DP2 - SPRINT 4 REPORTS

GI - 01

Yoana Dimitrova Penkova

Manuel Cañizares Juan

Diāna Bukša

Iván Menacho Gallardo

Álvaro Rubia Tapia

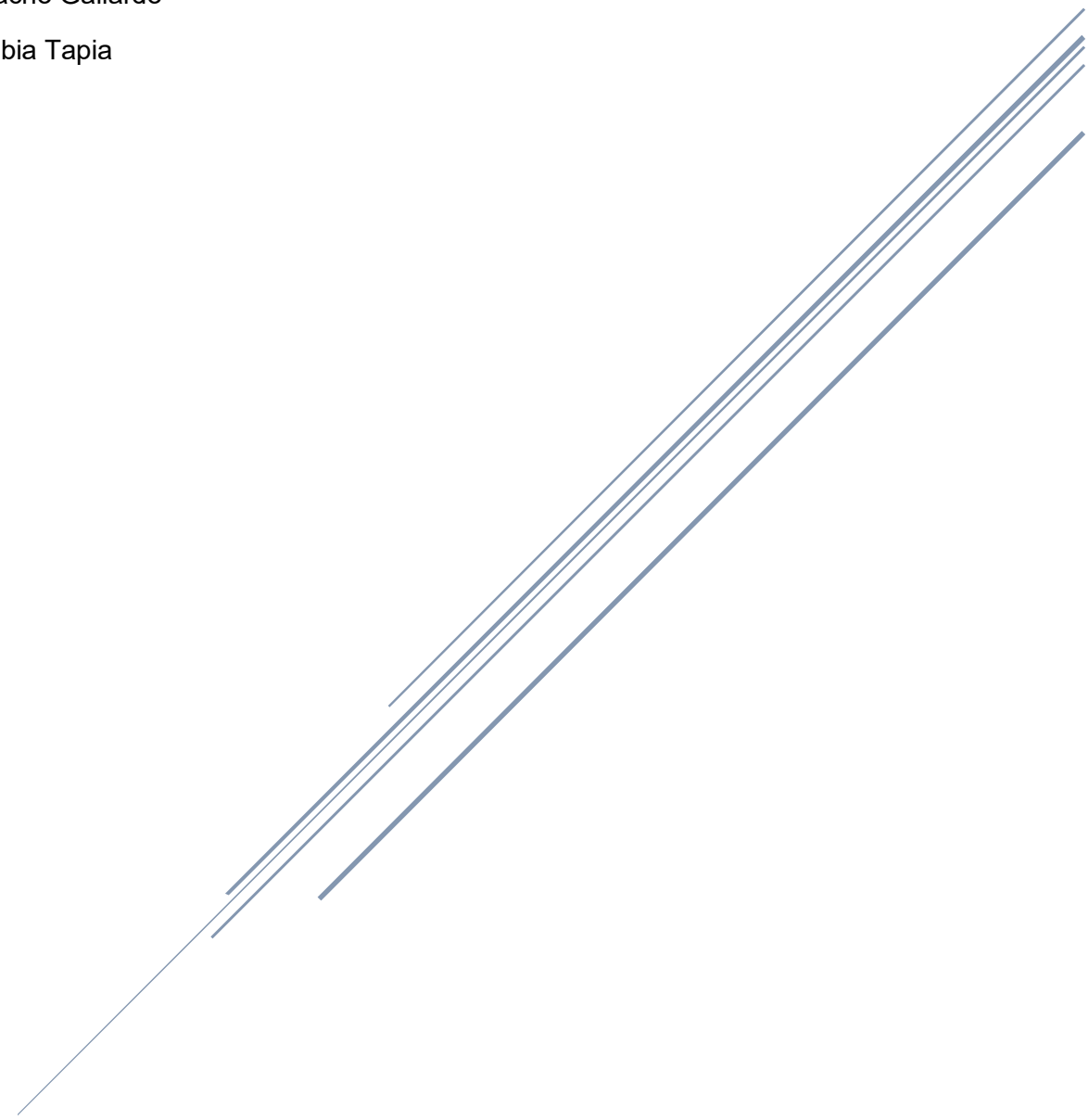

Table of Contents

Alvaro's report

For purpose of this deliverable, the task assigned were the next:

- Performance testing
- Profiling
- Refactoring
- Extra task with Gatling

The ask were performed in the order above, following the theory classes. Regarding the performance testing it not was difficult task not in complexity but in time consuming. This was due to the bad quality of my computer, so for the stress testing I took me a long time playing with the values until reaching an answer.

