**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Чорноморський національний університет   
імені Петра Могили**

**Факультет комп’ютерних наук**

**Кафедра інженерії програмного забезпечення**

**ЗВІТ**

**З ЛАБОРАТОРНОЇ РОБОТИ № 5**

з дисципліни «Професійна практика програмної інженерії»

на тему: «**Інтеграція з віддаленим сервером**»

Спеціальність 121 «Інженерія програмного забезпечення»

121 – ЛР.ПЗ.05 – 309.1717108

Виконали: студенти групи 309

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С. В. Котигробов

(підпис, ініціали та прізвище)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Є. Д. Стоєв

(підпис, ініціали та прізвище)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_М. Г. Пасіченко

(підпис, ініціали та прізвище)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_О. В. Конишев

(підпис, ініціали та прізвище)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

Перевірив: викладач кафедри ІПЗ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Є. О. Давиденко

(підпис, ініціали та прізвище)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

**Миколаїв – 2020**

**Зміст**

[**Product Backlog продукту** 3](#_Toc41481394)

[**Діаграми взаємодії для першого sprint** 5](#_Toc41481395)

[**Діаграми класів першого sprint** 7](#_Toc41481396)

[**Макет першого sprint** 11](#_Toc41481397)

[**Ознайомлення из проектом на ресурсі Trello** 11](#_Toc41481398)

[**Інтеграція проекту з системою Jenkins** 12](#_Toc41481399)

[**Інтеграціая проекту на власний віддалений сервер** 15](#_Toc41481400)

# **Product Backlog продукту**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Story | Tasks | Estimate | Story point | Sprint | Description |
| 1 | As a member, I can start the app and see the main activity of system | Create simple App | 150 | 8 | 1 | We need a model of our future app to see how develop it next |
| Design simple UI to review the programm | 70 | 13 |
| Create DataBase and tables for Users of app | 120 | 13 |
| Connect DB’s data and UI client’s activity for login/registration | 100 | 8 |
| 2 | As a user, I must be in touch with the teachers and be able to sign up for consultations, same I need to see all consultations and shedule | Create table with consultation’s characterisics | 70 | 21 | 2 | The most important part is keeping the separation of weeks |
| Marking up the activity of Consultations and creating its interface | 50 | 21 |
| Connect DB’s data and UI client’s activity “Consultations” | 90 | 21 |
| 3 | As a user, I can watch my personal lesson schedule at the university in order to make it easier for me to navigate the lesson schedule | Creating the basis in the database, setting up and linking tables for the correct work with data and information elements.  Creating tables, taking into account the division into clear and odd, so that the application correctly reports schedule changes | 130 | 89 | 3 |  |
| Create activity for displaying information from the database, marking up tables in the application and setting sizes for different device screens | 110 | 55 |
| Linking data from the database to the UI client and manipulating information. adjustment of display correctness taking into account the parity of weeks of study | 80 | 55 |
| 4 | As a user, I can see my personal information and photos in this app. Also I can login and register my account and stay online in app | Modernization of login/register activity, redesign UI and more powerful back-end. | 90 | 21 | 4 |  |
| Rework login system. Now server can give to user 2 tokens for online session. Every five minutes tokens are die. App sent refresh token to server for update 2 tokens and session extension | 110 | 55 |
| 5 | As a user, I can use app without any problems, everywhere and whenever. I have not think about loading data or smth else. I found wi-fi and app work correctly | For perfect access to our app we need create server point for back-end data and DB, so for start we are save all project and backup it. | 50 | 21 | 5 |  |
| After saving, we can create own server on one of laptops of member of team. Install Ubuntu server, Java soft, MySQL server and connect machine to the internet. | 90 | 55 |
| Creating users/privilege on server and customization access to data | 90 | 21 |
| Finally, install our project to the server and test it. Connect from other PCs, request\answer. | 80 | 21 |
| 6 | As a member, I have to understand what happened with project, I need information about status of programm | Generating documentation for front-end with java tools and public it to GitHub | 90 | 13 | 6 |  |
| Using swagger for help with generating docs for backend part of project and public it on GitHub too | 110 | 55 |

# Щоб переглянути наш GitHub перейдіть за посиланням(https://github.com/NarutaUzumaki/AppForUniver/tree/master). Ми маємо 2 репозиторії, один із кодом для Android застосунку, інший беспоредньо віпдовідає за бекенд частину проекту, нажаль її ми не можем залишити в посиланнях, оскільки в цілях безпеки було вирішено цей репозиторій залишити приватним для учасників і розробників системи, дякуємо за розуміння. Гілка з лабами(https://github.com/NarutaUzumaki/AppForUniver/tree/Documents/labs). Гілка з діаграмами(https://github.com/NarutaUzumaki/AppForUniver/tree/Documents/diagrams). Гілка з product backlog().

