UNIVERSITY OF TWENTE.

Requirement Analysis Document

Project Name: Ball on the Wall	**Team Members: Jan van Zwol, Vo Nhat Minh, Tran Duc Duc, Marjolein Bolten, Ho Hoang Phuoc, Daan Velthuis**
Team ID: 30	**Mentors: Puru Vaish & Venelina Pocheva**

1. Introduction

The product we are designing is an interactive augmented reality game where a virtual game is played by real world inputs. The game will consist of a projected image on a wall where there will be targets that the player needs to hit. The player can hit these targets by throwing a tennis ball at the wall that the image is projected on. This tennis ball will be tracked by two cameras. When the tennis ball hits the wall the Pi will use the cameras to pinpoint the location of impact (it also uses the cameras to notice when the ball has hit the wall) and use that information for the gamelogic.

1.1. **Purpose**:

The purpose of our application is for it to be a fun game but also to be a good way to train eye-hand coordination. We like the idea of making a game which includes augmented reality. This application is good for eye-hand coordination because of the throwing aspect and social interaction because you can challenge your friends to beat your high score.

1.2. **Limitations of the current system (If any)**:

There are other systems that will be able to track tennis balls however there is no fun element of the game in current systems that use the same technique and methods as us.

1.3. **Intended Audience**

The game is for children and adolescents. They will like this interactive game where they can throw balls at the wall. They can train their eye-hand coordination while having fun. With this intended audience we will keep in mind that certain imagery is not suited for them and will keep an eye on child friendliness.

1.4. **Define SMART Goals**:

This section is used to list down the target/expected results from the project. All the goals should be written in a SMART (Specific + Measurable + Attainable + Relevant + Time-bound) way.

Specific (What)	Measurable (Up to)	Attainable (How)	Relevant (Why)	Time-bound (When)
1. To be able to track a (tennis) ball in a live camera videostream	*By calculating the difference between the system's guess of location and the actual location we see on the screen.*	*By making use of video recognition software or techniques.*	*The game needs the input of the location where the tennis ball hits the wall.*	*By the end of sprint 3*
2. To create a user-friendly interface so that the user can easily play the game	*By asking possible future users to play the game and tell us how they like the interface (user testing)*	*Taking notes during the testing phase of possible future users.*	*Having a user-friendly interface makes it easier for the player to start and play the game, which enhances the experience.*	*By the end of sprint 3*
3. To create an integrated system with a sensor which can detect the ball throw game.	*To measure delay time, success rate/ errors on any integrated parts.*	*By choosing suitable hardwares (camera and projector) and algorithms.*	*To ensure the game works and minimize the delay/ errors as well as improve user experiences.*	*By the end of sprint 3.*
4. To use a security system that makes sure passwords are safe	*By testing with strong and weak passwords*	*By giving input of not safe passwords*	*Accounts need to be protected against attacks*	*By the end of sprint 4*
5. To add a competitive element to the game.	*By asking possible future users to play the game and tell us how they like the interface (user testing)*	*By introducing a scoreboard system in which users can compare their results with each other.*	*Competitive elements motivate the players more and make it more fun to play against friends.*	*By the end of sprint 4*

1.5. **Scope:** This section is required to write about the important resources to achieve the goals of your system. The technology used to develop your project (methods/algorithms, software requirements, hardware requirements), the duration of the project, and the project constraints should be included here. The project constraints can be any technical hiccups, lack of resources, internal and external conditions (boundary conditions), etc. that can help further to avoid the related problems in the future during execution. In short, you can utilize this section to write about the limitations and boundaries of your project.

- ***Software:***
  - + Programming language: Python for algorithm and backend, Javascript, HTML, CSS for frontend, SQL for database.
  - + Algorithms: Image processing, Object and movement recognition.
  - + Libraries: Python libraries for algorithm and database connectivity.
  - + Database: postgreSQL via utwente bronto.
- ***Hardware:***
  - + At the heart of the system is a Raspberry Pi 4. It is connected to a projector, which will display the interface on the wall.
  - + Sensors: we have two cameras. One of the cameras will be used to check when the ball hits the wall and the other camera will look where the ball has hit the interface.
  - + Projectors: we use one projector to display the game with the shooting objectives.
- ***Interfaces:***
  - + The program uses the Internet via WiFi to communicate with an online database, and HDMI to connect with the projector.
- ***Limitations:***
  - + The system has to be set up before you can play the game. This will take some time and also calibration every time you will play in a different environment.
  - + There is a possibility that the system will not detect the tennis ball, in that case the throw is a fail.
  - + There is a possibility that the speed of the tennis ball is too high in comparison with the number of frames the camera can process, then it can be impossible to detect at which moment the ball has hit the wall.
  - + The system depends on the light and background environment, which can affect the correctness of the camera when it captures the screen (the image on the wall). For example the system might react differently to a black background than to a white background.

