Shanoir NG

Software Design Description

Table des matières

[1 Project context 2](#_Toc501725578)

[1.1 Project presentation 2](#_Toc501725579)

[1.2 Project goals 2](#_Toc501725580)

[1.3 Main functionalities 2](#_Toc501725581)

[1.4 Application users 3](#_Toc501725582)

[2 Functional architecture 4](#_Toc501725583)

[2.1 Interaction with Shanoir 4](#_Toc501725584)

[2.2 Flows description 4](#_Toc501725585)

[2.3 Functional architecture schema of Shanoir NG 5](#_Toc501725586)

[2.3.1 User microservice 5](#_Toc501725587)

[2.3.2 Keycloak server 5](#_Toc501725588)

[2.3.3 Keycloak database 5](#_Toc501725589)

[2.3.4 Study microservice 5](#_Toc501725590)

[2.3.5 Study card microservice 6](#_Toc501725591)

[2.3.6 Dataset microservice 6](#_Toc501725592)

[2.3.7 Dataset file microservice 6](#_Toc501725593)

[2.3.8 Import microservice 6](#_Toc501725594)

[2.3.9 NifTI conversion microservice 6](#_Toc501725595)

[2.3.10 Score microservice 6](#_Toc501725596)

[2.3.11 DTM 6](#_Toc501725597)

[2.3.12 Small animal microservice 6](#_Toc501725598)

[3 Software architecture 6](#_Toc501725599)

[3.1 Software solution 6](#_Toc501725600)

[3.1.1 Frameworks and libraries 6](#_Toc501725601)

[3.1.2 Docker 7](#_Toc501725602)

[3.2 Components 7](#_Toc501725603)

[3.2.1 Front end 7](#_Toc501725604)

[3.2.2 Keycloak 7](#_Toc501725605)

[3.2.3 User management microservice 7](#_Toc501725606)

[3.2.4 Study microservice 7](#_Toc501725607)

[3.2.5 Study card microservice 8](#_Toc501725608)

[3.2.6 Dataset microservice 8](#_Toc501725609)

[3.2.7 Dataset file microservice 8](#_Toc501725610)

[3.2.8 Import microservice 8](#_Toc501725611)

[3.2.9 NifTI conversion microservice 8](#_Toc501725612)

[3.2.10 Score microservice 8](#_Toc501725613)

[3.2.11 DTM microservice 8](#_Toc501725614)

[3.2.12 Small animal microservice 8](#_Toc501725615)

[4 Security 8](#_Toc501725616)

[4.1 Authentication 8](#_Toc501725617)

[5 Technical architecture 8](#_Toc501725618)

[5.1 Software components 8](#_Toc501725619)

[5.2 Interactions between components and external systems 9](#_Toc501725620)

[5.2.1 Microservice with MySQL database 10](#_Toc501725621)

[5.3 Deployment 10](#_Toc501725622)

[6 Development environment 10](#_Toc501725623)

[6.1 Code structure 10](#_Toc501725624)

# Project context

## Project presentation

Shanoir (SHAring NeurOImaging Resources) is an open source neuroinformatics platform designed for sharing, archiving, searching and visualizing neuroimaging data. It provides a user-friendly secure web access and offers an intuitive workflow to facilitate the collecting and retrieving of neuroimaging data from multiple sources and a wizard to make the completion of metadata easy. Shanoir comes along many features such as anonymization of data, support for multi-centers clinical studies on subjects or groups of subjects.

Shanoir allows searchers, clinicians and engineers to carry out local or remote (cloud) research projects.

## Project goals

Shanoir NG (Next Generation) is the new version of Shanoir. It is based on a multi microservices architecture.

## Main functionalities

Shanoir NG main functionalities are:

* User management
* Study management
* Study card management
* Dataset management
* Dataset file management
* Import
* NifTI conversion
* Score management
* Data Transfer Module (DTM)
* Small animal

