UNIVERSIDAD INTERNACIONAL DEL ECUADOR

Facultad: Ciencias Técnicas

Carrera: Ingeniería en Ciberseguridad

Docente: Mónica Salazar

Asignatura: Lógica de Programación

Nombre del estudiante: Dayana Yaselga

Actividad: Aprendizaje Autónomo 1. Selección del Programa a desarrollar / Generación de Diagramas funcionales y Arquitectura de Software

Período académico: octubre 2024 – marzo 2025

Selección del Programa a desarrollar: Piedra, papel o tijera

Identificar el problema: Desarrollar un programa que permita al usuario jugar una partida de "Piedra, Papel o Tijera" contra la computadora.

Comprender el problema:

El juego deberá contar con reglas que permitirán elegir al ganador cuando el usuario seleccione una de las opciones entre piedra, papel o tijeras, la computadora deberá seleccionar aleatoriamente una opción y finalmente se mostrará el resultado de la partida al usuario si es ganador o si hubo un empate.

Identificar soluciones alternativas:

- Crear un juego en una interfaz de consola, donde el usuario mediante texto ingresa la opción de piedra, papel o tijeras.
- Desarrollar el juego con botones para cada opción de piedra, papel y tijeras, permitiendo al usuario seleccionar la opción visualmente y tener una retroalimentación rápida.
- Crear una aplicación web para el juego que permita interactuar con el usuario a través del navegador.

Seleccionar la mejor solución:

Desarrollo del juego con una interfaz visual y amigable para el usuario, se puede construir utilizando herramientas de desarrollo de interfaces gráficas como Python.

Listar los pasos de la solución seleccionada:

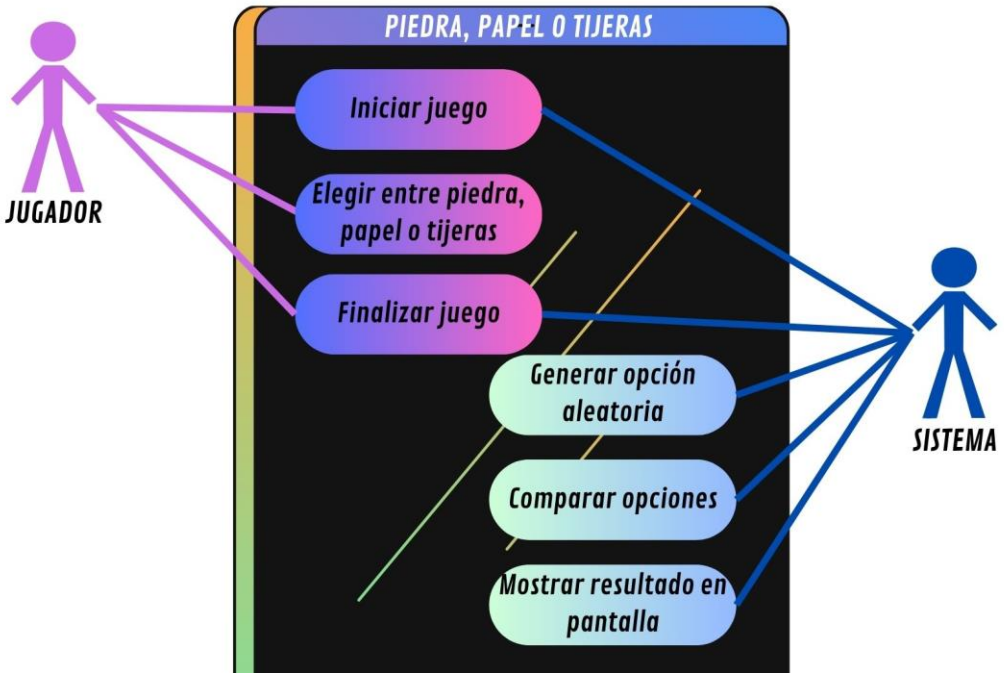
1. Realizar un diagrama de funcionalidad como es el diagrama de caso de uso que permitirá visualizar el comportamiento del usuario con el juego.
2. Crear la arquitectura del software utilizando el Modelo-Vista-Controlador (MVC), permitirá tener una vista macro del juego a desarrollar.
3. Diseñar la interfaz grafica del juego, con los tres botones de piedra, papel y tijeras.
4. Realizar pruebas a la aplicación, asegurando que funcione perfectamente y con los resultados esperados.

Evaluar/Probar la solución:

Finalmente revisar la interfaz del juego, asegurarse que los botones se encuentren bien ubicados y etiquetados, comprobar que funcione sin errores verificando que el juego responda correctamente.

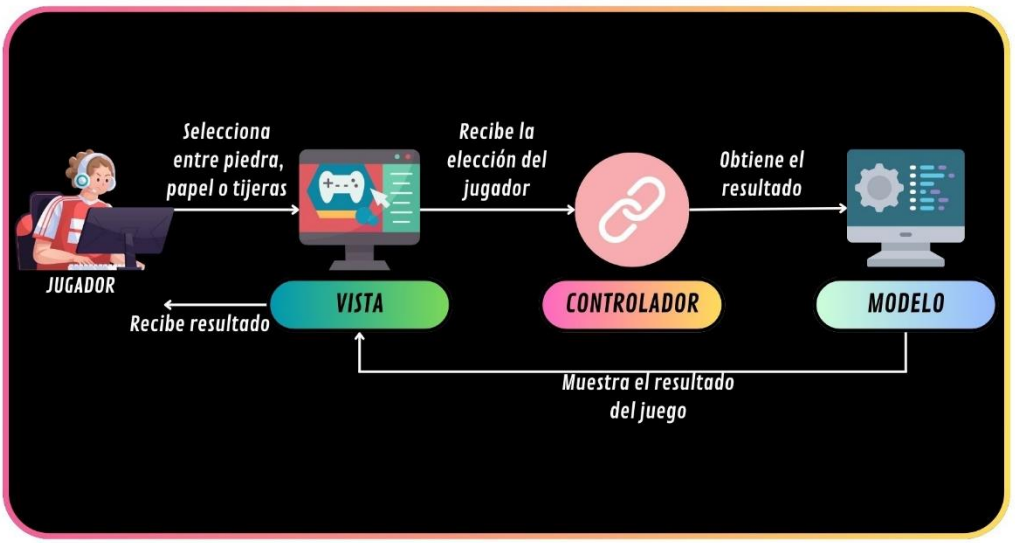
Diseño de diagrama de funcionalidad

Diagrama de caso de uso: permite modelar como los usuarios (actores), interactúan con el sistema, es una visión general para explicar un sistema a un público no técnico. (Lucidchart, 2018)



Diseño de diagrama de arquitectura de la aplicación

Modelo-Vista-Controlador (MVC): divide la aplicación en tres componentes: el modelo contiene los datos y la lógica, la vista se encarga de la presentación y el interfaz grafico de usuario y el controlador coordina las interacciones entre el modelo y la vista. Al tener los componentes separados permite que la aplicación sea fácil de corregir, mantener y expandir. (Hernandez, 2021).



Referencias:

Hernandez, R. (28 de Junio de 2021). *freeCodeCamp*. Obtenido de https://www.freecodecamp.org/espanol/news/el-modelo-de-arquitectura-view-controller-pattern/

Lucidchart. (7 de Febrero de 2018). *Lucidchart.* Obtenido de https://www.lucidchart.com/blog/es/tipos-de-diagramas-uml