**Logo, company name

Description automatically generated**

**Security Assessment Report**

**Dirk Johnson’s Snake Game**

Version N.1

May 1, 2023

Table of Contents

[1. Summary 3](#_Toc127779445)

[1. Assessment Scope 3](#_Toc127779446)

[2. Summary of Findings 3](#_Toc127779447)

[3. Summary of Recommendations 4](#_Toc127779448)

[2. Goals, Findings, and Recommendations 4](#_Toc127779449)

[1. Assessment Goals 4](#_Toc127779450)

[2. Detailed Findings 5](#_Toc127779451)

[3. Recommendations 5](#_Toc127779452)

[3. Methodology for the Security Control Assessment 5](#_Toc127779453)

[4. Figures and Code 7](#_Toc127779454)

[4.1.1 Process flow of System (this one just describes the process for requesting) 7](#_Toc127779455)

[4.1.2 Other figure of code 7](#_Toc127779456)

[5. Works Cited 7](#_Toc127779457)

# Summary

When this project was originally created, security was not even a concept considered. This was made as a test of skills during a methodology course teaching the C++ programming language. Once that class concluded and I had taken other courses to strengthen my proficiency in the C++ language I felt I needed to make some improvements upon it. This course provided me a perfect opportunity for this. Not only would it test my coding skills, but it would force me to think differently about my coding practices. Overall, I feel as though, through techniques learned in this class, I have strengthened the security aspect of this program and set up a plan for ways to further improve upon it in the future.

## Assessment Scope

What tools, platforms, OSes, Browsers, and software (including your own) was tested or used in testing? The tools I used to test this program against security concerns were CLion, TryHackMe.com, MacOS, Windows OS, and practices learned in this course. All these allowed me to determine where any weak points existed and helped plan ways to counteract and fix them.

## Summary of Findings

Of the findings discovered during our assessment, 3 were considered High risks, 7 Moderate risks, and 3 Low. The SWOT used for planning the assessment are broken down as shown in Figure 2.

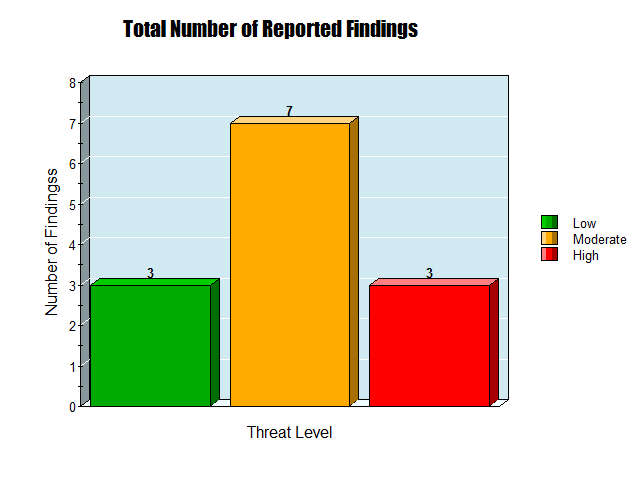


Figure 1. Findings by Risk Level

Above is a bar graph showing the breakdown of how many of each threat level concern was found during initial assessment of the project.

