Title-Exchange

Director(es):
Dianalin Neme Prada
Elkin Ferney Quintero Gomez
Ivan Rodrigo Romero Florez

Edison Elias Pelaez Ruiz
Mayo 2021

Universidad Antonio Nariño.
Facultad de ingeniería.
Especialización en ingeniería del software

Copyright © 2021 por Edison Peláez Ruiz. Todos los derechos reservados.

Dedicatoria

Dedico este proyecto a Dios el autor y consumador de la vida quien me permitió realizar esta especialización “Para infundir sagacidad en los inexpertos, conocimiento y discreción en los jóvenes, escuche esto el sabio, y aumente su saber, reciba dirección el entendido…”, a mi esposa e hija que son el motor en la vida para lograr un día mejor al anterior.

Abstract

Facilitate the exchange of digital video games, which is free and builds trust. According to a report from the newspaper El Tiempo, 48% of the software in Colombia is illegal (pirated), although the outlook is not good, at least the country is better located with reference to Venezuela, which has 89%. In the video game industry, it is not strange to talk about piracy openly, as was mentioned in the problem, the cost of an original video game is high, that is why many Gamers incur in piracy or the exchange of a video game, but with this the best solution is that we can make an exchange keeping the original game that belongs to us.

Tabla de Contenidos

Lista de tablas

Lista de figuras

Capítulo 1

La problemática

Descripción del problema

En la industria de los videojuegos hay mucho por ganar para las compañías y mucho por gastar para los usuarios, hablando de temas monetarios es una industria que económicamente percibe más ingresos que las industrias de la música y cine juntas, con lo cual se puede deducir que hay un gran grupo de Gamers dispuestos a usar su dinero para adquirir el último título estrenado en el año, pero hay un grupo aún más grande que tiene los siguientes problemas: 1. Capacidad de almacenamiento en la consola 2. Desinterés que se adquiere por un título después de haberlo jugado mucho 3. Mas importante aún es el costo que estos llegan a tener.

Figura 1 Costo del juego Mortal Kombat para Xbox One

Fuente: Ktronix.com

Generalmente una consola sale de fabrica con capacidad de almacenamiento entre 500GB y 1000GB esto suele ser engañoso ya que hay títulos como “Call of Duty: Warzone” O “Hallo Jefe Maestro” que llegan a pesar más de 100GB, Problema a la vista, quizás la solución sea adquirir un disco duro externo “Disco Duro Externo 2 Teras Toshiba Original USB 3.0 Estuche precio: $324.900”.

La experiencia de un Gamer se limita a cuánto tiempo se ha jugado a un título y sobre todo a que tan bueno se es en el mismo, la una va ligada a la otra, pero en el momento en el que aparece el desinterés, el juego sale a un segundo plano y se pierde en el olvido al punto de ser desinstalado de la consola ¿y el dinero invertido?, expone el principal problema al que se exponen los Gamers, al alto costo que los títulos tienen el comprar uno tras otro se convierte en el talón de Aquiles de aquellos Gamers sin tanto presupuesto.

Formulación del problema

¿Si existen tiendas físicas para el intercambio de videojuegos físicos, porque no puede existir un Web Site para el intercambio de videojuegos digitales que solucione el talón de Aquiles de los Gamers?

Capítulo 2

Los Objetivos

Objetivo General

Implementar un web site para el intercambio de videojuegos digitales que sea gratuito.

Objetivos Específicos

Crear una estructura de aplicación de alto nivel empleando el paradigma de programación orientado a objetos.

Implementar en el diseño los principios para la alta cohesión, bajo acoplamiento, encapsulamiento, polimorfismo y herencia.

Implementar módulo de PQR para que los usuarios puedan poner sus peticiones y/o sugerencias sobre el sitio web.

Digitalizar el intercambio de videojuegos de manera segura teniendo un validador (Externo async) que se encargue de la validez del título.

Generar el interés de los Gamers por títulos nuevos, a través de notificaciones de correo electrónico según sus gustos.

Crear un software de calidad, lo que implica que se apliquen los siguientes ítems sobre el software; buenas prácticas de desarrollo, refactorización y escalabilidad.

Capítulo 3

El marco de referencia.

Estado del arte

"Somos una comunidad de video jugadores dedicada a la venta, intercambio y noticias del mundo de los videojuegos. Nació de la idea de ofrecer videojuegos, consolas y accesorios sumado con una excelente atención al cliente, compartir conocimientos y asesoría responsable".

"IntercambioJuegos.cl es un Portal online de intercambio de videojuegos de segunda mano, es una comunidad donde puedes intercambiar tus juegos que no utilizas con otros usuarios".

Figura 2 Portal intercambios, página web para el intercambio juegos

fuente: ColombiaGamer.com

Figura 3 Logo Colombia Gamers

fuente: ColombiaGamer.com

Impacto en la sociedad

Impactar la sociedad Gamer es un poco difícil, puesto que esta se basa en las generaciones, gráficas, jugabilidad de los títulos y que tan adictivos pueden ser, entonces un intercambio de videojuegos no llega a ser tan impactante como se quisiera, aunque siempre suele aparecer un "pero…" en cualquier circunstancia de la vida y en esta ocasión no es la excepción.

El intercambio en físico existe de hace un tiempo, y el intercambio digital es una alternativa en Facebook o blogs para Gamers. Pero un website que gestione el intercambio de títulos digitales no existe ¡hasta ahora!, facilitar la vida del Gamer brindándole seguridad, confianza es el impacto a lograr con este proyecto

Impacto personal

El impacto personal de este proyecto se limita a la plena satisfacción de saber que el poco o mucho conocimiento que tenga un Gamer ingeniero de sistemas puede ser puesto a disposición y ayuda de la comunidad Gamer.

Cifras a nivel nacional

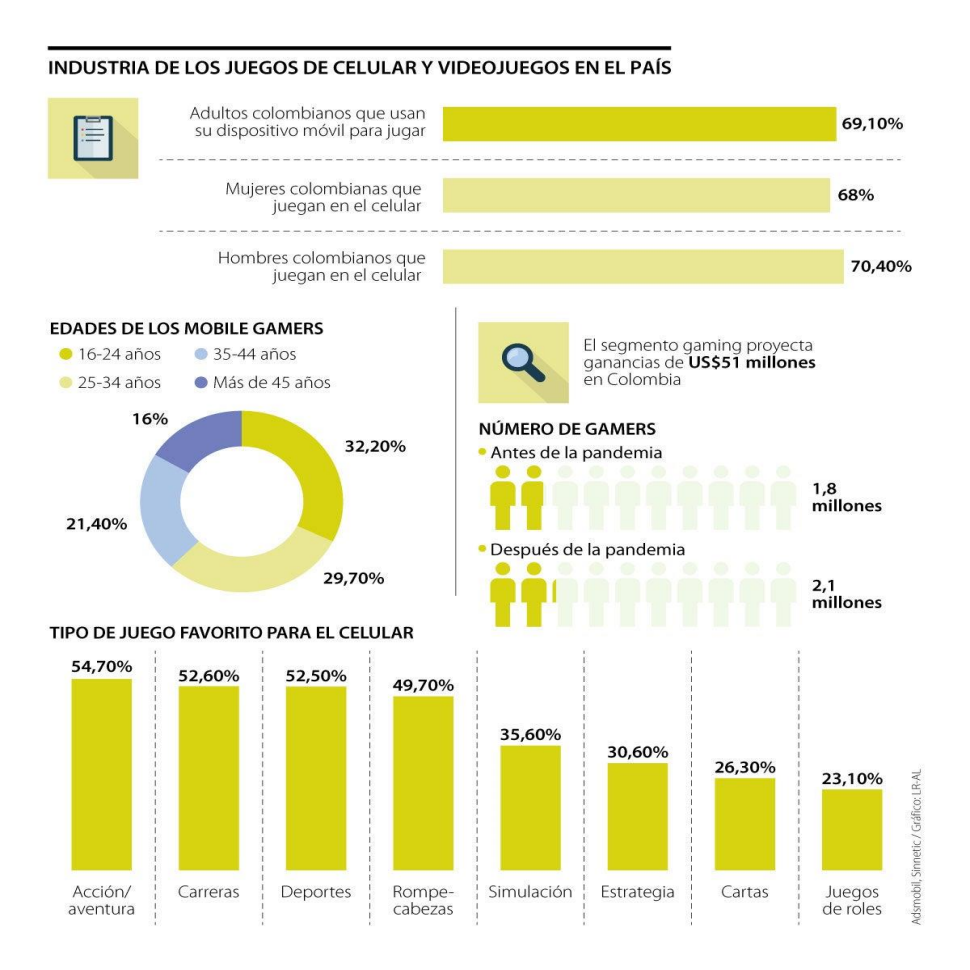


Figura 4 Cifras nacionales de la industria de los videojuegos en Colombia

Fuente: laRepublica.co

Crecimiento de la industria en Colombia

El apartado de los videojuegos en Colombia es bastante rentable para la industria Gamer, *"según el señor Gabriel Contreras, quien es CEO de Sinnetic comenta que la según las cifras a nivel mundial, la tendencia de la industria es al alza. En juegos móviles en celulares crece a 14%, en venta de consolas de videojuegos crece a 16%, de juegos por CD incrementa 8% y los juegos colaborativos online crecen 17%".*

Componente de innovación

Facilitar el intercambio de videojuegos digitales, que sea gratuito y genere confianza. Según un informe del periódico el Tiempo el 48% del software en Colombia es ilegal (pirata), aunque el panorama no es bueno por lo menos el país se encuentra mejor ubicado con referencia a Venezuela quien tiene un 89%. En la industria de los videojuegos no es nada extraño hablar de la piratería abiertamente, así como se mencionó en la problemática el coste de un videojuego original es elevado, por eso muchos Gamers incurrimos en la piratería o el intercambio de un videojuego, pero con esta solución lo mejor es que podemos hacer un intercambio conservando el juego original que nos pertenece.

Pero quiero ser claro y que en ningún momento quiero fomentar la piratería, como ya es conocido Xbox tiene su política clara de videojuegos, adjunto un código QR en el cual están las políticas oficiales:

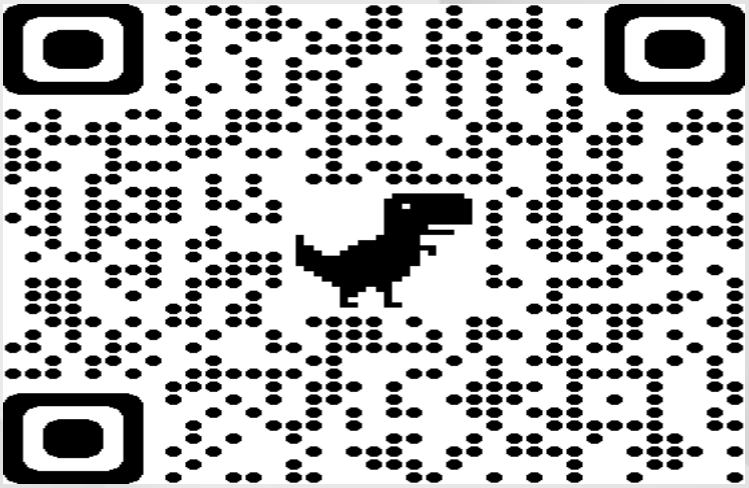


Figura 5 Código QR, para acceder a las políticas de Xbox

Fuente: propia

Capítulo 4

Marco Teórico.

Videojuegos

En breve la historia de los videojuegos se remonta a su inicio en el 52 cuando Alexander S. Douglas computarizo el juego a tres en raya. Mas adelante en el 58 William hizo el "Tennis for two" para entretener los visitantes de la exposición "Brookhaven National Laboratory", magnavox odyssey fue el primer sistema domestico de videojuegos creado por Ralph Baer, Albert Maricon y Ted Dabney en 1966.

