eClinic

Technical solution description

Bogdan Sukonnov

Saint-Petersburg 2019

Content.

[Project goals 3](#_Toc4127)

[Application description 3](#_Toc4128)

[Used Technologies 3](#_Toc4129)

[Database model 5](#_Toc4130)

[System infrastructure 5](#_Toc4131)

[System architecture 6](#_Toc4132)

[Additional features 13](#_Toc4133)

[Code quality 13](#_Toc4134)

Tests....................................................................................................................... 13

Sonar report: .......................................................................................................... 14

Build and deploy................................................................................................... 14

Future improvement. ............................................................................................ 15

# PROJECT GOALS

Application development for the medical rehabilitation clinic. The application should have cohesive data model, user friendly interface, separate access to different system’s part, reliable system.

# APPLICATION DESCRIPTION

Web application has three types of user: doctors and nurses. Admin has access to each part of the application and can do everything that the other users do.

Doctors can add and discharge patients, create, edit and cancel prescriptions. Also they have access to any information about patients, prescriptions and events.

Nurses has limited access to the service. After authentication a nurse can only see events, complete and cancel them.

There is an authentication mechanism in the system that control access to the service. Each user in application has access level that defines information for displaying and privileges.

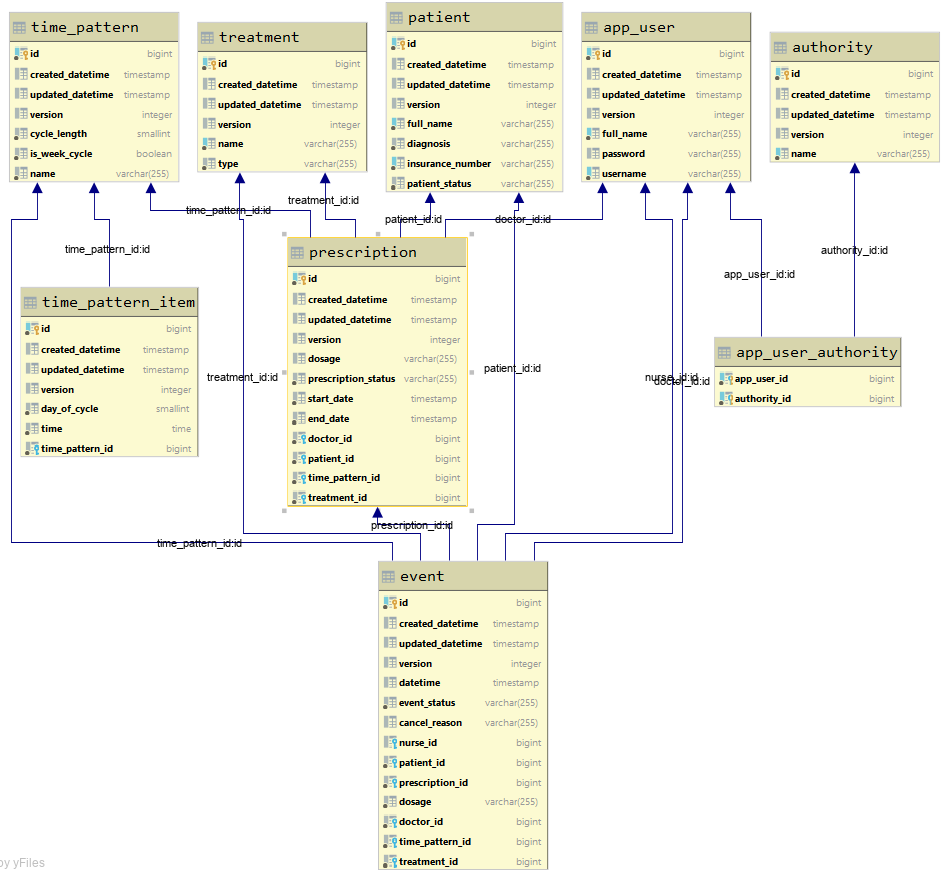
Data of users stores in a reliable database.

