Report

[INSERT NAME]

[INSERT NAME]	Version: 0.1
Report	Last modification: 26/05/19

Introduction

This report is the fourth document realized for this project. Its goal is to describe the context in which the project was carried out and the results obtained regarding the game to realize.

It has to present the different functionalities that the game contains in its final state. It should establish a description of implemented functionalities by the development team regarding priorities established at the beginning and according to the initial requests. This document also provides an overview of the project's problems and the skills acquired during the realization of this project.

I. Presentation of the project

A. Context

As part of the program of EFREI PARIS (École Française en Électronique et Informatique), and as part of our second year in the undergraduate program, students of the international session had the choice to working either has a four or five EFREI student group, or collaborate with Coventry students on a transversal project with the purpose of solving a social or environmental problem.
In our case, we chose to work with Coventry students. The team is composed of three EFREI students as well as three Coventry students.
The team decided to implement a gamified learning experience.

The project ends at the end of the fourth semester. An oral presentation is asked.

B. Analysis of what already exists

As this project is a game, we wrote it from scratch in C++. We however used a set of tools to help us in the process of creating this game. We used a game creation library called raylib, as well as a header only implementation of ECS called EnTT. We also used a library to take care of the storage and retrieval of data as a json format. We drew inspiration from our previous projects for this one.

C. Organisation and contributors

Vincent Mouillon – Team Leader and Lead Programmer

Jasper Dahil – Lead Designer
Ibukunoluwa Adegoke – Designer

Victor Tailleu – Programmer
Luca Vaio – Programmer
Vincent Lim – Programmer
Jesse Prescot – Programmer

D. Objective and general goal

The objective of the project is to realize a game that will teach the user how to code. This is a story-based game and the user unlocks the different lessons as he progresses in the game.

The objective of the project is to realize a game that teaches the user how to code and use common tools so that he becomes independent.

To do this, we choose to make a story game on data structures and other computer science related topics like variables, functions, the different major architectures, design patterns etc...

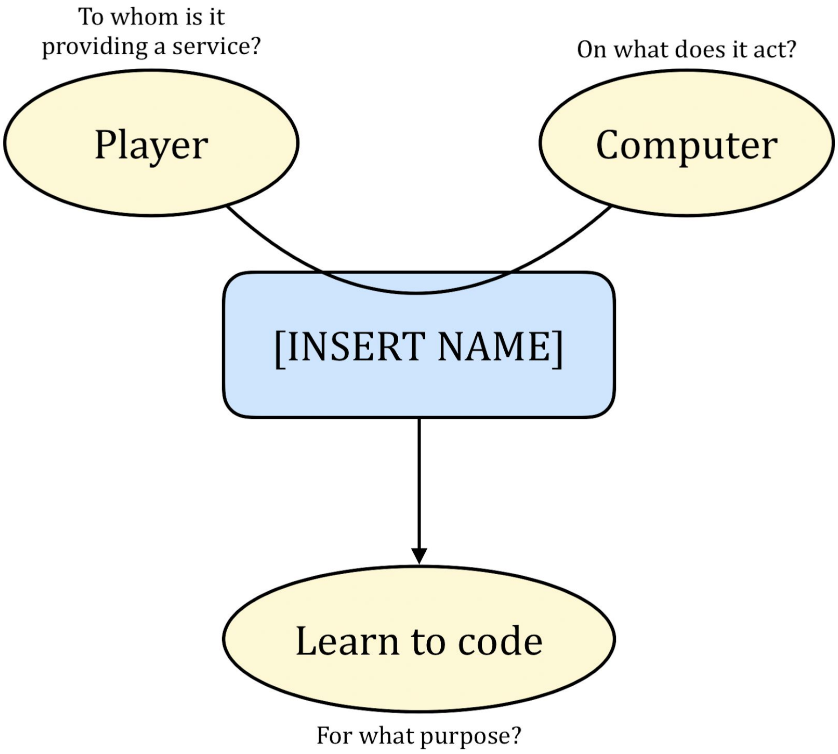
The user will have to retain the information. We will thus make him practice his reasoning skills and programming knowledge. To do so, we need to implement an editor inside our application with puzzles that the user will have to solve. We will consequently check his program with test functions. The user will be able to run his program.
if the user succeed in completing the puzzle, he will go to the next lesson of story based mission.

