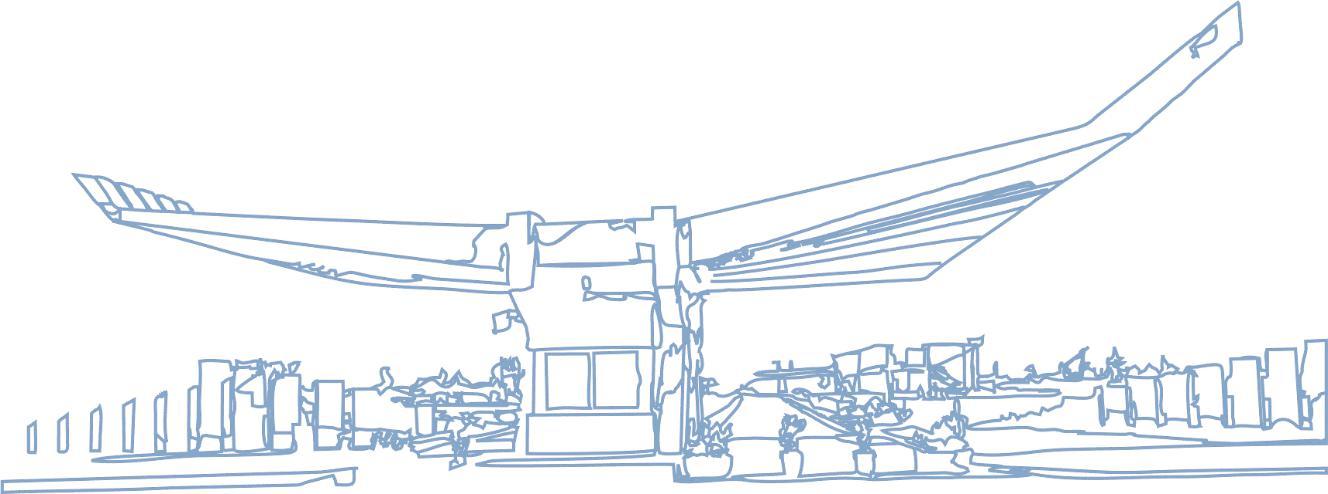
`

***theDOT Requirements Specification***

**INTRODUCTION TO SOFTWARE ENGINEERING**

**theDOT**



**Team Members:**

**Greisi Jaho**

**Era Mulla**

**Franceska Rrokaj**

**Kevin Hyska**

**Klajdi Shehu**

**Juljan Losha**



***theDOT Requirements Specification***

|  |  |  | **Table of Contents** |  |
| --- | --- | --- | --- | --- |
| [**1.**](#bookmark=id.1fob9te) | [**EXECUTIVE SUMMARY**](#bookmark=id.1fob9te) | | | **2** |
|  | [1.1](#bookmark=id.1fob9te) | [PROJECT OVERVIEW](#bookmark=id.1fob9te) | | 2 |
|  | [1.2](#bookmark=id.3znysh7) | [PURPOSE AND SCOPE OF THIS SPECIFICATION](#bookmark=id.3znysh7) | | 3 |
| [**2.**](#bookmark=id.3znysh7) | [**PRODUCT/SERVICE DESCRIPTION**](#bookmark=id.3znysh7) | | | **3** |
|  | 2.1 | PRODUCT CONTEXT | | 3 |
|  | [2.2](#bookmark=id.3znysh7) | [USER CHARACTERISTICS](#bookmark=id.3znysh7) | | 3 |
|  | [2.3](#bookmark=id.3znysh7) | [ASSUMPTIONS](#bookmark=id.3znysh7) | | 4 |
|  | [2.4](#bookmark=id.3znysh7) | [CONSTRAINT](#bookmark=id.3znysh7)S | | 6 |
|  | [2.5](#bookmark=id.2et92p0) | [DEPENDENCIES](#bookmark=id.2et92p0) | | 15 |
| [**3.**](#bookmark=id.2et92p0) |  | | **Risk Management + Requirements** | **7/ 16** |
|  | 3.1 | FUNCTIONAL REQUIREMENTS | | 17 |
|  | [3.2](#bookmark=id.3dy6vkm) | [NON-FUNCTIONAL REQUIREMENTS](#bookmark=id.3dy6vkm) | | 18 |
|  | [*3.2.1*](#bookmark=id.3dy6vkm) | | [*User Interface Requirements*](#bookmark=id.3dy6vkm) | *19* |
|  | [*3.2.2*](#bookmark=id.3dy6vkm) | | [*Usability*](#bookmark=id.3dy6vkm) | 20 |
|  | [*3.2.3*](#bookmark=id.3dy6vkm) | | [*Performance*](#bookmark=id.3dy6vkm) | 21 |
|  | *3.2.4* | | *Manageability/Maintainability* | 22 |
|  | *3.2.5* | | *Security* | 25 |
|  |  | |  |  |
|  | *3.2.7* | | *Other Non-Functional Requirements* | 26 |
|  | [3.3](#bookmark=id.1t3h5sf) | [DOMAIN REQUIREMENTS](#bookmark=id.1t3h5sf) | | 27 |
| 4. | **DESIGN THINKING METHODOLOGIES** | | | 28 |
|  | 4.1 | Negotiation | | 28 |
|  | 4.2 | Empathy | | 28 |
|  | 4.3 | Noticing | | 28 |
|  | 4.4 | GUI | | 29/30 |
| **5.** | [**SOFTWARE DESIGN**](#bookmark=id.4d34og8) | | | 31 |
|  | 5.1 | Use Case | | 31 |
|  | 5.2 | State Diagram | | 32 |
|  | 5.3 | Class Diagram | | 33 |
|  | | | |  |
|  |  | | |  |
|  |  | | |  |
|  |  | | |  |



Page 1



***theDOT Requirements Specification***

1. Executive Summary

***1.1*** ***Project Overview***

theDOT game is an educational coding game offering a kid-friendly introduction to programming concepts and problem solving.The idea is that kids from 3-6 years old can learn programming concepts while playing the game and having fun.

The game consists user using arrows as commands showing the fuzz (the character) which way to roll to get to the end of the maze.The game is self-guided and features a voiceover that prompts kids when a new concept, like loops, is introduced.