# **Діаграми взаємодії для першого sprint**

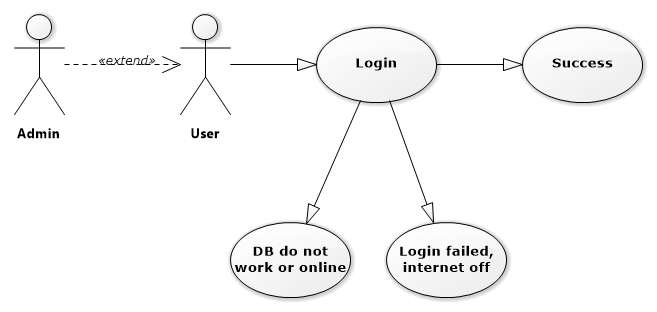
****

Рис.1 Прецедент логіну у застосунок

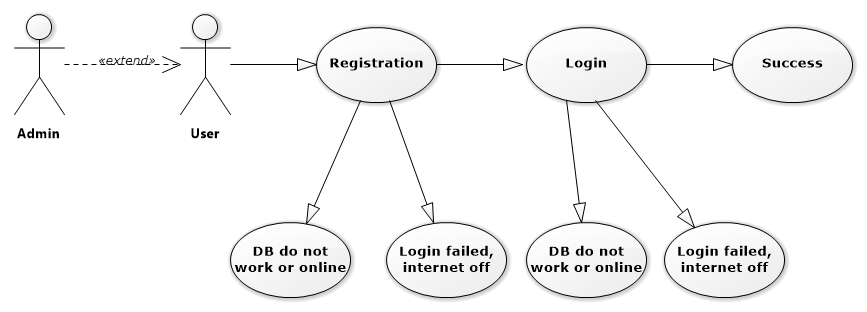
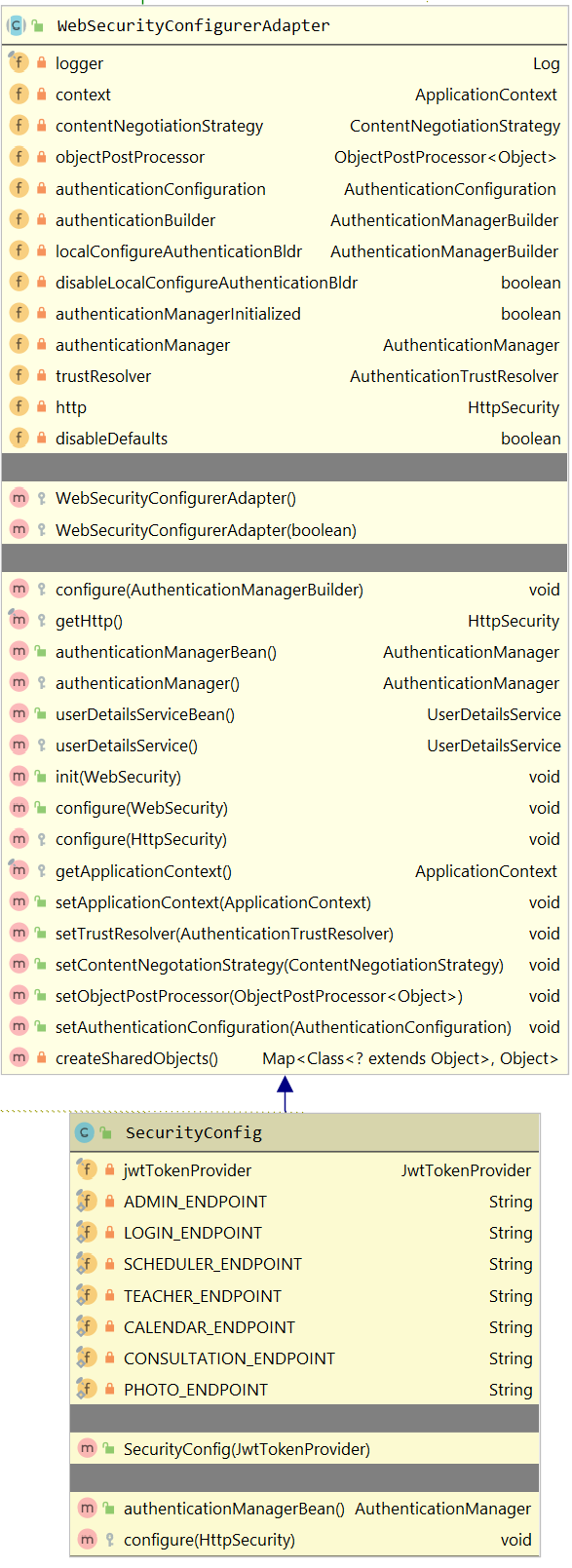