Figura 6 Juego a tres rayas

Fuente: es.wikipedia.org

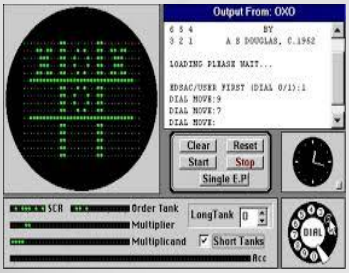


Figura 7 Juego a tres rayas de alexander d

Fuente: exevi.com

Figura 8 la primera consola Magnavox Odyssey

Fuente: parceladigital.com

Figura 9 Primer juego tennis for two

Fuente: muycomputer.com

Sin el ánimo de saltarse la historia o de profundizar mucho en ella, se encuentran videojuegos que han marcado la historia y que muchas generaciones conocen, tal es el caso de: PAC-MAN, Mario Bros, SEGA, Tetris, Call of Duty, Halo entre otros. Un videojuego es un software de interacción continua entre maquina (imágenes) y jugador, no llega a ser más complejo que definirlo como juego electrónico.

Videoconsolas

La consola es un sistema electrónico para el hogar que ejecuta videojuegos. Desde sus inicios al día de hoy las consolas han ido evolucionando con el pasar de los tiempos, numerosas marcas han pasado por el mundo de los videojuegos construyendo sus consolas propias, tal es el caso de Microsoft, Sony, Nintendo entre otras más, con sus conocidas consolas: Xbox, Xbox 360, Xbox One, PlayStation 2,3,4

Figura 10 Consola Xbox One

Fuente: parceladigital.com

Figura 11 Consola Xbox 360

Fuente: parceladigital.com

Figura 12 Consola PlayStation 3

Fuente: parceladigital.com

Figura 13 Consola PlayStation 4

Fuente: parceladigital.com

Generaciones

Las generaciones en el mundo de los videojuegos son primordiales ya que están imparten el punto de partida entre una época y otra, lo que conlleva que llegue mejoras en el apartado gráfico, potencia y nuevas opciones de jugabilidad de las consolas. En primera generación esta la ya mencionada Odyssey de Magnavox en 1972, La segunda generación da inicio con la llegada de consolas para televisores a color y de 8 bits, la Atari 2600 fue el referente de la generación, pero aproximadamente hubieron más de 8 o 9 consolas más que hacían la vez de competencia.

Figura 14 Consola Atari 2600

Fuente: google.com

Tercera generación Aparece con fuerza Nintendo con su Game Boy de 1989 pero inicio con 1983 cuando Nintendo lanzo la Famicom (NES).

Figura 15 Consola Game Boy

Fuente: google.com

Figura 16 Consola Famicom

Fuente: google.com

La cuarta generación Lo más relevante fue la consola Mega drive (Genesis) de SEGA, la más vendida SNES de Nintendo. En la Quinta Generación nace la icónica PlayStation de Sony que se convertiría en uno de los estandartes de los videojuegos, pero sin dejar de lado la famosa Nintendo 64.

Figura 17 Nintendo 64

Fuente: google.com

Figura 18 PlayStation

Fuente: google.com

Figura 19 Consola PlayStation 2

Fuente: google.com

Figura 20 Consola Xbox

Fuente: google.com

Sexta generación da a luz la rivalidad entre las tres grandes Sony, Microsoft, Nintendo quien de aquí en adelante comenzarían la batalla sin fin, modelos que suenan con fuerza PlayStation 2, Xbox, Game Cube

C#

C Sharp en ingles lenguaje de programación multiparadigma, lenguaje elegido por su la adaptación que tiene al paradigma de programación orientada a objetos, es sencillo de utilizar, moderno es adaptable a casi cualquier tipo de aplicaciones que se desee construir, en este caso servirá como lenguaje de desarrollo para el back y frontend

Net Core 3.1

Versión más reciente de net core, (LTS) con soporte a largo plazo esto indica que tendrá un soporte más longevo que sus antecesores, net core es Opensource esto facilita su manejabilidad y adaptabilidad con distintos sistemas operativos como Windows, Linux o Mac, permite ser escalable al estar separado por paquetes, es muy eficaz en el desarrollo de microservicios

Blazor

Marco de trabajo para crear una interfaz de usuario web interactiva del lado del cliente. Básicamente remplaza a Java Script por C# para hacer tareas interactivas, funciona en todos los navegadores actuales, se pueden asignar tareas sencillas dejándolas del lado del cliente y su principal cualidad es ser Opensource

SQL Server

Motor de base de datos confiable y seguro, capaz de soportar el ligero flujo de información que conlleva este proyecto en su versión Express. Este motor es compatible con todas las tecnologías (Microsoft) elegidas en este proyecto

UML

Lenguaje unificado de diagramación. Esencial para diagramar los casos de uso, de clases,

Formato de los videojuegos

Los videojuegos existen en dos formatos el físico (CD) y el digital este último se adquiere desde un store, que cambia dependiendo la compañía, se compra vía internet, se obtiene un código y este a su vuelta trae una clave de descarga única para cada usuario.

Intercambio

El principal problema que existe los intercambios es que están diseñados o pensados para el formato físico de los videojuegos dejando a un lado el formato digital.

Si existen algunos lugares físicos para el intercambio, así como páginas web, foros, Facebook entre otros pero su atención está en el formato físico, aparte cabe mencionar el mercado de la piratería que abunda mucho en estos tiempos, y esto hace que un juego original se consiga por menos del 10% del valor real en los mercados negros, nuevamente con el formato físico, y el formato digital elimino casi por completo el problema de la piratería pero añadió otro al Gamer, el costo del videojuego, Microsoft sabiendo de esto implemento una política para que los usuarios compartieran sus juegos en consolas de sus amigos, problema solucionado?, no, porque hay Gamer que usan diferentes consolas y eso hace que mi círculo de amigos se convierta en estrecho, y al momento de entrar a alguien con quien pueda intercambiar es difícil, de allí nace la idea de solucionar el problema con intercambios seguros de los títulos

Títulos

Los títulos son la representación figurativa, del videojuego digital, de allí es donde nace la idea de intercambio de títulos en inglés, la solución es simple, un usuario publica el título de un videojuego que requiere cambiar y otro usuario en la plataforma envía una solicitud de cambio por otro título y este podrá rechazarla o aceptarla, en dicho caso se comparten las credenciales de acceso a la cuenta de Microsoft para que el otro usuario pueda utilizar el videojuego en formato digital.

Capítulo 5

Metodología.

Kanban

El término Kanban proviene del japonés y significa algo como tarjeta, cartón o recibo, pero también contenedor. El Kanban, desarrollado en 1947 por Taiichi Ohno, el inventor del sistema de producción de Toyota, es una implementación del método de control conocido como principio pull o hol en la producción o fabricación. Se inspiró en el procedimiento en el supermercado: los consumidores se sirven a sí mismos y los empleados (proveedores) aseguran suficientes existencias en los estantes (almacenamiento intermedio) en función de la salida de mercancías. Una vez que los estantes están llenos, el ciclo comienza de nuevo. Como gerente de Microsoft, David Anderson transfirió el concepto a la tecnología de la información a mediados de la década de 2000 para poder llevar a cabo no solo procesos de producción, sino también proyectos de manera más rápida y eficiente. Junto con Scrum, Kanban es uno de los métodos de gestión ágil más utilizados.

Figura 21 Tablero Kanban de cómo se aplicó en una herramienta online

Fuente: propia

En la imágen anterior se explica lo siguiente: la imagen 22 tablero kanban muestra la aplicación de la metodología mediante una plataforma online (kanbantool), en dicha plataforma se crear una cuenta y ella predeterminadamente tiene un tablero con tres sesiones para realizar la aplicación de la metodología es decir se compone de la siguiente manera.

Columna de: “que hacer”, en esta columna se describen todas las tareas a relazarse dentro del proyecto de desarrollo o de gestión del proyecto, en esta parte se debe escribir todo lo relacionado con las tareas pendientes.

En la segunda sección se encuentra la columna de: “haciendo”, en esta sesión se relacionan las tareas que esta siendo desarrolladas en el momento, por lo cual se deben hacer y tener cuidado en no saturar el numero de tareas en esta parte.

Por ultimo se encuentra la columna de: “hecho”, en esta session se almacenan las tareas que están en estado culminado, es decir que ya se terminaron y dan paso que la sesión anterior se desocupe

Capítulo 6

Proceso de Software.

Descripción

En este capítulo aborda la manera de cómo se planeó el proyecto tanto para elegir la arquitectura, como para definir requerimientos funcionales y no funcionales, los respectivos diagramas de despliegue, secuencia clases, arquitectura de alto nivel, caso de uso relevante, con cada uno se logra un fin en específico para dar cumplimiento al proyecto, es decir todo lo mencionado anteriormente es relevante para el proyecto porque es la estructura principal para lograr la solución.

Requerimientos funcionales

Requerimiento funcional define lo que esperamos que deba hacer un sistema allí se describe la interacción entre el sistema y el entorno se detallan los servicios o funciones que proveerá el sistema por ejemplo este es un requisito funcional que detectamos con nuestro cliente algo que necesitamos es que el sistema deberá generar los reportes de los lugares con mayor riesgo de accidente y ese es un requisito funcional igual de importante.

El sistema enviará un correo electrónico cuando se registre alguna de las siguientes transacciones: registro del cliente, registro intercambio, finalización del cambio.

El sistema permitirá a los usuarios el ingresar la información descriptiva de los videojuegos y así mismo actualizarlos.

El sistema permitirá al usuario autorizado el aprobar y/o rechazar intercambios.

El sistema permitirá a los usuarios el solicitar el intercambio de un videojuego por otro, del gusto.

El sistema permitirá a los usuarios la actualización de la información registrada en su perfil personal.

Requerimientos no funcionales

Requerimiento no funcional el cual define cómo debe ser el sistema describe restricciones que limitan las elecciones para construir una solución por ende son atributos relacionados con la calidad como rendimiento escalabilidad fiabilidad disponibilidad mantenimiento seguridad etcétera un ejemplo el requerimiento no funcional puede ser el lenguaje de programación debe ser ya van alta velocidad de procesamiento de datos o una interfase gráfica de fácil lectura. En la siguiente tabla se describen los requerimientos no funcionales que están planeadas para este proyecto.

N°	Nombre	Descripción
1	Flexibilidad	Funcionamiento por módulos.
2	Usabilidad	El sistema de intercambio es capaz de cumplir los objetivos deseados.
3	Testeabilidad	Todos los módulos son aprobados utilizando el plan de pruebas, cada módulo depende de otro para mostrar y almacenar información.
4	Concurrencia	El sistema es capaz de registrar múltiples equipos de forma recurrente.
5	Tolerancia a fallos	Implementación de logs para tener un registro del sistema.

Tabla 1 Requerimientos no funcionales del Proyecto

Fuente: propia

Diseño y arquitectura

El diseño arquitectural elegido para desarrollar este sistema es el tripe D (Domain Driven Design) en donde el centro de la aplicación es el negocio, y una arquitectura en capas, en el grafico a continuación se describe la arquitectura y diseño a seguir para la construcción del software.