The intended Audience are kindergarten kids 3 to 6 years old.

***1.2*** ***Purpose and Scope of this Specification***

**In scope:**

**1.Project purpose statement**

We have created this game to make programming a part of children’s life. The main reason of

building a game like theDOT is to create a funny and educational environment for the children, without loosing their focus and making them ready for the future life.

**2. Background**

Firstly we created a very hardworking and faithful group. We build a schedule for meetings and respected them. The first task for all stakeholders was to make interviews with different targetgroups related to children such as: children, teachers, parents, siblings, cousins and friends. After making these meeting we came up with an idea for the game. So we build up the context of the game and then sketch a few parts of it. We created a main character which was round, and named the game “theDOT ''. After every meeting we concluded the entire idea of the game and all of us created a github account and started coding. We have changed a lot in this game every week, because we wanted to create a very simple, funny and educative work. The golden thing was adding the question task in the game. As we were working on codding the game, we were working on writing this documentation and presentation of the entire project.

**3. Objectives**

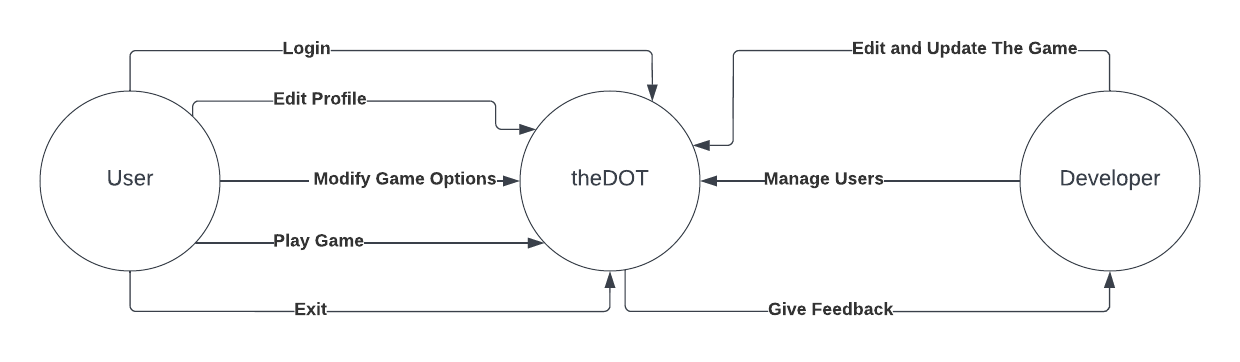
* To create a funny, creative, educational game.
* To build a helpful game for every child.
* To create a strong and manageable group.
* To be specific in tasks and requirements of the project.
* To be hardworking and responsible for every part of the project.
* To respect the meeting schedule and to be flexible.
* To enjoy this grateful time with the team.

**4.Key stakeholders and other players**

Th e stakeholders for this project are: Greisi Jaho (leader of the group), Era Mulla, Juljan Losha, Franceska Rrokaj, Klajdi Shehu, Kevin Hyska. Each of them has contributed in the project with their time, ideas, professional work and communication. There are a lot of other players in this project such as: the children, parents, teachers, other persons related to the children.

Page 2

***2.1*** ***Product Context***



***2.2*** ***User Characteristics***

***The kindergarten kids who will play the game:***

-Kids from ages 3-6.

-Education : kindergarten.

-Kids have a well known experience in playing games, they know what they prefer and what they require in a game.

-Can adapt to easy and understandable game instructions.

-Will be the ones to use the game more than other customers.

***Students:***

-The ones who will create the game and test it.

-The students are the one who will control the way the game works and make changes.

-Ages: 18-20.

-Experience in coding and creating a game.

-Experinece in c programming and python.

-Educating in software engineering.

-Intelligent users/customers, that do not have problems with understanding the game.

-Will use the game frequently in the developing and creating process.

***Others:***

-People of different ages who will be interested in playing the game.

-Different education and technical experiences. (According to the person who will use the product)

Page 3

***2.3*** ***Assumptions***

Behind this project there is a collateral work of students to create a special game called “theDOT”, for every child of kindergarten.

***a)Resource assumptions***

The first task for every stakeholder was to make an interview with different target groups such as: kindergarten children, teachers, parents, siblings and cousins or other people related to children. Two of our stakeholders (Era Mulla and Franceska Rrokaj in collaboration with Greisi Jaho), had do make an interview with “Albanet” kindergarten children and teachers. They were very welcoming and very interesting characters. The interview was filled with simple and understandable question for the children such as:

1- Do you use any electronic devices to play games?

2-How much time do you spend playing ?

3-What kind of games do you like to play?

4-Do you like colors, animals, interesting personages in your games?

5-Have you ever played any educational game before which include academic simple questions?

6-Were you focused during these kind of games?

After making these type of question to the children the answers were even more interesting such as:

1-We have tablets and our parents phones.

2-Yes, we spend a lot of time on our electronic devices.

3-We like to play games with guns, people running, animals, with a lot of colors, barbies, dolls or cooking.

4-Yes we have played educational games but we weren’t focused because the entire game was made by questions, it was boring.

The teacher was very calm and kind with all of them. After asking the same question to her relatining with childrens, the conclusion was that they weren’t focused and they aren’t patient during the educational games.



Page 4



The interview of (Klajdi Shehu) with siblings was interesting too, because of the answers that they gave about their brothers and sisters. They said that some of them were calm and very patient, but mostly they couldn’t focus on academic questions without having fun.

The interview of (Juli Losha) with the parents was focused on the time that their children spend with technological devices. The parents themselves give them those devices because they stay all day long playing games. They mostly like funny games, which keep them focused and academic questions make them nervous.