Рис. 2 Прецедент реєстрації нового користувача

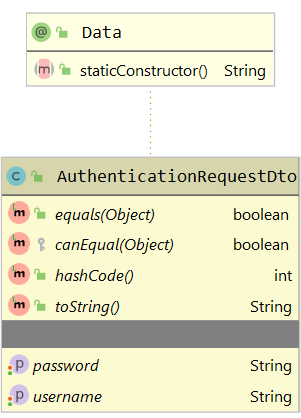
# 

# **Діаграми класів першого sprint**

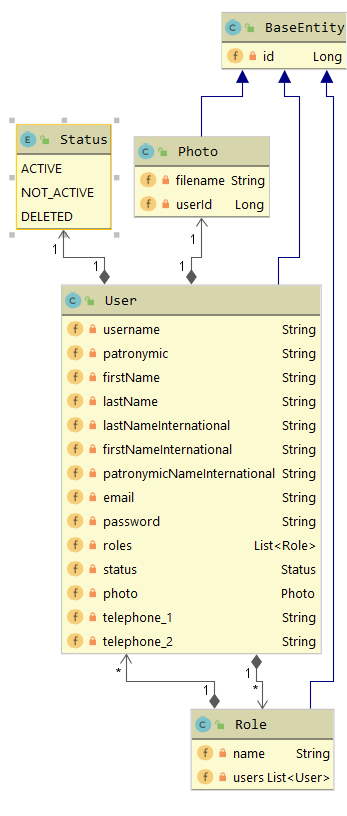
SecurityConfig



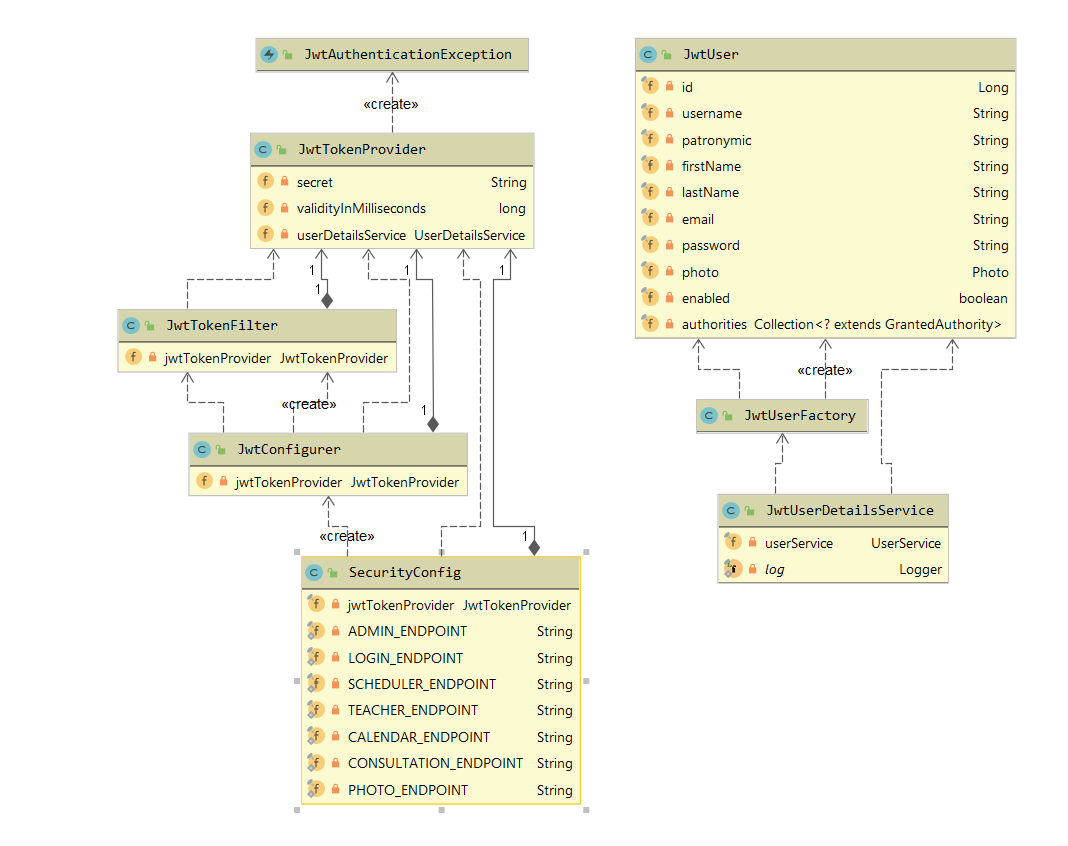
AuthenticationRequestDto



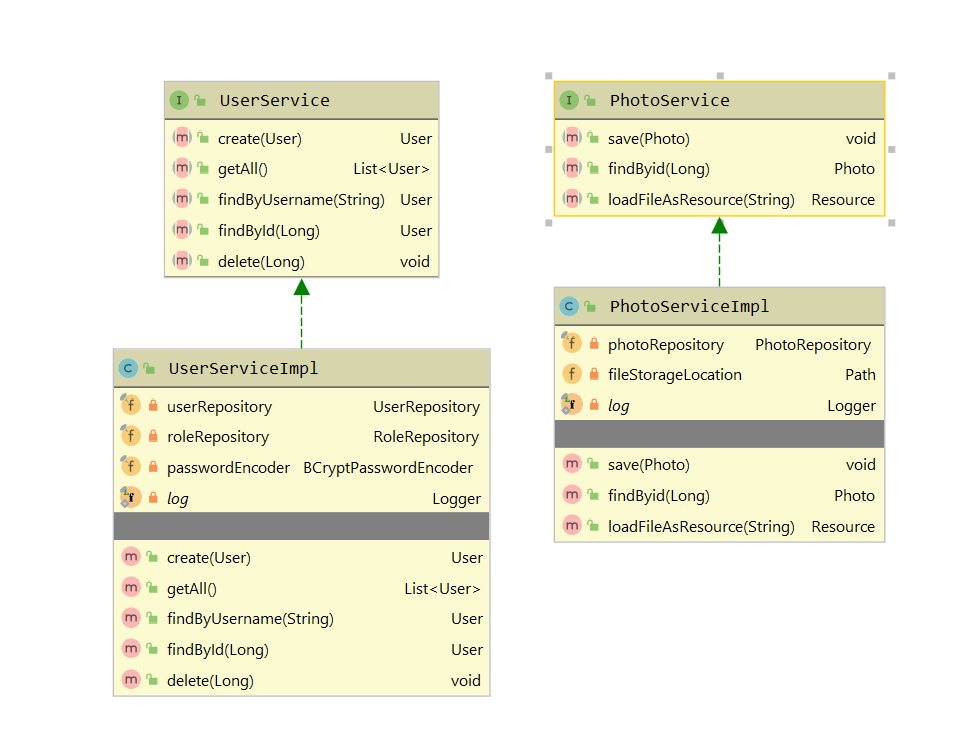
User



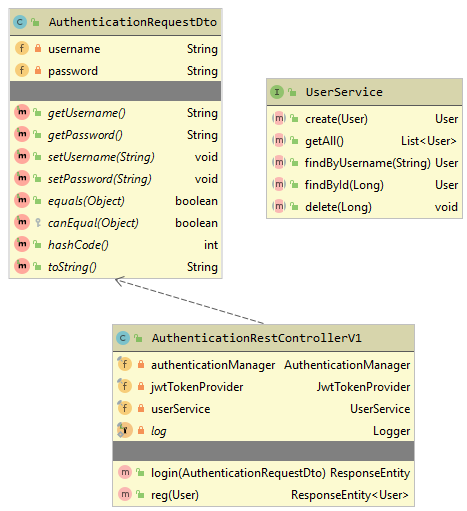
security



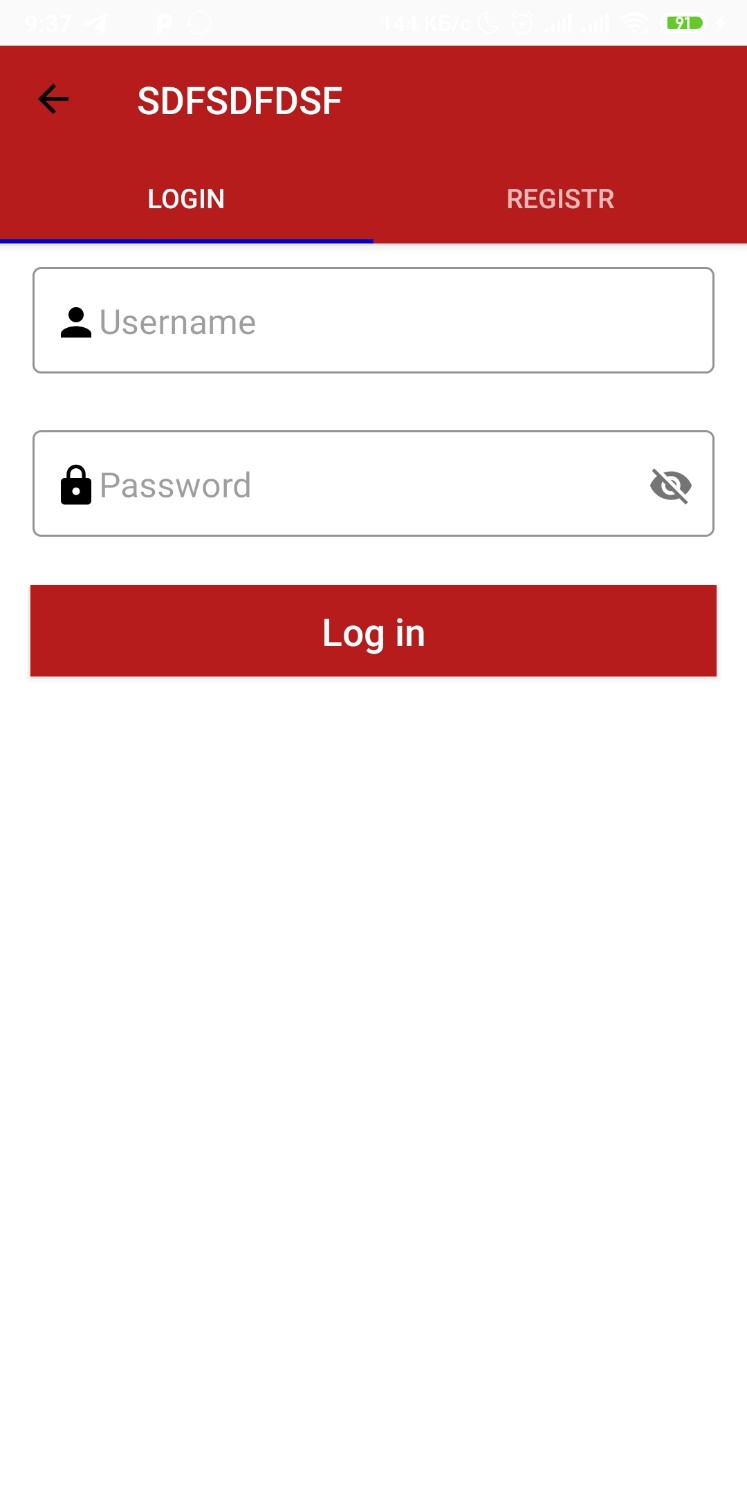
service



rest

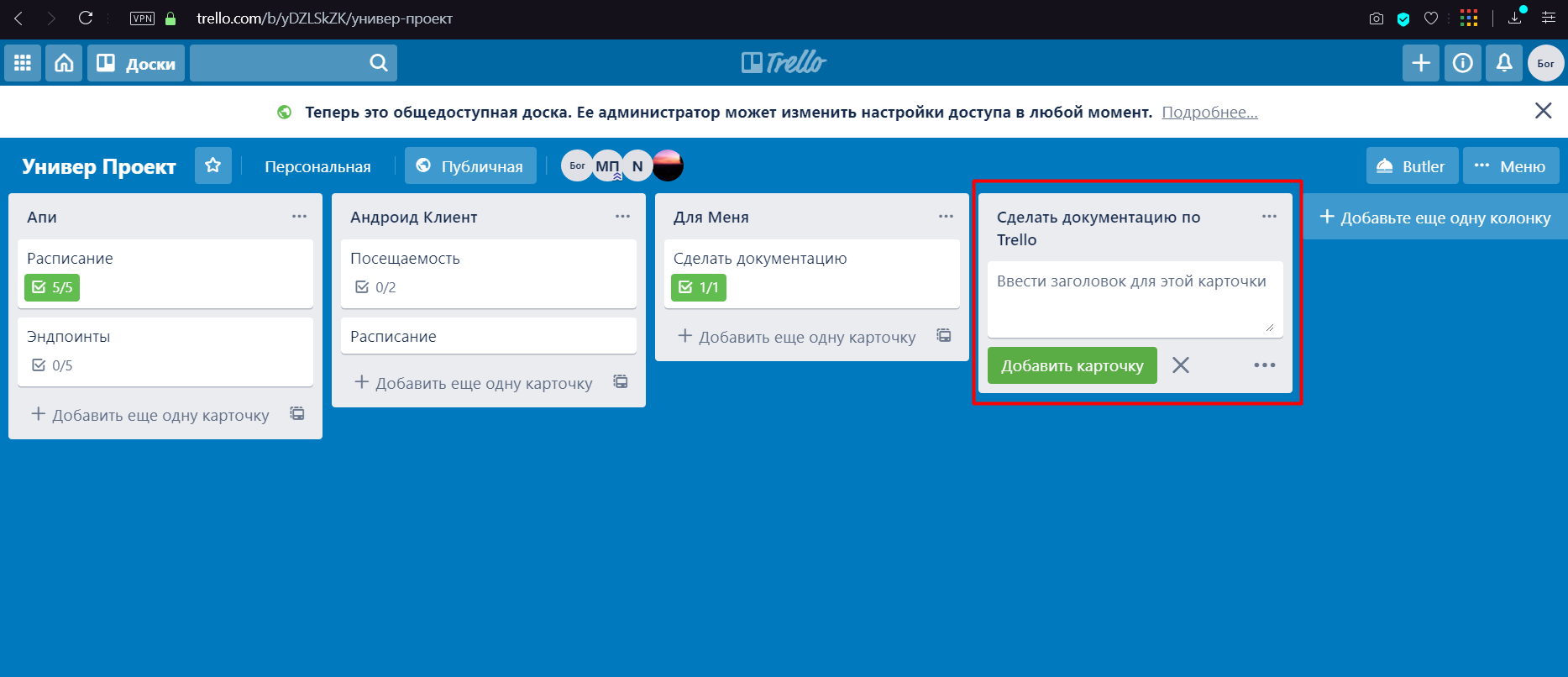


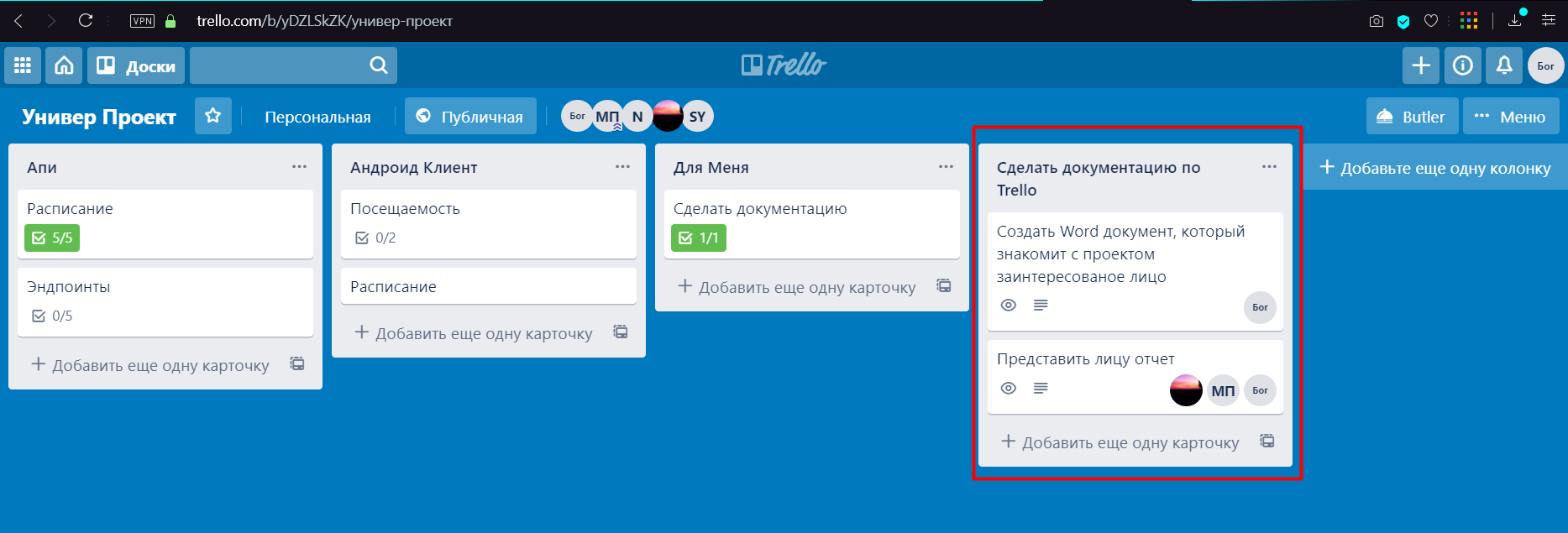
# **Макет першого sprint**



# **Ознайомлення из проектом на ресурсі Trello**

Trello – це облачний сервіс для керування проектам невелеких груп розробників, свого роду «Доска завдань».Сервіс дозволяє створювати задачі(«Таск») та керувати їми по ходу виконання.



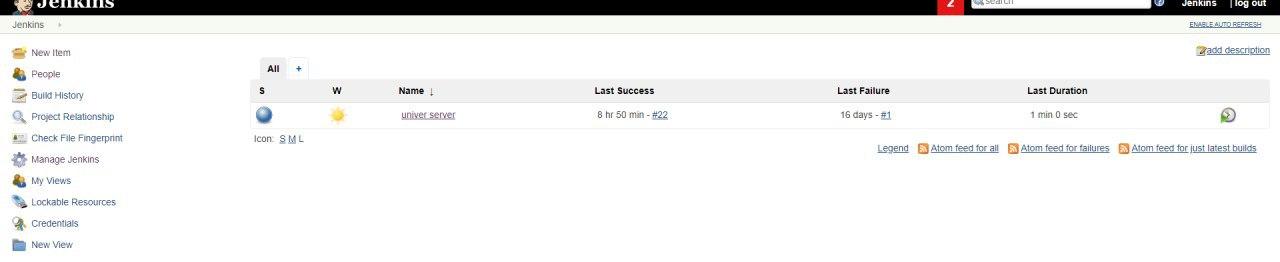
Кожен учасник проекту може свторювати задачі та контролювати її виконання. Також є можливість додавання до проекту інших учасників, та тримати їх у курсі виконная саме цієї частини.  
  
Ниприклад, коли цей звіт буде зданий, задачу презентації у Trello можна буде переводити у статус «Виконано».

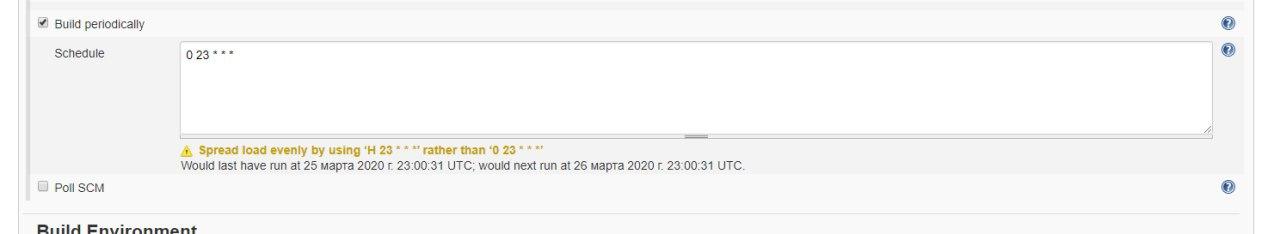
# **Інтеграція проекту з системою Jenkins**

**Jenkins** — [відкритий](https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%96%D1%82%D0%B8%D0%BA%D0%B0_%D0%B2%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BE%D0%B4%D1%83" \o "Політика відкритого коду) інструмент для [безперервної інтеграції](https://uk.wikipedia.org/wiki/%D0%91%D0%B5%D0%B7%D0%BF%D0%B5%D1%80%D0%B5%D1%80%D0%B2%D0%BD%D0%B0_%D1%96%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B0%D1%86%D1%96%D1%8F" \o "Безперервна інтеграція) написаний на [Java](https://uk.wikipedia.org/wiki/Java). Jenkins допомагає автоматизувати частину процесу [розробки ПЗ](https://uk.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F" \o "Розробка програмного забезпечення), яка не потребує участі людини, в таких частинах, як [неперервна інтеграція](https://uk.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BF%D0%B5%D1%80%D0%B5%D1%80%D0%B2%D0%BD%D0%B0_%D1%96%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B0%D1%86%D1%96%D1%8F" \o "Неперервна інтеграція) та розширює технічні можливості команд розробників з [неперервною доставкою](https://uk.wikipedia.org/wiki/Continuous_delivery" \o "Continuous delivery) продукту.