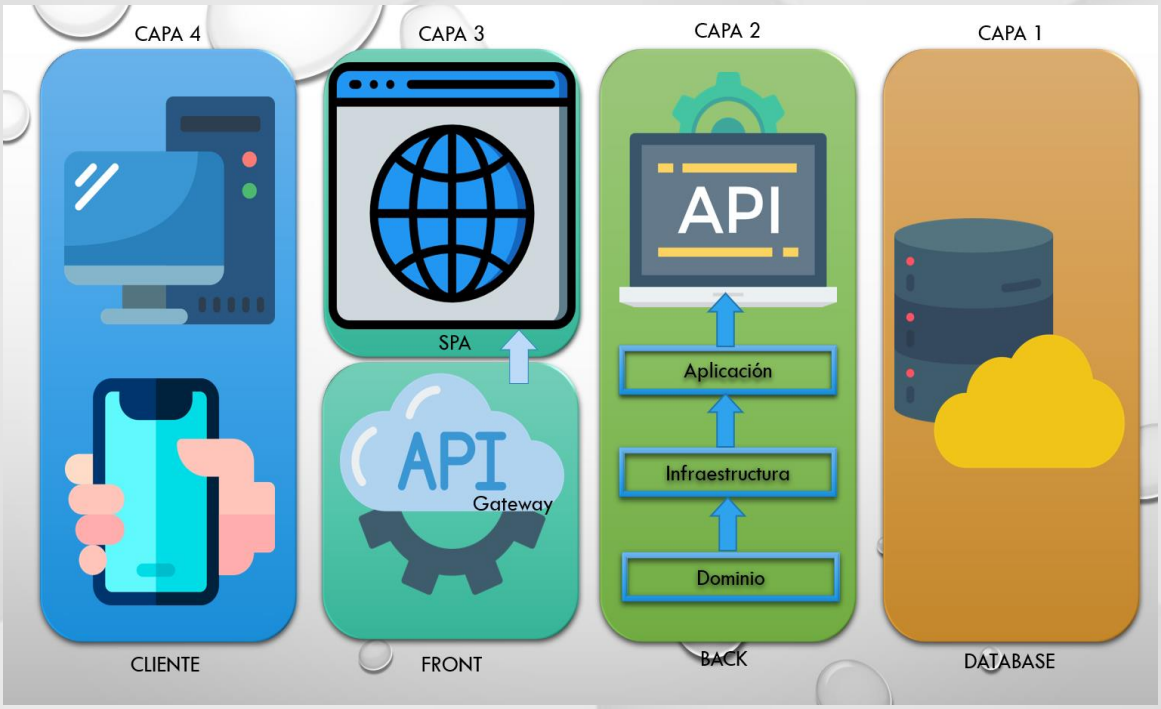


Figura 22 Diseño de Arquitectura Domain driven design

Fuente: propia

CAPA 1: En esta capa esta la persistencia a base de datos motor SQL SERVER.

CAPA 2: En esta capa esta embebida toda la lógica del back, es decir, el núcleo del negocio es el **dominio**, el acceso a data mediante el patrón repositorio esta es la **infraestructura**, y reglas de validación y lógica de componentes en **aplicación**, todo lo anterior se expone para su consumo en un **API**

CAPA 3: En esta capa, está la presentación dividida en dos partes, como primera medida un api Gateway que sea una fachada para orquestar la comunicación del front con el back, en segundo lugar, un SPA donde está la vista del cliente en el navegador.

Diagrama de despliegue

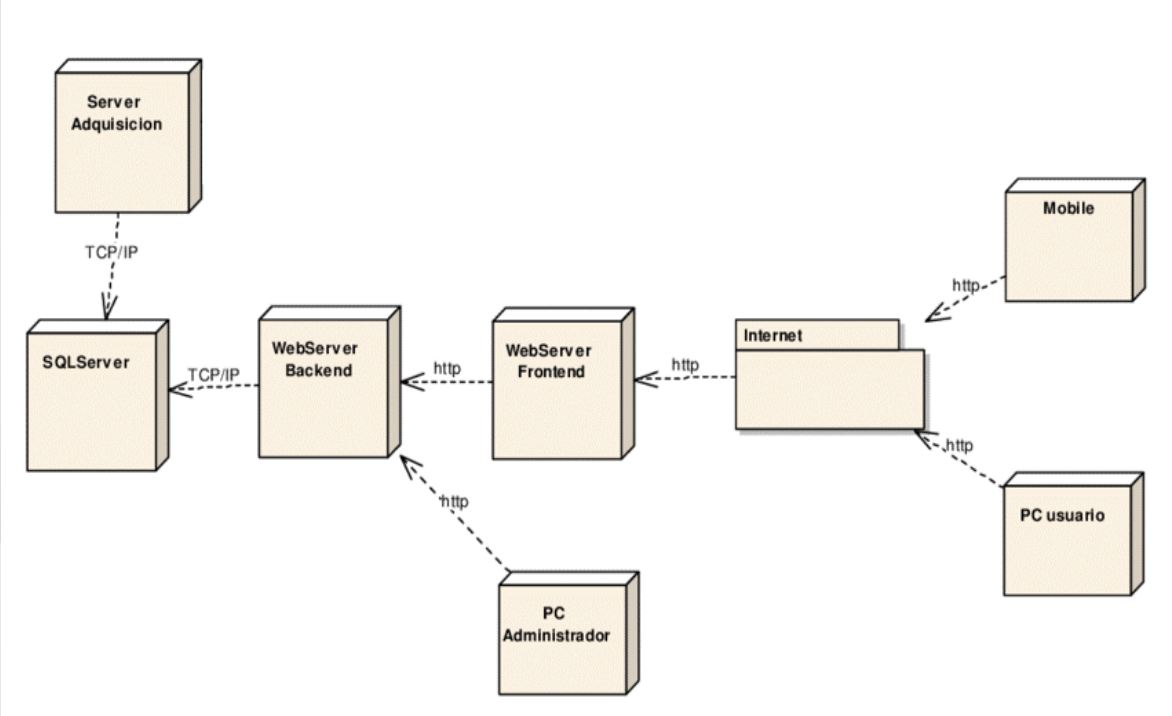


Figura 23 Diagrama de despliegue de la solución

Fuente: propia

Los dispositivos como móvil o pc se comunicarán con la aplicación mediante el explorador de internet, allí se aloja el FontEnd, este a su vez tendrá comunicación con el Backend, el back solo hará peticiones al motor de base de datos para la persistencia de datos.

Caso de uso arquitecturalmente relevante

El Caso de uso Cambiar título es el más relevante debido a que la razón de ser del sistema pasa directamente por este caso, se busca que un usuario pueda elegir entre los títulos disponibles e

intercambie su título por el de la elección, llenando el formulario de intercambio y enviando la solicitud al otro usuario para la aprobación

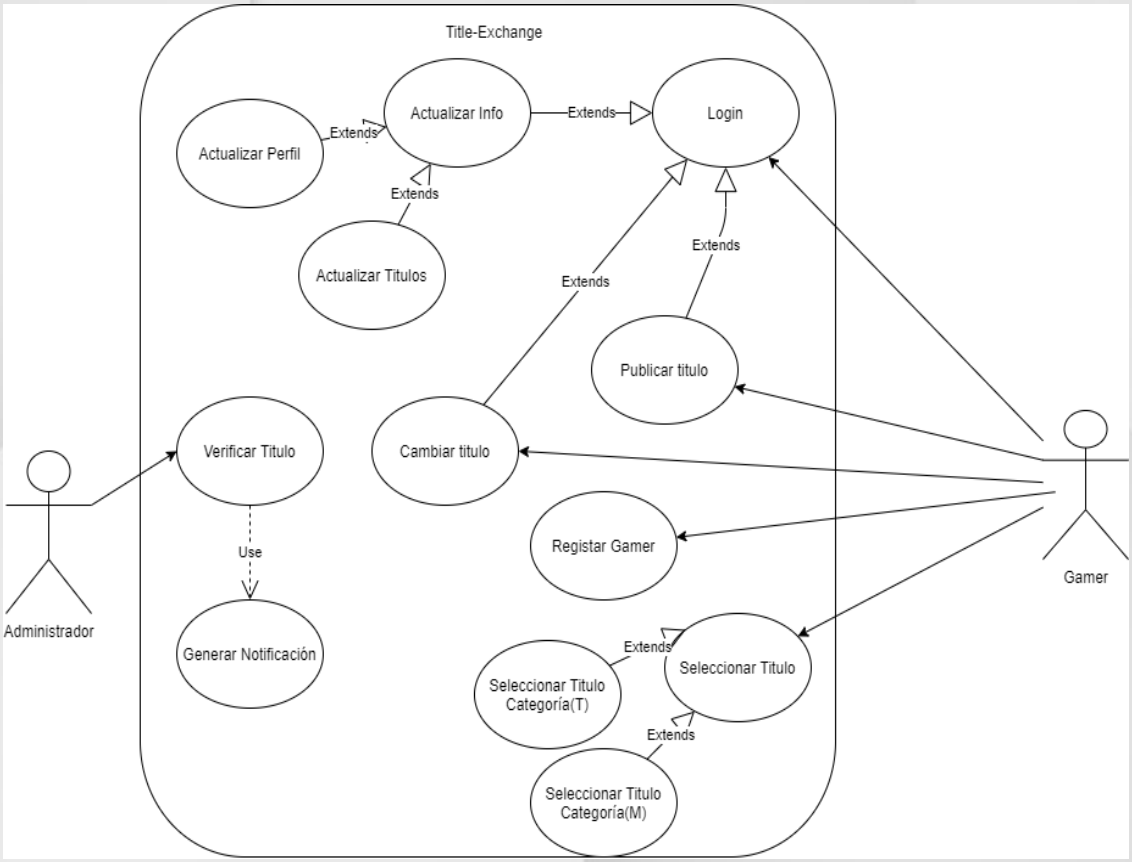


Figura 24 Diagrama de casos de uso del Proyecto

Fuente: propia

Diagrama de secuencia

Actor es el equivalente a usuario, este ingresa al sistema. Debe realizar el registro en el sistema al ingresar por primera vez, consulta el catálogo de los videojuegos publicados, elige un videojuego y solicita el intercambio, finaliza la tarea

Administrador Consulta catálogos, actualizar los catálogos y por último aprueba los intercambios siendo el validador externo Async

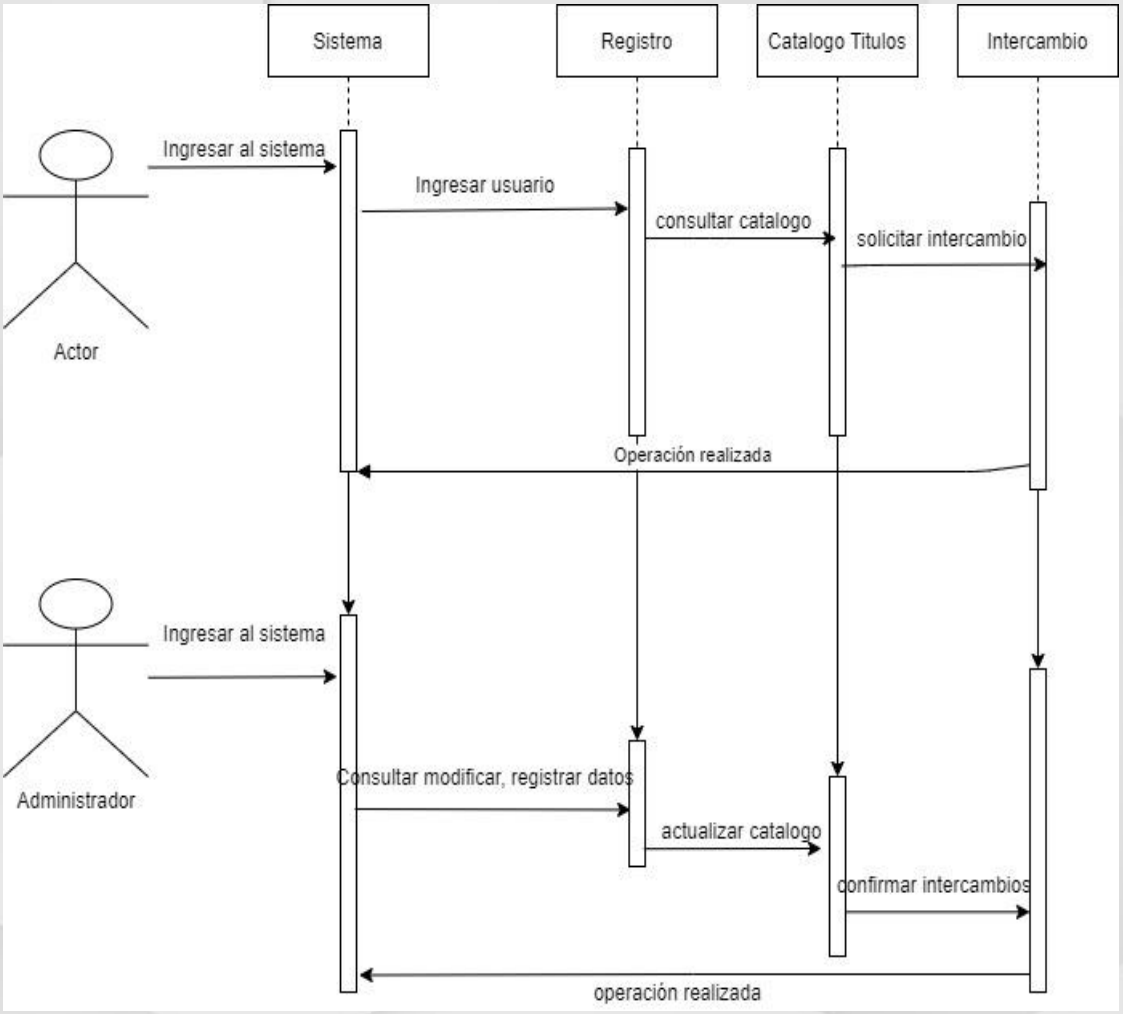


Figura 25 Diagrama de secuencia del proyecto

Fuente: propia

Diagrama de clases

En siguiente diagrama se definen las clases en donde se exponen de la siguiente manera Clase Usuario se relaciona con el administrador y con Gamer, un Gamer puede tener múltiples cambios, un

