## TCP SYN FLOOD ATTACK

## Team Formation

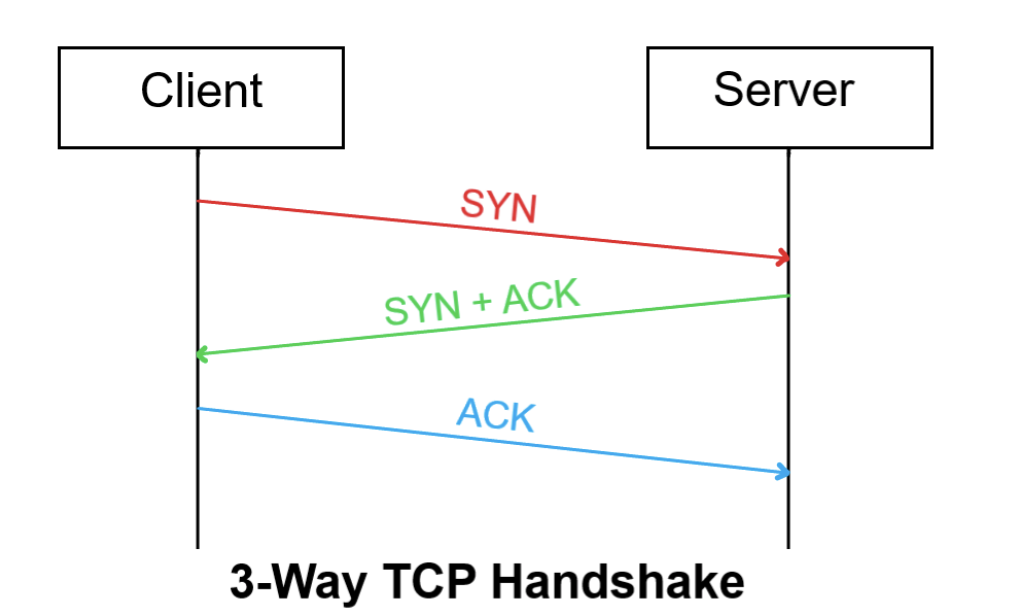
1. **Project purpose or justification-** **As part of this project, we will tell you about the essence of the TCP SYN Flood attack. In addition, you will learn how to simulate this malicious DoS attack for test purposes, also infamous as DoS attacks, are quite easy to carry out, are not always obvious and can cause serious failures in the computer system, which will inevitably lead to an increase in the downtime of your system resources.**
2. **How TCP SYN Flood Attack Works**

In our project, we will use a Denial of Service attack, also known as a DoS attack, which is quite simple to carry out, not always obvious and can cause serious failures in the computer system, which will inevitably lead to an increase in the downtime of your system resources. In an attack using SYN packet overflow (TCP SYN Flood), attackers (in this case, we) use a three-way TCP handshake to cause network and service failures. Attacks of this type can easily take you by surprise, as it is often difficult for system administrators to quickly identify them.

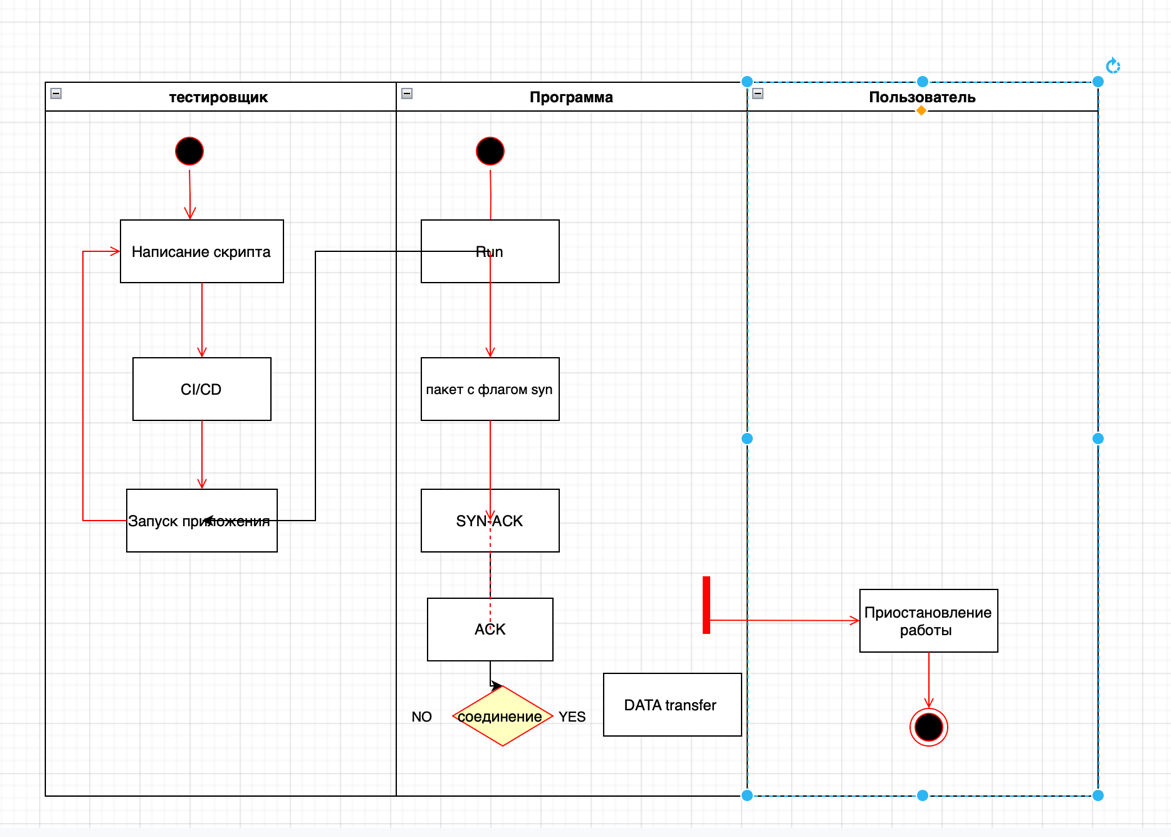
When a client tries to connect to the server using the TCP protocol (for example, when establishing an HTTP or HTTPS connection), he must immediately go through the three-way handshake procedure before data exchange between the client and the server becomes possible. Since the initiator of the TCP three-way handshake is always the client, he is the first to send a packet with the SYN flag to the server.