# USED TECHNOLOGIES

• Instruments:

1. IDE - IntelliJ IDEA
2. Maven
3. Docker Technologies:
4. Java 8
5. Spring
6. JSP
7. JPA
8. Hibernate
9. JavaScript
10. jQuery
11. JSF
12. REST
13. Spring security
14. ActiveMQ
15. Bootstrap4
16. PostgreSQL
17. EJB
18. Junit
19. Mockito
20. Jenkins
21. SonarQube
22. Tomcat
23. WildFly
24. Log4j2
25. Flyway

# DATABASE MODEL



# SYSTEM INFRASTRUCTURE

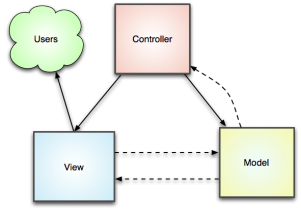
* Front-end (for doctors and nurses):
  1. Web-page structure - HTML, JSP
  2. Page-design – Bootstrap4
  3. Dynamic contents: JavaScript, Jquery
* Back-end (server based level):
  1. Application server - Tomcat
  2. Database - PostgreSQL
  3. Server logic - Spring Framework
* Client board application:
  1. Web-pages - JSF
  2. JMS - ActiveMQ
  3. Application server - WildFly

4) Server logic - EJB

5) WS - REST

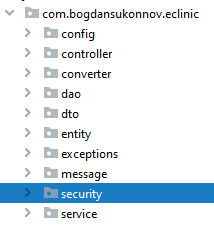
# SYSTEM ARCHITECTURE

Architecture of server-based part presented by MVC - design pattern.

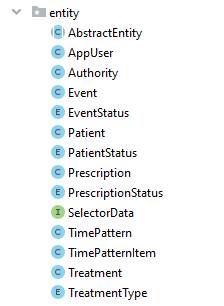


Class structure

According MVC-pattern application has next structure:



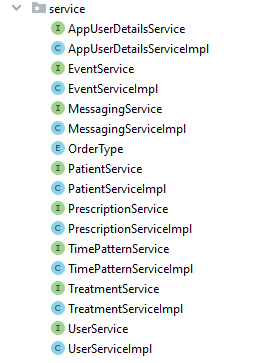
Model level:



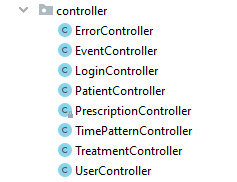
Model-service level:



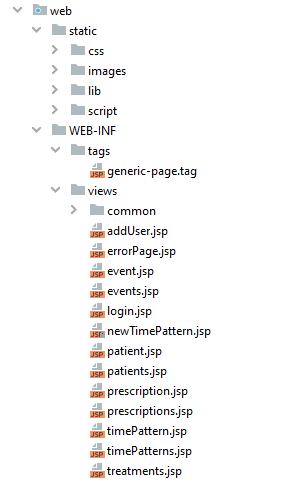
Service level:



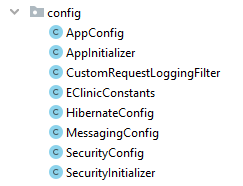
View-service level:



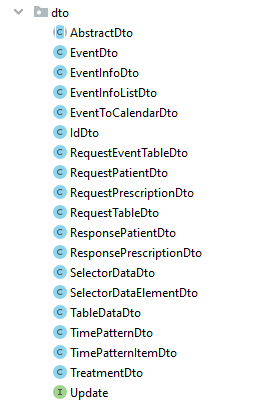
View level:



Configuration:

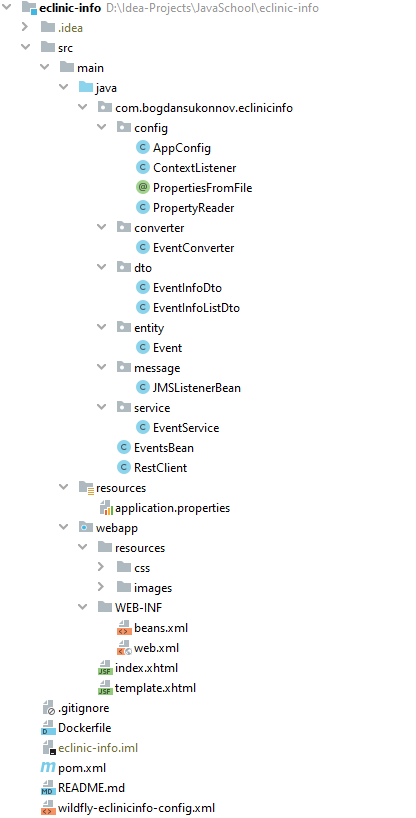


DTO:



Rest client applications:

1. 2.