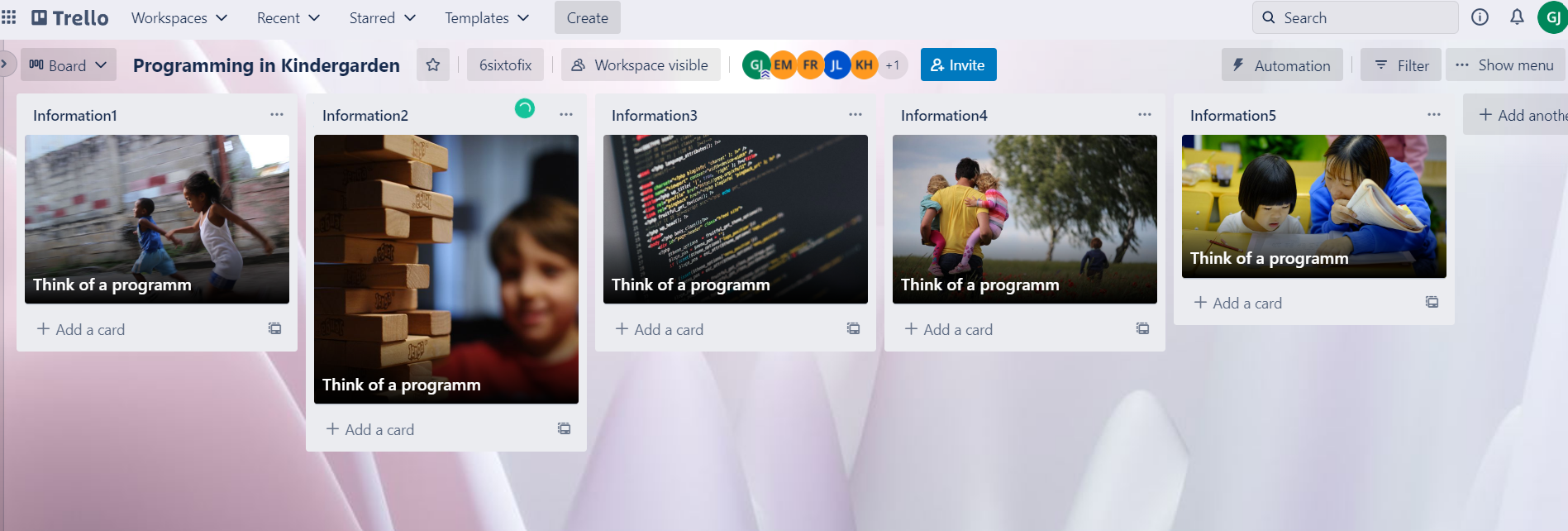
The other task made by (Kevin Hyska ) was to get information from the internet. The conclusion was the same, except some of the academic game marketing, which say that the children ere completely focused on those kinds of games.

***b)Technology assumptions***

In this project we have used laptops and mobile phones to gather informations that are needed in this project, but also to work. The game was created by Python language in Github from our group ”6toFix”. This is a very helpful platform, because it allowed us to keep track of everyone's job. In one case that one of us changed the color of the background, we directly managed to get it back how it was because it was better that way. So github has helped us a lot.

Slack was another platform, where we shared the meeting timetables and we communicated about different parts of the project.

Trello was another platform that helped us give requirements about the project. Each member of the group had one requirement for the week and it has an deadline, which everyone should respect.



***c)Time-based assumptions***

We fixed our time by chopping the requirements and tasks like said before in weeks. Each of us had one task each week.

Page 5

******

***d)Quality/specification assumptions***

There were some specific tasks wich needed to be redone, such as classdiagram, or state diagram. They were changing during the idea of creating a game, where everychild needed to feel comfortable and safe. So one in the class diagram, we added another class called “Teacher”, because we thought it was necessary the direction that a teacher would give to their children, about answering the programming question.

***e)Location/environmental assumptions***

The group had his meeting at “Epoka University”. It was a place where everyone felt comfortable and ofcourse ready to work on the project. But sometimes the group has met on coffee bars and restaurant, to discuss aboot the project.

***2.4*** ***Constraints***

**a)Access**

theDot is available in AppStore and in GooglePlay. The user can access the game by downloading it in these platforms, which can be already available on their mobiles.

*Steps on how to download theDot from these platforms:*

1. **Click** the icon of “AppStore”,”GooglePlay”.
2. **Search** the name of the game “theDot”.
3. After reading the important information, click **Download.**
4. Wait a couple of seconds up to 1 minute and click **Open**.

The user can access theDot by searching it on google chrome or Safari by clicking the name and implementing the necessary personal information.

Page 6



**b) Management**

**Risk Management**

It has to do with identifying risks that may come up during the time the project is created, and drawing up plans to minimise their effect on the project.

*There are three types of risks which could affect our project:*

1.Project risks -> the ones which affect schedule or resources.

2.Technical risks -> the ones which affect the quality or performance of the software (game) that we are developing.

3.Business risks -> the ones which affect the organisation developing the software.

| **Risk** | **Risk Type** | **Risk Probability** | **Risk seriousness**  **(Effects)** |
| --- | --- | --- | --- |
| Staff Turnover | Project | Moderate  (25-50%) | Serious impact on cost + schedule |
| Staff illness | Project | Moderate  (25-50%) | Serious |
| Recruitment problems | Project  Technical | High  (50-75%) | Catastrophic |
| Required training for staff is not available | Project  Technical | Moderate  (25-50%) | Tolerable |

Page 7

****

| **Risk** | **Strategy** |
| --- | --- |
| Staff Turnover | 1. Organise teams so that information about all activities is dispersed. 2. Give more meaningful and fun work for staff to do. 3. Assign backup staff for every critical engineer. 4. Offer flexibility. |
| Staff illness  (Minimisation strategy ) | 1. Reorganise team and change each person’s assignments. 2. Make sure everyone understands each other’s jobs. 3. Overlap-> extend the assignments of the ill staff to other people in the project. |
| Recruitment problems | 1. Since this is a high cost risk, lots of patience, time, skill and money is needed to find the right staff. It is a time consuming endeavour, so we opt to hire the services of a third-party outsourcing provider. 2. Alert customer of potential difficulties and the possibility of delays. 3. Investigate buying- in components. |
| Required training for staff is not available | 1. Try to make tasks easier and understandable for staff. 2. Create teams to work together and help each other. |