After receiving such a SYN packet, the server responds by confirming receipt of the request and sending its own SYN packet request with the SYN and ACK flags set. The client, in turn, after receiving a confirmation packet and a request from the server, sends a packet to the server with the ACK flag set, which confirms that both hosts agree to create a connection. After such an "exchange of handshakes", the connection is considered established, and data can be transferred between hosts.

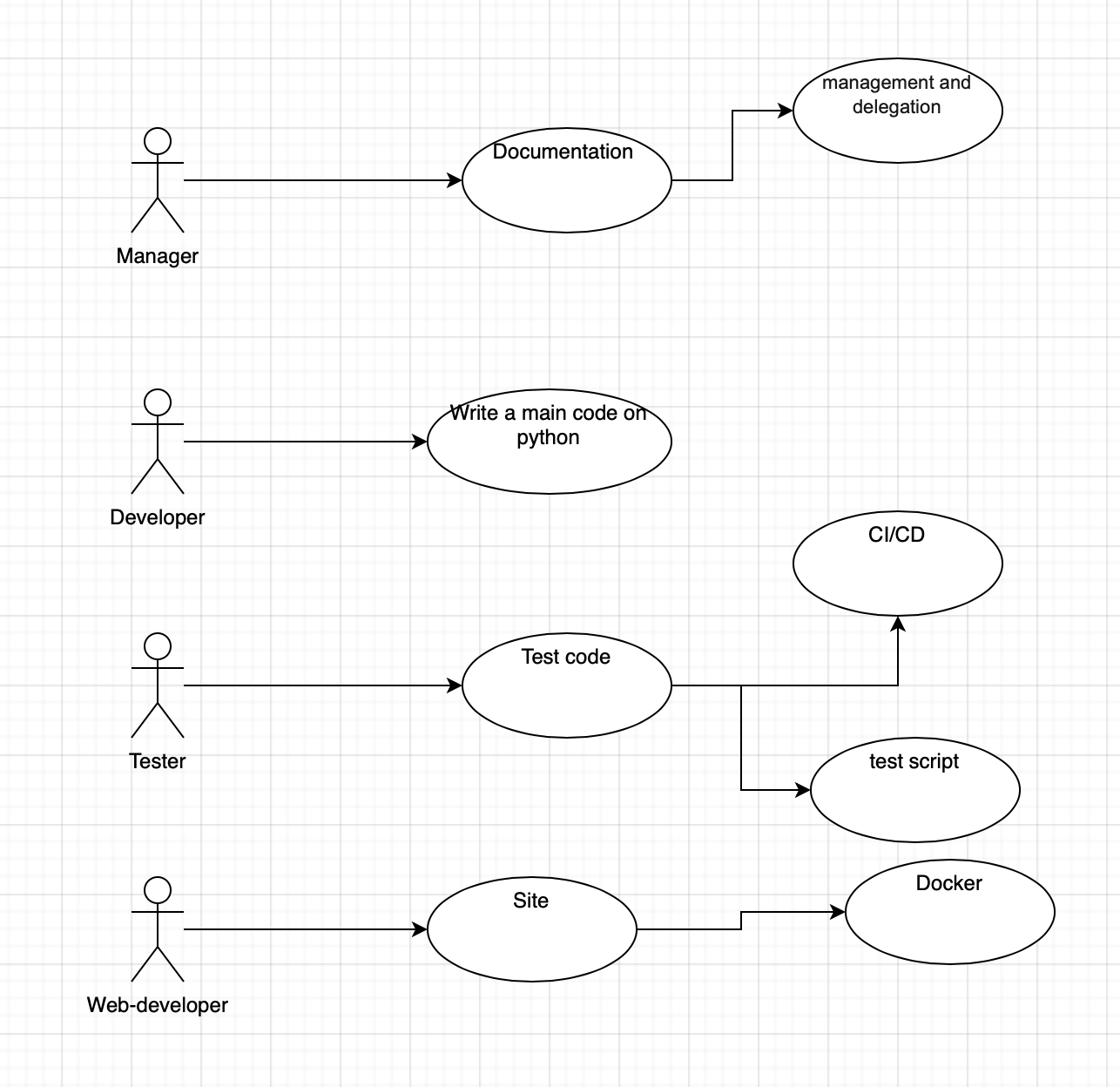
When conducting a TCP SYN Flood attack, we will intensively send a large number of SYN packets with fake IP addresses to the server. This forces the server to respond by sending a SYN-ACK packet in response to each such false request, allocating some resources and leaving its ports "half-open" in anticipation of numerous responses (packets with the ACK flag set) from hosts that do not actually exist, and they, accordingly, will not send confirmations.