2. **Product features:**

This section describes the functionality that you want to have in your product such as the components used for the application and its functionality, appearance, performance in terms of speed/time, etc. You can specify them in the form of functional and non-functional requirements. A minimum number of 7 requirements (9 in case of selecting an existing application) is to be expected for your application. That includes functional as well as non-functional requirements cumulatively. However, it is highly probable that you will need more than the minimum amount to fully cover all the requirements.

2.1 **Functional requirements:**

Write the requirements that are directly connected with the functionality of the application.

- Users can register a new account and log in to the system. *(1- Security by Design Checklist)*
- The system can detect when and where a tennis ball hits the wall and objectives.
- The system can control the projector to update the game situation correctly in real time
- Users are kept updated with the current score while playing.
- The system can update the online scoreboard through the Internet.

2.2 **Nonfunctional requirements:**

Write the requirements that are not the specific actions for your application but improve the quality of the system. This can be related to the storage capacity, performance requirements, Security requirements *(Refer to the checklist given in SBD Document-Phase 1)*, etc.

- The system will be able to process user data (login credentials, score) no longer than 5s.
- The user interface should be user friendly with pleasing interaction.
- The updating game situation's delay should be less than 3s.

- ***Security by Design Checklist:***

- Users need to create an account to claim their score. (*1- Security by Design Checklist)*
- The passwords for registering should have high security standards. *(2- Security by Design Checklist)*
- Only authorized people can access the database. *(3- Security by Design Checklist)*
- User information should only be used for game purposes. *(4- Security by Design Checklist)*
- The passwords of the accounts should be hash-ed before storing in the database. *(5- Security by Design Checklist)*
- Losing the raspberry Pi should not mean losing the user information. *(6 - Security by Design Checklist)*
- The input of the camera should be well protected, such that it cannot be misused. *(7- Security by Design Checklist)*
- The user input such as username/ password should be sanitized *(8- Security by Design Checklist)*

3. **Conclusion:**
This document gives an explanation about all the requirements needed for our game: **Ball on the Wall**. The most important part of the project will be **tracking the tennis ball through space.** We will use **two cameras** to achieve this. We will also have to allocate time to making a visually attractive interface.

4. **Reference**: List the existing literature (documents/articles/blogs/research papers) references you have considered for finalizing the project idea.

Raspberry pi connected to a camera: http://domoticx.com/raspberry-pi-webcams-gebruiken-via-usb/

Raspberry pi connected to a projector:
https://projects-raspberry.com/connect-raspberry-pi-to-projector-or-tv/

Image processing algorithm: https://opencv.org/releases/

Module 5 -Computer Systems (2021-22)

Project

UNIVERSITY OF TWENTE.

Security by Design Checklist
(Requirement Analysis Phase 1)

Team ID: 30	**Team Members: Jan van Zwol, Vo Nhat Minh, Tran Duc Duc, Marjolein Bolten, Ho Hoang Phuoc, Daan Velthuis**
Project Name: Ball on the wall	**Mentor(s): Puru Vaish and Venelina Pocheva**

Steps to be performed:

i) You should select a minimum of one security mechanism from each of the security requirements from authentication and authorization both (auditing is not included here).

ii) The auditing requirements should be considered as suggested in the table according to your application. Other than the normal check on protecting log files, backup files, etc, you should also think about the GDPR obligations, software licensing, etc. in line with your application.

iii) The given security mechanisms are for your inspiration. You can select other mechanisms also according to the requirement of your application. For example: If you select "authentication" as one of the security requirements, the mechanism can be logging/password checking, biometric, OAuth, etc. The same is applicable for authorization and auditing.

iv) Justify the reason to select a particular mechanism for the requirements in the given column 'C'.

v) Write supplement requirement(s) in the form of a user story or Abuse case for the application (refer to the example given on the table, column 'D'). (The supplement requirements should be according to the goals and non-functional requirement (s) identified for your application.)

vi) Write the possible risks involved for the supplement requirements (refer to the example given in the table, column 'E').

vii) Write the resources/mechanisms/tools to avoid/mitigate those risks for security controls (refer to the example of the column heading "Appropriate Security Control" (column 'F')).

viii) This document must be reviewed with the team members and approved by your mentor(s)/TAs.

ix) Put a tick mark in the last column for all verified items.

x) This document should be appended to the Software Requirement Specification (SRS) document.

Follow these 5 points for each of the Security Mechanisms and write them under Appropriate Security Controls

i) Supplement security requirements to avoid risk.

ii) Write the requirement of the resources to mitigate such risks. For example: The type of Authentication software, security tokens, password management software, etc.

iii) Devise a plan/method (tentative) to work on the identified risks.

iv) Review the documentation within your team.

v) Approve the document by your mentor.

