargetId": {
          "Ref": "StateTableAutoScalingWriteTarget"
        },
        "TargetTrackingScalingPolicyConfiguration": {
          "TargetValue": 75.0,
          "ScaleInCooldown": 60,
          "ScaleOutCooldown": 60,
          "PredefinedMetricSpecification": {
            "PredefinedMetricType": "DynamoDBWriteCapacityUtilization"
          }
        }
      },
      "DependsOn": [
        "StateTableAutoScalingReadPolicy"
      ]
    },
    "ConfigTable": {
```

```
    "Type": "AWS::DynamoDB::Table",
    "Properties": {
      "AttributeDefinitions": [
        {
          "AttributeName": "type",
          "AttributeType": "S"
        },
        {
          "AttributeName": "name",
          "AttributeType": "S"
        }
      ],
      "KeySchema": [
        {
          "AttributeName": "type",
          "KeyType": "HASH"
        },
        {
          "AttributeName": "name",
          "KeyType": "RANGE"
        }
      ],
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 2,
        "WriteCapacityUnits": 2
      }
    }
  },
  "ConfigTableAutoScalingReadTarget": {
    "Type": "AWS::ApplicationAutoScaling::ScalableTarget",
    "Properties": {
      "MaxCapacity": {
        "Fn::FindInMap": [
          "Settings",
          "DynamoDbTableCapacities",
          "ConfigTableMaxRead"
```

```
            ]
        },
        "MinCapacity": {
            "Fn::FindInMap": [
                "Settings",
                "DynamoDbTableCapacities",
                "ConfigTableMinRead"
            ]
        },
        "ResourceId": {
            "Fn::Join": [
                "/",
                [
                    "table",
                    {
                        "Ref": "ConfigTable"
                    }
                ]
            ]
        },
        "RoleARN": {
            "Fn::GetAtt": [
                "SchedulerDynamoDBScalingRole",
                "Arn"
            ]
        },
        "ScalableDimension": "dynamodb:table:ReadCapacityUnits",
        "ServiceNamespace": "dynamodb"
    }
},
"ConfigurationTableAutoScalingReadPolicy": {
    "Type": "AWS::ApplicationAutoScaling::ScalingPolicy",
    "Properties": {
        "PolicyName": {
            "Fn::Join": [
                "-",
```

```
            [
              {
                "Ref": "AWS::StackName"
              },
              "ConfigTableAutoScalingReadPolicy"
            ]
          ]
        },
        "PolicyType": "TargetTrackingScaling",
        "ScalingTargetId": {
          "Ref": "ConfigTableAutoScalingReadTarget"
        },
        "TargetTrackingScalingPolicyConfiguration": {
          "TargetValue": 75.0,
          "ScaleInCooldown": 60,
          "ScaleOutCooldown": 60,
          "PredefinedMetricSpecification": {
            "PredefinedMetricType": "DynamoDBReadCapacityUtilization"
          }
        }
      },
      "DependsOn": "StateTableAutoScalingWritePolicy"
    },
    "ConfigTableAutoScalingWriteTarget": {
      "Type": "AWS::ApplicationAutoScaling::ScalableTarget",
      "Properties": {
        "MaxCapacity": {
          "Fn::FindInMap": [
            "Settings",
            "DynamoDbTableCapacities",
            "ConfigTableMaxWrite"
          ]
        },
        "MinCapacity": {
          "Fn::FindInMap": [
            "Settings",
```

```
            "DynamoDbTableCapacities",
            "ConfigTableMinWrite"
          ]
        },
        "ResourceId": {
          "Fn::Join": [
            "/",
            [
              "table",
              {
                "Ref": "ConfigTable"
              }
            ]
          ]
        },
        "RoleARN": {
          "Fn::GetAtt": [
            "SchedulerDynamoDBScalingRole",
            "Arn"
          ]
        },
        "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
        "ServiceNamespace": "dynamodb"
      }
    },
    "ConfigTableAutoScalingWritePolicy": {
      "Type": "AWS::ApplicationAutoScaling::ScalingPolicy",
      "Properties": {
        "PolicyName": {
          "Fn::Join": [
            "-",
            [
              {
                "Ref": "AWS::StackName"
              },
              "ConfigTableAutoScalingWritePolicy"
```

```
            ]
          ]
        },
        "PolicyType": "TargetTrackingScaling",
        "ScalingTargetId": {
          "Ref": "ConfigTableAutoScalingWriteTarget"
        },
        "TargetTrackingScalingPolicyConfiguration": {
          "TargetValue": 75.0,
          "ScaleInCooldown": 60,
          "ScaleOutCooldown": 60,
          "PredefinedMetricSpecification": {
            "PredefinedMetricType": "DynamoDBWriteCapacityUtilization"
          }
        }
      },
      "DependsOn": [
        "ConfigurationTableAutoScalingReadPolicy"
      ]
    },
    "SchedulerLogGroup": {
      "Type": "AWS::Logs::LogGroup",
      "Properties": {
        "LogGroupName": {
          "Fn::Join": [
            "-",
            [
              {
                "Ref": "AWS::StackName"
              },
              "logs"
            ]
          ]
        },
        "RetentionInDays": {
          "Ref": "LogRetentionDays"
```