Once the performance testing was done, I proceeded to the profiling. My choice was to use visualvm instead of glowroot because the last one gave me an error I could not solve at the time and because of other subjects I did not have spare time. Visualvm gave me the necessary data to work with so it was not a disadvantage.

The extra task was not a challenge at all due to one of our teammate tha finish it sooner than she needed so we had a template to work with.

The performing was done extremely fast because for the subject PSG2 I had been in charge for the same issue and I was already familiar with SonarCloud and the methodology I had to follow.

Last, I was in charge of studying all the refactoring reports and merging them into one for the final deliverable. It was time consuming because of the different format each member had in their reports.

DESCRIPTION	HOURS
Performance	10h 41m
Profiling	4h 17m
Refactoring	2h 07m
A+ with Gatling	1h 22m
My part in this report	47m
Refactoring report for the final delivery	3h 10m
TOTAL	**22h 24m**

Yoana's report

In this deliverable the majority of the tasks were using external tools to measure our system. Those tasks were the following:

- Performance testing
- Profiling
- Refactoring
- Extra task with Gatling

Regarding the performance it was very intuitive to work with Gatling and all the explanations provided by the teachers. It was interesting to see which were the limits of our system taking into account the hardware and software I was using to test.

Concerning profiling, I chose the user story that had the worst performance result and it was US - 011 Homeless Pet Management. Giving the initial results Glowroot gave me, after some refinements of code such as using the lazy fetch and establishing a few cache files the performance was much more adequate. I'll be commenting about the profiling a little in the extra paragraph.

As for the refactoring we used SonarCloud to see what was the state of our project in terms of code smells, vulnerabilities and issues. From the explanations SonarCloud provided I was able to reduce the code smells by around 150 in only one refactoring session and that, in my opinion, is a very good outcome.

And finally with respect to the extra task, we chose the one about the feeders with Gatling. It consisted of making the inserting part of the scenarios (if you had any) more real because the scala file would be taking values from a very long csv file with random generated values. I didn't have any issues with it.

In general, I think this deliverable had less workload but it is very important to perform each one of these tasks in a real environment as well. The effort I've put into the sprint is specified in the following table:

DESCRIPTION	HOURS
Performance	9h 7m
Profiling	4h 4m
Refactoring	3h 59m
A+ with Gatling	1h 37m
My part in this report	23m
GI - 01 report for the final delivery	1h 45m
TOTAL	**20h 55m**

Ivan's and Manuel's reports

Although this deliverable we had completely independent tasks, we keep working together because it was easier for both of us.

This deliverable was defined by the use of external applications like Gatling, Mockaroo, GrowRoot and SonarCloud. Because of that, we spend a considerably high amount of time investigating the different tools we needed to use. Also a difference between this deliverable and the previous one was the establishment of deadlines, so we could have a more consistent workflow.

We started with the performance testing, at first it was very confusing, even after having all videos seen. The main problem we both have was that we didn't fully understand how to search the correct thresholds for Load and Stress test. As we had a deadline, we developed a sketch of the document. After discussion with the rest of the group and having a confirmation by the teacher, we were fix our errors, although having to sacrifice more time.

Concerning profiling, we spend a lot of time searching documentation, but the testing itself wasn't neither difficult nor time consuming. Ivan selected the US-009 Vet Updates Medical Record, whereas Manuel chose US-001.

- Ivan, after seeing the results of Glowroot, localized the huge amount of query calls as the main problem for the performance. He decided to use lazy fetch and create cache for medical record to reduce the amount of queries. While the time some queries took increased, the overall performance of the page improved drastically.
- Manuel, using Glowroot too, found that during the creation of a medicine, there were plenty of queries regarding an entity as a field of the medicine entity. First, he changed the fetch type but it was not enough, so he cached this secondary entity and the number of queries of this lessened considerably.