For even greater understanding, I created a uml activity diagram, this diagram reflects business processes, logic of procedures and work flows - transitions from one activity to another

****

To understand the roles and their relationship, I created a Use-case diagram

****

1. **Measurable project objectives and related success criteria**:

|  |  |
| --- | --- |
| **Objective** | **Success criteria** |
| **Write code in the python programming language to simulate a TCP SYN Flood attack** | **we will get a working project that, in a test environment, is capable of stopping the work of the site for a while.** |
| **Using Docker,create a site for testing the attack** | **We will simplify the work of the team and create a website through docker, since we will not need to download a separate environment for work** |
| **Also, to automate the project, we will start working with CI/CD, creating tests and pipelines** | **improving the convenience, frequency and reliability of deploying software or product changes** |

1. **High-level requirements:**
   1. **Show how easy it is to create and use tcp syn flood (that is, many can use it to interrupt the work of many web applications)**
   2. **Possible use on different platforms**
2. **Assumptions and constraints**
   1. **We have limitations in the form of, this project will continue to work at the test level, that is, it is not aimed at breaking any sites on the Internet**
3. **High-level risks** - **sites with weak protection may suspend their work for a while when used for tests**
4. **Summary milestone schedule**
   1. **Writing a project in pycharm**
   2. **at the same time starting the development of the site**
   3. **connecting ci/cd**

**Team and roles:**

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Skillsets, Interests** |
| **Tester** | **Aidos** | * CI/CD * Work with GITLAB * Preparation of a test plan. * Writing test documentation and suggestions for software improvement. * Writing test scripts (test cases). * Testing the functionality of the software. |
| **Developer** | **Nail** | * • Set access rights * • Conduct network monitoring * • Provide network security * • Maintain the software of servers and workstations in working order. |
| **Manager** | **Nazym** | * management and delegation * calculating the budget of the information environment * the bridge between the technical staff and the client |
| **Web-Developer** | **Nikita** | * Backend-by the server part of the product * Frontend-visual design of the project * Docker-automation of application deployment and management |

**GITHUB**

Аs version control we chose github

Because they found such advantages as :

High performance.

Advanced integration tools with other VCS, in particular, with CVS, SVN and Mercurial. In addition to multidirectional repository converters, the bundled software tools allow developers to use git when hosting a central repository in SVN or CVS, in addition, git can simulate a cvs server, providing work through client applications and support in development environments that specifically do not support git.

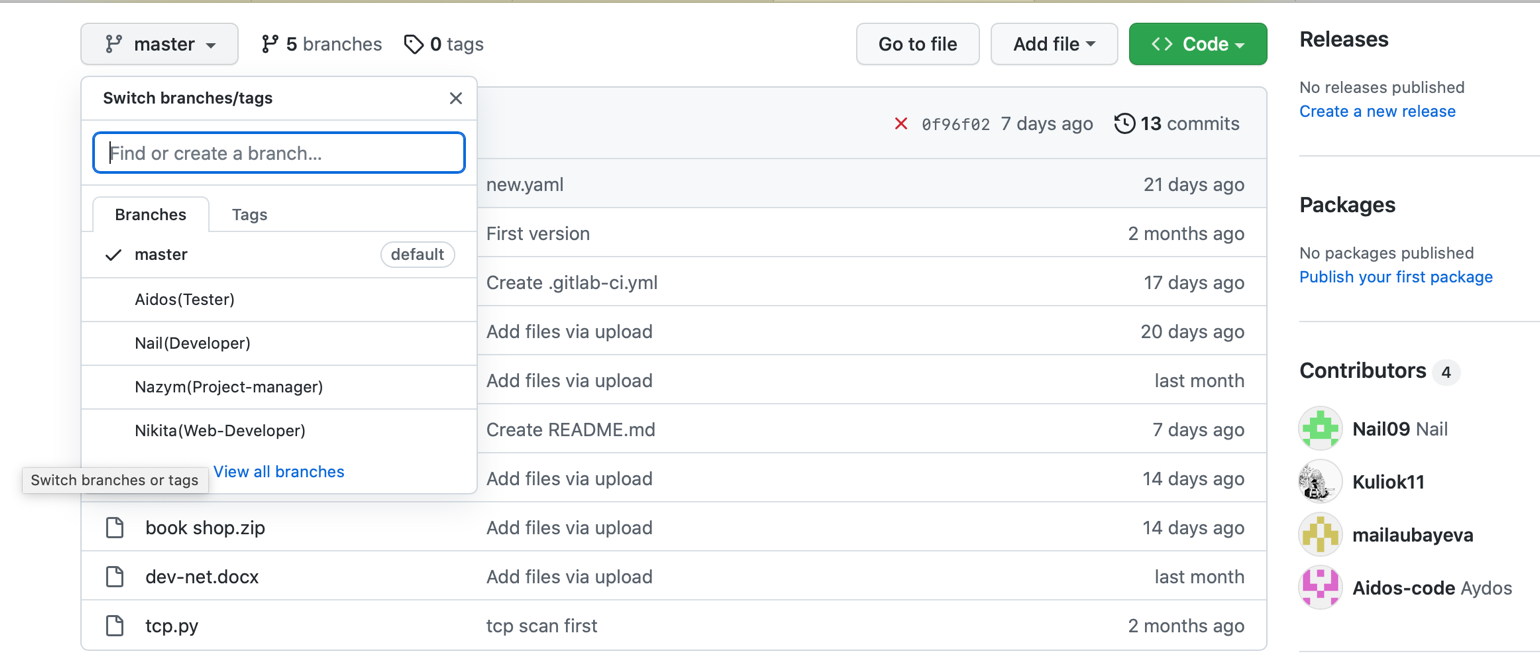
A well-thought-out command system that allows you to conveniently embed git into scripts.

