

2019

# PROJECT REPORT : JAVA



Group 9

CESI.eXia - Lingolsheim

03/06/2019

# SUMMARY

---

<b>SUMMARY .....</b>	<b>1</b>
INTRODUCTION .....	2
<b>I.    SUBJECT ANALYSIS &amp; SCHEDULE SETUP .....</b>	<b>2</b>
<i>a.    Subject analysis.....</i>	<i>2</i>
<i>b.    Schedule setup.....</i>	<i>3</i>
<b>II.   UML MODELING.....</b>	<b>4</b>
<b>III.  DATABASE CREATION .....</b>	<b>5</b>
<i>a.    The database connection.....</i>	<i>5</i>
<i>b.    The database overview.....</i>	<i>5</i>
<b>IV.  PROGRAMMING AND RESULTS .....</b>	<b>6</b>
<b>V.   CONCLUSION .....</b>	<b>7</b>

## Introduction

During this project, we had to produce a lot of diagrams, programs and some other reports linked with the rest of the work we had to do.

We, at first, structured our way of seeing the project in order to be the fastest possible but also the most efficient. We weren't, because of our marks, a very good group in java but we managed to almost finish the project on time.

We're going to introduce you the timeline of the project and what we've done until the deadline.

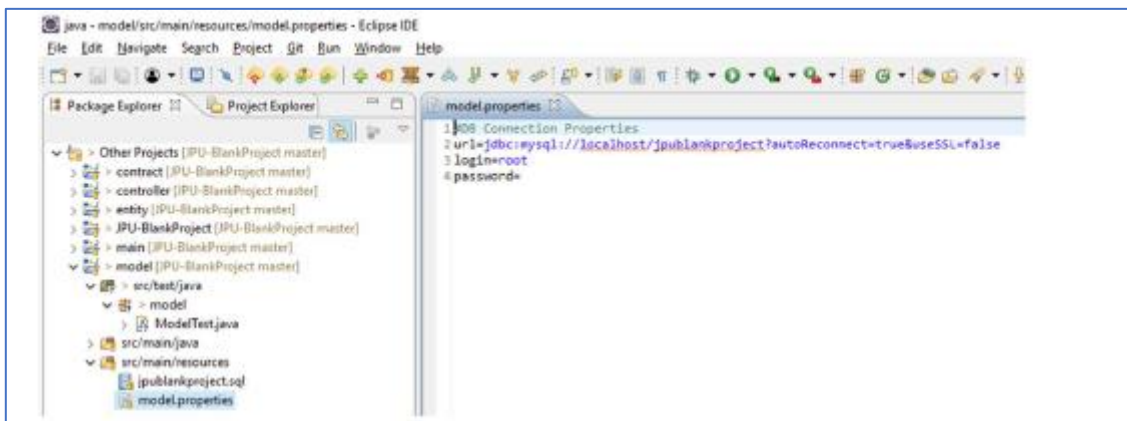
## I. Subject analysis & Schedule setup

### a. Subject analysis

We're beginning with this part because it is the first step of every project we've made so far. We work as a team and not alone so we need to discuss a lot about the project and what it is asked to us in the end, in another words, the project objectives.

At first, we read the subject and brainstormed together to get some different insights about the project.

We gathered some information, files and other things. For example, they gave us a basis to work with in the project. We followed what they asked us to do with the project (essentially with the maven configuration).



The screenshot shows the Eclipse IDE interface. On the left, the Package Explorer displays a project structure for 'JPU-BlankProject master'. The 'model' package is expanded, showing 'src/test/java' and 'src/main/java' sub-packages. The 'src/main/resources' package is also visible, containing 'ipubblankproject.sql' and 'model.properties'. On the right, the 'model.properties' file is open, showing a JDBC connection string: 'url=jdbc:mysql://localhost/jpubblankproject?autoReconnect=true&useSSL=false', 'login=root', and 'password='.