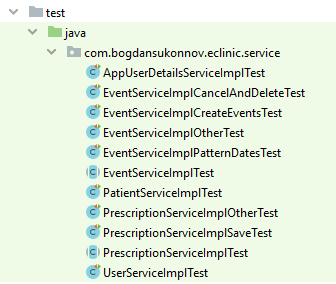
# ADDITIONAL FEATURES

1.Docker. All parts of the application are stored in docker containers and starts up by docker-compose command

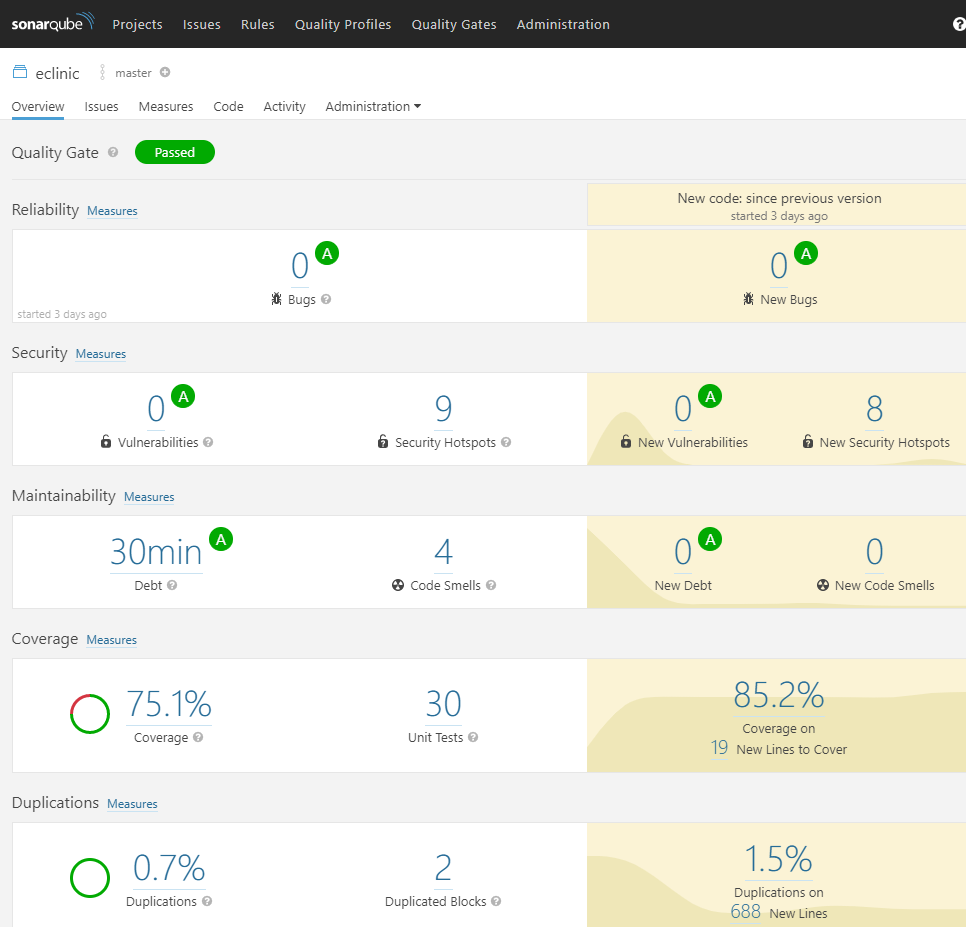
2. Jenkins + SonarCube. Jenkins starts build automatically after the push to bitbucket repository and check code with SonarScanner plugin.

# CODE QUALITY

Tests:



Sonar report:



## BUILD AND DEPLOY

To build the application run the command in a terminal *mvn clean install flyway:migrate*

Docker-compose up command is a part of the install maven goal.

## FUTURE IMPROVEMENT

1. Refactoring and optimization code.
2. Adding new functionality