Page 8

| **Risk** | **Risk Type** | **Risk Probability** | **Risk seriousness**  **(Effects)** |
| --- | --- | --- | --- |
| Management change | Project | Moderate  (25-50%) | Tolerable |
| Organisational risks | Project  Business | High  (50-75%) | Serious |
| Organisational financial problems force reductions in the project budget | Business | Low  (10-25%) | Catastrophic |
| The organisation is structured so that different management are responsible for the project | Project  Technical | High  (50-75%) | Serious |

| **Risk** | **Strategy** |
| --- | --- |
| Management change | 1. There will be a change of organisational management with different priorities. 2. Choosing the right people to manage different things, which they are more specialised about. |
| Organisational financial problems force reductions in the project budget | 1. Being careful in spending project budget. 2. Spending money only on important things. 3. Creating/ Having a “just in case” budget beforehand, would be very good in these cases. |
| The organisation is structured so that different management are responsible for the project | 1. Creating an organisation with qualified and professional people. 2. Making sure everyone knows what is doing and they’re the right person for their tasks. 3. Different management are responsible for the project, which means the work is divided equally in all organisation members. |

Page 9

| **Risk** | **Risk Type** | **Risk Probability** | **Risk Seriousness**  **(Effect)** |
| --- | --- | --- | --- |
| Hardware unavailability | Project | High  (50-75%) | Serious |
| Technology change | Business | High  (50-75%) | Catastrophic |
| The database used in the system cannot process as many transactions per seconds as expected | Project  Application | Moderate  (25-50%) | Serious |
| Software components which should be reused contain defects which limit their functionality | Project  Application  Business | High  (50-75%) | Catastrophic |
| The time required to develop the software is underestimated | Project  Application  Business | High  (50-75%) | Catastrophic |
| The size of the software is underestimated | Project and product | High  (50-75%) | Tolerable |

Page 10



| **Risk** | **Strategy** |
| --- | --- |
| Hardware unavailability | 1. Check for other hardware available that can do the job. 2. Find the right hardware which can overplay the role of missing one. |
| Technology change | 1. Communicate with the customer about the change and decide what's the best option. 2. Choose the best technology which suits the project more. |
| The database used in the system cannot process as many transactions per seconds as expected | 1. Make changes in the database so everything is okay. 2. Investigate the possibility of being a higher-performance database. 3. Change database to a database more fit for the project. |
| Software components which should be reused contain defects which limit their functionality | 1. Work with the team to fix defects and make the software work better. 2. Create another software to use. |
| The time required to develop the software is underestimated | 1. Alert customer about possible delays. 2. Organise team members to cover as much work as possible, to develop software faster. 3. Investigate the use of a program generator. |
| The size of the software is underestimated | 1. Check and get rid of anything unimportant. 2. Let customer know that the size of software and if not pleased make changes. |

Page 11

| **Risk** | **Strategy** |
| --- | --- |
| Hardware unavailability | 1. Check for other hardware available that can do the job. 2. Find the right hardware which can overplay the role of missing one. |
| Technology change | 1. Communicate with the customer about the change and decide what's the best option. 2. Choose the best technology which suits the project more. |
| The database used in the system cannot process as many transactions per seconds as expected | 1. Make changes in the database so everything is okay. 2. Investigate the possibility of being a higher-performance database. 3. Change database to a database more fit for the project. |
| Software components which should be reused contain defects which limit their functionality | 1. Work with the team to fix defects and make the software work better. 2. Create another software to use. |
| The time required to develop the software is underestimated | 1. Alert customer about possible delays. 2. Organise team members to cover as much work as possible, to develop software faster. 3. Investigate the use of a program generator. |
| The size of the software is underestimated | 1. Check and get rid of anything unimportant. 2. Let customer know the size of the software and if not pleased make changes. |



Page 12



| **Risk** | **Risk Type** | **Risk Probability** | **Risk seriousness**  **(Effects)** |
| --- | --- | --- | --- |
| Requirements changes  (Changes to requirements which require major design work are proposed) | Project and product | Moderate  (25-50%) | Serious |
| Customers fail to understand the impact of requirements changes | Project | Moderate  (25-50%) | Tolerable |
| Produce competition | Business | High  (50-75%) | Serious |
| CASE tool under-performance | Product | Moderate  (25-50%) | Tolerable |
| CASE tools cannot be integrated | Project and product | High  (50-75%) | Tolerable |
| The code generated by CASE tools in inefficient | Project and product | Moderate  (25-50%) | Insignificant |

Page 13



| **Risk** | **Strategy** |
| --- | --- |
| Requirements changes | 1. Communicate with the customer to understand his complaints. 2. Organise team members to work in groups with the changes, for a quicker job. 3. Make the changes that the customer asks for. |
| Customers fail to understand the impact of requirements changes | 1. Communicate with the customer to make him understand what he’s asking for. 2. Give the customer insight of what actually happens in the developing of the project. |
| Produce competition | 1. Make changes that make the system and product better than the competitive product. 2. Investigate the possibility of investing more time and money in the project to make it better. 3. Check the resemblances and get rid of them. |
| CASE tools cannot be integrated | 1. Make changes that will help in integrating CASE tools. 2. Group CASE tools to integrate well with each other. |
| The code generated by CASE tools in inefficient | 1. Organise team to work in changing the code to an efficient one. |

Page 14

**c)Security**

theDOT is a high secure game. The first requirement of the game for the user is to enter a name and a STRONG password. The password should include at least 8 characters such as: letters,numbers and symbols. There is a detector to check if the password is strong enough so the user can change it if it is necessary.

theDOT has a limit of 200 mb on the disk space. Itis a simple game, which does not require much space to be build. theDOT has a very simple and understandable design for kindergarten children. The space is 40x60 rectangle in the shade of blue. There are two 5x15 white rectangles located 1 cm away from the borders. There is also a white ball with a radius of 5 mm which tends to move randomly in the space. The programm language which create this game is PYTHON, and it is used by the group in the GIThub. GITub is a great platform, where a group is welcomed to work together and share the work.

***2.5*** ***Dependencies***

This new product presents the need to download distinct features on the IDE such as PyGame Library, specific libraries and extensions to be executed.

This new product presents the need to have access on mobile phones, laptops, tablets, PC, and other technological devices.

