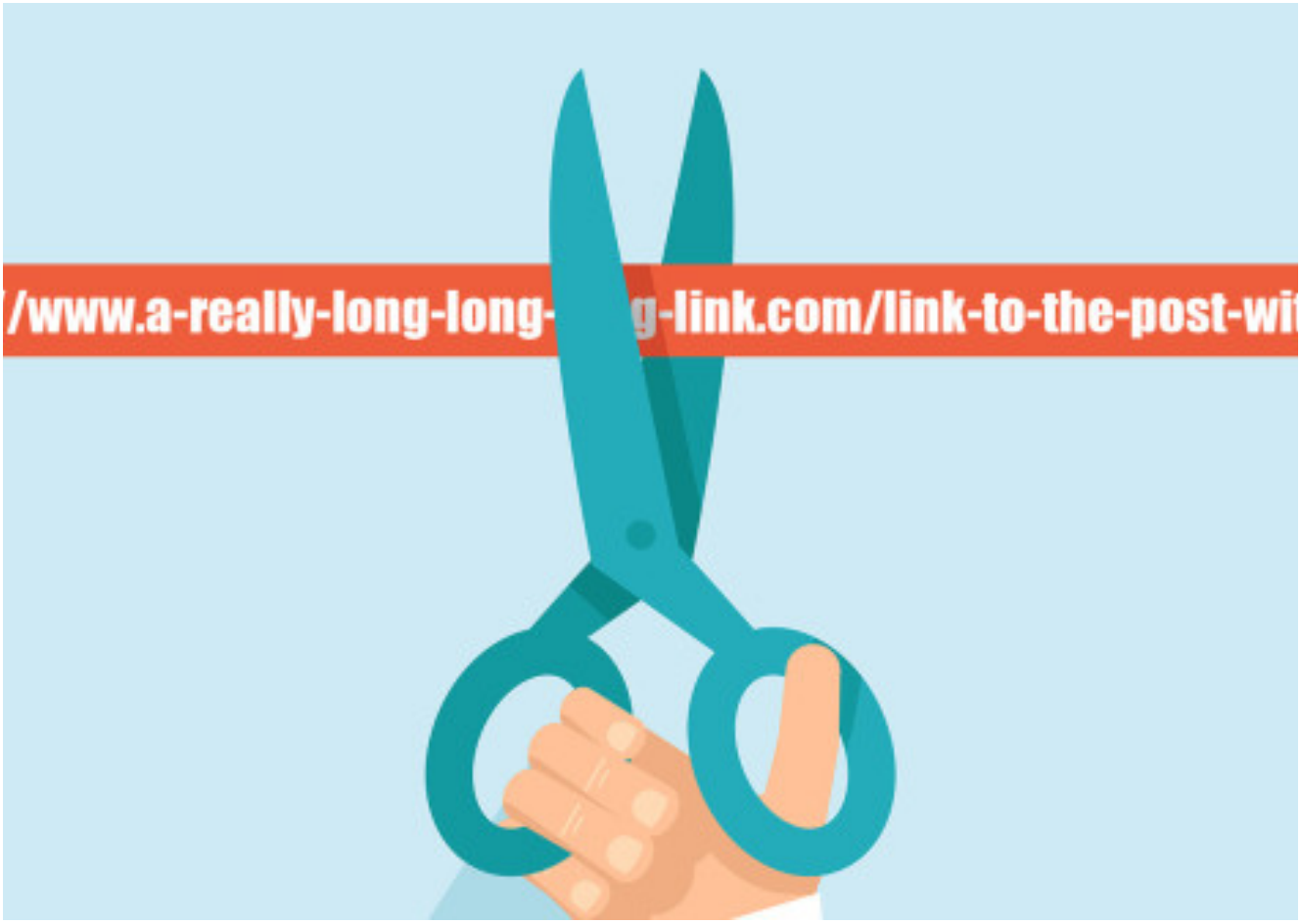
URL Shortener

A URL shortening service using AWS Services

Nick Jiang <congtao.jiang@outlook.com>
https://www.linkedin.com/in/congtao-jiang-27155a9/

