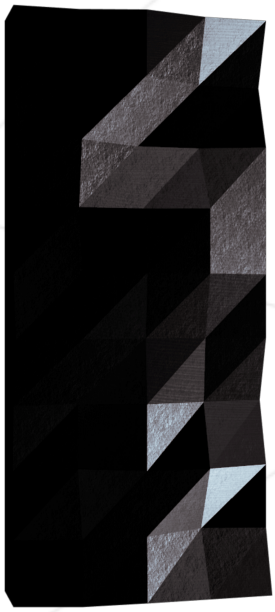


dopanic

Unfortunately the Magrathean whale was not alive,
now we are all in panic.

Summary: Getting crazy.

Version: 1.0

Contents

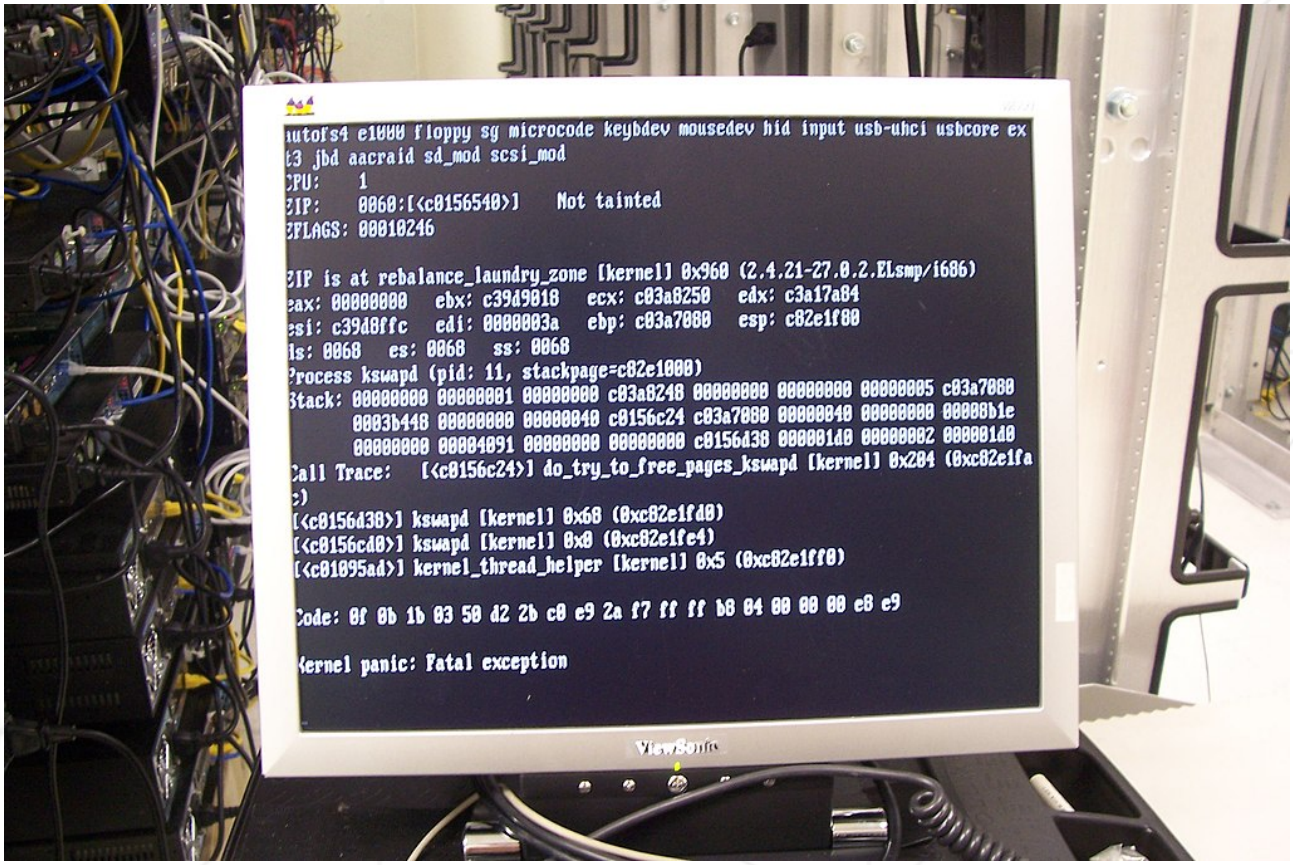
Chapter I

Foreword

Centaurs are half-human, half-horse creatures in Greek mythology. They have the body of a horse and the torso, head and arms of a man. They were considered to be the children of Ixion, king of the Lapiths, and Nephele, a cloud made in the image of Hera. According to a different myth, however, they were all born from the union of a single Centaurus with the Magnesian mares.