****

Page 15

****

**3. Requirements**

***3.1*** ***Functional Requirements***

*This table is create by 3 main headers such as: User story which include the actions that the user can take related to the System. Functional Requirement which describe what must the system allow or not and the priority.*

|  | User story | Functional Requirement | Priority |
| --- | --- | --- | --- |
| User specification | User should specify its post. | The system must allow user to choose between teacher or developer. | * Must have |
| Creatin profile | User should create a profile. | The system must allow the user to create a profile by entering e-mail,pasword,age. | * Must have |
| Uploading student’s reports. | User should edit and upload the profile with student’s reports. | The system must allow the user to apload the students personal information. | * Must have |
| Log in | User should log in the system. | The system must allow the user to log in by entering email and password . | * Must have |
| Start Playing | User should play the game. | The system must allow the user to play the game after filling the brackets required. | * Must have |
| Repeating the level | User should respond to the programming question, to repat the level. | The system must allow the user to repeat the level after responding to the programing question correctly. | * Must have |
| Win the game | User should collect star coins to win the game. | The system must allow the user to pass the level after collectin enough star coins. | * Must have |
| Errors | User find an error during playing. | The system must allow the developer fix the error. | * Must have |

Page 17

******

***[theDOT] Requirements Specification***

***3.2*** ***Non-Functional Requirements***

|  | User story | Functional Requirement | Priority |
| --- | --- | --- | --- |

| Modify | User want to modify the character and background. | When the modify button is pressed the system shall allow the user to modify the character and the background. | * Shall have |
| --- | --- | --- | --- |
| Inserting colors. | User want to add colors to the character and the background. | When the “select color” button is pressed the system shall allow the user to add colors to the character and the background. | * Shall have |
| Inserting size. | User want to change the size of the character and the background. | When the “select size” button is pressed the system shall allow the user to change the size of the character and the background. | * Shall have |
| Inserting country | User want to change the country of the background that he is. | When the “select country” button is pressed the system shall allow the user to change the country and add a flag in the background | * Shall have |
| Update the level | User want to update the current level after passing the previews one. | When the “next level” button is pressed the system shall allow the user to pass to the next level. | * Shall have |
| Exit game. | User want to exit the game. | When the “exit” button is pressed the system shall allow the user to exit the game. | * Shall have |



Page 18



**3.2.1 User Interface Requirements**

In addition to functions required, describe the characteristics of each interface between the product and its users (e.g., required screen formats/organization, report layouts, menu structures, error and other messages, or function keys).

At the moment when theDOT is opened there is a page with three main buttons at the top of the page such as: profile,play,modify and log in. Each of them requires important information:

1. If the user click the profile button, it will be required: name, surname,e-mail, age, password and security question.

2.If the user click modify button,it will be required to choose between the character and the background. In the case that the user chooses the character, there arse two alternative: size and color; in each of them the user can choose whatever he/she idealise. If theuser chooses background, then two main alternatives will pop up: color and country. If he/she chooses country, than the flag of the country will be added in the corner of the background.

3.If the user click log in button, then there will be some information required such as: name,surname,paassword.

4.If the user click play button the game will start.

At the moment that the user play the game and press command to make thDOT move and collect coins there comes to an end when the character arrives at the “FINISH” sign. Then there are two possibilities:

a)If the user wins there will be aploadet a page where the character is surrounded by musical instrument which play music, and there are two buttons: Next Level and EXIT game.

b)If the user doesn’t win then it will show up a page for the programming question. There will show 4 alternatives, and if the user select the right answer, he/she can repeat the level. The user could also press exit button.

At the moment when the user finds an error in the game he/she can clich the “?” button and click on “report a problem”: the type of problem and the time when the problem was fond will be required.

Page 19

**3.2.2 Usability**

**User Documentation**

The user of theDOT may be a middleaged person, but this game is build for 4-6 aged children, so the manual usability of this game is very simple and understandable.

* **Profile:**

1. Click profile
2. Click name and enter it in the bracket.
3. Click surname and enter it in the bracket.
4. Click e-mail and password and eneter them in the specific places, by respecting the rules.

* **Log in:**

1. Click log in.
2. Enter e-mail
3. Enter password

* **Modify:**

1. Click character or background.
2. Choose between size, color and country.
3. Select the prefered one.

* **Play:**

1. Clic play.
2. Click start.
3. Put commands into the cubes: If the character should move into the right direction, the user should choose the right arrow and so on.
4. The character should collect coins and arrive at the finish line.

* **Win:**

1. The user win the game.
2. Click next to play in the next level.
3. Click Exit to exit the game.

* **Loose**:

1. The user loose the game.
2. Click retry level
3. Click “QUESTION”
4. Select an alternative.
5. Click repeat level to repeat.
6. Click Exit to exit game.

Page 20



**3.2.3 Performance**

The user can use the game for 24 hours/day.

For every section the user do not need to wait more than 1-2 seconds. The game generate fast and every system action is evaluate in a specific time.

The user actions depend on his way of solving the game. The longest time he/she could use is 10 second, so the use can answer the question.

**3.2.4** **Manageability/Maintainability**

3.2.1.1 Monitoring

Service health monitoring:

- Computer health checks and middleware of services’ configuration are done through ASP.NetCore.

Failure conditions:

According to the severity of the user actions, the failures are classified as:

-Minor: Closure of the game by the user while it has not finished yet.

-Major: Code is not running properly, even though there is an absence of syntax error and a lack of

memory limit problems.

-Hazardous: Primary data failures come as a result of data corruption by hackers.

Error (fault) detection:

● There should be a proactive detection for monitoring, meaning that potential risks should be

identified and a contingency plan should be established to respond to errors in an effective

manner.

● There should be an estimation of the remaining life of the component equipment and an

evaluation of extension to determine fault location.

● Automatic diagnostic systems (ADS) are the next step to pure and simple monitoring. They

should receive information and manage the “knowledge bank” of the system, where

information from physical parameters is crossed and integrated.



Page 21



● Continuous information about the system should be collected, provided by the monitoring.

system, in a short amount of time, on the behaviour of the equipment of a production system

and its supervision data, which will allow to determine if a device is properly operating or is at

Risk.

Logging and correction:

● A log management system should be set up, which stores, gathers, sorts log data from different

sources in only one location.