cambio tiene un título que está dentro de una categoría, el usuario administrador estar encargado de aprobar o rechazar los intercambios por validez de data

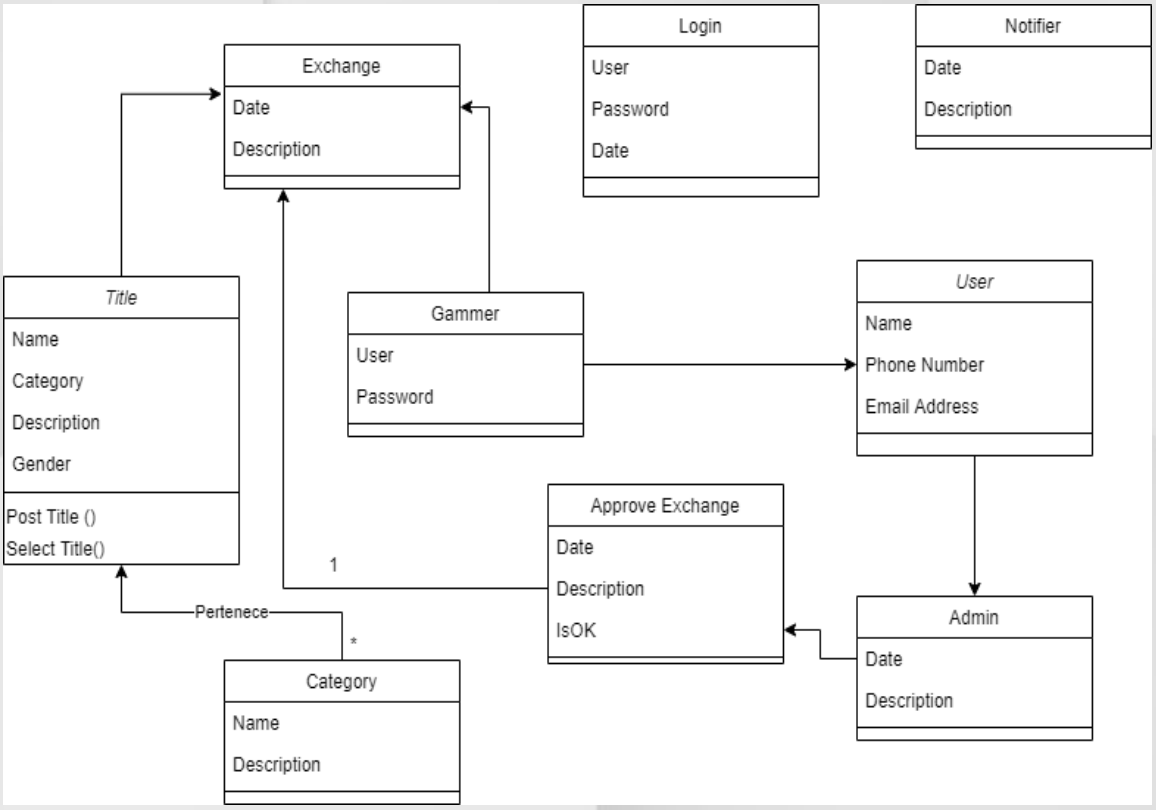


Figura 26 Diagrama de clases del proyecto

Fuente: propia

Arquitectura de alto nivel

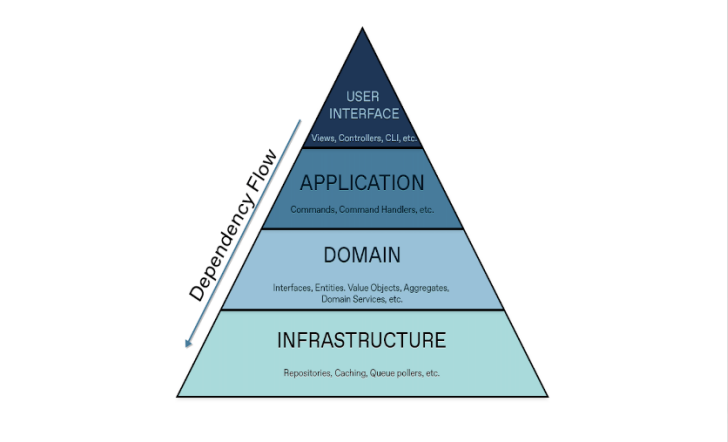


Figura 27 Diseño piramidal del estilo arquitectural Domain Driven Desing

Fuente: medium.com

En la imagen anterior se explica a alto nivel el flujo del diseño de arquitectura DDD, en donde en la base del sistema es la infraestructura allí se encuentran los repositorios entre otros que se encargan de sostener el diseño, en el siguiente nivel se encuentra el dominio donde se almacenan las interfaces, entidades, objetos de valor, servicios de dominio etc…. En el siguiente nivel se ubica la aplicación es decir comandos o la llamada lógica de negocio, por último, estaría la interfaz de usuario donde se tendría las vistas, controles y demás. La flexibilidad en este diseño permite usar patrones de desarrollo como; Repositorio en la persistencia de datos, inyección de dependencias. Este diseño se basa en el dominio ignorando en todo sentido la tecnología de aplicación es decir este proyecto en implementación no necesita de una tecnología en específico es abierta a implementarse en cualquiera. Las clases mencionadas en el diagrama de clases se encuentran en el dominio, en infraestructura se encuentran los repositorios y el contexto, por último, aplicación contiene los servicios con la lógica, esta es la mejor manera de ilustrar la manera en la cual se implementa, ya que no se ha mencionado en ningún momento un lenguaje como java, Python, C, C# etc…. o marcos de trabajo como .Net, BootSprint y otros. Lo importe es definir el flujo de trabajo según las entidades y procesos que requiera el sistema.

Capítulo 7

Construcción.

Estructura de la solución

Lo siguiente es desarrollado dentro del IDE Visual Studio Community, también se muestran las capas que se construyeron, capa de dominio donde se ubican los modelos que representan las entidades de la base de datos y adicionalmente las interfaces del patrón repositorio para el acceso a los datos. Capa de infraestructura tiene la dependencia al dominio, guarda la lógica de construcción de tablas, tiene el contexto para conectar a la base de datos con el framework, Entity Framework Core, a su vez la lógica de los repositorios. La capa de aplicación tiene dependencia de dominio e infraestructura se agregan los servicios e interfaces para ser expuestas al API.

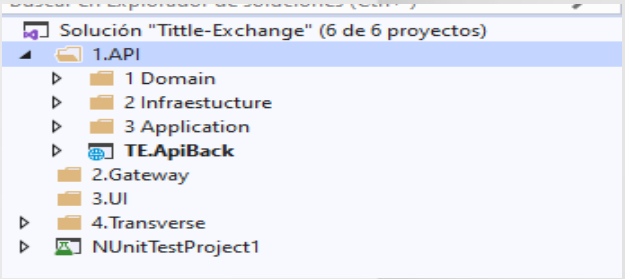


Figura 28 Arquitectura general y distribución del código en el proyecto

Fuente: propia

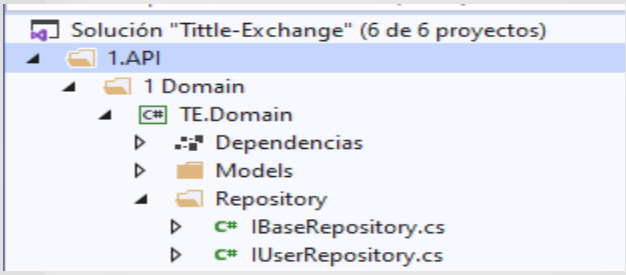


Figura 29 Dominio, no depende de infraestructura ni aplicación

Fuente: propia

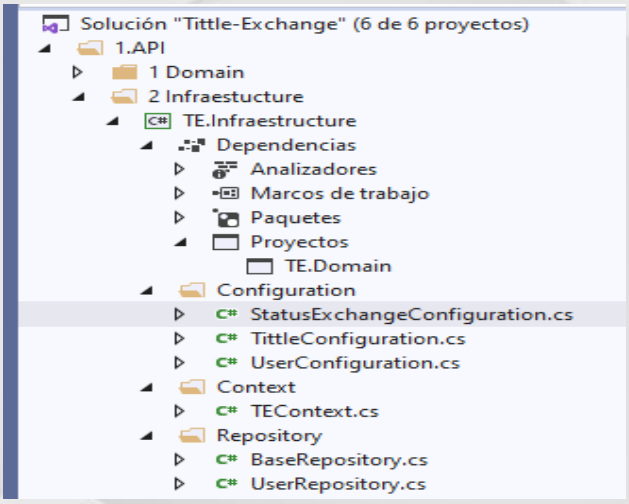


Figura 30 Infraestructura, Depende de dominio, pero no depende de aplicación

Fuente: propia

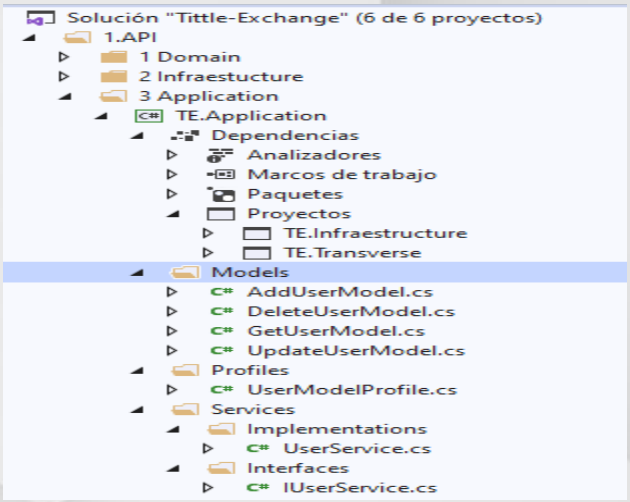


Figura 31 Aplicación, contiene la lógica del negocio y depende de dominio e infraestructura

Fuente: propia

Capítulo 8

Pruebas.