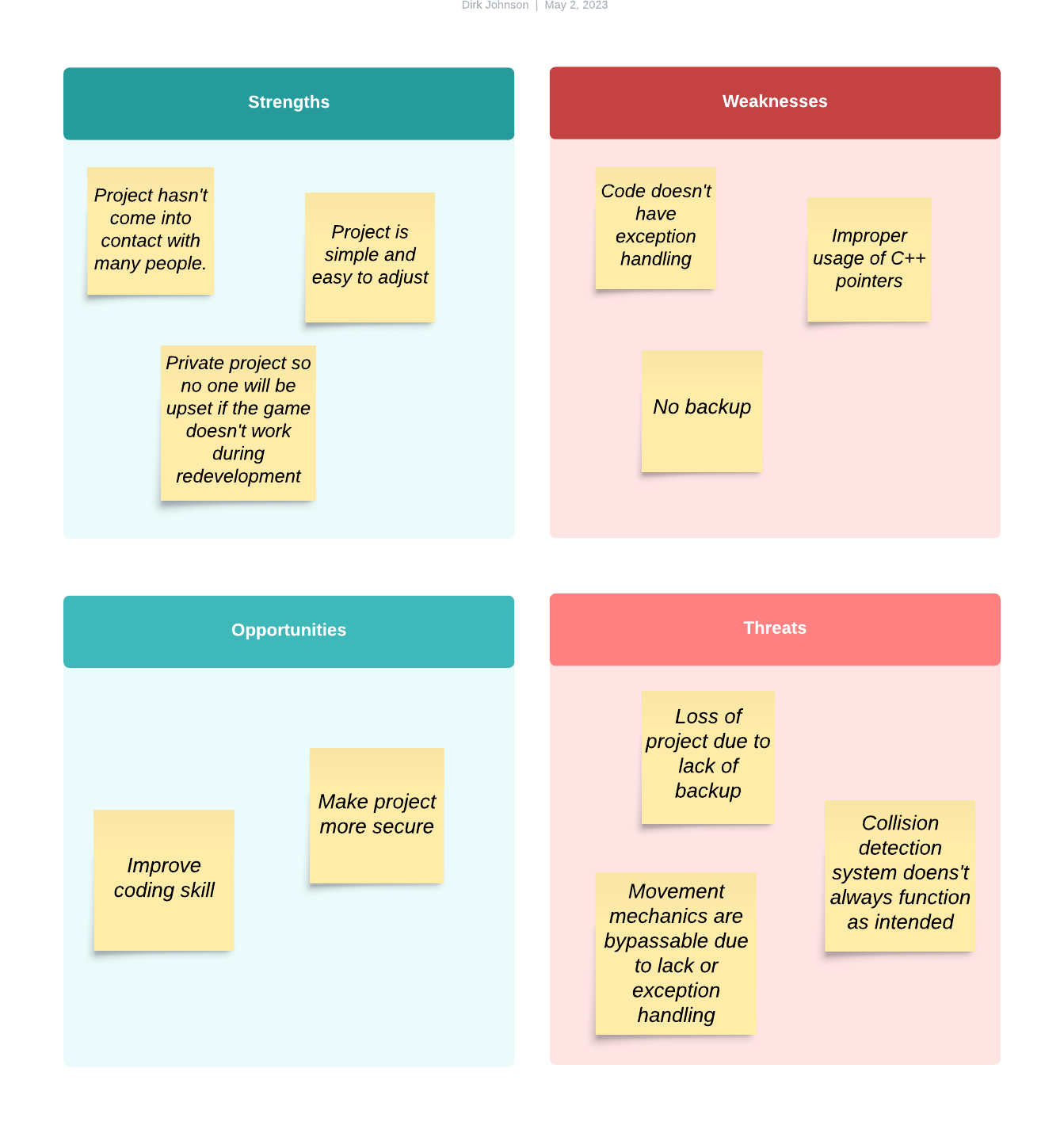


Figure 2. SWOT

Pictured above is the initial SWOT analysis of this project outlining strengths, weaknesses, opportunities, and threats the coincide with the project.

## Summary of Recommendations

# Goals, Findings, and Recommendations

## Assessment Goals

The purpose of this assessment was to do the following:

* Ensure the project had at least a basic level of security implemented.
* Discover where in my programming practices there were security holes and how to adjust my methods moving forward to include security.
* Expand upon my programming ability to include a more robust computer security mindset when developing future projects.

## Detailed Findings

We will begin with the low-level security concerns found during assessment. These include third party libraries not being up to date with updates, encryption and authentication methods not being used to access code, and no plan set for how to deal with potential bad internal actors. These concerns are more to deal with the code being functional due to these concerns dealing with lack of updates and bad actors possibly deleting or irreversibly altering the code. These were all determined to be low level threats because they didn’t prove to be both a high priority and very concerning. Next were the moderate level concerns. These were the most plentiful and included lack of access controls to only allow authorized actors to make changes to the code, lack of a backup policy, lack of exception and error handling, lack of a process for logging changes to code, and lack of security testing done. These were considered as moderate level concerns because they had a decent level of concern attached to each of them should they be exploited. Lastly were the high-level concerns. These included three different concerns that all revolved around the project not being available on a secure cloud-based platform, such as GitHub.

## Recommendations

Many of the concerns mentioned above can being quickly remedied with the simple addition of the project to a GitHub repository and simple setup of access controls on said repository. This would deal with six different concerns, those being cloud based access controls, cloud based sensitive information hiding, platform user groups for code changing, backup policy, logging of changes made to project, and encryption and authentication of connection to cloud-based platform. This would be considered an easy ease of fix because it is simple and only involves setting up a GitHub repository and creating access controls. As for the other concerns, third party libraries can be set up to automatically update periodically, easy ease of fix, exception and error handling can be fixed by testing the code and adjusting where necessary, this would likely be a very difficult ease of fix concern since it involves not only reprograming certain portions of the project, but it also involves finding the problems in the first place. Internal actor threats can be addressed with a policy where people working on the project get their permissions revoked to view only as soon as they are taken off the project, easy ease of fix. We can also ensure there is a level of security testing by periodically doing security assessments to determine any weaknesses not found in the previous assessment. This would be a moderately difficult ease of fix because it involves going through a whole assessment process which takes time and a great level of evaluation with testing.