October 27, 2019



Workable Solution

URLs of Web UIs and Restful API Endpoints

- End User Web UI

https://shortenit.s3-ap-southeast-1.amazonaws.com/web/index.html

- Custom Domain Admin Web UI

https://shortenit.s3-ap-southeast-1.amazonaws.com/web/index-domain.html

- Redirect Restful API Endpoint

https://h919hqbhel.execute-api.ap-southeast-1.amazonaws.com/njdev/NJRedirect-URL

- Shorten URL Restful API Endpoint

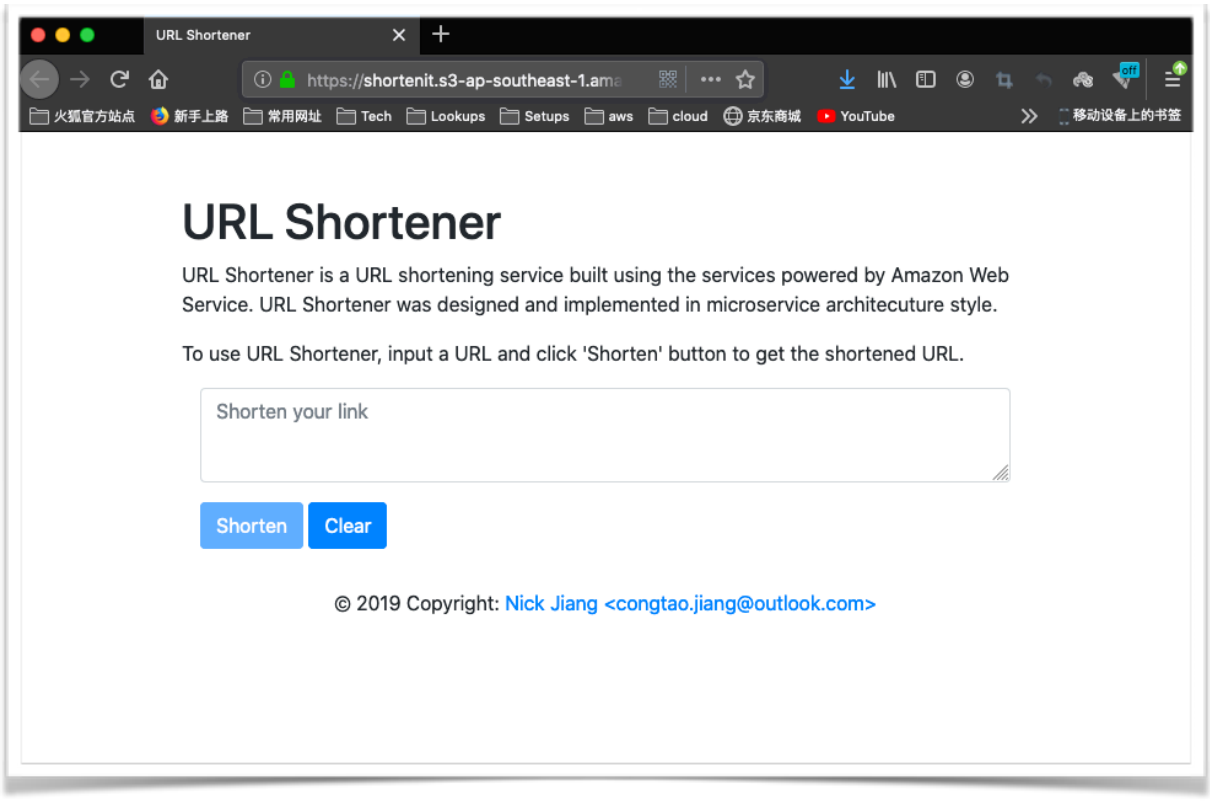
https://dycg2mna2f.execute-api.ap-southeast-1.amazonaws.com/njdev/NJShorten-Url