High-quality web interface "out of the box".

Git repositories can be distributed and updated by system-wide file archiving and updating utilities, such as rsync, due to the fact that commit changes and synchronization do not change existing data files, but

only add new ones. Any web server is enough to distribute the repository over the network.

<https://github.com/Nail09/devnet/tree/master>

**For each team member we have branches**

**TCP SYN FLOOD(Nail)**

**Basically, our attack code is written in python, using libraries such as:**

import socket

import random

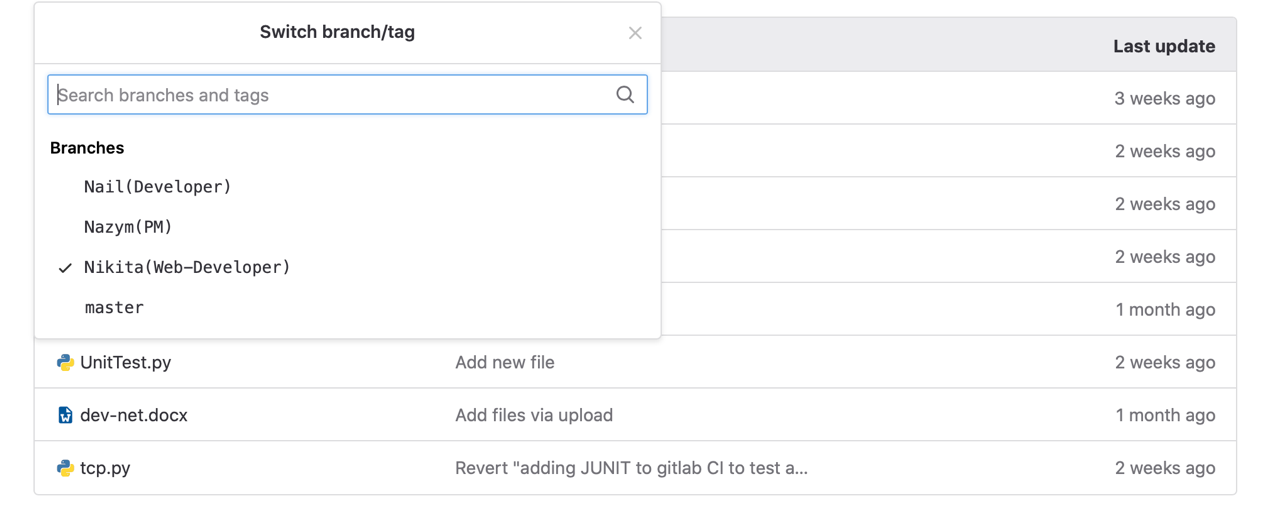
import threading

**This code suspends the operation of the web server for a while, the site that we specified**

https://github.com/Nail09/devnet/blob/master/tcp.py

**GITLAB(Aidos part/CI/CD)**

in gitlab, we also have branches, but since the main project is on github, information from the main one that Aidos works with is duplicated in each branch

****

CI/CD is one of the DevOps practices that allows developers to deploy software changes more often and more reliably, minimize errors, increase the build rate and the quality of the product being developed. It is a combination of continuous integration and continuous delivery

CI, or continuous integration, is the process of continuous software development with integration into the main branch. Automatically collects the software, tests it and notifies if something goes wrong.

CD, or continuous delivery, is the process of constant software delivery to the consumer. Ensures the development of the project in small parts and ensures that it can be released at any time without additional manual checks.