● Log data should be analysed in case studies so that computer-generated records are reviewed,

**3.2.1.2 Maintenance**

*Maintenance attributes:*

● Modularity: The application is modularized (separated) into a menu, 10 levels, 1 questions per level, a “Exit” button and a “Next Level ” or “Repet Level” button. No more space is allocated for more questions.

● Defect tracking tools: bugherd.com

● Code restructuring tools: RefactorIt.

● Search engines: Google.

● Quality measurement tool: Clarity

● Productivity measurement tool: Space

● Hardware configuration item(s): configparses

● Software configuration item(s): config (Python initialization configuration)

Page 22



**3.2.1.3 Operations**

**Periods of interactive and unattended operations:**

**Interactive operations**: When the user clicks the “Login” button, adds the username and password, and then proceeds to play the game..

**Unattended operations:** The user does not have to close the app to restart the application nor does he have to calculate the score himself.

**Data processing support functions:**

● The supplied data should be correct and relevant, thus validated.

● Items should be arranged sequentially and in different sets.

● Data should be reduced to its main points through statistical and automatic summarization.

● Multiple pieces of data should be combined.

● Data should be collected, organized, interpreted and fully presented.

● Summary data and computer information should be reported.

● Data should have a hierarchy (classification) into various categories.

Page 23



**Backup and recovery operations:**

● If the application is closed, it should be opened for normal reuse again; if open, recovered tablespaces should be brought online.

● These operations should enable the user to migrate from an older version of the code to a newer one.

● If recovery manager backup files of the app are lost, these should be restored with the python-sleuthkit-recover command. Safety considerations and requirements:

● Sensitive data should not be distributed between third party mediators.

● No sensitive data should be allowed in backups.

● Data should not be stored outside the app’s storage system.

● Data should not be stored for a long amount of time and memory should be cleaned fast.

● Passwords must not be exposed through the interface.

● The application should not be clicked for more than 3 times per second. It may close or lag as a result of this.

● No failure should shut down the app completely.

**Disaster recovery and business resumption:**

● Using NAKIVO & Replication backup, man-made disasters and outages should be fully recovered.

● Interruptions to the normal operations should be minimized.

● The extent of disruption and damages should be limited.

● The economic impact of the interruption should be minimized.

● Alternative means of operations should be established in advance.

● Customers should be guided with emergency procedures for smooth and rapid service restoration.

● Business resumption should be carried out through this 7-step procedure: Leader notification-> Setting a disaster recovery team -> Determining the degree of disaster-> Monitoring progress -> Backup site and schedules’ establishment -> Contacting the personnel (all teammates) -> Notifying users about the period of time up to which the code will start working again.

Page 24

****

**3.2.5** **Security**

3.2.5.1 Protection

**Encryption:**

● Asymmetric cryptography should be used and a private key alongside with it (not shared with anyone).

● RSA (Rivers-Sharmin-Adleman) algorithm usage should be used to protect insecure data.

**Data integrity checks:**

● Data should be secured critically and accurately.

● No unintended changes to data should be made.

● No malicious programs should harm it.

● Data should be protected from unauthorized parties.

**Firewalls/Antiviruses:**

● Unsolicited and unwanted incoming network traffic should be blocked.

● Access should be validated by assessing incoming traffic for malware.

● Windows Firewall should be used.

● McAfee should be installed in Windows to prevent malware.

● Antivirus should be updated each time it checks for updates.

**Auditing**:

● A script should be run over the application to spot trends/irregularities/fraudulent entries.

● It will be useful for big numbers and large data amounts.

**Elimination of exposure to external networks:**

● There should be DNS lookups for any exposure to external websites.

● IP address should be googled to see if it has been changed; if so, it is a sign of fraud/theft by a user.

● Checking with your ISP (Internet Service Provider), IPS (in-plane switching panels) are the panels that should be asked for.

**Secure remote access:**

● A combination of security processes/solutions should be designed to prevent unauthorized access and security data loss.

● Security endpoints should be established for users who use the app in more than one device.

● All users must be reminded not to click on any unsolicited material (pop-up blocks).

Page 25



**3.2.5.2 Authorization and Authentication**

**Knowledge factors:**

● PIN address, the username/passwords should be authenticatable.

● Data and information should be displayed before accessing a security system.

**Ownership factors:**

● They should be maintained through a device that belong to the correct user.

● Specific information should be displayed before accessing the user’s system, such as e-mails or one-time passwords.

**Inherence factors:**

● They should be biometric (fingerprints/thumbprints/facial recognition), but they are not used into our application.

**Location factors:**

● IP addresses should identify users attempting to copy another user’s IP address.

**Behaviour factors:**

● Actions of the user to gain access to the system should apply the picture selection forms (in case other forms fail), such as Captcha.

Page 26



***theDOT Requirements Specification***

***3.3*** ***Domain Requirements***

The start of the program will have the start=time.time() function, whereas the ending one end=time.time().

Refresh time frequency: 60HZ

Size: 200 mb

Time restart after failure: 0.2-0.35 secs

RAM chips’ size used: 256 mb

Training time in Python coding: 2.5 weeks for each teammate to get enough knowledge on codding the game.

Help frames: Tkinter frame, used as a foundation class to apply widgets.

MTBF=Infinityy

MTTF=1

Amount cost: 600$

Goal target cost 700$

Percentage of system cost: 62%

Page 27

1. **Design thinking methodologies**

4.1 ***Negotiation***

* The team negotiated in terms of each member's preferences towards the game .Some members wanted different features involved ,for example: different sounds, different number of levels and different color of background etc. In the end all members needed to agree on something that was the best option for the product/game ,in other words we needed to negotiate with each other and all other stakeholders in order to bring the best product possible.

4.2 ***Empathy***

* Coming up with solution that would satisfay all parties involved in the proces of making the game was hard, so members needed to show empathy towards each other ideas and their preferences. By showing empathy towards each other, the members came up with productive ideas which helped with shaping the game, and the overall process of creating the games was easier.

4.3 ***Noticing***

* The game is for kids from 3-6 years old, members needed to pay close attention to all details to complete all requirements, no matter how small or unimportanrt they might seem. These details start from the login page, making the game more entertaining for the kids and adding the sounds at each level pass.