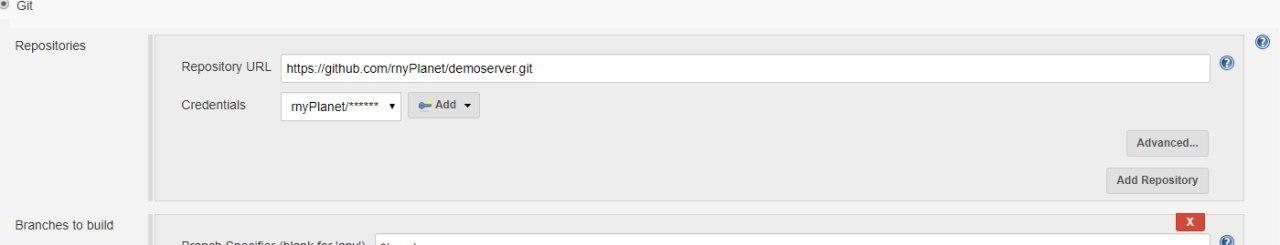
Нижче представлена головна сторінка тестів для застосунку. Jenkins буде сам збирати проект та тестувати його кожен день за допомогою репозиторія на GitHub.



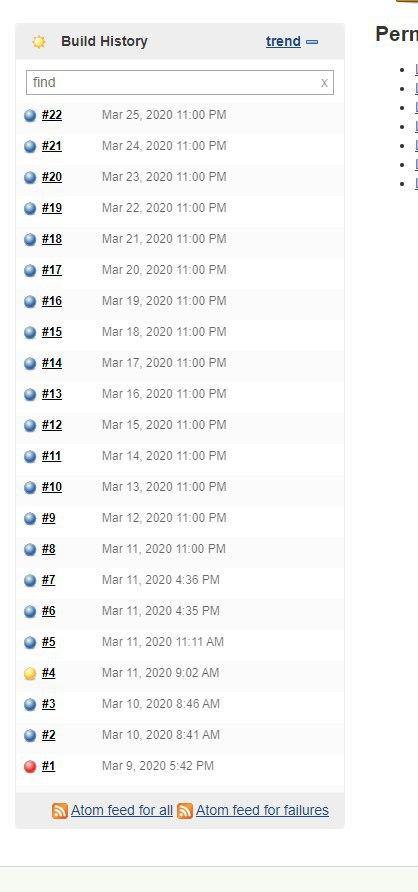
На сторінці з проектами відображаються всі застосунку розробника що включені до системи з короткою статистикою по кожному. Останній вдалий build, останній провальний build, ім’я, остання тривалість.  


Система Jenkins тестує наш сервер із back-end частиною застосунку через GitHub.

Збірка та тестування проекту буде влаштовуватись кожен день об 23:00 та зберігатись у статистици результатів.



Нижче наведена статистика тестувань за попередній місяць.

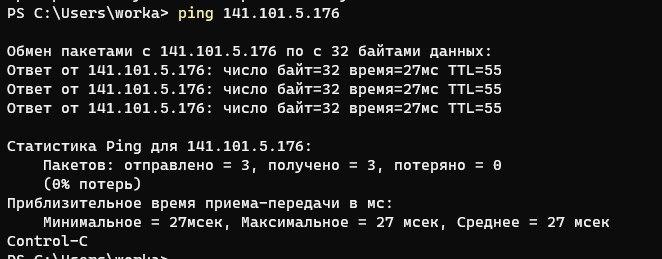


# **Інтеграціая проекту на власний віддалений сервер**

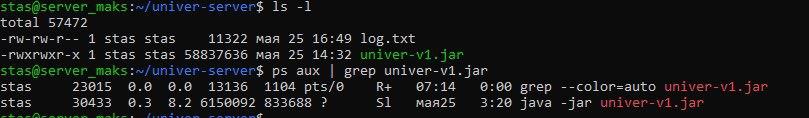
Оскільки інтеграція проектів із серверами або хостами коштує грошей командою було вирішено влаштувати власний сервер на старому лептопі одного із учасників проекту. Сервер встановлено на лептопі Fujitsu S751.Характеристики серверу наведені нижче.

* Процесор - Intel(R) Core(TM) i3-2310M CPU @ 2.10GHz
* ОЗУ - 10гб
* Сховище - 1ТБ

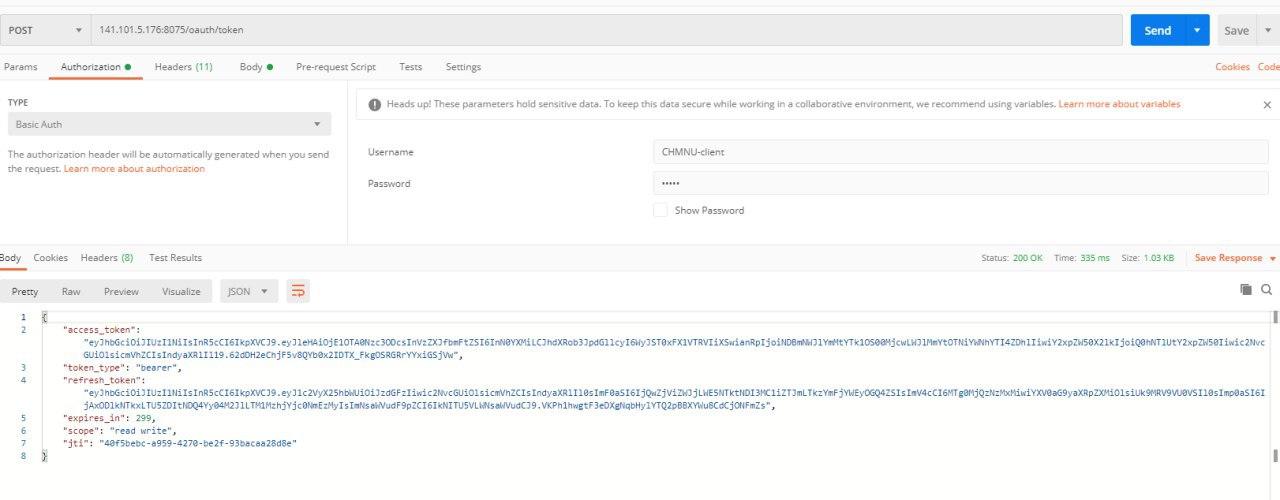
Щоб запустити сервер на машину було встановлено Ubuntu server, переналаштовано мережеву карту, та встановлено відповідний софт для комфортної роботи. Відповідальна за сервер людина дає доступ бекенд-програмісту до інструментів серверу, після цього там розміщується наш бекенд код. Перевіримо наш сервер звичайни ping’ом з іншого ПК та впевнимося в його коректній роботі.