# Methodology for the Security Control Assessment

**3.1.1 Risk Level Assessment**

Each project risk has been assigned a Risk Level value of High, Moderate, or Low. The rating is an assessment of the priority with which each project risk will be viewed. The definitions in Table 1 apply to risk level assessment values (based on probability and severity of risk). While Table 2 describes the estimation values used for a risk’s “ease-of-fix”.

Table - Risk Values

| Rating | Definition of Risk Rating |
| --- | --- |
| High Risk | Exploitation of the technical or procedural vulnerability will cause substantial harm to the business processes. Significant political, financial, and legal damage is likely to result |
| Moderate Risk | Exploitation of the technical or procedural vulnerability will significantly impact the confidentiality, integrity and/or availability of the system, or data. Exploitation of the vulnerability may cause moderate financial loss or public embarrassment to organization. |
| Low Risk | Exploitation of the technical or procedural vulnerability will cause minimal impact to operations. The confidentiality, integrity and availability of sensitive information are not at risk of compromise. Exploitation of the vulnerability may cause slight financial loss or public embarrassment |

Table - Ease of Fix Definitions

| Rating | Definition of Risk Rating |
| --- | --- |
| Easy | The corrective action(s) can be completed quickly with minimal resources, and without causing disruption to the system or data |
| Moderately Difficult | Remediation efforts will likely cause a noticeable service disruption   * A vendor patch or major configuration change may be required to close the vulnerability * An upgrade to a different version of the software may be required to address the impact severity * The system may require a reconfiguration to mitigate the threat exposure * Corrective action may require construction or significant alterations to the manner in which business is undertaken |
| Very Difficult | The high risk of substantial service disruption makes it impractical to complete the corrective action for mission critical systems without careful scheduling   * An obscure, hard-to-find vendor patch may be required to close the vulnerability * Significant, time-consuming configuration changes may be required to address the threat exposure or impact severity * Corrective action requires major construction or redesign of an entire business process |
| No Known Fix | No known solution to the problem currently exists. The Risk may require the Business Owner to:   * Discontinue use of the software or protocol * Isolate the information system within the enterprise, thereby eliminating reliance on the system   In some cases, the vulnerability is due to a design-level flaw that cannot be resolved through the application of vendor patches or the reconfiguration of the system. If the system is critical and must be used to support on-going business functions, no less than quarterly monitoring shall be conducted by the Business Owner, and reviewed by IS Management, to validate that security incidents have not occurred |

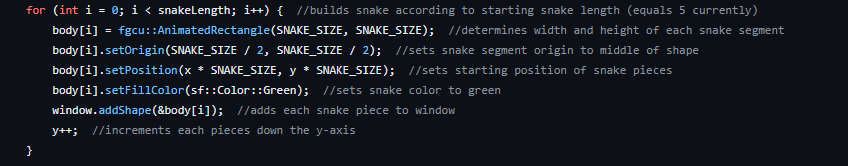
**3.1.2 Tests and Analyses**

Testing done on this project included overflow testing, error and exception testing, injection testing, threat analysis, white/black box testing, and quality assurance testing.

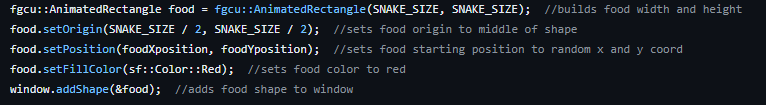
**3.1.3 Tools**

Tools used to complete the testing mentioned above included CLion, Linux command line, white/grey/black box testing methods, TryHackMe.com, and LucidChart.com. These tools allowed me to test security concerns, fix those concerns, and document the process of remedying them.