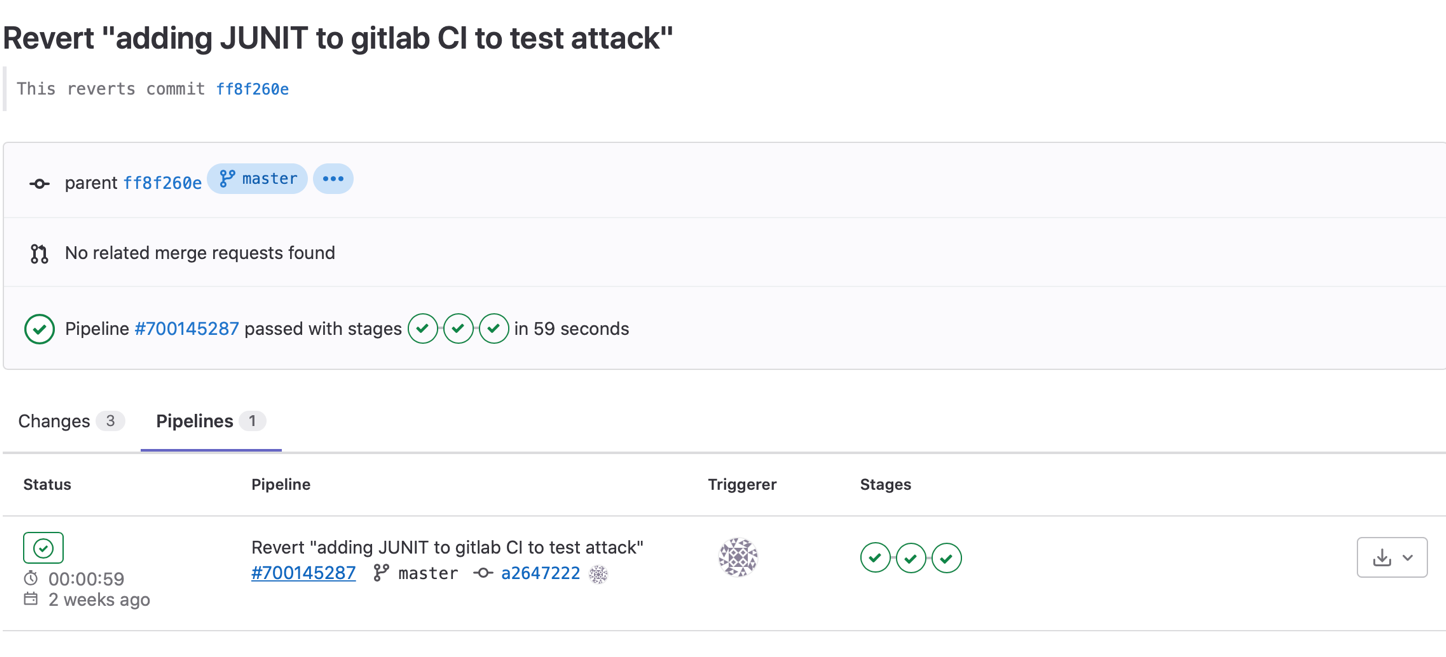
In this project, the task was to create a specific CI/CD for easier startup and automation. What exactly did our CI/CD do:

firstly, with any project, we need to endlessly update parts of the code, improve and refine the code. This solution suits us perfectly, because when setting up, it runs the test itself, checks the logic of our code, in case of passing the check, it then sends it to the "deploy", that is, it deploys itself directly on the server. First, the code goes through the build stage, after we prescribed a simple "Unit test", and also pinged the site in working order, and after the attack, the difference was clear, in the first case we received an answer, in the second there was no answer, the last stage is deployment, the advantage is that it is carried out continuously.

GitLab was a great fit for us, in which we managed to create our own pipeline without any difficulties and run it for our code.