- Custom Domain Restful API Endpoint

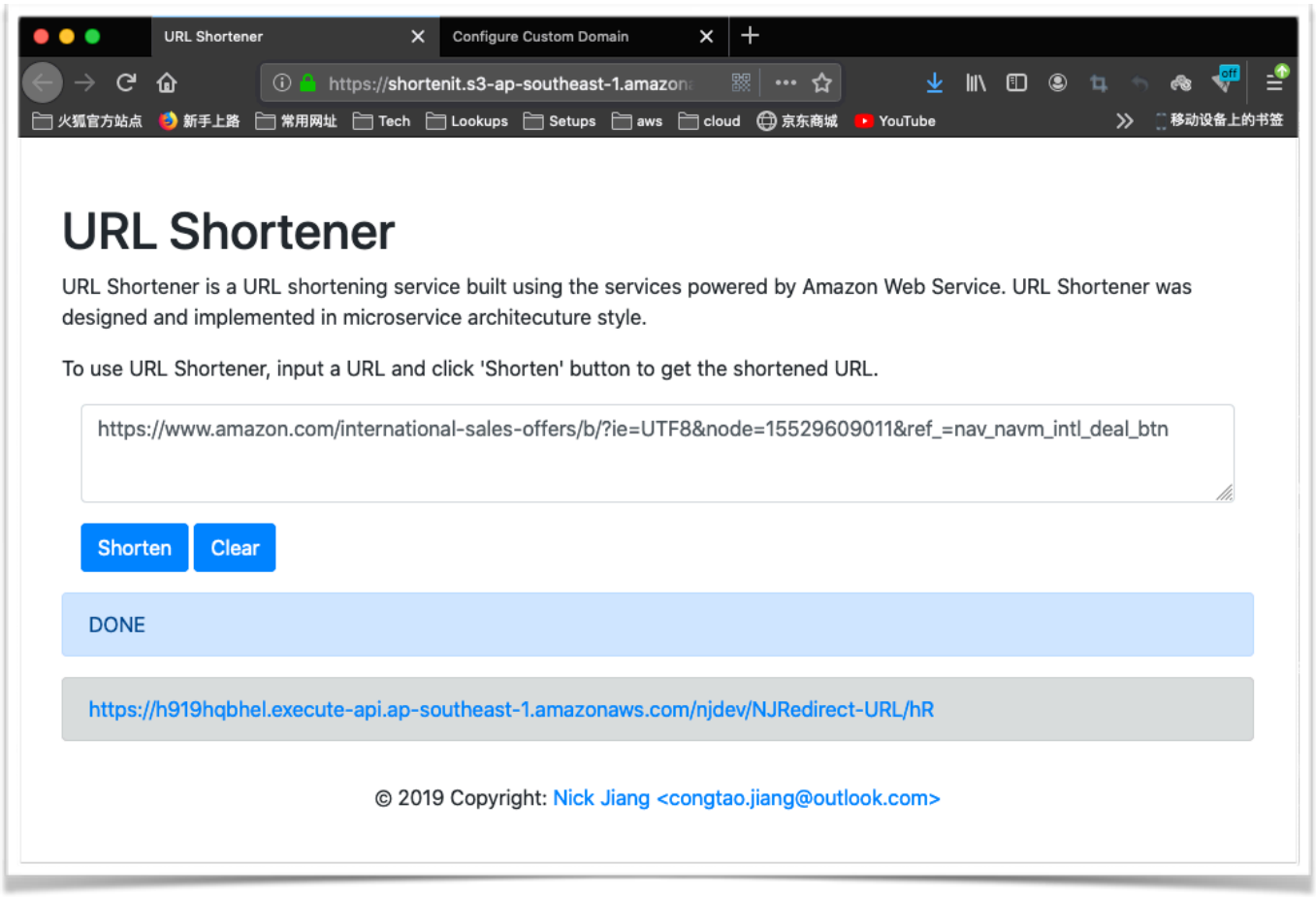
https://ndirsdrhbf.execute-api.ap-southeast-1.amazonaws.com/njdev/NJSetup-CustomFqdn

User Interface - End User Web UI

Open in browser the End User Web URL (https://shortenit.s3-ap-southeast-1.amazonaws.com/web/index.html), the URL shortener UI will show up like