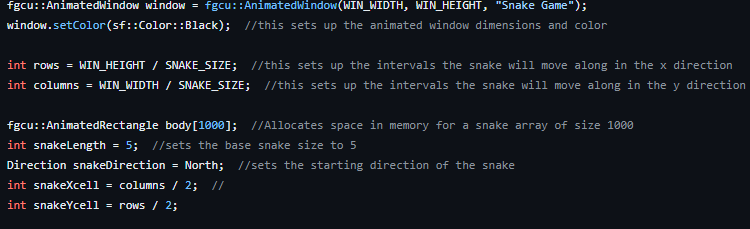
# Figures and Code



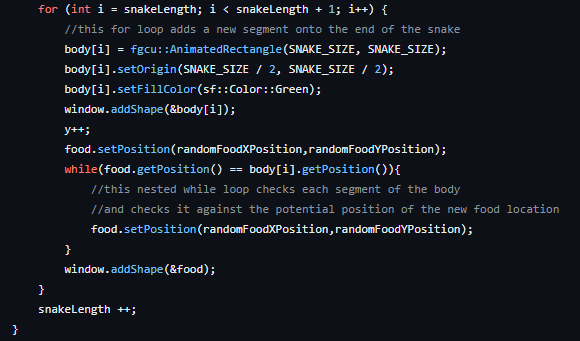
Pictures above is a section of the code that created the body of the snake. This would have been better set up as an array as opposed to the method shown.



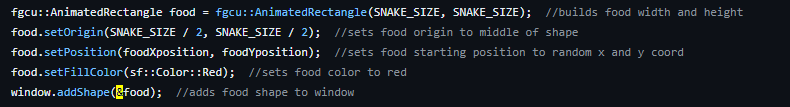
The code featured above breaks down the creation of the food entity in the game. This could have been done better using smart pointers referencing the AnimatedRectangle class in the SFML library.



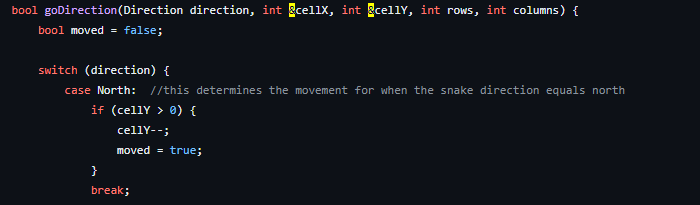
The code above lays out the creation of the window on which the game is played and the body of the snake. This should have been done utilizing smart pointers instead of creating an array.



Above is a section of code that adds segments to the snake every time it eats the food. This is done here using references pointers. This should be done utilizing C++ smart pointers instead.