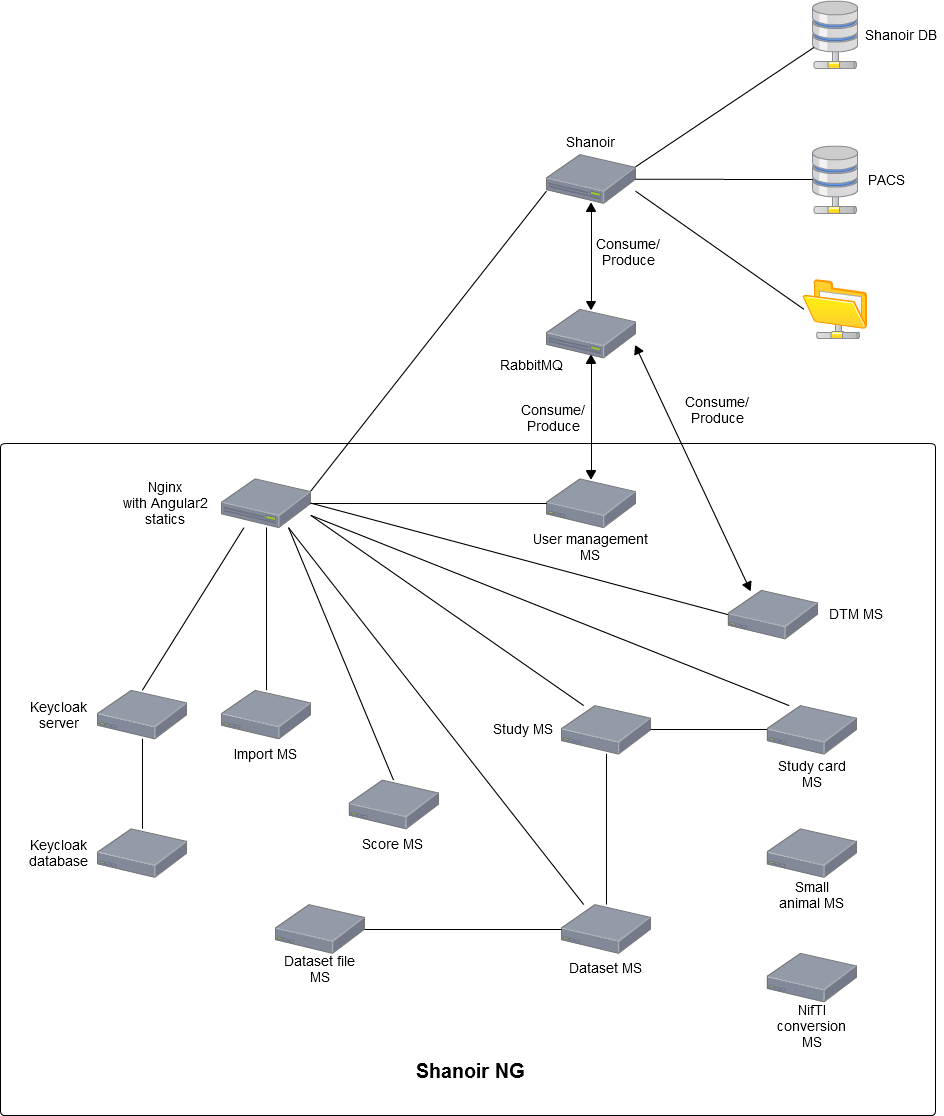
## Application users

Target population is:

* Shanoir administrators

# Functional architecture

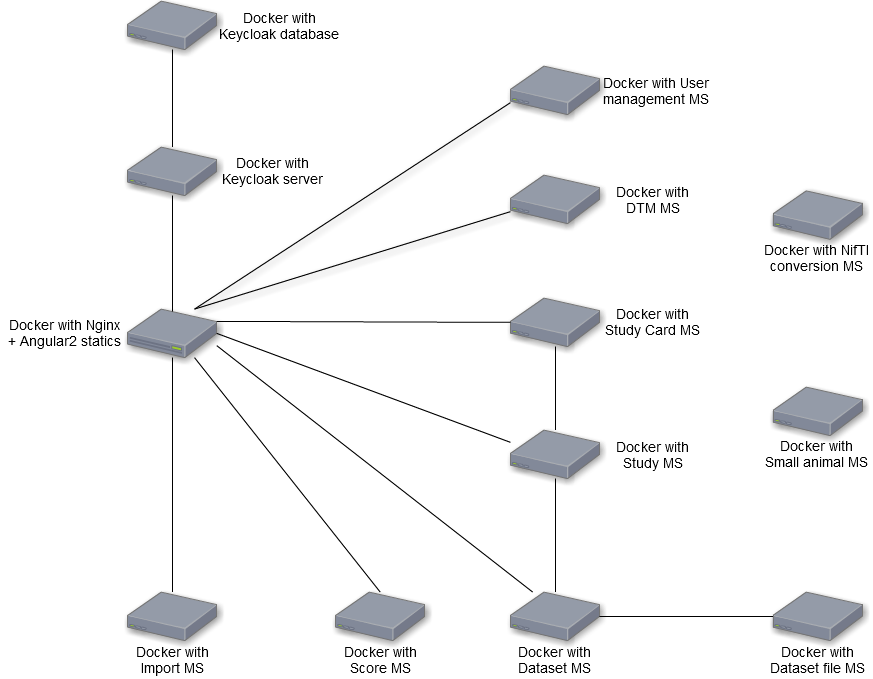
## Interaction with Shanoir



## Flows description

Shanoir NG communicates with Shanoir through RabbitMQ server.

## Functional architecture schema of Shanoir NG



### User microservice

This microservice goal is to manage all application users. It allows to create, update or remove users and manages rights and access.

### Keycloak server

Keycloak is an open source identity and access management solution. It is used to manage login and users session.

### Keycloak database

Database for Keycloak server data.

### Study microservice

This microservice is responsible for study management. It is used to manage:

* Centers
* Acquisition equipments
* Coils
* Investigators
* Studies
* Subjects and groups of subjects

Main functionalities are:

* Managing centers (list, create, delete, update)
* Managing investigators (list, create, delete, update)
* Managing acquisition equipment (list, create, delete, update)
* Managing coils (list, create, delete, update)List, create, view, update and delete studies
* Managing studies (list, create, delete, update)List, create, view, update and delete studies
* Manage requests to join study
* Download report
* Export study

### Study card microservice

This microservice is used to manage:

* Study cards

### Dataset microservice

* Examination
* Dataset acquisition
* Dataset

### Dataset file microservice

### Import microservice

### NifTI conversion microservice

### Score microservice

### DTM

Data transfer module (DTM) microservice goal is to transfer data from VIP platform to Shanoir.

VIP process pipelines based on Shanoir dataset and calculates scores. DTM gets results and scores of each pipeline, transfers them to Shanoir abd links them to entry dataset.

### Small animal microservice

# Software architecture

Shanoir NG is based on microservices. One microservice corresponds to backend of one functionality.

Application front end is apart and pages for all functionalities are grouped on Nginx server.

User sessions and passwords are managed by a Keycloak server.

Each microservices, Keycloak server and Nginx are embedded in a Docker.

## Software solution

### Frameworks and libraries

Shanoir NG uses following frameworks and libraries:

|  |  |  |
| --- | --- | --- |
| Framework / Library | Version | Description |
| Spring Boot | 1.4.1 | Spring Boot is Spring's convention-over-configuration solution for creating stand-alone, production-grade Spring-based Applications that you can "just run". It takes an opinionated view of the Spring platform and third-party libraries so you can get started with minimum fuss. Most Spring Boot applications need very little Spring configuration. |
| Angular 2 | 2.4.9 | Development platform for building mobile and desktop web applications. |

### Docker

Docker is an open-source project that automates the deployment of Linux applications inside software containers.

Docker containers wrap up a piece of software in a complete filesystem that contains everything it needs to run: code, runtime, system tools, system libraries – anything you can install on a server. This guarantees that it will always run the same, regardless of the environment it is running in.

## Components

### Front end

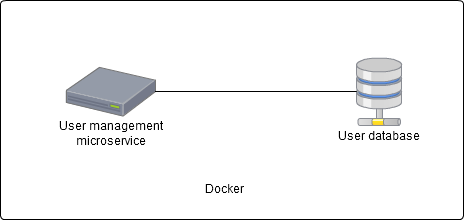
Front end is developed with Angular 2 and is placed on Nginx server.

### Keycloak

Keycloak is a third-party software, used to secure the application.

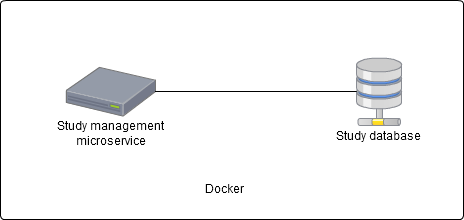
### User management microservice

User management microservice is based on Spring Boot framework. Data are stored in a MySQL database.