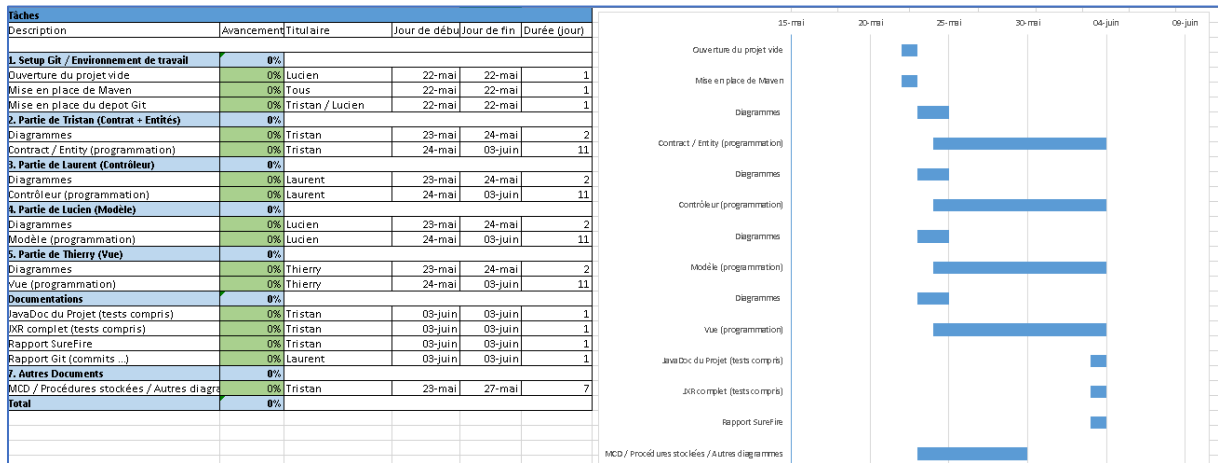
Lancez ensuite un :

- update project
- build compile
- build package
- build install
- build site
- build site:site
- build site:stage

## OOP&UML PROJECT – GROUP 9 - BOULDERDASH

### b. Schedule setup

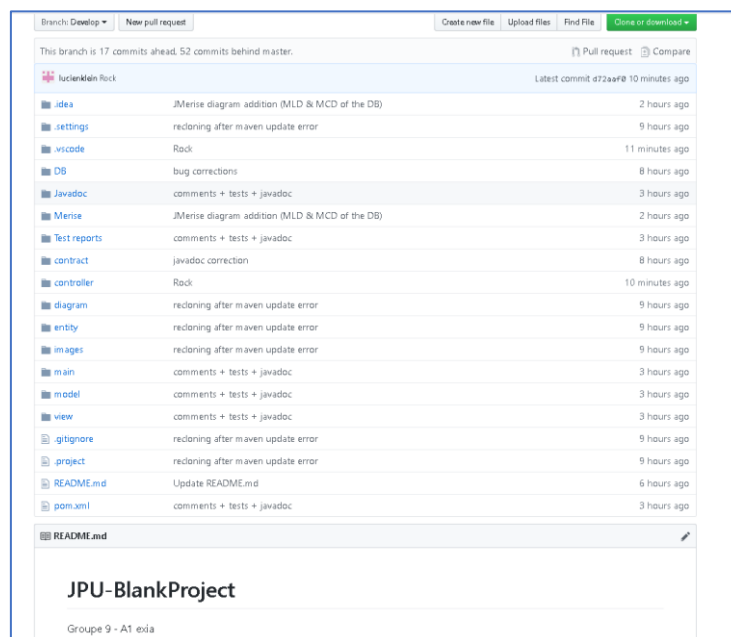
And then we had to share the tasks equally within the members of the team. We chose to do a Gant diagram to show the timelines and the deadlines for each task.



After we finished this task, we directly began to work in order to lose the less time possible.

We said before that we were working as team, and to have each member to work on his computer and have every time the last version of it, we were told to use GitHub, a version control tool (git is the software). We created then the master branch and we made a little mistake because we began to code on this branch, which is highly not recommended. We created sometimes later another branch; the branch develops to contain all the work in progress programming.