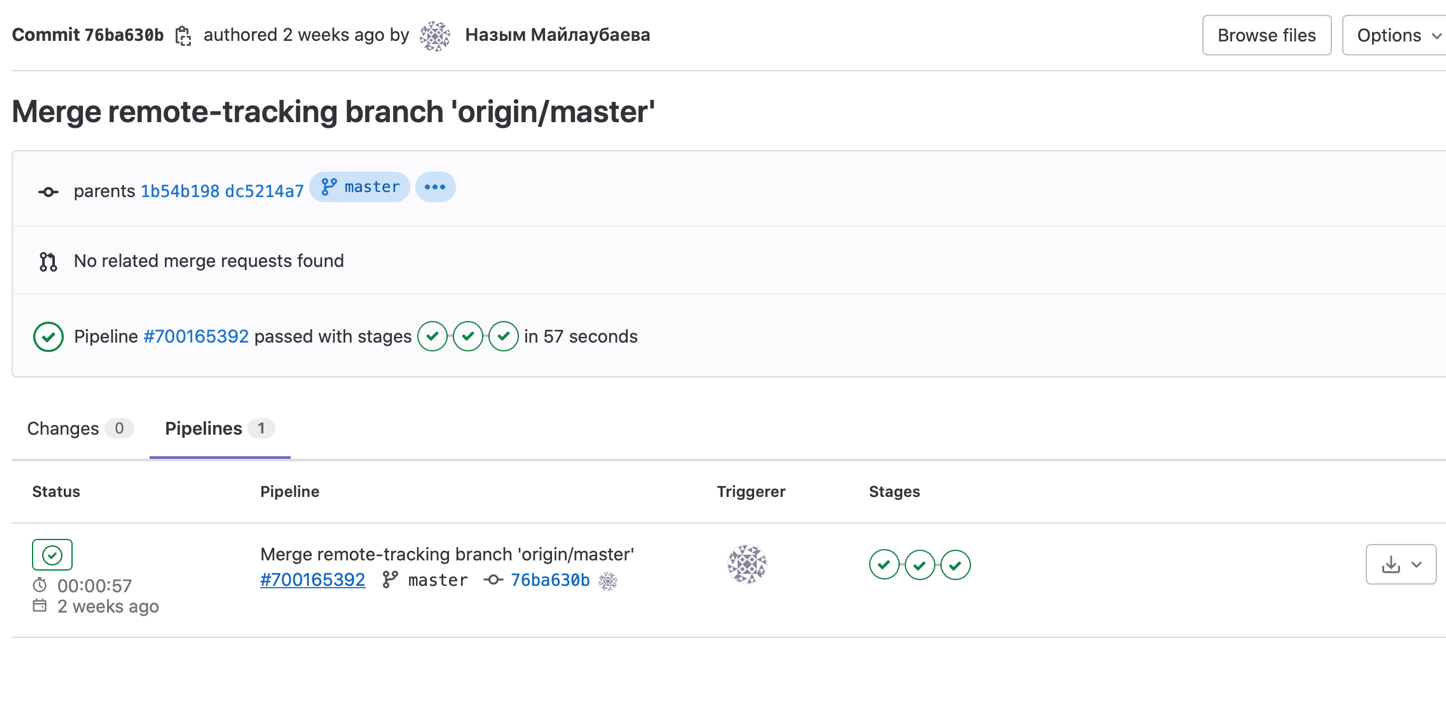
1. **UNITTEST**

**We are going to use Unit Testing. Unit Testing tests individual components of code to validate that each component works as expected. To run the tests I have created a new file called “test\_main.py” within the root of the my project which imports the unittest library and contains the tests I have created. this test check that the returned values from the functions in my Python code return values that are strings**.



**2.REMOTE-TRACKING**

**Remote-tracking branches are references to the state of remote branches. They're local references that you can't move; It moves them for you whenever you do any network communication, to make sure they accurately represent the state of the remote repository.**

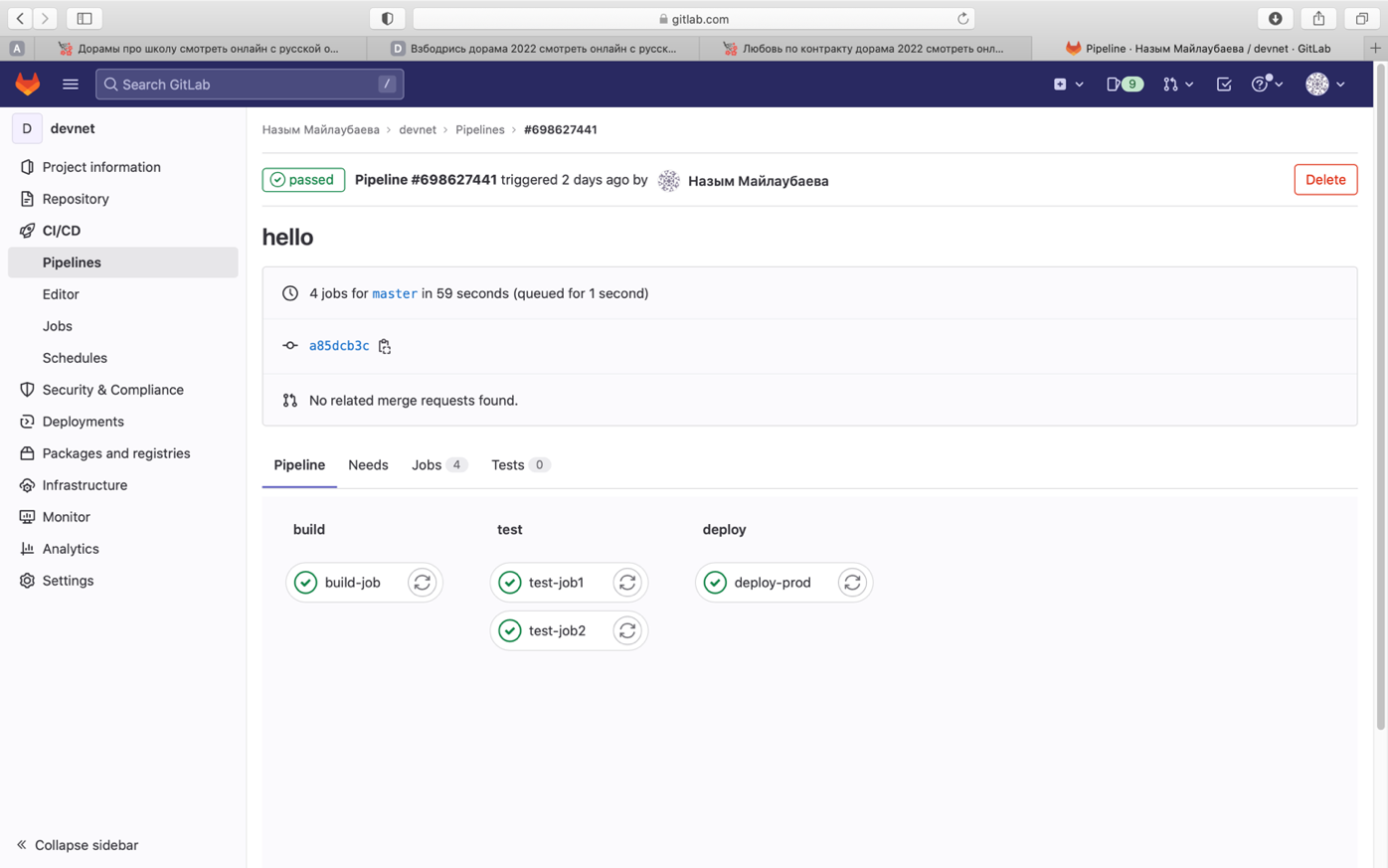
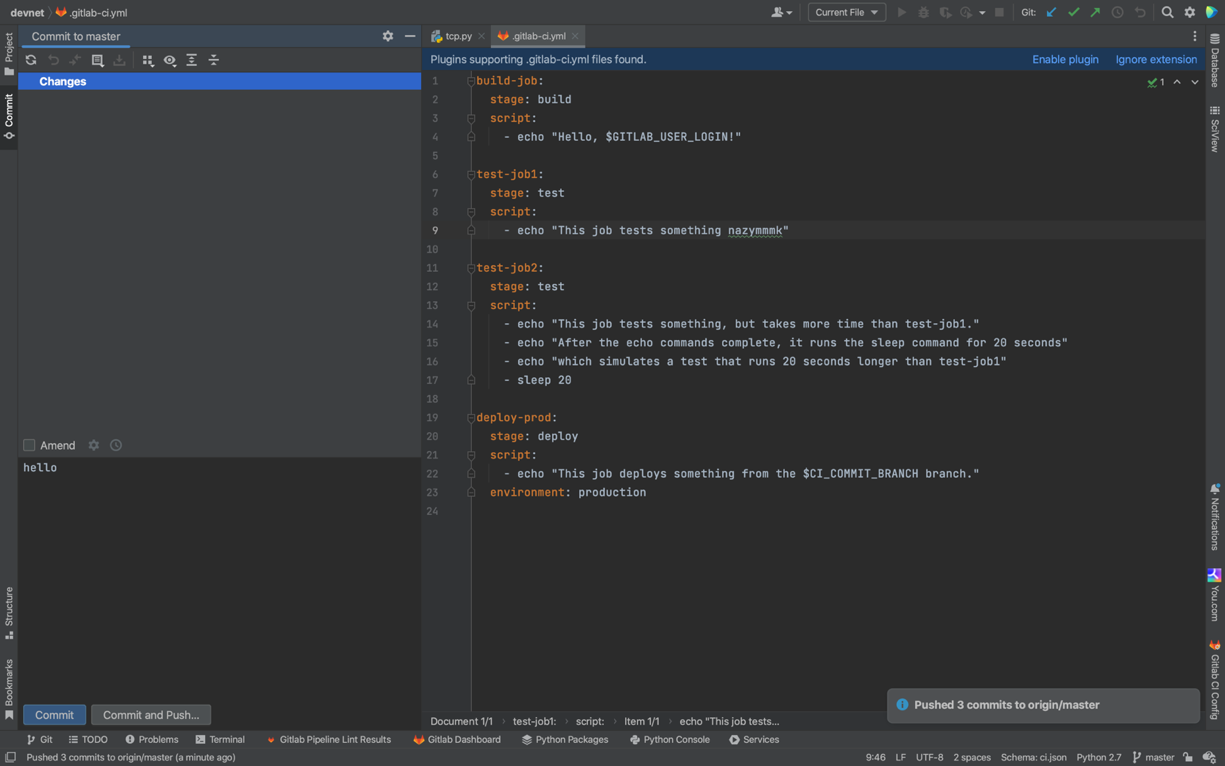