Pruebas Backend

Las pruebas en el apartado del back se realizan con el criterio de buscar errores no controlados a modelos de entrada incorrectos, cambios en la lógica por los campos y el comportamiento en general, todo esto ayudado del framework swagger para el API del back, desde allí se podrá probar los modelos de entrada y las repuesta que el back respondería en diferentes escenarios

Integridad en los datos, en la siguiente prueba se adjunta una imagen en donde se demuestra la recepción de data inesperada, es decir se esta ingresando un valor de cadena para una entrada de tipo numérica,

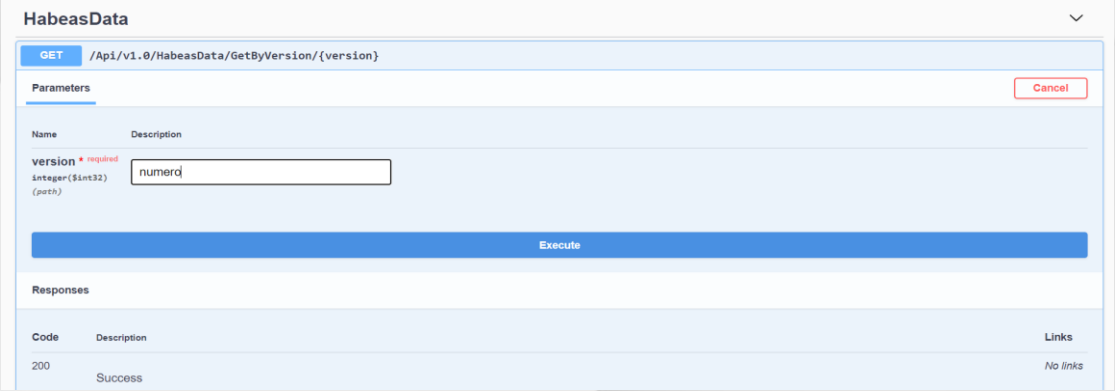


Figura 32 Back, prueba de endpoint habeasData

Fuente: propia

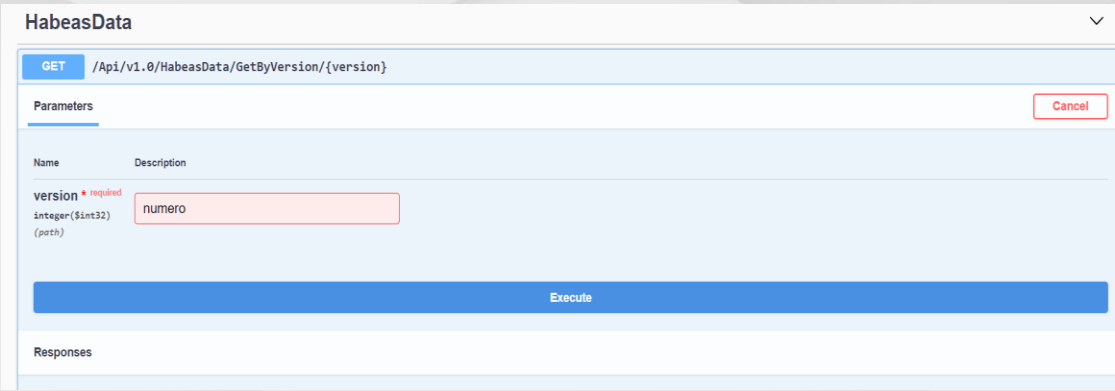


Figura 33 Back, prueba de endpoint habeasData con error

Fuente: propia

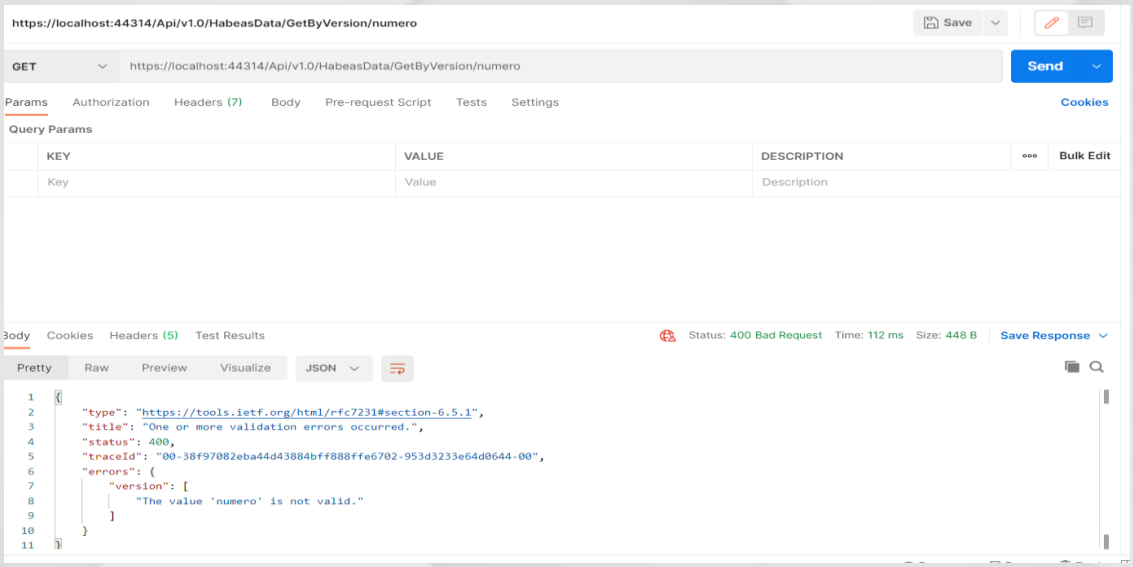


Figura 34 Back, prueba de endpoint habeasData desde postman

Fuente: propia

En la siguiente prueba se envían datos no esperados, igualmente el sistema tiene una respuesta adecuada para que siempre halla disponibilidad en el back

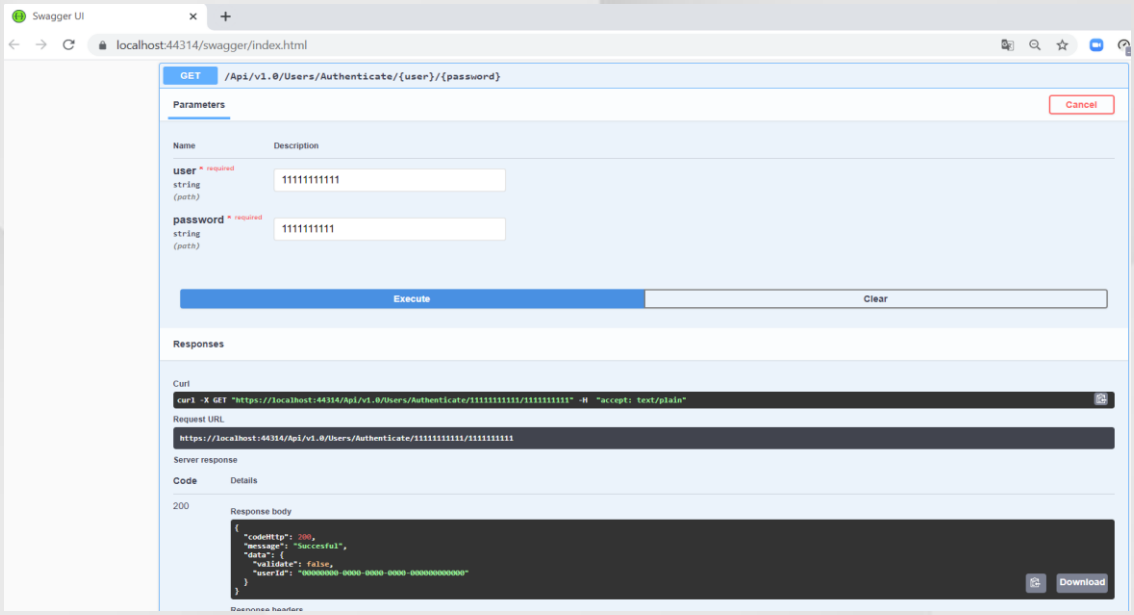


Figura 35 Back, prueba de endpoint authenticate para un usuario

Fuente: propia

Errores no controlados por el sistema, en el siguiente caso se producirá una excepción no contralada con el fin de ver que responde el sistema. En la respuesta controlada se devuelve el mensaje de error en el sistema. Esto ocurrirá en cualquier endpoint debido al uso de una clase middleware para la recepción de errores no controlados.

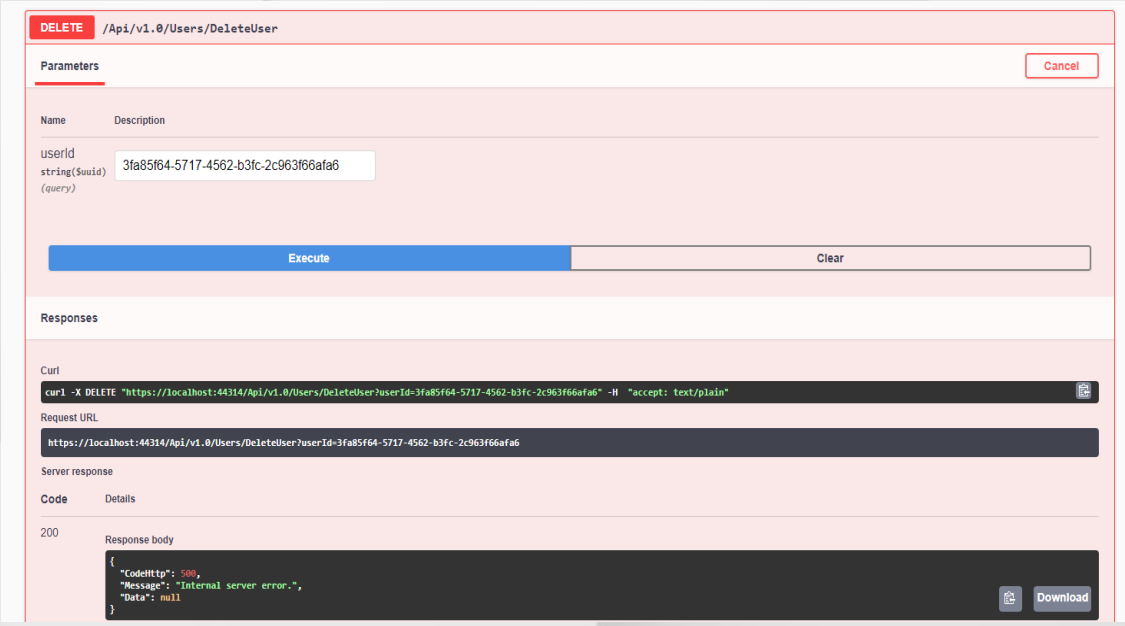


Figura 36 Back, prueba de endpoint DeleteUser con un error no controlado por sistema

Fuente: propia

Estructura de respuesta para todos los casos es igual, siendo dinámica la data de cada endpoint