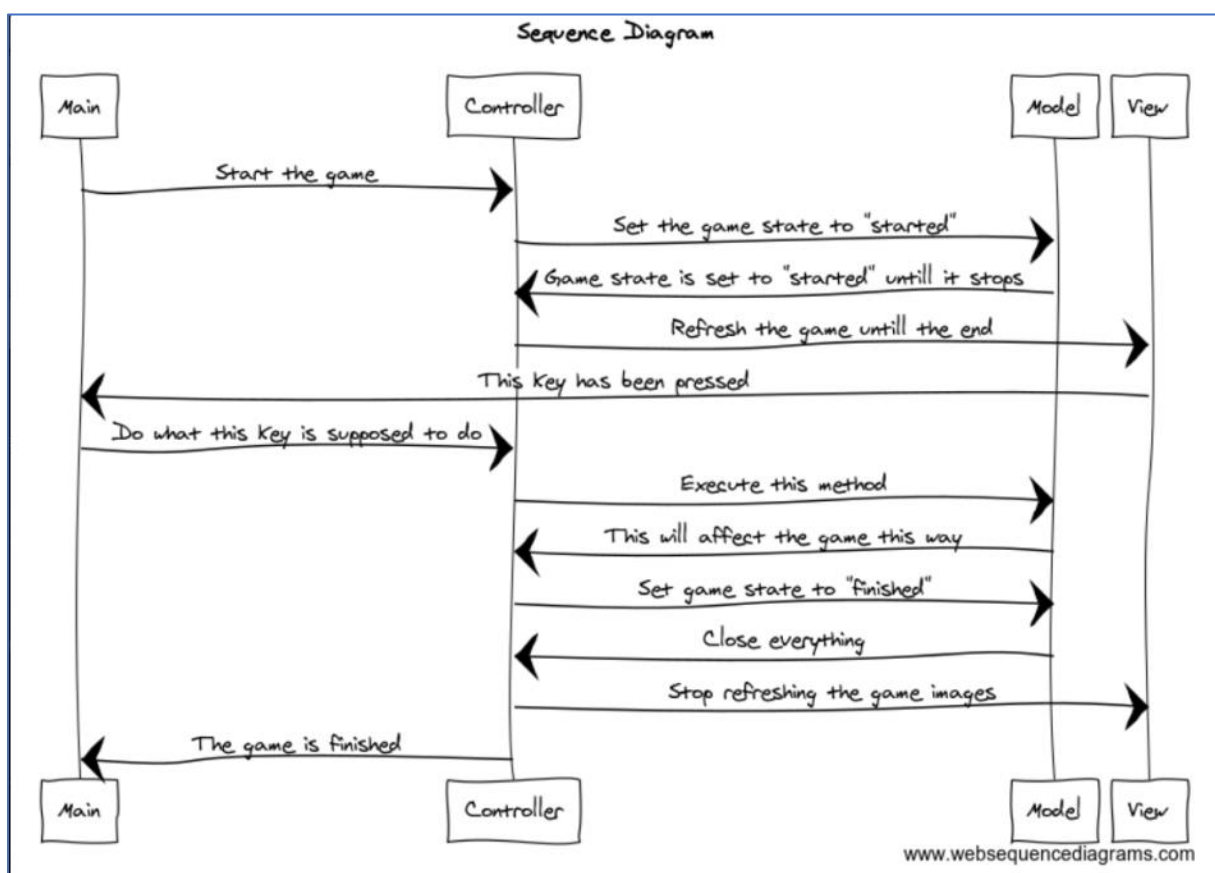
With GitHub we were able to have the last version of the project, even if someone modified the code.



## II. UML Modeling

We had to create diagrams (class, package, sequence and components) to have an overview of what our project was going to look in the end. This part was made even before beginning to program because it shows the structure of the project. You cannot begin something without having a basis. So we created these diagrams to be able to program in the best conditions possible.

Here is an example of the diagrams we had to do.