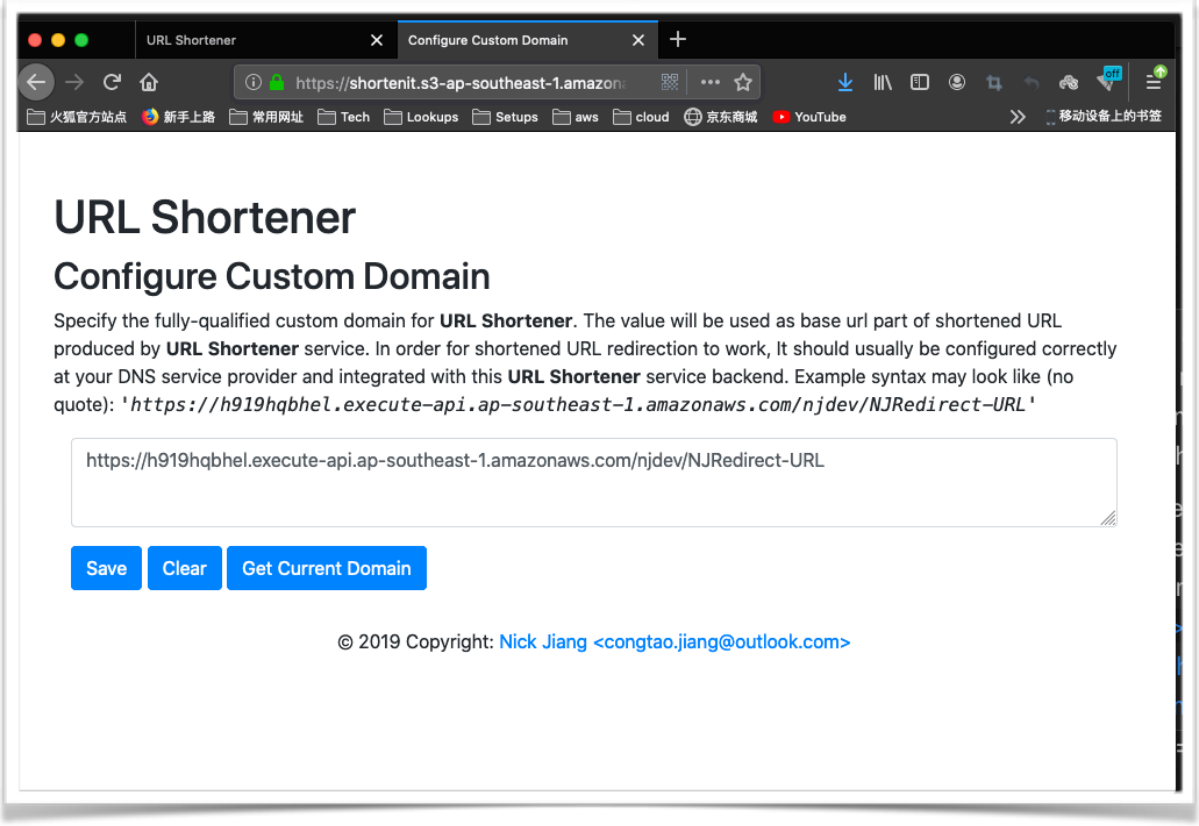
Input any long web URL in text box, and click **Shorten** button, the shortened version of the long web URL will be generated like



Click the generated link at the bottom of the page, the same page at the provided web URL will open in new browser tab.