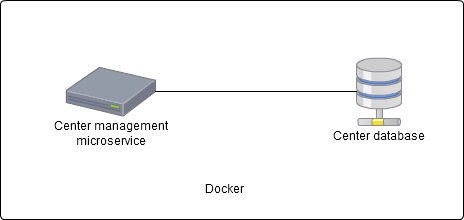
### Study microservice

Study management microservice is based on Spring Boot framework. Data are stored in a MySQL database.



### Study card microservice

Study card management microservice is based on Spring Boot framework. Data are stored in a MySQL database.



### Dataset microservice

### Dataset file microservice

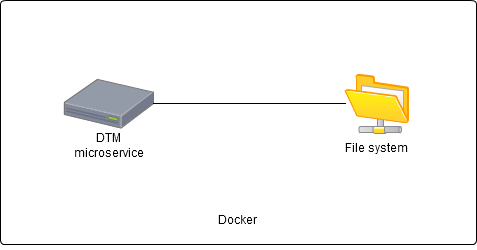
### Import microservice

### NifTI conversion microservice

### Score microservice

### DTM microservice

DTM microservice is based on Spring Boot framework. It uses documents in file system to process data.



### Small animal microservice

# Security

## Authentication

To manage user authentication, session management and password policy and to ensure security we use a third-party software: Keycloak.

Keycloak (<http://www.keycloak.org/>) is an open source identity and access management solution. It is used to manage login and users sessions with tokens.

A token is:

* Generated if login success
* Used as a proof of session
* Crypted by Keycloak server with private key
* Uncrypted by microservices with public key

It contains:

* Session id
* User id
* User roles

# Technical architecture

## Software components

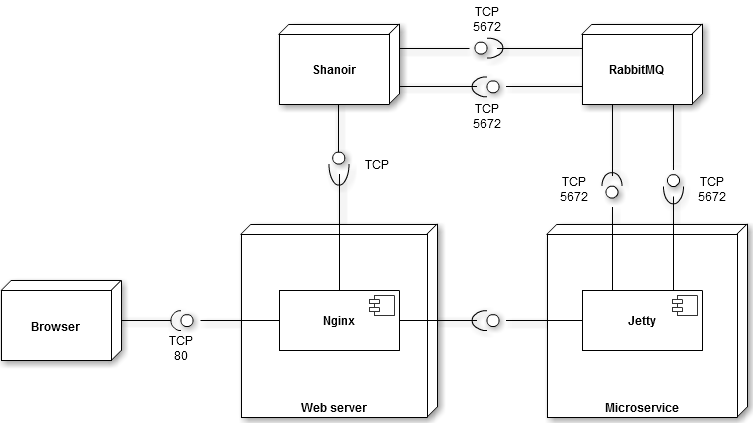
Shanoir NG is based on five software components:

* Docker
* Nginx server
* Jetty server
* MySQL database



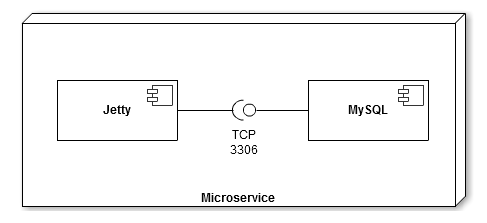
|  |  |  |
| --- | --- | --- |
| **Component** | **Version** | **Role** |
| Docker |  | Open-source project that automates the deployment of Linux applications inside software containers. |
| Nginx |  | Nginx is configured as reverse proxy, allowing to isolate application server from internet.  Nginx is also used:   * For TLS certification |
| Jetty |  | Application server hosting microservices |
| MySQL |  | Application database |

## Interactions between components and external systems



|  |  |  |
| --- | --- | --- |
| **Components** | **Exposed interfaces** | **Used interfaces** |
| Nginx | Nginx exposes an http interface to internet, which is used by clients | IP address of microservice and port of microservice Jetty server  Shanoir inteface |
| Jetty | IP address xxx  Port yyy | RabbitMQ interface |
| Shanoir | Port of Tomcat server |  |
| RabbitMQ | RabbitMQ port |  |

### Microservice with MySQL database



|  |  |  |
| --- | --- | --- |
| **Components** | **Exposed interfaces** | **Used interfaces** |
| Jetty |  | MySQL database connector |
| MySQL | MySQL database connector |  |

## Deployment

# Development environment

## Code structure

Code is divided in many projects. Each microservice is defined in one project. All code projects are managed with Maven.

Apache Maven (https://maven.apache.org/) is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