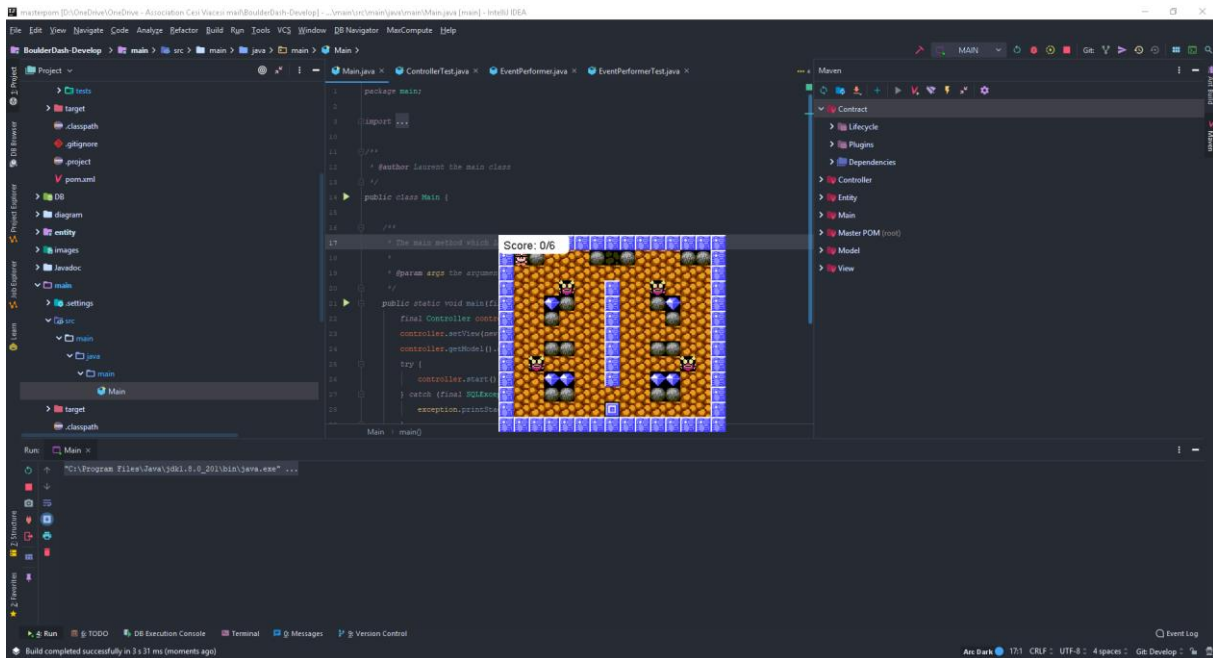
This diagram is the sequence diagram, it shows what our program is going to do. But this one was not the first one we've made, we modified, upgraded it during the project because of the changes we did.

All the different diagrams we had to do are available in the vpp folder in the GitHub repository.



## IV. Programming and results

We began to program the boulderdash game with creating the classes, the interfaces and everything we might need to do our game (everything is detailed in the Javadoc) and we obtained this in the end.



We created our own maps and in them, monsters were implemented, players, diamonds or boulders for example. And our character was digging the dirt to get all the diamonds and get to the exit to win. But if a boulder fell on his head, or he entered in contact with a monster, he would die. We could have copied the original game's maps, but we preferred to create ours.

We are not developing this part so much because everything is commented in the Javadoc (see the GitHub repository).

## V. Conclusion

---

### **Tristan:**

I do not like so much the java language, practical wise only, because I understood every theoretical part. But I had a lot of difficulties with programming even if I liked the OOP principle. I still managed to do it during the project because I was able to help myself with the workshops we've made. And even if I missed completely my test, I had a group with hard workers and the dynamic was very nice in the group. So even if we didn't manage to have a complete game, we've made something nice and I'm proud of me and the team.

### **Laurent:**

For me, this project has been one of the best, my group was active and dynamic, and everyone was working. Also, as I was the project manager for the first time, I learnt a lot about the fact of guiding a group to an common objective and the project itself was for me a little challenge that I enjoyed a lot trying to overcome

### **Thierry:**

All the group was very active. Even if we had some individuals difficult, the group was also helpful. The project itself was very interesting, the game conception process is something complete and learning friendly.

### **Lucien:**

This project allowed me to better understand the object-oriented programming and thus to acquire new competence in Java.

I quite quickly understood the subject, unlike the BDD project, but imagine the programs in the whole was rather complicated.

One of the biggest difficulties was working in a group and respecting the MCV design pattern. We had a lot of trouble working together, indeed we leave each one of his side without really knowing what others were doing. And everyone else was leaving us a little in all directions, which had the end to have a project incomprehensible and computable. So personally, I left our old programs in Java, dogfighter and insane vehicles, to be able to create the structure of our program. But it was not the best of these solutions, indeed even adapting to the maximum, I could not have a good structure. And like everyone else, so I started all over again to have a better structure.

I think that on the organization of the first part of the project, we could have done a lot better. But we were able to raise our heads and realize that we are going in all directions, and so we were not afraid to start all over again and work even harder to finish the project on time.

This project was a great experience and taught me a lot of things both personally and programmatically.