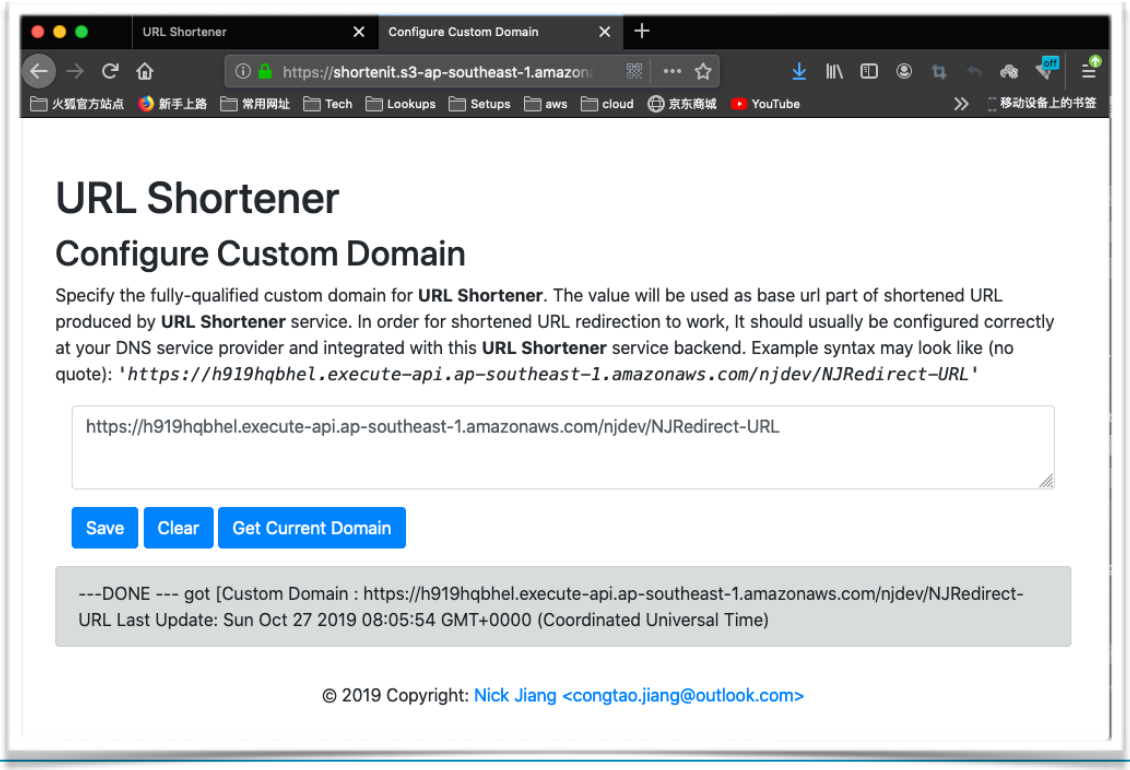
User Interface - Custom Domain Admin UI

Open in browser the Custom Domain Admin Web UI (https://shortenit.s3-ap-southeast-1.amazonaws.com/web/index-domain.html), the custom domain configuration UI for URL Shortener service will show up like the following. In the current implementation for functionality demonstration purpose, the auto-generated stage API URL from AWS API Gateway is used, in production, the shorter domain name needed to be acquired from DNS service provider and should be used for better experience.



Input the fully-qualified domain as depicted on the above image and click **Save** button to apply into the URL Shortener service.

Click **Get Current Domain** button to query the domain name that is in effect in URL Shortener service, e.g.



URL Shortener

Introduction

URL Shortener service provides the similar functionality as https://bitly.com to shorten a web URL that is long and hard to remember into a short URL that is easy to remember and convenient to share via IM, SMS and to be communicated over phone.

The URL Shortener is designed and implemented using micro-service architecture and technology stack from Amazon Web Services, it is highly scalable and performant, compared to traditional VM based solution.

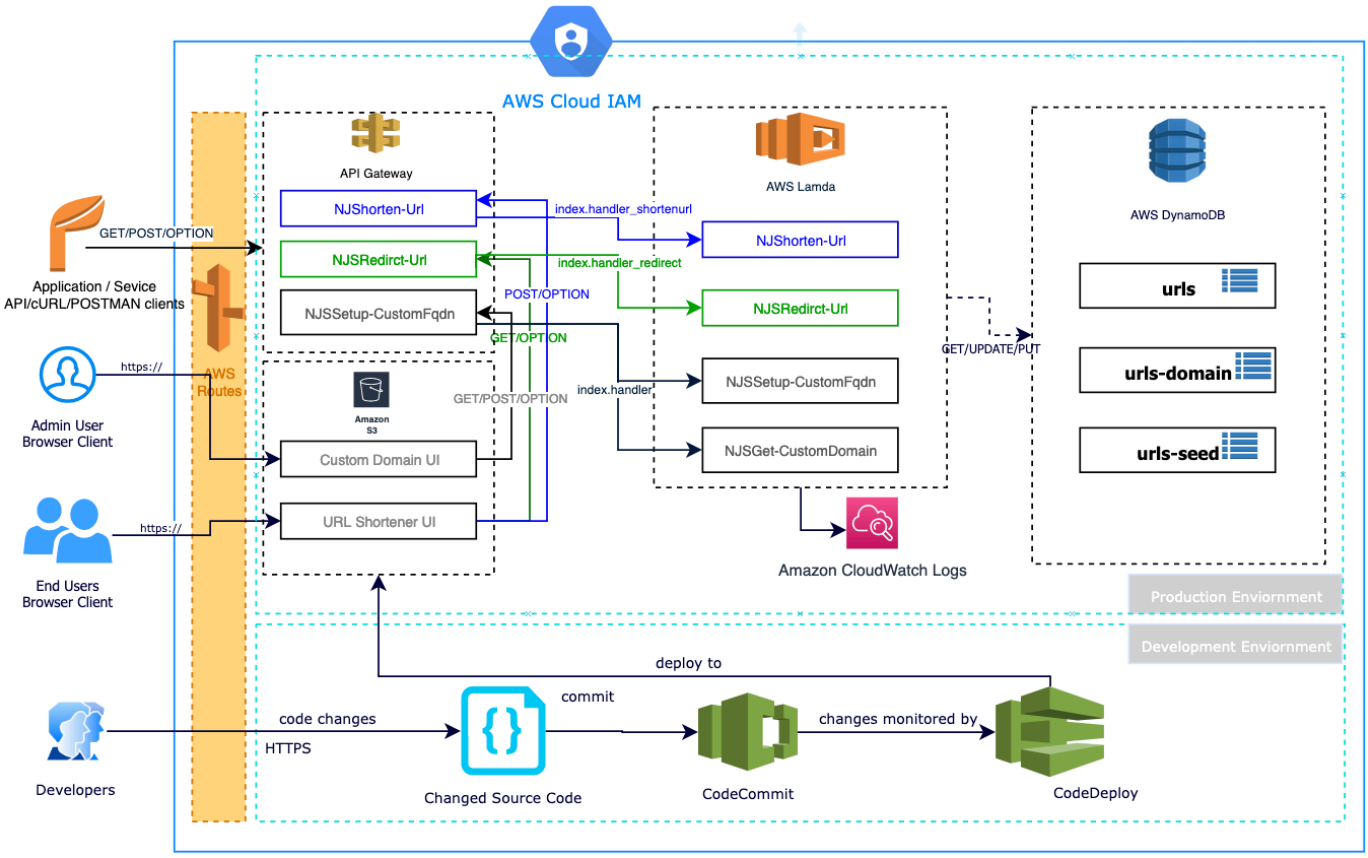
URL Shortener can be used in the following ways