But why making a story out of it ?
We really think that making something involving helps in learning and retaining concepts. Making it really concrete and having tangible consequences to your actions helps remembering what you did.

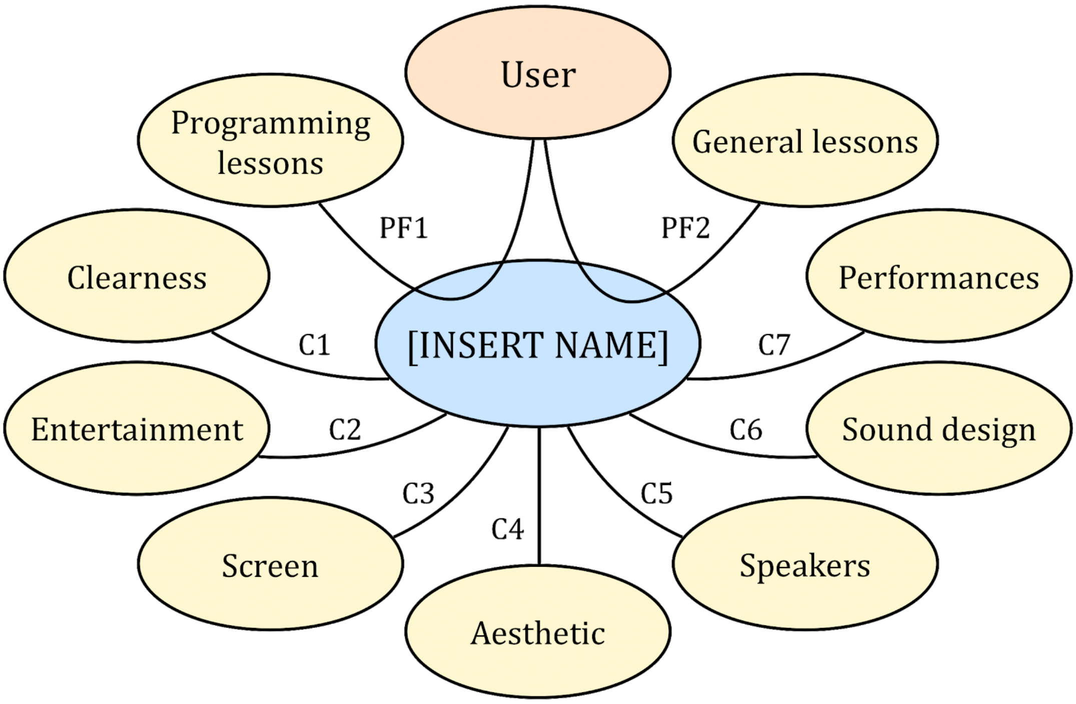
The code also needs to be flexible so that the addition of new lessons or new gameplay won’t cause important modification of the code.

II. Functional description

A. Bull chart



B. Octopus diagram



C. Table of functionalities

Name	Functions	Criteria	Implementation
PF1	Teach the user how to code	- Teach a programming language syntax - Make an editor - Test the user's code or knowledge - Teach the user data structures - Teach the user major program architectures	- Teach C - Be able to write in a text file or drag and dropping the file the user created - Test functions for each data structures - Teach: - Variables - Pointers - Arrays - Multidimensional arrays - SLLs (singly linked list) - DLLs (doubly linked list) - Binary Trees - Balanced Binary Trees - Search Binary Trees - AVLs - Trees with an "infinite" number of children (acyclic directed graph) - Graphs - Teach: - OOP - ECS - Functional Programming - Programming patterns
PF2	Make the user independant	- Present the user the major editors, IDEs, languages and tools	- Initiate to Emacs, Vim, Sublime, Atom, VS code - Initiate to Visual Studio, Code::Blocks, Clion - C/C++, Python, Java, Haskell - gcc, gdb, valgrind, make, cmake, raylib, entt, SDL
C1	The actions proposed to the player has to be clear and coherent	- Have a good tutorial - The lessons need to be really clear - Help menu	- Having a help and hint menu for any mission - Having an intuitive AI
C2	The actions proposed to the player	- The lessons must be playful and enjoyable for the user	- Develop at least 1 story driven missions at the beginning of each world