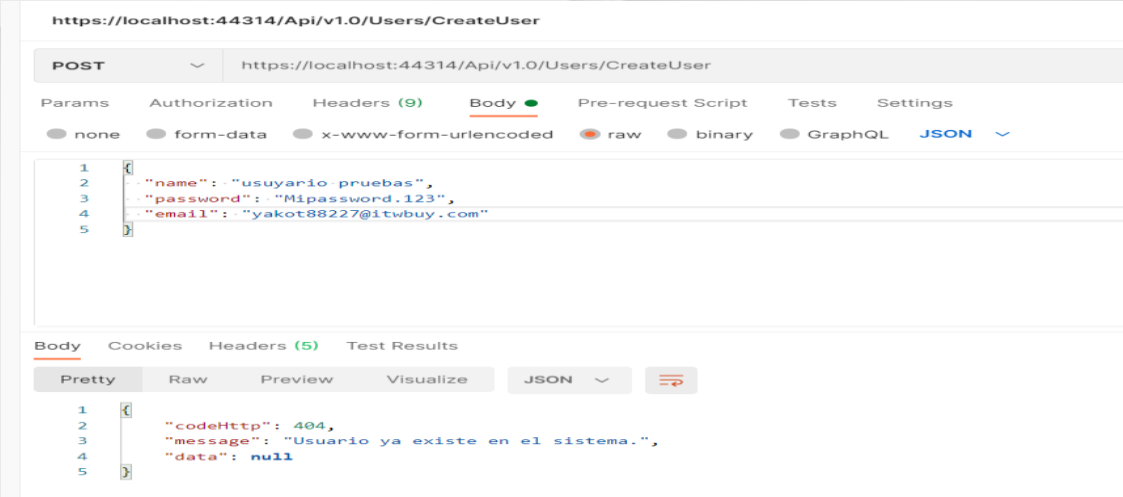


Figura 37 Back, prueba de endpoint CreateUser desde Postman

Fuente: propia

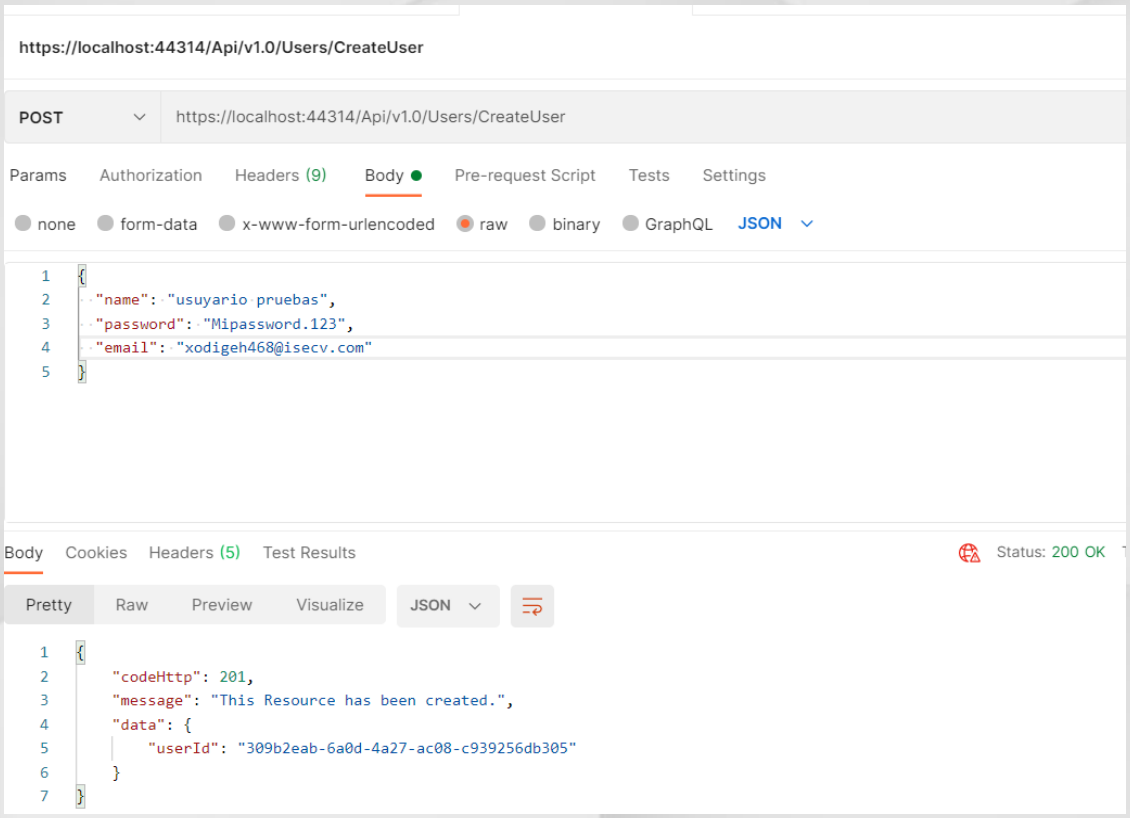


Figura 38 Back, prueba de endpoint CreateUser desde Postman satisfactoria

Fuente: propia

Pruebas Frontend

La idea de pruebas en el front es basarse en que la comunicación de front y back este correcta, y el mapeo de los datos sea correcto desde el front para que envié los datos de manera correcta, y así mismo los lea con el modelo esperado según corresponda

En primera medida se prueba que el front consulte al back por los términos y condiciones para los usuarios.

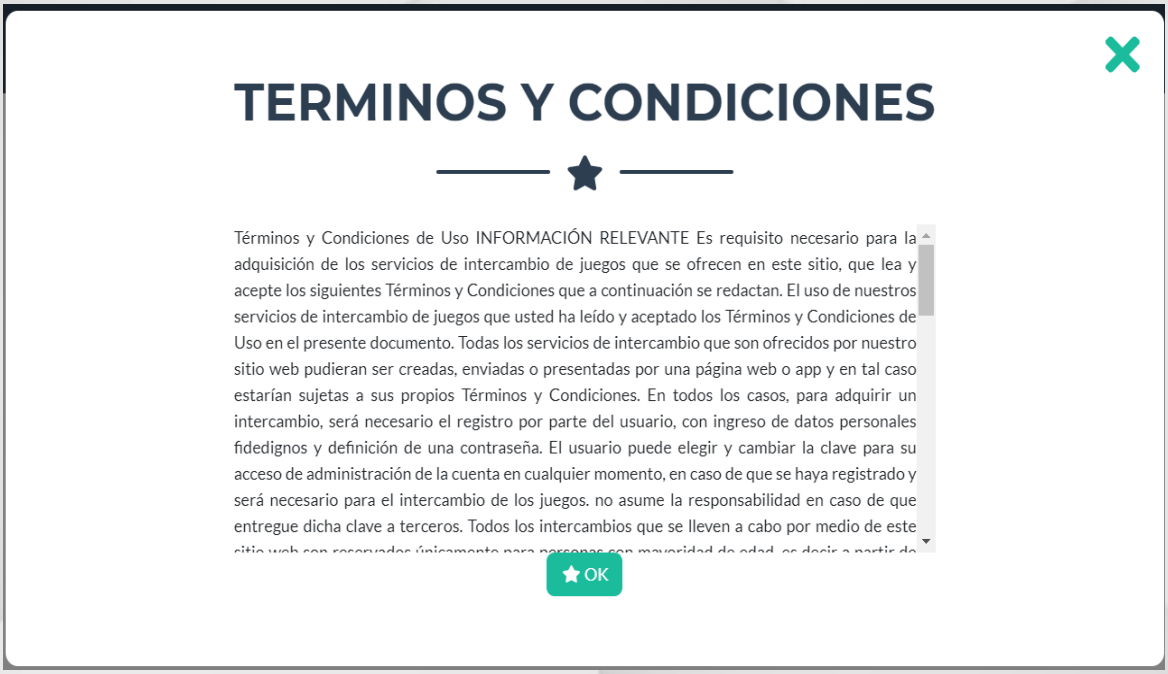
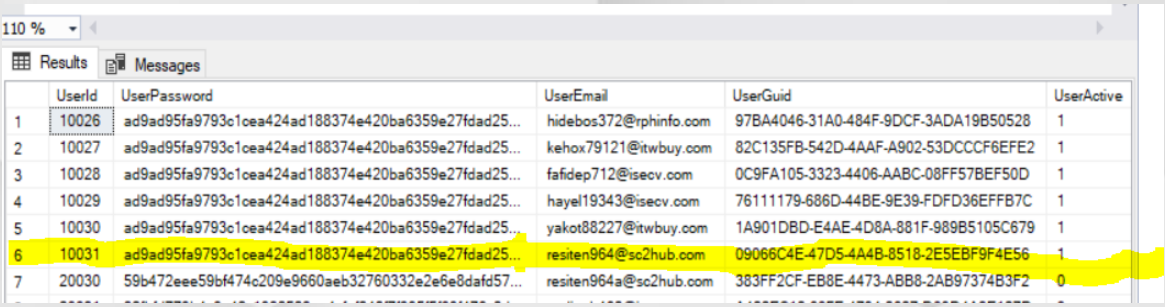


Figura 39 Front, modal términos y condiciones de la pagina

Fuente: propia

Seguido a esto se hace una petición al back para registrar un usuario nuevo.



	UserId	UserPassword	UserEmail	UserGuid	UserActive
1	10026	ad9ad95fa9793c1cea424ad188374e420ba6359e27fdad25...	hidebos372@rphinfo.com	97BA4046-31A0-484F-9DCF-3ADA19B50528	1
2	10027	ad9ad95fa9793c1cea424ad188374e420ba6359e27fdad25...	kehox79121@itwbuy.com	82C135FB-542D-4AAF-A902-53DCCCF6EFE2	1
3	10028	ad9ad95fa9793c1cea424ad188374e420ba6359e27fdad25...	fafidep712@isecv.com	0C9FA105-3323-4406-AABC-08FF57BEF50D	1
4	10029	ad9ad95fa9793c1cea424ad188374e420ba6359e27fdad25...	hayel19343@isecv.com	76111179-686D-44BE-9E39-FDFD36EFFB7C	1
5	10030	ad9ad95fa9793c1cea424ad188374e420ba6359e27fdad25...	yakot88227@itwbuy.com	1A901DBD-E4AE-4D8A-881F-989B5105C679	1
6	10031	ad9ad95fa9793c1cea424ad188374e420ba6359e27fdad25...	resiten964@sc2hub.com	09066C4E-47D5-4A4B-8518-2E5EBF9F4E56	1
7	20030	59b472eee59bf474c209e9660aeb32760332e2e6e8dafd57...	resiten964a@sc2hub.com	383FF2CF-EB8E-4473-ABB8-2AB97374B3F2	0