Page 28



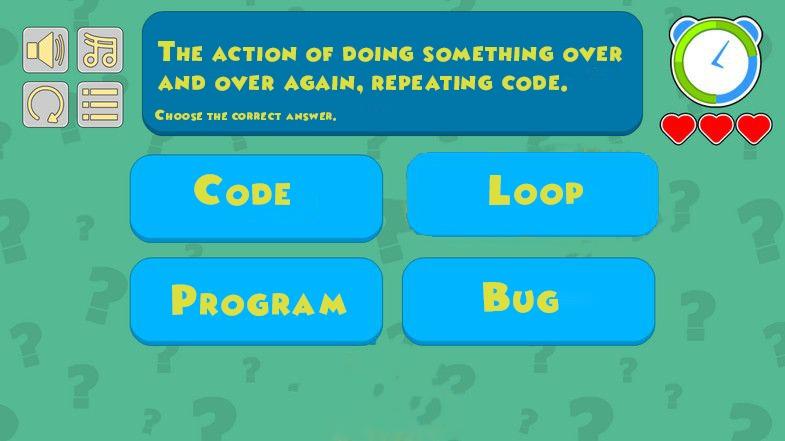
**4.4 GUI (Screenshots)**





Page 29





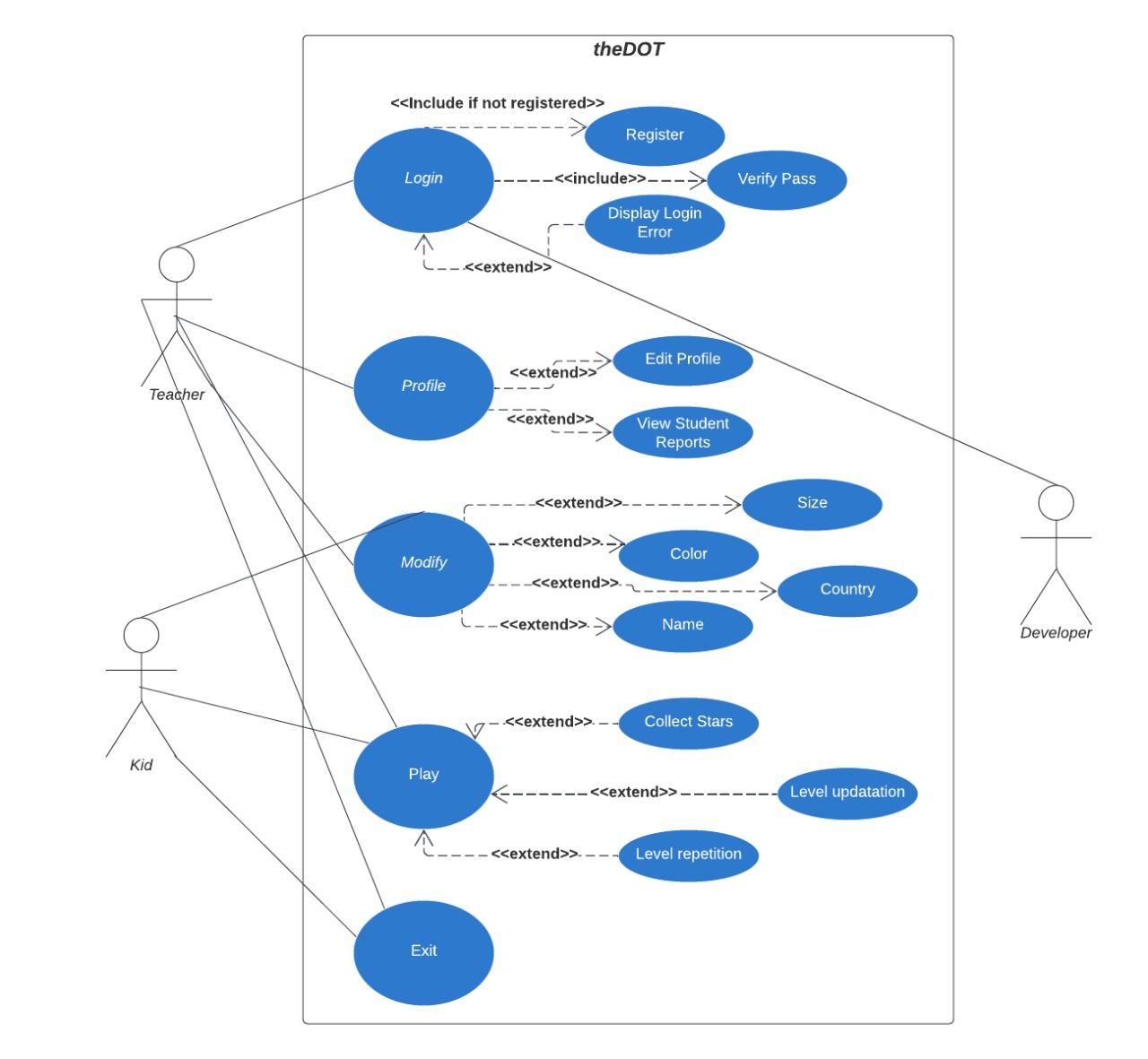


Page 30

****

**5. Software Design**

5.1 Use Case



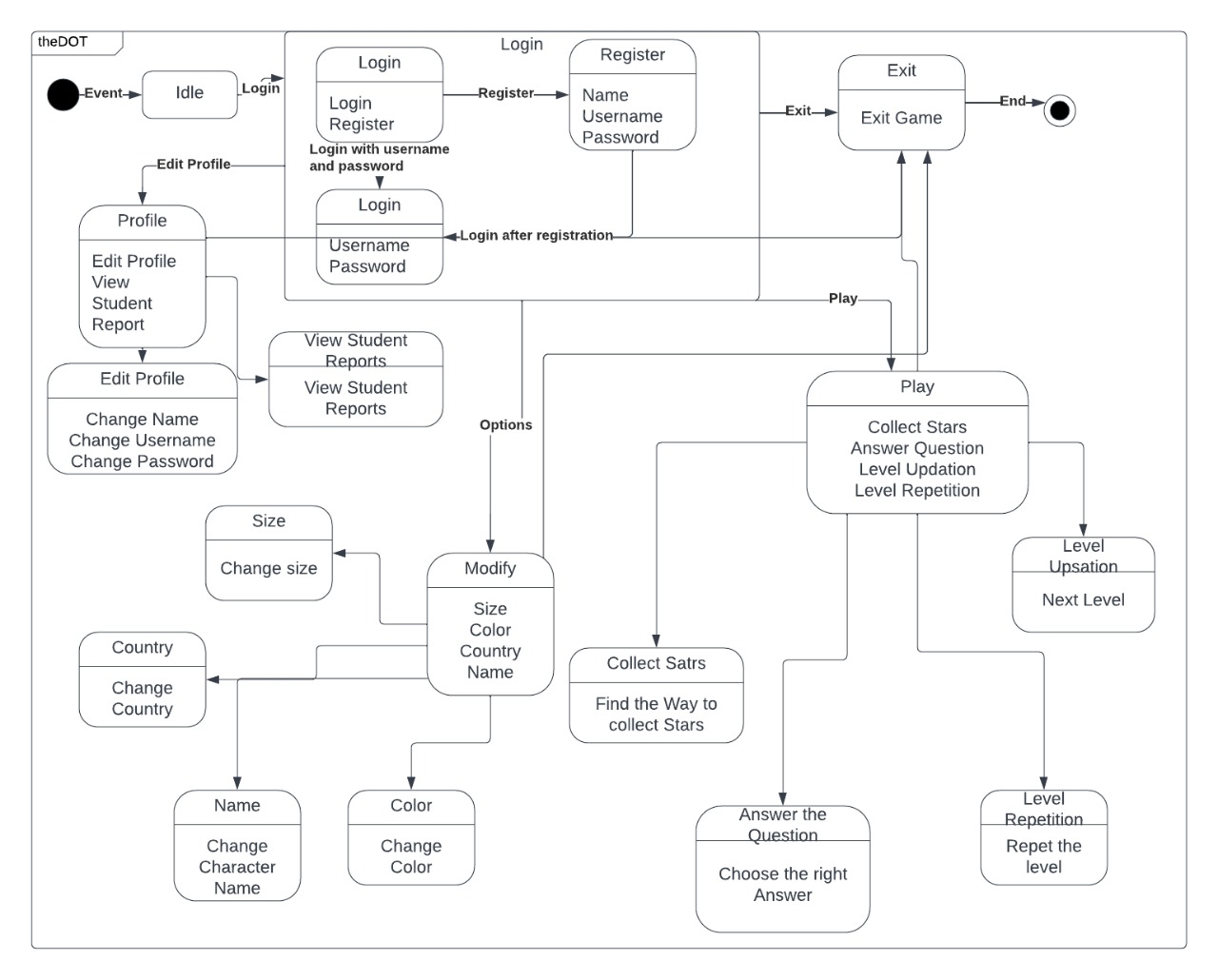
**Explanation:**

There are three main actors for theDOT programm: teacher, child(kid) and the developer. Teacher have acces to all 4 cases such as: log in, profile,modify,play and exit. The child has acces on modify,play and exit. The developer has acces only in log in, and then fixin the error that has shown up. Each of these cases is related to other ones. Log in does include Register, verify password and it does extend Display log in errors. Profile extend edit profile and view student reports. Modify extend size,color,country and name of the character or the background. Play extend collect stars, level updation and level repetition. In the end is Exit.

Page 31

****

**5.2** **State Diagram**



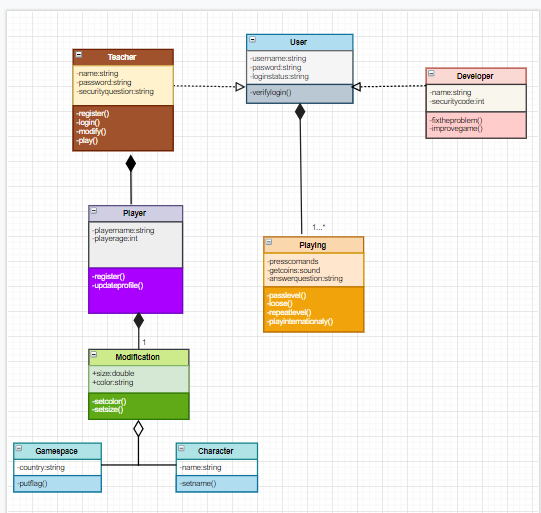
**Explanation:**

The first event is to log in (idle) , log inhas two main parts, the log in which requires username and password and register which requires name, username, password. This is related with edit profile, which has the edit profile and view studen report parts. It is also related with playing, which have four more actions the user can take such as: collect stars, level updation, level repetition and answer the question. It is also related with modify, which has four more action that the user can taike such as: size,country,color and name. It is also related with exit, which exit the game.

Page 32

****

**5.3 Class Diagram**



***EXPLANATION:***

1.There is the main class called “User”. It does have the attributes of creatine a username a password and a log in status, such as player or developer. There is a function called verifylogin().

2.There are two other classes which are implemented with the “User”; the “Teacher” class and the “Developer “class.

a)”Teacherr” class has attributes of creating a player name, player password, the player age and of course a security question. There are three functions such as: register, log in and update profile.

b)”Developer” class has two attributes of the name and the security code and there are two functions such as fix the problem (if there are such cases) and improve the game.

Page 33

c)There is another composition with the “Teacher” class, which is called “Player”, at it does have two attributes such as: play name and play age and two functions called: register(),update profile().

There is also a class which is a composition of the “User” and it is called “Playing”. It has three attributes such as press command(so the loop has the main focus), get coins(to win the game) and answer question- (programming question)

3.There is a composition of the “Player” and it is called “Modification“, which has two public attributes of size and color and two private functions such as: set color and set size.

a)There are two subclasses which are implemented with the “Modification”, one is called “Gamespace” which has the attribute of choosing the country, where you want to play and the function of putting the counttrie’s flag and the other one is called “Character”, which has the attribute of the choosing the name of the character and the function of setting a name.

Page 34