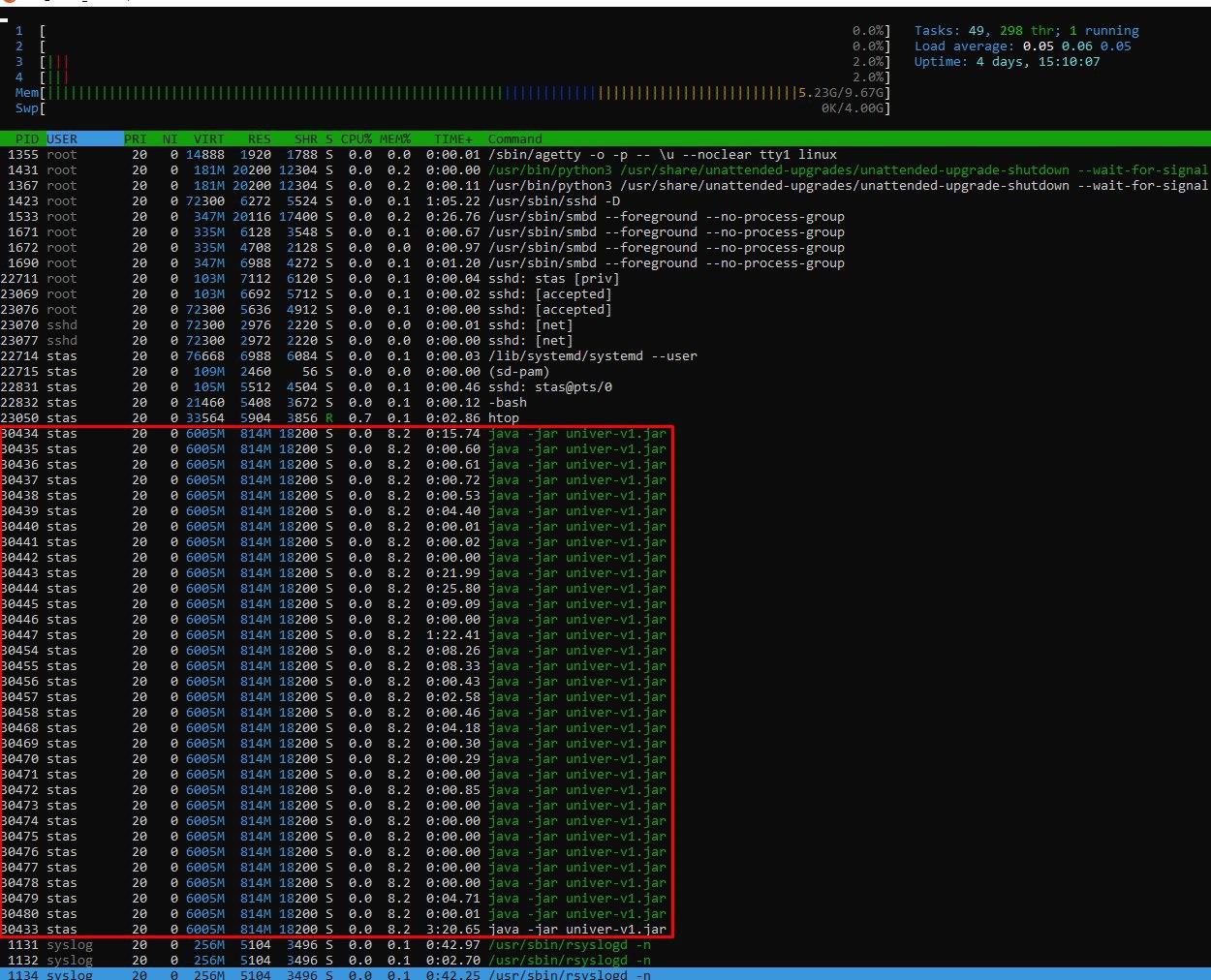
Сервер готовий до роботи, надалі можна встановлювати на нього проект та користуватися. Нижче представлено скріншот наявності проекту на сервері після встановлення.



Спробуємо протестувати наш застосунок та переглянемо чи змінилась коректність роботи проекту після інтеграції.

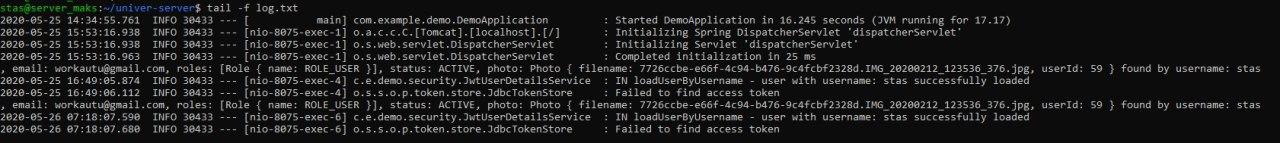


Сервер працює, це можна помітити, поглянувши на консоль, що демонтсрує отримання пакетів та токенів у json форматі, отже ми отримуємо інформацію. Теперь впевнимося чи саме сервер дає нам ці дані, а не випадкова генерація у коді проекту. Переглянувши скріншот, наведений нижче, спокійно продовжуємо роботу, адже сервер дійсно реагує на запити клієнту та на відповідає нам.