- By web UI, pasted link and shorten it to get a short URL
- By Restful API, to be easily integrated into the third-party applications

URL Shortener has been implemented using the following AWS services

- **AWS S3** : S3 is used to host a static website / front-end UI of URL Shortener
- **AWS DynamoDB**: DynamoDB is used to provide persistent data storage for URL Shortener service
- **AWS Lambda**: the service backend logics are implemented as a collection of AWS Lambda functions
- **AWS IAM**: IAM service is used to manage access to DynamoDB storage and S3
- **AWS API Gateway**: API Gateway is used to expose, deploy and manage APIs of URL Shortener service, enforce access policy, like CORS
- **AWS Developer Tools**: CodeCommit, CodeDeploy

Service Design - Architecture



API Design

Refer to "Restful API Design" section for detail design of the following APIs:

- **NJShorten-Url**
  https://dycg2mna2f.execute-api.ap-southeast-1.amazonaws.com/njdev/NJShorten-Url

- **NJRedirect-URL-API**
  https://h919hqbhel.execute-api.ap-southeast-1.amazonaws.com/njdev/NJRedirect-URL

- **NJSetup-CustomFqdn-API**
  https://ndirsdrhbf.execute-api.ap-southeast-1.amazonaws.com/njdev/NJSetup-CustomFqdn

Approach

As illustrated in Service Design diagram, URL Shortener adopted micro-service architecture using AWS API Gateway, AWS Lambda , fast and high performant AWS DynamoDB No-SQL database. Compared to traditional VM based solution, this architecture does not require EC2

virtual machine instance provisioning, scheduling and maintenance efforts, there is no OS and runtime components patching for daily operations.

Business Outcome

Potential business outcomes or benefits

- Better URLs/links management for usability improvement for sharing
- Get flexibilities of tracking visits/traffics to pages, e.g. who is visiting you pages. More control to deliver what link to which channel
- Have more capabilities of doing links/urls customization to meet you branding requirements

Go Live

The following need to be considered before go to production

- API endpoint (e.g. set custom domain) need to be authenticated and authorized before servicing the requests
- Custom domain admin UI need to be internal access protected
- Access-Control-Allow-Origin of API methods' (OPTIONS, GET, POST) response header property need to be configured properly to prevent CORS attack
- Performance test need to be conducted to ensure the proper configurations of Write Capacity Unit and Read Capacity Unit are applied to cater for the expected performance goal
- Test any traffic rate limit/throttling from S3, API Gateway and DynamoDB for the selected service plan
- Full CICD pipeline (w/ acceptance tests) need to be established to manage code changes commit to deploy releases into production