**3.CONNECTION WITH PYCHARM TEST**

the test added a pipeline to test the connection, while empty

since our code is written in python, we used pycharm, in this step I transferred it immediately from gitlab to pycharm

Made changes to the code and these changes came out in a previously created pipeline, with an attached message



**DOCKER, WEB DEVELOPMENT(Nikita)**

Our development team currently has 4 people(Nazym,Aidos,Nail,Nikita), each of whom should have an up-to-date local image of the site. Given the fact that we have people working on Windows, Linux, macOS, it usually takes a lot of time to help everyone figure out how to set up a web server and other non-obvious things from the field of working with the environment.

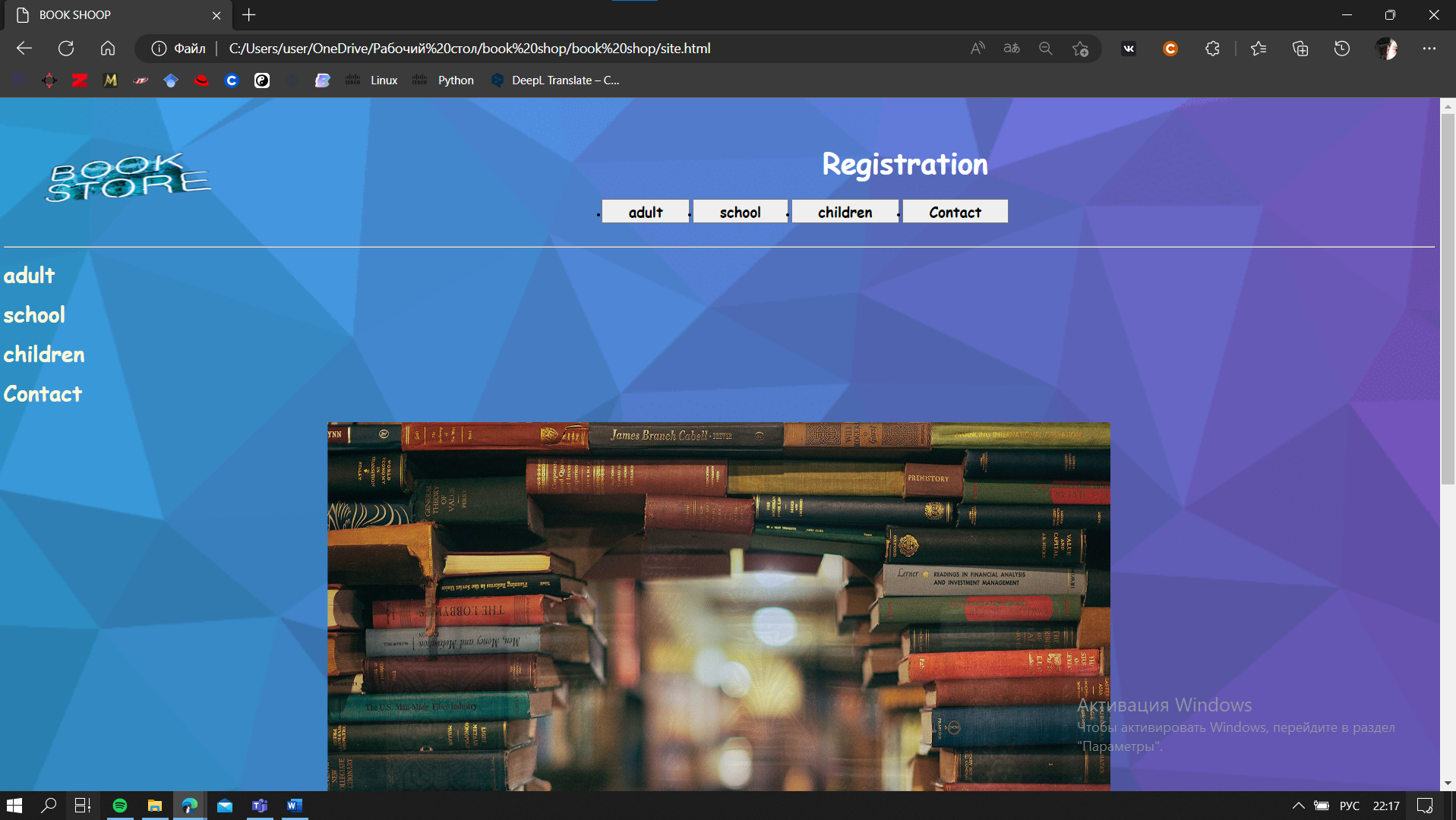
Docker is a tool that allows you to create processes isolated in separate containers, some kind of viral operating systems. Each process (php, nginx, mysql and others) runs in a separate environment, but is connected to other processes by means of the Docker itself.