```
        }
      }
    },
    "InstanceSchedulerSnsTopic": {
      "Type": "AWS::SNS::Topic",
      "Properties": {
        "DisplayName": {
          "Ref": "AWS::StackName"
        }
      }
    },
    "Main": {
      "Type": "AWS::Lambda::Function",
      "Properties": {
        "FunctionName": {
          "Fn::Join": [
            "-",
            [
              {
                "Ref": "AWS::StackName"
              },
              "InstanceSchedulerMain"
            ]
          ]
        },
        "Code": {
          "S3Bucket": {
            "Fn::Join": [
              "-",
              [
                "solutions",
                {
                  "Ref": "AWS::Region"
                }
              ]
            ]
```

```
    },
    "S3Key": "aws-instance-scheduler/latest/instance-scheduler-2.2.2.0.zip"
},
"Handler": "main.lambda_handler",
"Runtime": "python2.7",
"Role": {
    "Fn::GetAtt": [
        "SchedulerRole",
        "Arn"
    ]
},
"Environment": {
    "Variables": {
        "CONFIG_TABLE": {
            "Ref": "ConfigTable"
        },
        "TAG_NAME": {
            "Ref": "TagName"
        },
        "STATE_TABLE": {
            "Ref": "StateTable"
        },
        "LOG_GROUP": {
            "Ref": "SchedulerLogGroup"
        },
        "ACCOUNT": {
            "Ref": "AWS::AccountId"
        },
        "ISSUES_TOPIC_ARN": {
            "Ref": "InstanceSchedulerSnsTopic"
        },
        "STACK_NAME": {
            "Ref": "AWS::StackName"
        },
        "BOTO_RETRY": "5,10,30,0.25",
        "ENV_BOTO_RETRY_LOGGING": "False",
```

```
"SEND_METRICS": {
   "Fn::FindInMap": [
      "TrueFalse",
      {
         "Ref": "SendAnonymousData"
      },
      "Value"
   ]
},
"SOLUTION_ID": {
   "Fn::FindInMap": [
      "Settings",
      "Metrics",
      "SolutionId"
   ]
},
"TRACE": {
   "Fn::FindInMap": [
      "TrueFalse",
      {
         "Ref": "Trace"
      },
      "Value"
   ]
},
"USER_AGENT": {
   "Fn::Join": [
      "-",
      [
         "InstanceScheduler",
         {
            "Ref": "AWS::StackName"
         },
         "2.2.2.0"
      ]
   ]
```

```
            },
            "METRICS_URL": {
               "Fn::FindInMap": [
                  "Settings",
                  "Metrics",
                  "Url"
               ]
            }
         }
      },
      "MemorySize": {
         "Ref": "MemorySize"
      },
      "Timeout": 300,
      "Description": "ECS and RDS instance scheduler, version 2.2.2.0"
   },
   "DependsOn": [
      "SchedulerPolicy",
      "SchedulerRole"
   ]
},
"SchedulerInvokePermission": {
   "Type": "AWS::Lambda::Permission",
   "Properties": {
      "FunctionName": {
         "Fn::Join": [
            ":",
            [
               "arn:aws:lambda",
               {
                  "Ref": "AWS::Region"
               },
               {
                  "Ref": "AWS::AccountId"
               },
               "function",
```

```
                {
                  "Fn::Join": [
                    "-",
                    [
                      {
                        "Ref": "AWS::StackName"
                      },
                      "InstanceSchedulerMain"
                    ]
                  ]
                }
              ]
            ]
          },
          "Action": "lambda:InvokeFunction",
          "Principal": "events.amazonaws.com",
          "SourceArn": {
            "Fn::GetAtt": [
              "SchedulerRule",
              "Arn"
            ]
          }
        },
        "DependsOn": "Main"
      },
      "SchedulerRule": {
        "Type": "AWS::Events::Rule",
        "Properties": {
          "Description": "Instance Scheduler - Rule to trigger instance for scheduler function version 2.2.2.0",
          "ScheduleExpression": {
            "Fn::FindInMap": [
              "Timeouts",
              {
                "Ref": "SchedulerFrequency"
              },
              "Value"
```

```
            ]
        },
        "State": {
            "Fn::FindInMap": [
                "EnabledDisabled",
                {
                    "Ref": "SchedulingActive"
                },
                "Value"
            ]
        },
        "Targets": [
            {
                "Id": "Instance-Scheduler-Main",
                "Arn": {
                    "Fn::GetAtt": [
                        "Main",
                        "Arn"
                    ]
                }
            }
        ]
    }
},
"SchedulerConfigHelper": {
    "Type": "Custom::ServiceSetup",
    "Properties": {
        "ServiceToken": {
            "Fn::GetAtt": [
                "Main",
                "Arn"
            ]
        },
        "timeout": "120",
        "config_table": {
            "Ref": "ConfigTable"
```

