Know your network – general overview

Version 1.0

Table of Contents

[Intro 3](#_Toc463907438)

[List and function of modules 4](#_Toc463907439)

[Description and web application diagrams 5](#_Toc463907440)

[Sequence diagram 5](#_Toc463907441)

[Class diagram and Javadoc 5](#_Toc463907442)

[List and function of used frameworks 6](#_Toc463907443)

[Logging 7](#_Toc463907444)

[Improvements 8](#_Toc463907445)

[General improvements 8](#_Toc463907446)

[getIPAdminInfo method improvement 8](#_Toc463907447)

# Intro

This document is general overview of Know your network java web application. In the next few chapters you will find brief description of application, modules, diagrams, etc.

In the last chapter there is a list of improvements that can be made.

Basic architechture of application: Frontend (Static HTML) and Backend (Java REST service) part.

As a building tool for creating *war* artefact is selected Maven. It is also used as for dependency management. Building is enabled throw Maven Profiles (dev, test, prod). “Test” Profile uses Stub files with data within an application as an resource.

# List and function of modules

Name of main maven module is **KynKite**.

Child maven modules:

**KynKiteWebApp** –Web application module with exposed REST service

**KynKiteDocumentation** – Documentation module (Diagrams, Javadoc , etc.)

# Description and web application diagrams

## Sequence diagram

Sequence diagram is created using Plant UML and is located in folder *diagrams/sequence.*

## Class diagram and Javadoc

Javadoc and class diagram is located in documentation module in folder Javadoc. By opening *javadoc/index.html* you will see from general information about application to specialized reports:

* Dependency Convergence
* Dependency Information
* Dependency Management
* Project Modules
* Plugin Management
* JavaDocs
* CPD
* PMD

# List and function of used frameworks

Elastic search API (version 2.3.3) – **Important!** It has to be the same version as Elastic search server .

Junit (version 4.12) – Used for creating test

Spring framework (version 4.2.1.RELEASE) – Used for creating REST service and application layer structure

Jackson framework (version 2.4.1) – Used for created JSON java objects and JSON marchaling and unmarchaling.

# Logging

Logback framework is used for logging facilities.

About this framework: <http://logback.qos.ch/>

Configuration for this framework is in the logback.xml file. For the development environment, loback.xml is in the src/main/resources folder and there are two defined appenders. ConsoleAppender id defined for IDE console and RollingFileAppender is defined for log file located on file system.

Configuration for other environments is in: src/main/resources/logback/{environment}/logback.xml. Configuration for environment must be updated before deploying application to that environment.

# Improvements

## General improvements

It should consider a need for a transaction management.

Posibility of using JSON server for creating JSON mock data.

<http://www.betterpixels.co.uk/projects/2015/05/09/mock-up-your-rest-api-with-json-server/>

<https://coligo.io/create-mock-rest-api-with-json-server/>

There is a possibility of dependency conflicts that should be resolved.

## getIPAdminInfo method improvement

Instructions for creating this method was to reverse engineer php call to two curl commands:

1. curl -k -E /var/kyn/ssl/kyn.srv.allianz\_base64.cer --key /var/kyn/ssl/kyn.srv.allianz.key 'https://ipadmin.ip.allianz/ipadmin/call.html?do=attribute\_by\_ipv4&ip=192.168.5.2&nm=32&realm=1
2. curl -k -E /var/kyn/ssl/kyn.srv.allianz\_base64.cer --key /var/kyn/ssl/kyn.srv.allianz.key 'https://ipadmin.ip.allianz/ipadmin/call.html?do=attribute\_by\_ipv4&ip=192.168.5.2&nm=32&realm=1&type=hostname

In this case certificate, key and url is hard-coded, and curl must be defined on server path.

Since java is able to do http get via HttpClient, I've created second method getIPAdminInfoHttp.

1. URLs

For now urls are defined in properties file.

It would be better to put it as a url jndi resoruce on server.

By using jndi resource there is no need to change and build application if urls are changed.

Just change resource and restart application.

1. SSL

SSL should be configured on server.

By using server side configuration there is no need to change and build application if certificates are changed/expired.

If it is decided to manage ssl from java, certificates should be put in java keystore and additional code should be written in FileSystemDaoRepository.executeHttpCall method.

FileSystemDaoRepository should be renamed. (primary usage for that class was to execute curl call).