	has to be enjoyable and entertaining	- There must be a story to keep the players attention	
C3	Display the game to the screen correctly	- No display bug - Screen option menu - Support common screen resolutions	- Two modes: windowed and fullscreen - Support 720p 1080p
C4	Has to be visually attractive	- Good graphics - Interesting Visual design - Clear UI/UX	- Unifying style - 1 texture for each data structure - 1 background for each world
C5	Playing music and effect correctly through the speakers	- No sound bug - Sound option menu	- Sound and music options
C6	Has to be auditively pleasing	- Good sound effects - Good ambiance music	- 2/3 ambiant musics - at least 10 sound effects
C7	Has to perform well	- The game must be playable	- Always more than 60 frames per seconds
C8	Adapt to the starting knowledge of the player	- have an option to skip certain steps without skipping the story	- A test step at the end of each world accessible from the point of entering the world

III. Priorities

The C1 constraint was addressed immediately as it has to be addressed at the design stage. The C3 constraint should be kept in mind during the developpement time of the project. The several modes of rendering shouldn’t be a priority at first. The PF1 function is separated into multiple parts for the ease of implementation. The lessons shall be implemented in the correct order. The C2 class shouldn’t be forgotten at the design stage as it will be impacting on the all game.

IV. Progress

Function to implement	**Contributors**	**What was done**	**Not done**
PF1 Teach the user	Vincent Mouillon Vincent Lim	Beginning of the scripts	Finish all scripts
PF2 make the user independent	Vincent Mouillon Vincent Lim	Beginning of the scripts	Finish all scripts
C1 Coherent experience	Vincent Mouillon Vincent Lim	Design the menus	Design the missions
C2 Enjoyable experience	Vincent Mouillon Vincent Lim	Involve the user into the teaching experience	Verify that the lessons aren't repetitive and are entertaining
C3 Display to screen	Vincent Mouillon Vincent Lim Victor Taillieu Luca Vaio Ibukunoluwa Adegoke Jesse Prescot	Implement each screen with raylib	Video option menu
C4 Attractive look	Jasper Dahil Ibukunoluwa Adegoke	Design the intro screen and animations Create assets for the game such as characters, buttons, backgrounds	Finish the design of all different screens
C5 Sound	Vincent Mouillon Vincent Lim Victor Taillieu Luca Vaio Ibukunoluwa Adegoke Jesse Prescot	Beginning of sound implementation	Implement sound in each screen Adding a sound option menu
C6 Attractive sound	Vincent Mouillon Vincent Lim Luca Vaio	Create sound assets for the game (music and FX)	Implement sounds in the game
C7 Performance	Vincent Mouillon Vincent Lim Victor Taillieu	Good architecture (ECS design and graphical library Raylib)	Nothing

	Luca Vaio Ibukunoluwa Adegoke Jesse Prescot	Well optimized screens	
C8 Adapt to player	Vincent Mouillon Vincent Lim	Designed level select to choose what to learn (skip unnecessary lessons)	Design the tests of each world

V. Description of problems

There is multiple problems that were brought up by this project:

- How to keep the player interested during the all experience:
    - The experience shall be compelling and fun but this is really hard to do for lessons.
    - The story should be emotionally realistic which is really difficult to capture.
    - The player should have goals.
- We didn’t have the time to do a text editor in the program. So we implemented drag and drop.
- It is difficult to implement fun and interactive lessons for multiple data structures without doing a tutorial with too much text each lesson.
- It was difficult and long to make coherent and good looking assets considering the time limit.
- The separation of the work was a big issue. As we were six people from two different country, the organization was tough. Moreover, we didn’t have the same deadline for the project so we were fewer at the end.

VI. Assets and skills acquired

A. Knowledge

The coventry students did a course on data structures and on C++. As EFREI students, we were taught the C++ language and Object Oriented Programming during the semester. It was helpful for the realization of the project.
The team learned how to use Raylib as well as EnTT to be able to work on this project. They also trained on OOP and data structures as well as ECS design.
The team learned to work together also we are not in the same country. It was quite difficult to organize us and to cooperate efficiently to finish the project in time.

B. Ressources

We didn’t asked for a budget because until now, it wasn’t needed. We may need some in the future if we want to distribute the game by hosting the installer and/or embed it on a website.

VII. Glossary