Security Policy	**Confidentiality, Integrity, and Availability**					
Security Requirements	**Security mechanisms (List down for your application)**	**Remarks on why you considered these requirements? (in a brief)**	**Supplement requirements for your application (user story/Abuse case)**	**Risk identification/Threat Assessment (at least one risk identification/abuse case)**	**Appropriate Security Controls**	**Tick ✓if you have applied the given security controls as suggested in the left column**
Non-repudiation	1. Making accounts	To make sure people can play for themself and have their own high scores, we have to create accounts. To make sure that people only play on their own account and thus deserve their own high scores, the accounts should be secured by a password.	The system lets new users register a new and unique account to play the game and keep their own records. **User story**: As a user, I want to use my own account to play the game. **Abuse case:** As an attacker, I can fake the account to claim other user's score	i) People can have the same username ii) People can have unsafe passwords iii) People can mistyped password when in registration process	Method to work on the identified risks. - Make sure no duplicate username is chosen - Make sure users type a strong password with at least 1 upper-case letter, 1 special character and 1 number. - Make sure user type password 2 times when creating account	
Authentication	2. Checking password	For granting access to multiple users and for users to have their individual profiles and store their scores, we need to authenticate their username with a password. A more sophisticated authentication is not required, as other (augmented reality) games also do not have this two factor authentication.	The system verifies that there are no default passwords used by the user. The system should also sanitize the password so that attackers cannot inject malicious codes. Requirement: To access the application, one should require authentication. **User story:** As a user, I can enter my user name and passwords to access the application. **Abuse Case:** As an attacker, I can enter the default passwords to access the application.	i) The length of the passwords is more than 1023 characters. ii) You enter a wrong password more than 3 times.	Method to work on the identified risks: - Make sure the user type input password not more than 1023 characters by showing an error message. - If they type the password wrong 3 times their account will be locked for 15 mins.	
Authorization	3. Access control policies role-based for DB system.	To make sure only people with admin roles can access the database system.	**User story:** As project manager I don't want random people to access the system database without permission. **Abuse case:** - As an attacker, I can modify the scoreboard. -As an attacker, I can login.	i) Some non-admin users can access the database to modify it. ii) Some non-admin users can try to login as admin.	Method to work on the identified risks: - Make sure only 1 account have the admin privilege - The username for the admin account is unique and the password is not easy to predict. The password for the admin account will be changed every 15 mins.	

Audit	4. Personal identifiable information must be processed following GDPR compliant and must not be used for any other purposes than for the game only.	To make sure the database is complying with GDPR.	**User story:** As a user I want my database to comply with the GDPR. **Abuse case:** As someone who has access to the database I want to sell the user data or use it for unintended and illegal purposes.	i) Admin sells users data for money. ii) Users data is not automatically deleted after X days. iii) No clear legal documents for the use of users data.	Method to work on the identified risks: - GDPR, term of services is shown to the user - legal documents to prevent admin abuse	
Confidentiality	5. Hashing the passwords	User's plaintext password can be stolen if the attacker can access the database. Furthermore, people often use the same passwords for different systems, which makes it become a password leak problem.	The system will only store hashed passwords in the database and compare hashed passwords for verification. **User story:** The user fills in their password in the password input field unhashed. **Abuse case:** As an attacker, I can somehow get access to the database and view users' passwords. I can log in on every account on the system and can try this password on other systems as well.	i) An attacker finds the unhashed password with a bruteforce. ii) An attacker finds the unhashed password with a dictionary attack.	Method to work on the identified risks: - The password will be hashed using SHA512 - We will hash the passwords on the client side when registering and logging in such that it will become impossible to intercept the users plaintext passwords.	
	6. Storing user data (login information) and the information of Raspberry Pi in different database	If the information on the Pi and the user data are stored at the same place, losing the hardware can make you lose all your user information when someone gets a hold of your Pi.	**User story:** As a user my data is stored in a database. **Abuse case:** As an attacker, I have to get access to the database and the Raspberry Pi to get all the information instead of only stealing the Raspberry Pi.	i) People can look at the database outside of the game environment if it is saved locally on the pi ii) People can alter the database outside of the game environment if it is saved locally	Method to work on the identified risks: - We will make sure that we have a seperate database phpPgAdmin which is not connected to the Raspberry Pi.	
	7. Camera input cannot be accessed from outside	The cameras film the users personal environment, which is private information. Therefore it should not be accessible online.	**User story:** As a user I use the cameras to play the game. **Abuse case:** As an attacker, I can access the camera and view the private personal environment of the player.	i) An attacker can access the camera and view the personal environment of the user.	Method to work on the identified risks: - We will make sure that the Raspberry Pi will be restricted in the data it can send, such that it will become impossible to access the input of the camera.	
Integrity	8. Sanitizing user input e.g. usernames, passwords, etc.	When the user input is not sanitized a malicious user can use these input fields to perform sql injection and destroy, alter or steal user data. This would be a huge security flaw.	The system makes sure that user input will not be interpreted as an sql command. **User story:** "As a user I can enter my input in the input fields" **Abuse case:** "As an attacker I can use the input fields to submit SQL commands to the database."	SQL injections by malicious users who: i) alter data ii) delete data iii) download data	Method to work on the identified risks: - Sanitizing user input or using prepared statements.	

Ho Hoang Phuoc: yes

Jan van Zwol: yes

Vo Nhat Minh: yes

Tran Duc Duc: yes

Daan Velthuis: yes

Marjolein Bolten: yes

Puru Vaish:
Venelina Pocheva: Yes