Regarding refactorization, both of us found very little code smells in own classes, so we decided to work a little bit differently.

- Ivan, as he only had very unremarkable problem, decided to search in the main classes and try to reduce as many code smells as possible. In the end, He reduced the code smells by 6.
- Manuel had a few of smells regarding his code, so he decided to improve the Petclinic base code and reduced 40 code smells.

Finally, regarding the extra task, the group decided to do it about feeders with gatling. It was fairly easy, and we didn't had any significant problem with them. Ivan choose to implement US-003 with feeders and Manuel chose US-001

The last remark is about the reports. As we had plenty of reports between performance, refactorization, profiling and the extra task, we decided to divide it so one person will be in charge of putting together one document.

- Ivan was in charge of the performance reports, whilst it was not difficult at all, was a very long and tedious job. That was because of the different formats used by each member of the group, and the necessity of creating a cohesive document out of very different analysis.
- Manuel was in charge of the profiling reports merging, a task a little bit tedious due to finding a common format between all the documents, but actually sufficient easy.

In general, we think we did a good job The only notable problem we had were the deadlines.. While the concept of having deadlines within the group would seem as a good idea (And we both think it is), the deadlines were too close together and we had a huge amount of things to do apart from this subject, which results in “Crunchs” and having to redo some documents at the end of the deliverable.
Tables representing time spent on each of the tasks within this sprint:

Iván’s effort:

TASK DESCRIPTION	**HOURS SPENT**
Ivan’s Performance testing	13h 24min
Iván’s Profiling	5h 12min
Iván’s Refactoring	54min
A+ with Gatling	1h 20min
Fixing errors	1h 43min
Performance Testing Report for the final delivery	5h 56min
Writing this report	50m
TOTAL	**29h 16min**

Manuel’s effort:

TASK DESCRIPTION	**HOURS SPENT**
Manuel’s Performance testing	14h 13min
Manuel’s Profiling	5h 39min
Manuel’s Refactoring	2h 44min
A+ with Gatling	1h 40min
Profiling Report for the final delivery	2h 51min
Writing this report	50m
TOTAL	**27h 57min**

Diana's report

As for this deliverable, the coding part that refers to implementing user stories had already been finished, so now all focus was on the performance testing, profiling, refactoring and A+ task. Performance testing definitely took the biggest amount of the time, but other than that I did not encounter any other problems with that. At first, when trying stress testing, I had chosen very big number of users, that were trying to perform user stories at once, to test what is the maximum capacity of our system, therefore that was why it took that much time, since I reduced this number slowly, until I reached the border where system is no longer crashing (is not showing errors) and can actually proceed with the given number of users. At first I could not understand why these numbers differ from user stories, but then, after checking Task Manager I realized that some other programs were open and running, that consumed both CPU and memory, so after ending these programs, results were more even.

The profiling went good, except that at first I had a bit of connection problems with Glowroot, but with the help of my colleagues, they were solved quickly and everything worked just fine. I think that in general, profiling could be one of the simplest ways how to improve website's performance, since it is easy to run it (as, for example, Glowroot), but at the same time it shows a lot of information about the performance of the website, for each step, so it is quite easy to distinguish where is the problem, which action is taking quite a time to load etc, therefore to focus on that one thing.

As about the refactoring, we used SonarCloud, what was also presented in the lectures. In general, the biggest part of the bug smells were quite simple, as for example, removing unused parts of the code, including imports and comments or defining a variable instead of declaring it every time we use it. After we all were done with refactoring, it was reduced by more than 60%, so I think that we did a good job in that. Table representing time spent on each of the tasks within this sprint:

DESCRIPTION	HOURS
Performance testing	14h 20 min
Profiling	2h
Refactoring	2h 10 min
A+ with Gatling	30 min
Writing this report	25 m
Merging of all reports	1h 20 min
TOTAL	**20h 45min**