Docker does all the steps for us to install packages, configure services and configuration files. In fact, he will automatically deploy the entire environment for us, which will be ready to work almost immediately

This is the first step towards automating our project

This website for an attack for TCP syn flood this attack is an attack that involves sending a large number of SYN requests in a fairly short time.

We made a website like a bookstore!



**Final Deliverables**

Reflect on the type of issues you have faced while working on this project. How did you find solutions and what have you learned?

the problems we encountered, the first is that we first chose another platform for implementing ci/cd, in github itself, github actions, but could not set up tests through this service in any way and switched to gitlab

Which requirements did we and our team achieve in implementation?

* Web App with Docker
* CI/CD completion
* Work as a team in GitHub

1. What development and testing tools did you use?

* DitHub
* Gitlab
* Diagrams.net

Strategy/Project Plan

Provide a brief description of your common team’s strategy for completing this project.

* Promote a respectful environment.
* Open the lines of communication.
* Articulate a clear, achievable vision.
* Define team roles.
* Trust in your team's efficacy.
* Lean on your team's strengths.
* Reward and acknowledge excellence.
* Create a safe space.

**Reflect on the type of issues you have faced while working on this project. How did you find solutions and what have you learned?**

Due to technical problems, atomization and adding CI/CD in the beginning, we used GITHUB actions, but without getting the necessary results, we switched to GitLab

also, since we had different operating systems with data transfer and work in general, we used Docker files for this

If we talk about soft problems, since we worked at different times, there were problems with determining which task was done or not, we solved this problem using CI/CD

Team activities and reflection

Deliverable/Rubric: Your manager is interested in knowing how everyone worked together as a team. Please do not forget to provide your evidence of team collaboration on github or similar system version tool.

Below is a list of additional questions from your manager:

**a. What did you enjoy about working as a team? What worked well?**

As a result, teamwork is a treasure trove of opportunities for me. I don't need to be in charge, I just need a chance to say a word of encouragement to someone else, a chance to help someone go further than they thought possible.

Teamwork is about collectively achieving more than we could ever achieve alone.

**b. What team problems did you encounter and how did you resolve them?**

* Ineffective work methods - we paid special attention to issues such as.
* What are the relationships within and outside the team;
* Is there a high degree of efficiency in the management of resources;
* Whether the process of decision analysis is well designed;
* How new tasks are evaluated;
* What are the criteria for measuring effectiveness.

**c. What technical problems did you encounterand how did you resolve them?**

We ran into a lot of problems such as not being able to install and download applications and files, problems in the code, and misunderstandings about our work, which led to a complete change in our work.

**d. How was each team member held accountable individually and for the team as a whole?**

Our team's communications. Their frequency, structure, and openness depended on the specific tasks and the specific team. If we needed to review some important procedures and make a decision, we did the routine work on short notice. If we needed to think about a situation, a meeting or an online meeting was enough.

**e. What was your team's decision-making process?**

The decision-making process was the empowered team's choice of the most rational effective of the alternative actions carried out within our authority and aimed at achieving the team's goals.

The team, made the decision and we had to focus on three tasks:

* conduct research on the problem;
* generate ideas to solve the problem;
* choose (make) a decision.

**f. Overall, how were the team dynamics and what were any lessons learned?**

Problem solving is a group, team effort. Where we learned a few things about teamwork.

* Take your time to understand the problem.
* Get to the root causes.
* Eliminate the causes.
* Record the results.