Source Code and Access Credential

Attached with Email

Restful API Design

NJShorten-Url

https://dycg2mna2f.execute-api.ap-southeast-1.amazonaws.com/njdev/NJShorten-Url

```
---
swagger: "2.0"
info:
  version: "2019-10-26T14:16:03Z"
  title: "NJShorten-Url-API"
host: "dycg2mna2f.execute-api.ap-southeast-1.amazonaws.com"
```

```
basePath: "/njdev"
schemes:
- "https"
paths:
  /NJShorten-Url:
    post:
      produces:
      - "application/json"
      responses:
        200:
          description: "200 response"
          schema:
            $ref: "#/definitions/Empty"
          headers:
            Access-Control-Allow-Origin:
              type: "string"
      x-amazon-apigateway-integration:
        uri: "arn:aws:apigateway:ap-southeast-1:lambda:path/2015-03-31/functions/arn:aws:lambda:ap-
southeast-1:916483224774:function:NJShorten-Url/invocations"
        responses:
          default:
            statusCode: "200"
            responseParameters:
              method.response.header.Access-Control-Allow-Origin: "'*'"
        passthroughBehavior: "when_no_match"
        httpMethod: "POST"
        contentHandling: "CONVERT_TO_TEXT"
        type: "aws"
    options:
      consumes:
      - "application/json"
      produces:
      - "application/json"
      responses:
        200:
          description: "200 response"
          schema:
            $ref: "#/definitions/Empty"
          headers:
            Access-Control-Allow-Origin:
              type: "string"
            Access-Control-Allow-Methods:
              type: "string"
            Access-Control-Allow-Headers:
              type: "string"
      x-amazon-apigateway-integration:
        responses:
          default:
            statusCode: "200"
            responseParameters:
              method.response.header.Access-Control-Allow-Methods:
"'DELETE,GET,HEAD,OPTIONS,PATCH,POST,PUT'"
              method.response.header.Access-Control-Allow-Headers: "'Content-Type,Authorization,X-Amz-
Date,X-Api-Key,X-Amz-Security-Token'"
              method.response.header.Access-Control-Allow-Origin: "'*'"
        requestTemplates:
          application/json: "{\"statusCode\": 200}"
        passthroughBehavior: "when_no_match"
        type: "mock"
definitions:
  Empty:
    type: "object"
    title: "Empty Schema”
```

NJRedirect-URL-API

https://h919hqbhel.execute-api.ap-southeast-1.amazonaws.com/njdev/NJRedirect-URL

```
---
swagger: "2.0"
info:
  version: "2019-10-24T13:35:56Z"
  title: "NJRedirect-URL-API"
host: "h919hqbhel.execute-api.ap-southeast-1.amazonaws.com"
basePath: "/njdev"
schemes:
- "https"
paths:
  /NJRedirect-URL:
    options:
      consumes:
      - "application/json"
      produces:
      - "application/json"
      responses:
        200:
          description: "200 response"
          schema:
            $ref: "#/definitions/Empty"
          headers:
            Access-Control-Allow-Origin:
              type: "string"
            Access-Control-Allow-Methods:
              type: "string"
            Access-Control-Allow-Headers:
              type: "string"
      x-amazon-apigateway-integration:
        responses:
          default:
            statusCode: "200"
            responseParameters:
              method.response.header.Access-Control-Allow-Methods: "'DELETE,GET,HEAD,OPTIONS,PATCH,POST,PUT'"
              method.response.header.Access-Control-Allow-Headers: "'Content-Type,Authorization,X-Amz-Date,X-Api-Key,X-Amz-Security-
Token'"
              method.response.header.Access-Control-Allow-Origin: "'*'"
        requestTemplates:
          application/json: "{\"statusCode\": 200}"
        passthroughBehavior: "when_no_match"
        type: "mock"
  /NJRedirect-URL/{urlkey}:
    get:
      consumes:
      - "application/json"
      produces:
      - "application/json"
      parameters:
      - name: "urlkey"
        in: "path"
        required: true
        type: "string"
      responses:
        301:
          description: "301 response"
          schema:
```

```
          $ref: "#/definitions/Empty"
        headers:
          Location:
            type: "string"
    x-amazon-apigateway-integration:
      uri: "arn:aws:apigateway:ap-southeast-1:lambda:path/2015-03-31/functions/arn:aws:lambda:ap-
southeast-1:916483224774:function:NJRedirect-URL/invocations"
      responses:
        default:
          statusCode: "301"
          responseParameters:
            method.response.header.Location: "integration.response.body.location"
      requestTemplates:
        application/json: "{\n   \"urlkey\": \"$input.params('urlkey')\"\n}"
      passthroughBehavior: "when_no_templates"
      httpMethod: "POST"
      contentHandling: "CONVERT_TO_TEXT"
      type: "aws"
  options:
    consumes:
    - "application/json"
    produces:
    - "application/json"
    responses:
      200:
        description: "200 response"
        schema:
          $ref: "#/definitions/Empty"
        headers:
          Access-Control-Allow-Origin:
            type: "string"
          Access-Control-Allow-Methods:
            type: "string"
          Access-Control-Allow-Headers:
            type: "string"
    x-amazon-apigateway-integration:
      responses:
        default:
          statusCode: "200"
          responseParameters:
            method.response.header.Access-Control-Allow-Methods: "'DELETE,GET,HEAD,OPTIONS,PATCH,POST,PUT'"
            method.response.header.Access-Control-Allow-Headers: "'Content-Type,Authorization,X-Amz-Date,X-Api-Key,X-Amz-Security-
Token'"
            method.response.header.Access-Control-Allow-Origin: "'*'"
      requestTemplates:
        application/json: "{\"statusCode\": 200}"
      passthroughBehavior: "when_no_match"
      type: "mock"
definitions:
  Empty:
    type: "object"
    title: "Empty Schema"
```

