Service Application Documentation

Component Identifier Service

A REST Server for managing the generation and assignment of Terminology Component Identifiers. Supports SNOMED CT Identifiers and other identifier Schemes. Pre-bundled implementations for optional generation of legacy identifiers in SNOMED CT (SNOMEDIDs and CTV3IDs)

Installation

This application requires a MySQL database running on localhost. A new Database with name "idservice" needs to be available before running the application.

Clone this project with:

```
$ clone https://github.com/termMed/component-identifier-service.git .
```

Cd to the application location and execute the schema creation scripts:

```
cd /opt/component-identifier-service
npm install
node config/createSchema.js dbuser=your_db_user dbpass=your_db_pass
node config/initializeIdSchemes.js dbuser=your_db_user dbpass=your_db_pass // Optional, only
if Additional ids will be used
```

Start the Service:

```
node app.js dbuser=your_db_user dbpass=your_db_pass &
```

Now the web service and the admin tool will be available:

- REST Api: http://localhost:3000/api
- Swagger Docs: http://localhost:3000/docs
- Admin tool: http://localhost:3000/admin

Component Identifiers

The application supports to basic types of component identifiers, SCTIDS and generic Identifier Schemes. SCTIDs are assigned based on namespaces, Namcespaces can be created and managed with the Api. Identifier Schemes represent additional identifiers, like SNOMEDIDs (legacy ids from previous SNOMED Versions) and CTV3IDs (legacy Ids from the UK NHS Read Codes). Other identifiers can be added using code extension points without the need for alterations in the data structure or the Api.

Current Deployed API

https://cis.ihtsdotools.org/docs/

The Identifier record

The application stores related metadata for each idenfier generated or registered in the satabase, the model for a SCTID Identifer Record is:

```
"SCTIDRecord" : {
          "properties": {
              "sctid": {
                  "type": "string"
              },
              "sequence": {
                  "type": "integer"
              },
              "namespace": {
                  "type": "integer"
              },
              "partitionId": {
                  "type": "string"
              },
              "checkDigit": {
                  "type": "integer"
              },
              "systemId": {
                  "type": "string"
              },
              "status": {
                  "type": "string"
              },
              "author": {
                  "type": "string"
              },
              "software": {
                  "type": "string"
              },
              "expirationDate": {
                  "type": "string"
              },
              "comment": {
                  "type": "string"
              },
              "additionalIds": {
                "type": "array",
                "items": {
                  "$ref": "#/definitions/SchemeIdRecord"
                }
              }
          }
      }
```

A similar model is used for Scheme Identifiers.

The identifiers life cyle

Identifiers are generated or regsitered using the Api, and they will change status to represent publications, reservations and other events that may make the identifer to be available again or to be defitively linked with a terminology component.

The set of valid statuses and actions are represented in the State Machine diagram included in this git project as a pdf file.

Authentication

The Application is integrated with Atlassian Crowd for authentication. Admin permissions are assgined using Crowd groups, and resources permissions.

Example REST Api calls

Integrating this service into an application will require to perform http calls to the Api, for example:

Retrieving the available namespaces for a user

GET http://localhost:3000/api/users/USERNAME/namespaces/?token=hdaskjdhakjdgy7

Generating a new SCTID for a concept in namespace 1000179

POST http://localhost:3000/api/sct/generate?token=hdaskjdhakjdgy7

(Ids metadata, like namespace and parititon is passed in the body)

See the full API Documentation in the Swagger Docs.