**Design Document**

1. **CloudFormation Template Design for IAM**
   1. **IAM.yaml**

Purpose : This template will invoke all the nested templates and will avoid creation of single template for all the resources.Any IAM resource that is created will be called in the childstacks

**AWSTemplateFormatVersion**: Gives Information about the version of Cloud

**Description:** 'AWS CloudFormation IAM Master Template'

**Resources:**

**Users:** this child stack will create the user

**Type : "AWS::CloudFormation::Stack" -** the type mentioned for nested CloudFormation template

**Properties:** Properties of the childstack refered to

**Parameters:** Hold values for Inputs required for each child stack.

**Username:** “Username of the user that will be created”

**Password:** “Password for the user to login

**TemplateURL:** Path to the CreateUserStack.yaml

**Tags:** For tagging this resource

- **Key:** Name

**Value:** ChildStack01

**CreateUserStack02:** this child stack will create the user

**Type : "AWS::CloudFormation::Stack" -** the type mentioned for nested CloudFormation template

**Properties:** Properties of the childstack refered to

**Parameters:** Hold values for Inputs required for each child stack.

**Username:** “Username of another user that will be created”

**Password:** “Password for the user to login

**TemplateURL:** Path to the CreateUserStack.yaml

**Tags:** For tagging this resource

- **Key:** Name

**Value:** ChildStack02

**CreateGroupStack:**

**Type : "AWS::CloudFormation::Stack"**

**Properties:**

**Parameters:** Hold values for Inputs required for each child stack.

**TemplateURL:** Path to the CreateGroupStack.yaml

**Tags:**

- **Key:** Name

**Value:** ChildStack02

**RoleCreationStack:**

**Parameters:** Hold values for Inputs required for each child stack.

**Type : "AWS::CloudFormation::Stack"**

**Properties:**

**TemplateURL:**

**Tags:**

- **Key:** Name

**Value:** ChildStack03

**Outputs**: Outputs will be the outputs of the all child stacks .

**KeyAcces:** Gives the Access key and Secret Access Key for the new user created

**Value:**

**Fn::GetAtt:** ChildStack.Outputs.AccessKey –To refer the output of the childstack

**Description:** AWSAccessKeyId of new user

**KeySecret:**

**Value:**

Fn::GetAtt: ChildStack.Outputs.SecretKey

Description: AWSSecretKey of new user

* 1. **CreateUserTemplate.yaml**

Purpose:Users can be created following the template and creating individual resource for a individual user.

Each User will be created using the follwing template :

**Parameters:** Input parameter for the stack can be the Username /Password

**Resources:** List of resources

**NewUser:** Name of the IAM resource for user creation

**Type: AWS::IAM::User –** Type of the IAM resource

**Properties:**

**Path: '/' -** path to the user created

**Policies:** Policy for the created user

**Fn::ImportValue: "UserPolicyStack-ManagedPolicies" –** referestothe policies for the user in a separate stack.

**LoginProfile:** User login detail for console acccess

**Password:** Password for user

**PasswordResetRequired: yes –** Binary value to reset the password for the user.

**UserName:** required name for the user to be created

**NewKeys:** Defing access keys for the users

**Type: AWS::IAM::AccessKey –** Type of IAM Resource for Access Keys.

**Properties:** Properties for access key binding to user

**UserName:** the user name can be refered from the username crearted using !Ref UserName.

**Outputs:** Refers to output for this stack which is the access key and secret access key

**AccessKey:**

**Value: !Ref 'NewKeys'** Refers to key resource

**Description:** Decribes the details of the output

**SecretKey:**

**Value: !GetAtt [NewKeys, SecretAccessKey]**

**Description:** Decribes the details of the output

* 1. **CreateRole.yaml**

Purpose:Roles will be created for either for Services or Application specific roles –In case of NAPI – Roles like Patient, Application Admin ,Researcher and more as per defined in the Requirement by attaching the righ policy and giving a unique name to the role

To create a role the following format will be followed and will be called in the master template:

**Parameters:**

**Resource:**

**Role:**  Name of the resource for creation of Role

**Type: "AWS::IAM::Role" -** Type of IAM resource.

**Properties:** Properties for roles

**AssumeRolePolicyDocument:** Policy for the service which will be assuming the role

**Version: "2012-10-17" -** Policy details for the service to assume role

**Statement:**

**- Effect: "Allow"**

**Principal:**

**Service:**

**- ""**

**Action:**

**- "sts:AssumeRole"**

**Path: "/"**

**ManagedPolicyArns:**If policy arns are available list of policy arn can be provided

**- String**

**Path:**

**Policies:** List of policies that will be attched to the role can be refered from the

**- Policies**

**RoleName:** Name of the role to be created

* 1. **CreateGroup.yaml**

Purpose:This template will be used to create group and add users to this group

**Parameters:**

**Resources:**

**mygroup:** Name of the resource for creation of Group

**Type: AWS::IAM::Group** Type of IAM resource.

**Properties:**

**Path: "/AdminGroup/"**

**Policies:** Policies to be attached to the group

**addUserToGroup:** Name of the resource

**Type: AWS::IAM::UserToGroupAddition** Type of IAM resource

**Properties:**

**GroupName:** Refer to the groupname exsiting

**Users:** List of user to be added to group.These can be either existing users.

* 1. **Template format for policy**

Purpose:The policies can be written in JSON/YAML format , but they will be converted into JSON format by AWS CloudFormation.These policies can refered where the Fn:ImportValue is called.

**Parameters:**

**Resources:**

**Policy:**

**Type: AWS::IAM::ManagedPolicy**

**Properties:**

**Path:** path for IAM policy

**ManagedPolicyName:** Name of the Policy

**PolicyDocument:** Permissions that are rquired to be added.

**Version: '2012-10-17'**

**Statement:**

**- Effect: Allow**

**Action:**

**Resource:**

**Users:** Names of users to be added to the to the policy

**Roles:** Names of Roles to be added to the to the policy

**Groups:** Names of Groups to be added to the to the policy

**Outputs:** Output of the policy is exported and can be used to refer this policy by using Fn::ImportValue.

**ManagedPolicy:**

**Value: !Ref ManagedPolicy**

**Export:**

**Name: !Sub**

**-"$(AWS::StackName}-MangedPolicy"**