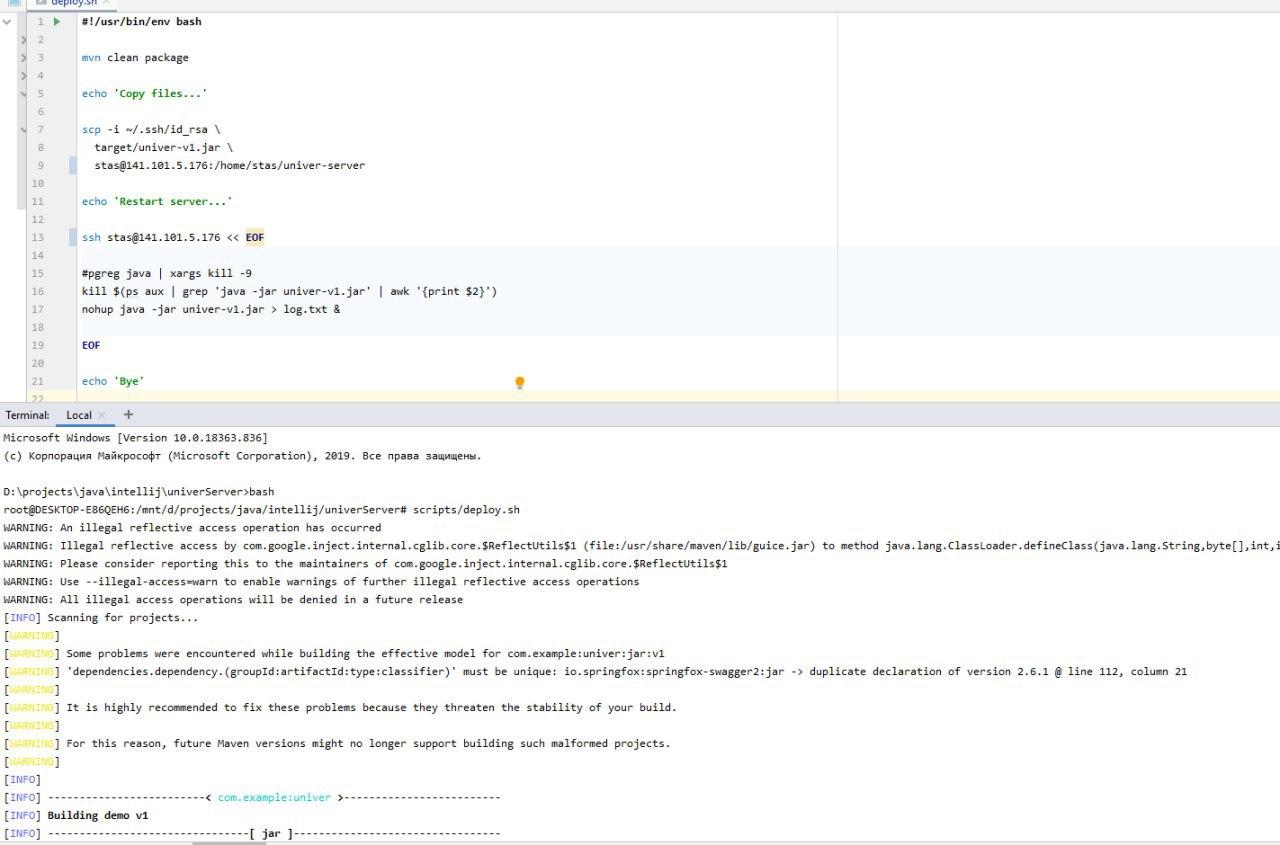


Нижче представлені лог-файли для ознайомлення із роботою віддаленого серверу. Варто відмітити, що власний сервер дає багато переваг:

1. Тотальний контроль над проектом
2. Захищеність від стороніх софтів
3. Незалежність від умов\цін\правил хостерів та компаній, що надають послуги у даній сфері
4. Швидкий доступ до серверу та всіх його інструментів
5. Вільний доступ до проекту через сервер без обмежень часу



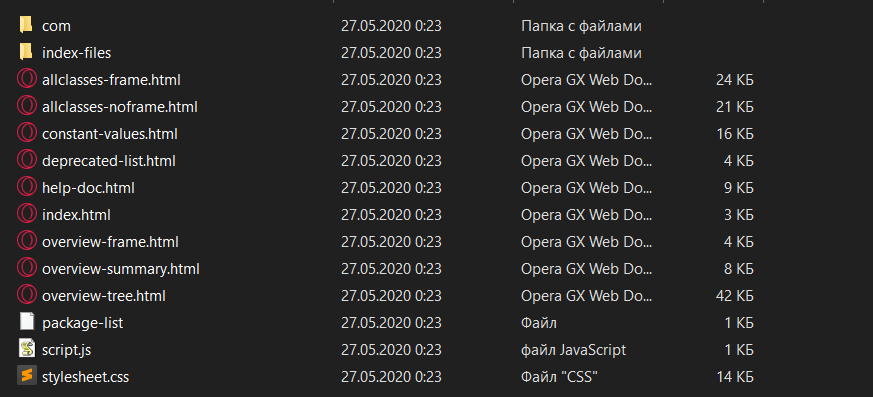
Також, згадуючи про інтеграцію з проектом Jenkins, варто відмітити, що обув написаний скрипт, що власноруч збирає, тестує та відправляє проект на сервер. Якщо проект вже є на сервері та запущений, він зупиняє виконання, очищається та перевстановлюється на віддалений сервер. Ця практика дозволяє запобігти старим помилкам та некоректностям у збірці, а також тримати проект під жорстким контролем програмістів. Скрпит-автозбирач наведено нижче.



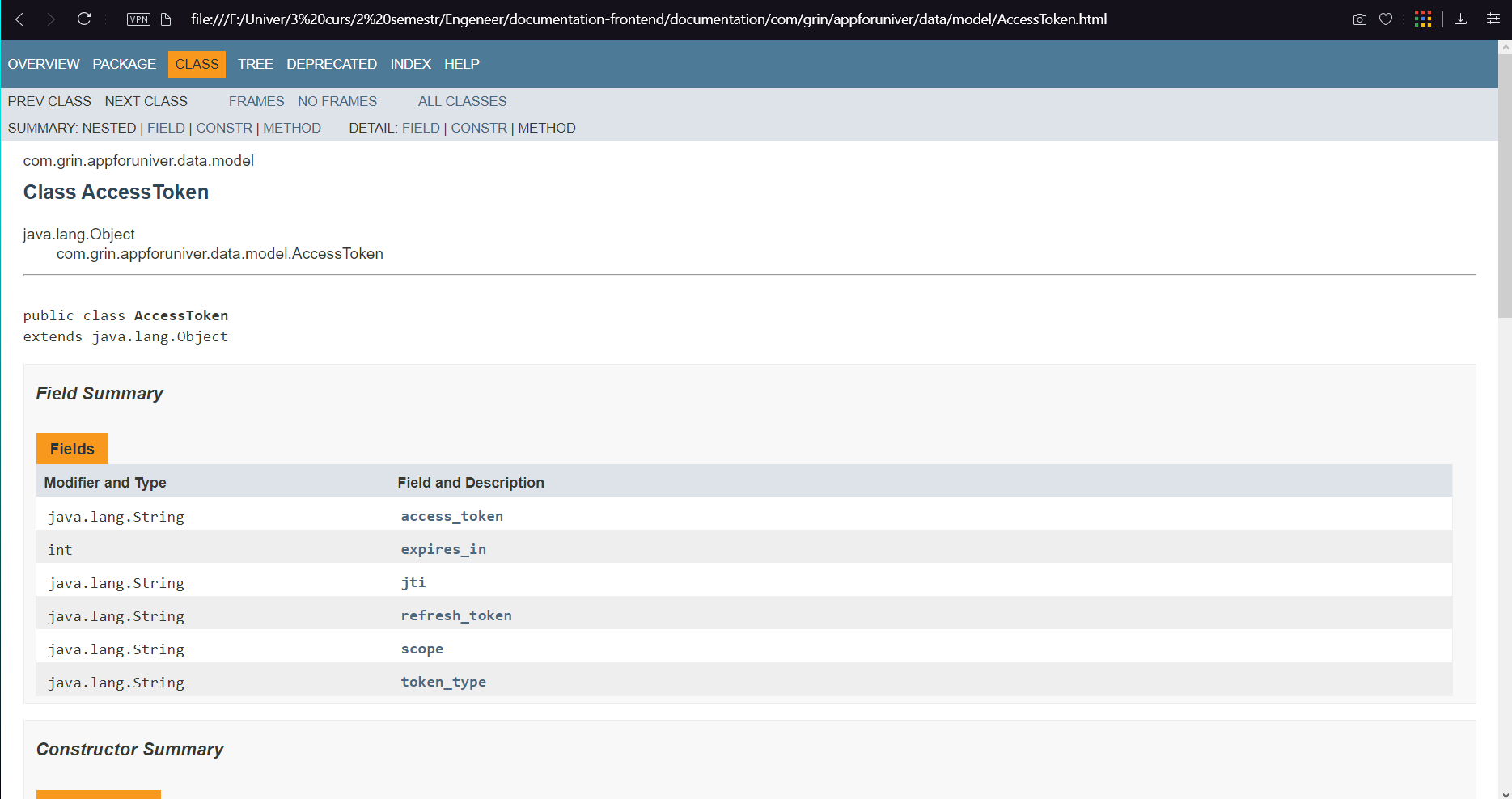
# **Генерація документації**

## ***Front-end***

В ході даного завдання спочатку були розмічені класи фронт-енд частини проекту. Документація генерувалася за допомогою JavaDoc. Оскільки не було представлено ніяких прикладів стосовно докумаентації проектів на java, кінцевий результат може відрізнятися від бажаного викладачем, дякуємо за розуміння. Команда достатньо довго працювала щоб сгенерувати документація, адже проект за весь час обріс не малою кількістю коду на логічних елементів загалом, і поставлене завдання виконала. Як виглядає кінцева документація наведено далі.