```
},
"tagname": {
   "Ref": "TagName"
},
"default_timezone": {
   "Ref": "DefaultTimezone"
},
"use_metrics": {
   "Fn::FindInMap": [
      "TrueFalse",
      {
         "Ref": "UseCloudWatchMetrics"
      },
      "Value"
   ]
},
"scheduled_services": {
   "Fn::Split": [
      ",",
      {
         "Fn::FindInMap": [
            "Services",
            {
               "Ref": "ScheduledServices"
            },
            "Value"
         ]
      }
   ]
},
"regions": {
   "Ref": "Regions"
},
"cross_account_roles": {
   "Ref": "CrossAccountRoles"
},
```

```
        "schedule_lambda_account": {
          "Fn::FindInMap": [
            "TrueFalse",
            {
              "Ref": "ScheduleLambdaAccount"
            },
            "Value"
          ]
        },
        "trace": {
          "Fn::FindInMap": [
            "TrueFalse",
            {
              "Ref": "Trace"
            },
            "Value"
          ]
        },
        "log_retention_days": {
          "Ref": "LogRetentionDays"
        },
        "started_tags": {
          "Ref": "StartedTags"
        },
        "stopped_tags": {
          "Ref": "StoppedTags"
        },
        "stack_version": "2.2.2.0"
      },
      "DependsOn": [
        "Main",
        "ConfigTable",
        "SchedulerLogGroup"
      ]
    }
},
```

```
  "Outputs": {
    "AccountId": {
      "Value": {
        "Ref": "AWS::AccountId"
      },
      "Description": "Account to give access to when creating cross-account access role fro cross account scenario
"
    },
    "ConfigurationTable": {
      "Value": {
        "Ref": "ConfigTable"
      },
      "Description": "Name of the DynomoDB configuration table"
    },
    "IssueSnsTopicArn": {
      "Value": {
        "Ref": "InstanceSchedulerSnsTopic"
      },
      "Description": "Topic to subscribe to for notifications of errors and warnings"
    },
    "ServiceInstanceScheduleServiceToken": {
      "Value": {
        "Fn::GetAtt": [
          "Main",
          "Arn"
        ]
      },
      "Description": "Arn to use as ServiceToken property for custom resource type
Custom::ServiceInstanceSchedule"
    }
  }
}
```

Python Code for Lambda Function:

```
import json
import os
import traceback
from datetime import datetime
from time import time
import configuration
import util

import boto_retry

from requesthandlers.admin_cli_request_handler import AdminCliRequestHandler
from requesthandlers.cloudwatch_event_handler import CloudWatchEventHandler
from requesthandlers.schedule_resource_handler import ScheduleResourceHandler
from requesthandlers.scheduler_request_handler import SchedulerRequestHandler
from requesthandlers.scheduler_setup_handler import SchedulerSetupHandler
from util.logger import Logger

LOG_STREAM = "InstanceScheduler-{:0>4d}{:0>2d}{:0>2d}"

def lambda_handler(event, context):
    try:
        dt = datetime.utcnow()
        log_stream = LOG_STREAM.format(dt.year, dt.month, dt.day)
        result = {}
        with Logger(logstream=log_stream, buffersize=20, context=context,
                    debug=util.as_bool(os.getenv(configuration.ENV_TRACE, False))) as logger:

            logger.info("InstanceScheduler, version %version%")

            logger.debug("Event is {}", util.safe_json(event, indent=3))

            for handler_type in [SchedulerRequestHandler,
                                 SchedulerSetupHandler,
                                 ScheduleResourceHandler,
                                 AdminCliRequestHandler,
```

```
                    CloudWatchEventHandler]:


            if handler_type.is_handling_request(event):
                start = time()
                handler = handler_type(event, context)
                logger.info("Handler is {}".format(handler_type.__name__))
                try:
                    result = handler.handle_request()
                except Exception as e:
                    logger.error("Error handling request {} by handler {}: ({})\n{}", json.dumps(event),
handler_type.__name__,
                                 e, traceback.format_exc())
                execution_time = round(float((time() - start)), 3)
                logger.info("Handling took {} seconds", execution_time)
                return result
        logger.debug("Request was not handled, no handler was able to handle this type of request {}",
json.dumps(event))
    finally:
        configuration.unload_scheduler_configuration()
```

- Python code is packed along with resources and stored in zip format over S3 and the key is given in script.

FUTURE IMPROVEMENTS

- Using Git for versioning and as repository, Jenkins can be used to monitor the change in git and trigger the job for updating the stack
- Dockerizing i.e conversion of servers into application loaded container

PRESENTATION LINK

https://s3.amazonaws.com/encodetcs/inframind+Chanakya_Sowjanya.pptx