NJSetup-CustomFqdn-API

https://ndirsdrhbf.execute-api.ap-southeast-1.amazonaws.com/njdev/NJSetup-CustomFqdn

```
---
swagger: "2.0"
info:
  version: "2019-10-27T07:43:21Z"
  title: "NJSetup-CustomFqdn-API"
host: "ndirsdrhbf.execute-api.ap-southeast-1.amazonaws.com"
basePath: "/njdev"
schemes:
- "https"
paths:
  /NJSetup-CustomFqdn:
    get:
      produces:
      - "application/json"
      responses:
        200:
          description: "200 response"
          schema:
            $ref: "#/definitions/Empty"
          headers:
            Access-Control-Allow-Origin:
              type: "string"
      x-amazon-apigateway-integration:
        uri: "arn:aws:apigateway:ap-southeast-1:lambda:path/2015-03-31/functions/arn:aws:lambda:ap-
southeast-1:916483224774:function:NJGet-CustomDomain/invocations"
        responses:
          default:
            statusCode: "200"
            responseParameters:
              method.response.header.Access-Control-Allow-Origin: "'*'"
        passthroughBehavior: "when_no_match"
        httpMethod: "POST"
        contentHandling: "CONVERT_TO_TEXT"
        type: "aws"
    post:
      produces:
      - "application/json"
      responses:
        200:
          description: "200 response"
          schema:
            $ref: "#/definitions/Empty"
          headers:
            Access-Control-Allow-Origin:
              type: "string"
      x-amazon-apigateway-integration:
        uri: "arn:aws:apigateway:ap-southeast-1:lambda:path/2015-03-31/functions/arn:aws:lambda:ap-
southeast-1:916483224774:function:NJSetup-CustomFqdn/invocations"
        responses:
          default:
            statusCode: "200"
            responseParameters:
              method.response.header.Access-Control-Allow-Origin: "'*'"
        passthroughBehavior: "when_no_match"
        httpMethod: "POST"
```

```
      contentHandling: "CONVERT_TO_TEXT"
      type: "aws"
    options:
      consumes:
      - "application/json"
      produces:
      - "application/json"
      responses:
        200:
          description: "200 response"
          schema:
            $ref: "#/definitions/Empty"
          headers:
            Access-Control-Allow-Origin:
              type: "string"
            Access-Control-Allow-Methods:
              type: "string"
            Access-Control-Allow-Headers:
              type: "string"
      x-amazon-apigateway-integration:
        responses:
          default:
            statusCode: "200"
            responseParameters:
              method.response.header.Access-Control-Allow-Methods: "'DELETE,GET,HEAD,OPTIONS,PATCH,POST,PUT'"
              method.response.header.Access-Control-Allow-Headers: "'Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-
Token'"
              method.response.header.Access-Control-Allow-Origin: "'*'"
        requestTemplates:
          application/json: "{\"statusCode\": 200}"
        passthroughBehavior: "when_no_match"
        type: "mock"
definitions:
  Empty:
    type: "object"
    title: "Empty Schema"
```