Figura 40 BD, Registro de usuario en la tabla usuario

Fuente: propia

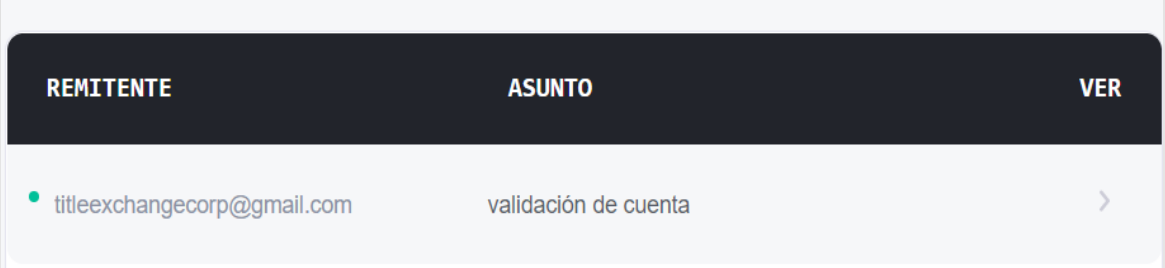


Figura 41 Notificación de validación de cuenta

Fuente: propia

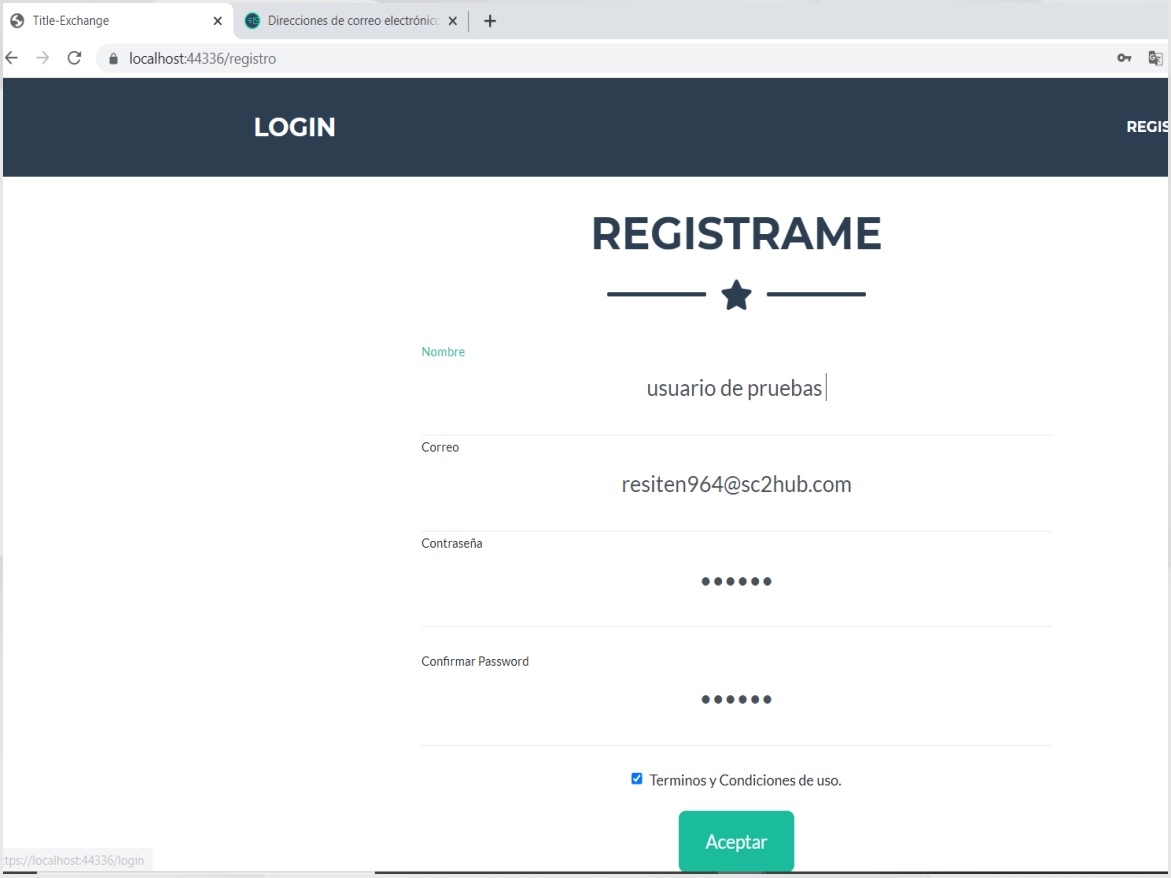


Figura 42 Front, formulario de registro

Fuente: propia

Se requiere que el back cargue todos los tilos guardados, por lo cual se hace la petición al back

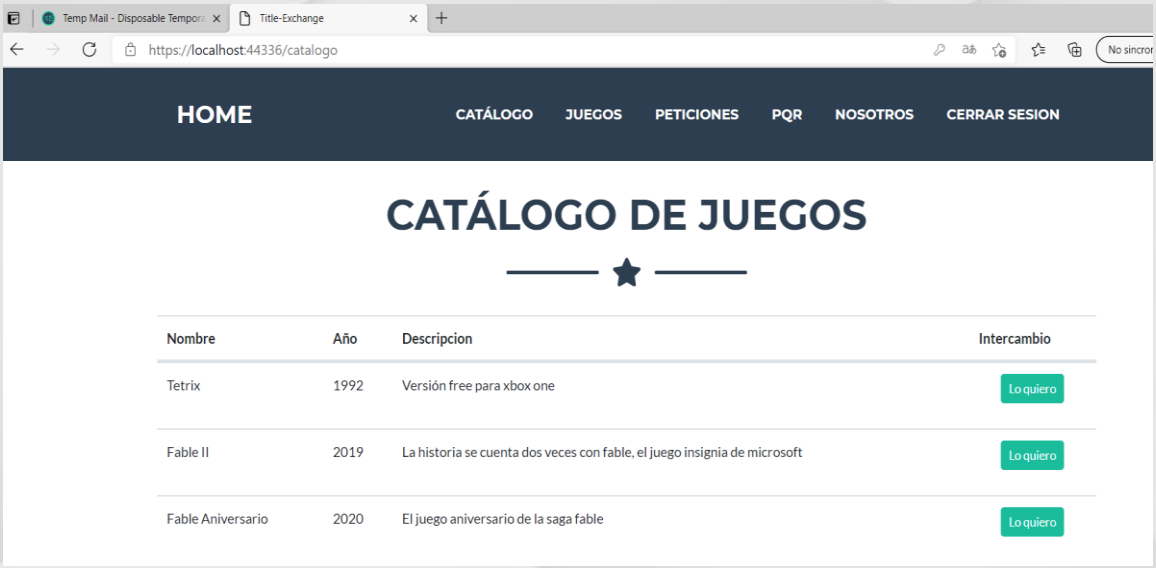


Figura 43 Front, catálogo de juegos registrados en el back

Fuente: propia

Pruebas Casos de uso

Login: para este caso de uso se debe validar que el usuario pueda ingresar a su cuenta con usuario y contraseña que definió al momento de crear la cuenta

Figura 44 Front, formulario del login

Fuente: propia

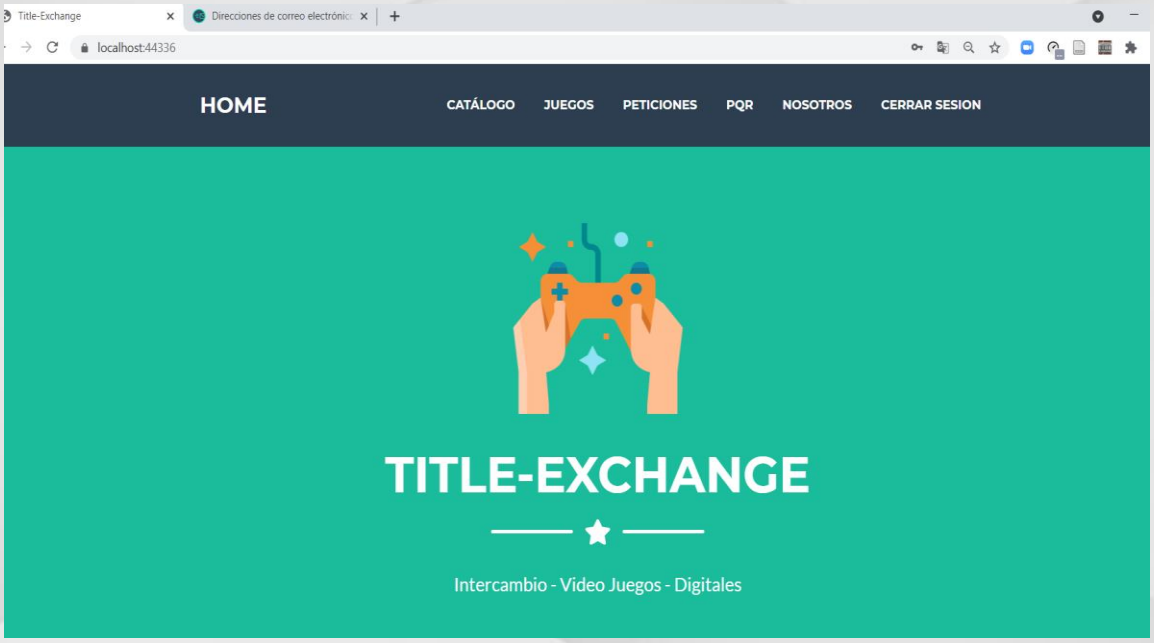


Figura 45 Front, index pagina

Fuente: propia

Publicar título: para publicar el título de un videojuego es necesario que el usuario este logeado en el sistema, ingrese al apartado de publicar título, e ingrese la información del formulario y publique.

Figura 46 Front, pagina para ver y agregar títulos de juegos

Fuente: propia

Figura 47 Front modal para agregar un juego

Fuente: propia

Figura 48 Front, página recargada después de guardar un juego

Fuente: propia

Registrar: En este caso de uso se debe validar que un usuario se pueda registrar en el sistema