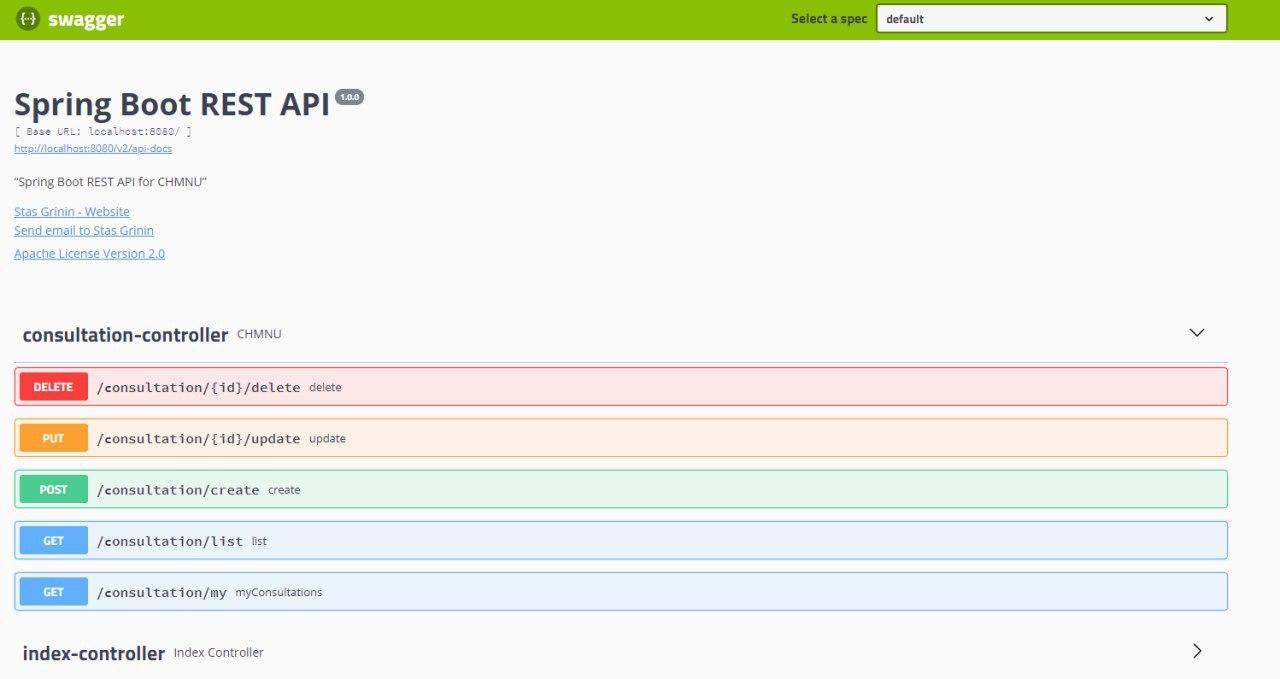
Нижче наведено скріншот згенерованого файлу документації одного із класів, що працює із token, що отримується від бекенд частини.

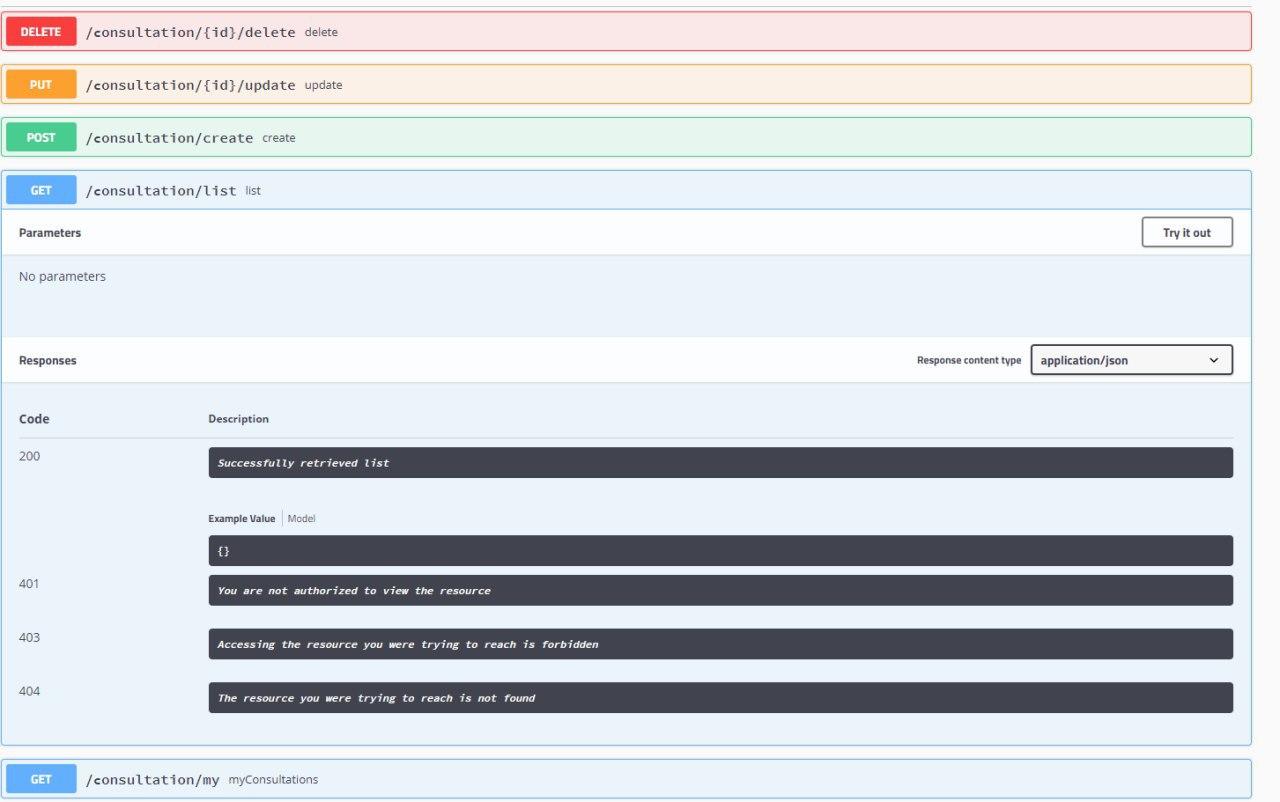


## ***Back-end***

В ході цього завдання виникли деякі складнощі, і для того щоб сгенерувати документацію бекенд частини проекту необхідно зупиняти сервер та перезавантажувати проект. В ціляї безпеки і невтрати коду та даних було вирішено відкласти це, якщо викладач потребує, документація буде завершена повністю, дякуємо за розуміння.

Щоб отримати документацію ми скористалися допоміжним інструментом swagger, що значно полегшив роботу генерації документації із бекенд. Нижче наведено один з сгенерованих файлів.





Також представимо наш розмічений бекенд код та відмітимо, що результати задовольняють потреби.