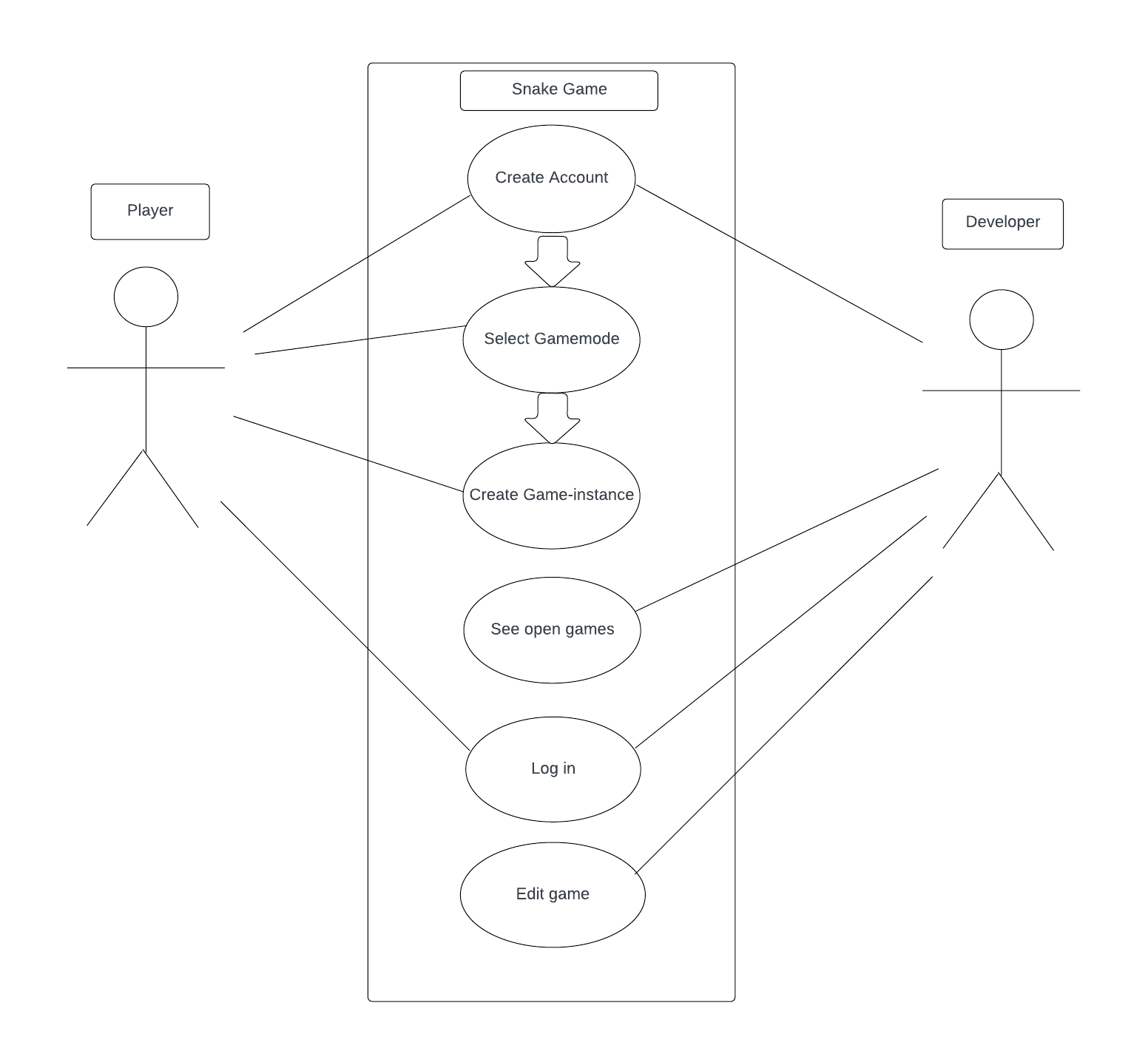


Here is another instance of where regular pointers are used instead of smart pointers.

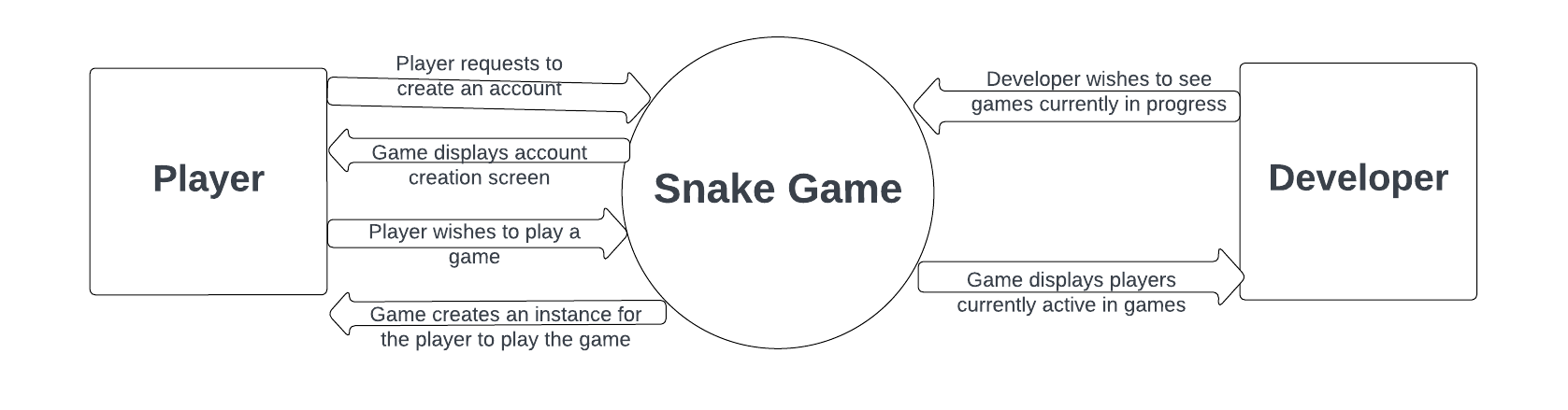


And another instance of regular pointers as opposed to smart pointers.

### Process or Data flow of System, use-cases, security checklist, graphs, etc.



Pictured above is a use case diagram depicting the flow of the project when in use.



Pictured above is the data flow diagram depicting the flow of data between the game and players and developers.

# Works Cited

**-https://fgcu.instructure.com/courses/536860/files/48829562?module\_item\_id=11675455.**

**-LucidChart.com**

**-https://fgcu.instructure.com/courses/536860/files/48856087?module\_item\_id=11679236**

**-https://fgcu.instructure.com/courses/536860/files/48983128?module\_item\_id=11693372**

**-https://fgcu.instructure.com/courses/536860/files/49204642?module\_item\_id=11719110**

**-https://fgcu.instructure.com/courses/536860/files/49285477?module\_item\_id=11729884**

**-https://web.microsoftstream.com/video/f48ac5f7-a367-4881-a994-5188e4382cfe**

**-Computer Security in the Real World Butler W. Lampson Microsoft**

**-https://www.hanselman.com/blog/whats-the-difference-between-a-console-a-terminal-and-a-shell**