Chapter II

Foreword II



A kernel panic (sometimes abbreviated as KP) is a safety measure taken by an operating system's kernel upon detecting an internal fatal error in which either it is unable to safely recover or continuing to run the system would have a higher risk of major data loss. The term is largely specific to Unix and Unix-like systems. For Microsoft Windows operating systems the equivalent term is "Stop error", resulting in a bug check screen that presents the bug check code on a blue background in Windows (colloquially known as a "Blue Screen of Death" or BSoD), or on a green background on the Xbox One platform and some Windows Insider builds.

Chapter III

Introduction

If you got here it's because you learned a lot and managed to overcome many challenges. First of all, congratulations.

Now, you must have already noticed that throughout your game, you had to go through several challenges that cut several areas of technology that exist in the market today and even then there are still ideas left and you want to implement new things. An important thing to understand is that we can always implement new things, almost infinitely and without setting a limit and an objective and understanding why this is interesting to our player, we don't stop adding new things.

This project implements new things, looking to make it even cooler and add cool new technologies to your game, but it doesn't change it that much at the end of the day. Here you have the opportunity to learn that the possibilities are endless and you can achieve anything. Anyway, everything takes time and dedication to achieve and here you can expand your horizons in different ways.

Chapter IV

General instructions

- You can change your game in several ways from here, think about which one appeals to you the most. Either way, you need to say why you made the decisions you did.
- As always, new subjects will be introduced to you, don't be afraid to read a lot of documentation.
- The features that already exist in your project cannot be removed, you will only have to improve them by reimplementing them.
- You must ensure that your tests and CI/CD are working properly with the new changes introduced.

Chapter V

Mandatory part

V.1 Common features

- You should put NGINX[1] in front of all your services to serve them. That means you will have one more container in your game, just for NGINX. This container must be the only one exposed to the world and must send the necessary requests to the other applications through the internal Docker network. You can take the image of Dockerhub.

- NGINX should always redirect to port 443[2] to ensure TLS/SSL secure connection. You should also implement the routes needed for the other functionality below here.
- The NGINX container needs to ensure that the logs generated by NGINX are saved even if the container stops.
- NGINX logs will always be analyzed by GoAccess[3] and will be available to be accessed and analyzed at any time. GoAccess can be installed on 42 São Paulo's server and tested and evaluated from there.
- It's now official: chat with others in the community and let them play your game, ask for feedback and implement it if you feel it's appropriate. You need to collect at least one thing from other people and implement it.
- You need to configure Uptime Robot[4] to ensure your application is working 100% of the time and if not, receive an alert so you can take action.

[1] https://www.nginx.com/
[2] https://sectigostore.com/blog/port-443-everything-you-need-to-know-about-https-443/
[3] https://goaccess.io/
[4] https://uptimerobot.com/

V.2 Serverside features

- You must implement an authentication similar to the 42's API[5].
- Serverside must be public. Which means that other people can access it if they have access to a key to perform authentication.
- Likewise, the game must be playable also only through HTTP requests, which means that the serverside can work alone, even if there is no clientside. If someone else wants, you can use your API to make another clientside, just using your application.
- It may not be possible to do many requests per second on your serverside, you need to set a maximum limit for this. Whether the limit is per user, global or something else is up to you.
- You can check the hardware's ability to withstand DOS attack by running your serverside with tools like Bombardier[6].
- In NGINX, your serverside must be accessed via the '/api' route. Other routes must be concatenated into this same route, for example '/api/play'.
- You'll need to collect data on what your application is doing and store it in a database of your choice to be consumed by Metabase[7] (see below). See what is important to be stored and that has an important meaning to better understand how players play your game.

[5] https://api.intra.42.fr/apidoc
[6] https://github.com/codesenberg/bombardier
[7] https://www.metabase.com/

Chapter VI

Bonus part

For the bonus to be valid, the mandatory part must be perfect.

The software you used until then was installed by hand. and if you need to migrate server and install them again, would that be an easy task? Probably not.

For the bonus, you will have to write an Ansible playbook that will install everything on the server provided by 42 São Paulo necessary for your application to work, from installing software to ensuring everything is running as needed. This includes GitHub Actions, the command line software used, Docker and what you need for your application to run with just one command.

In short, you will write a playbook that will install all the software on a server with one command using Ansible. Your playbook file name should be ‘install.yaml‘ and you should know how to run it manually.

`You can use VirtualBox to test the playbook.`

Chapter VII

Turn-in and peer-evaluation

VII.1 Turn-in

Turn in your work on your repo Git. Only the work included on your repo will be reviewed during the evaluation.

VII.2 Peer-evaluation

Another group will evaluate your game and your code and will positively welcome the quality of it.