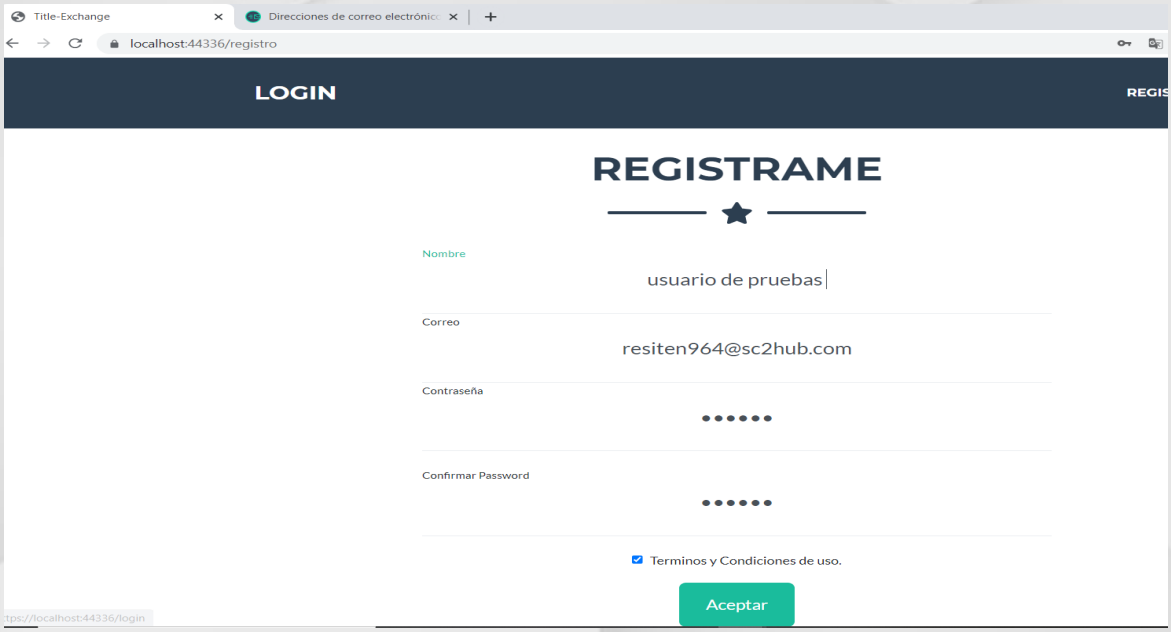


Figura 49 Front, formulario de registro al validar

Fuente: propia

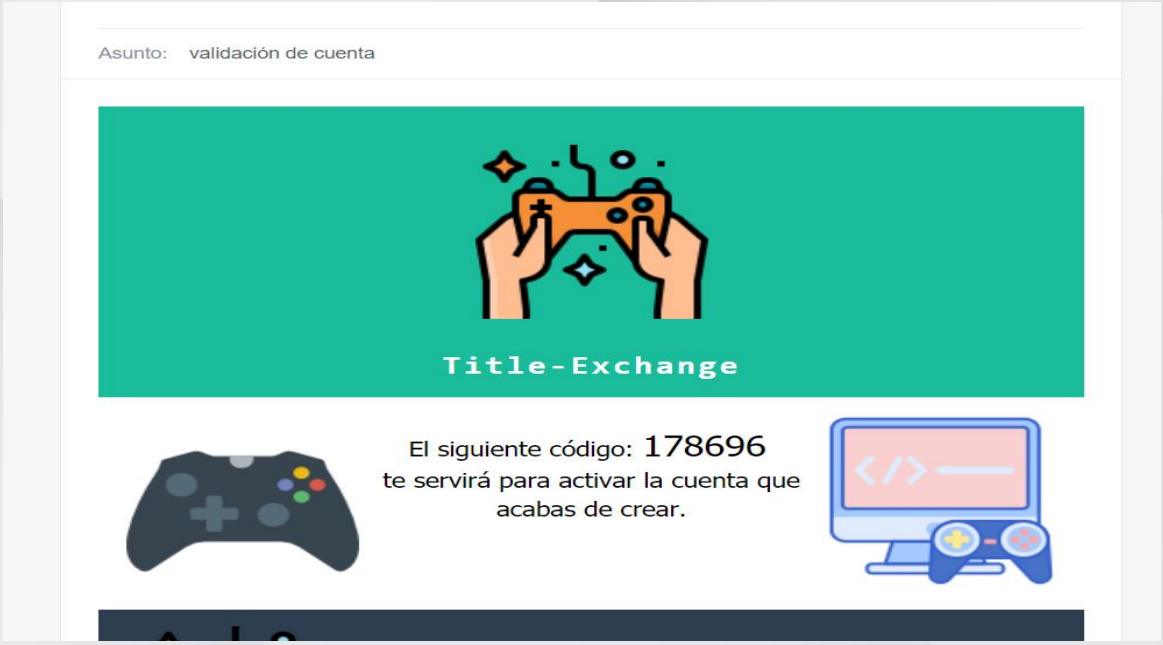


Figura 50 Front, OTP en el correo

Fuente: propia

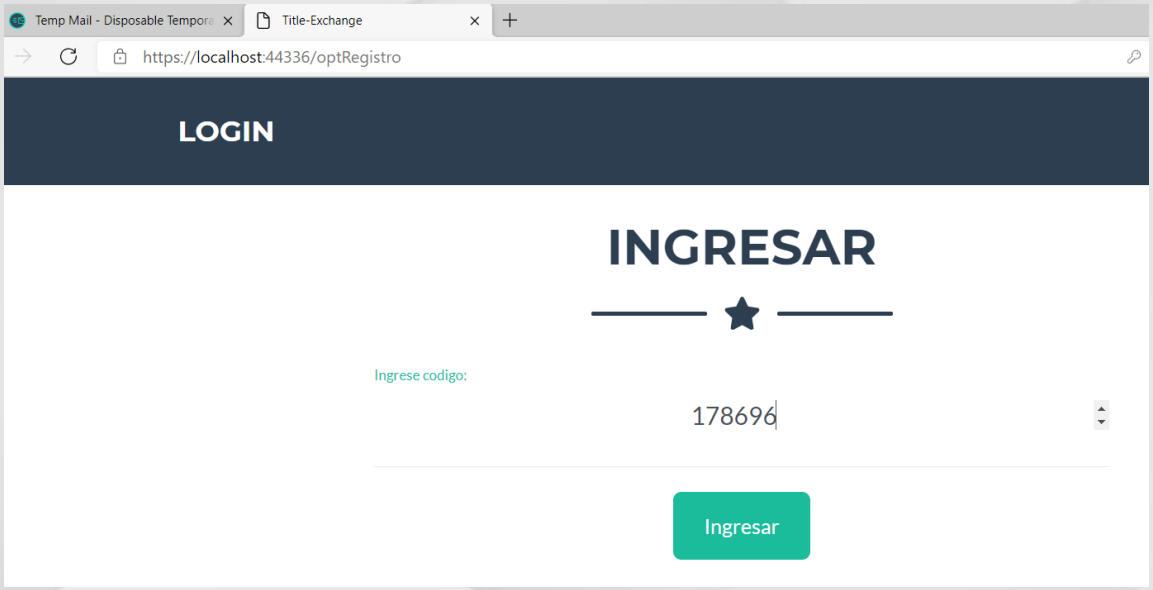


Figura 51 Front, página para ingresar el otp

Fuente: propia

Cambiar título: aquí el usuario debe hacer login, seleccionar el título a cambiar y el titulo que propone para el cambio, se enviara una notificación de cambio a el otro usuario

Figura 52 Front, catálogo de juegos registrados

Fuente: propia

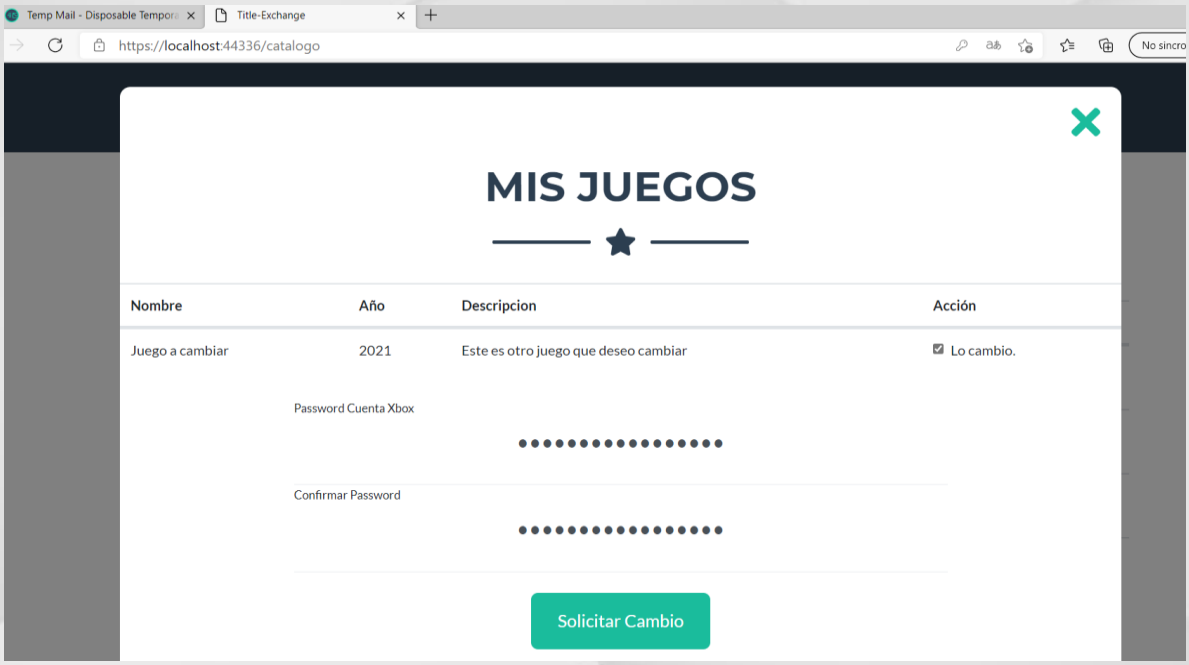


Figura 53 Front, modal cambio de juego

Fuente: propia

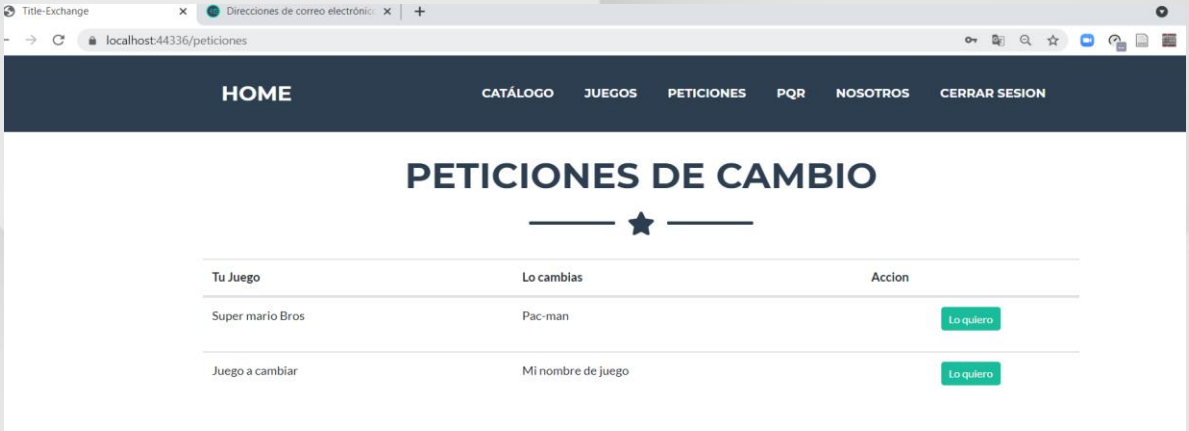


Figura 54 Front, página de peticiones de cambio

Fuente: propia

, si ambos usuarios están de acuerdo con el cambio, se deberá registrar los datos en el formulario de intercambio.

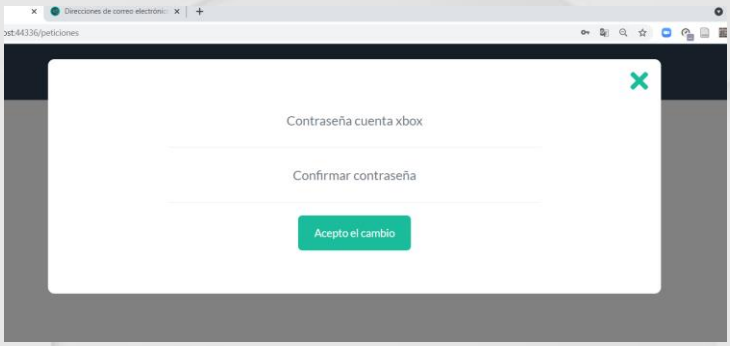


Figura 55 Front, formulario de cambio

Fuente: propia

Verificar título y completar cambio: Un validador externo asíncrono, se encargará de validar que los datos de los títulos de ambos usuarios sean válidos. De ser así se procederá a aprobar el cambio, de caso contrario se rechazará y por último se notificará a los usuarios vía mail.

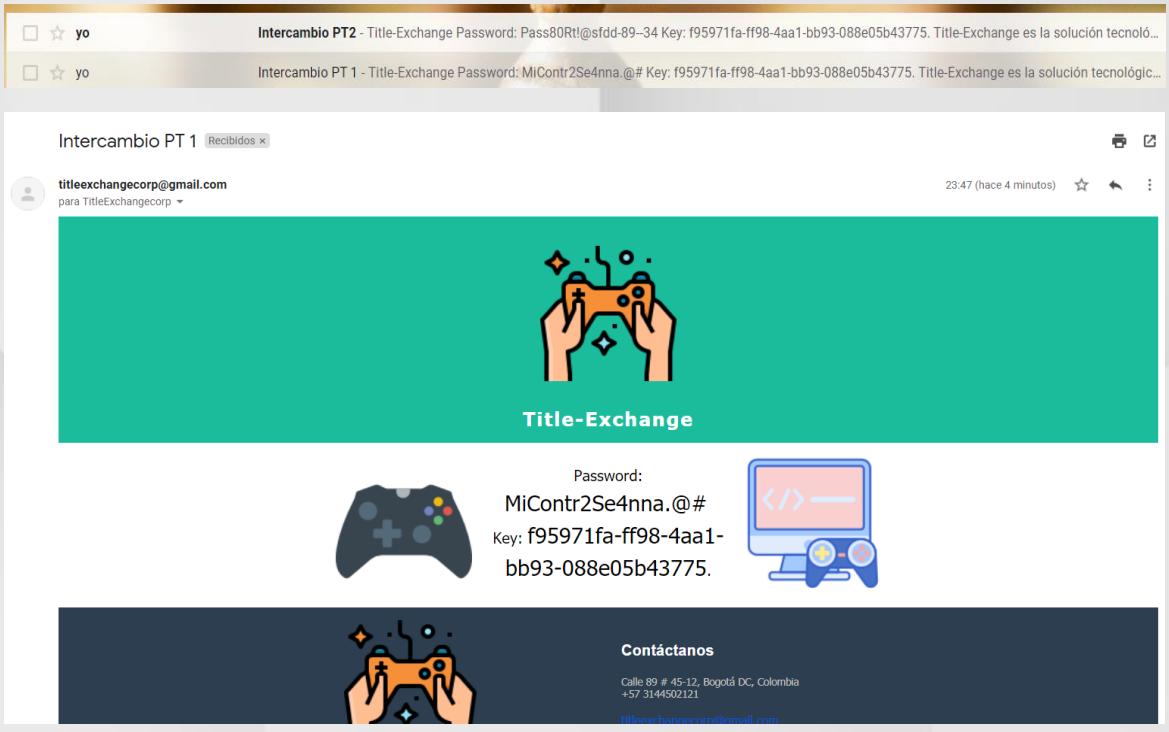


Figura 56 correo de intercambio parte 1

Fuente: propia

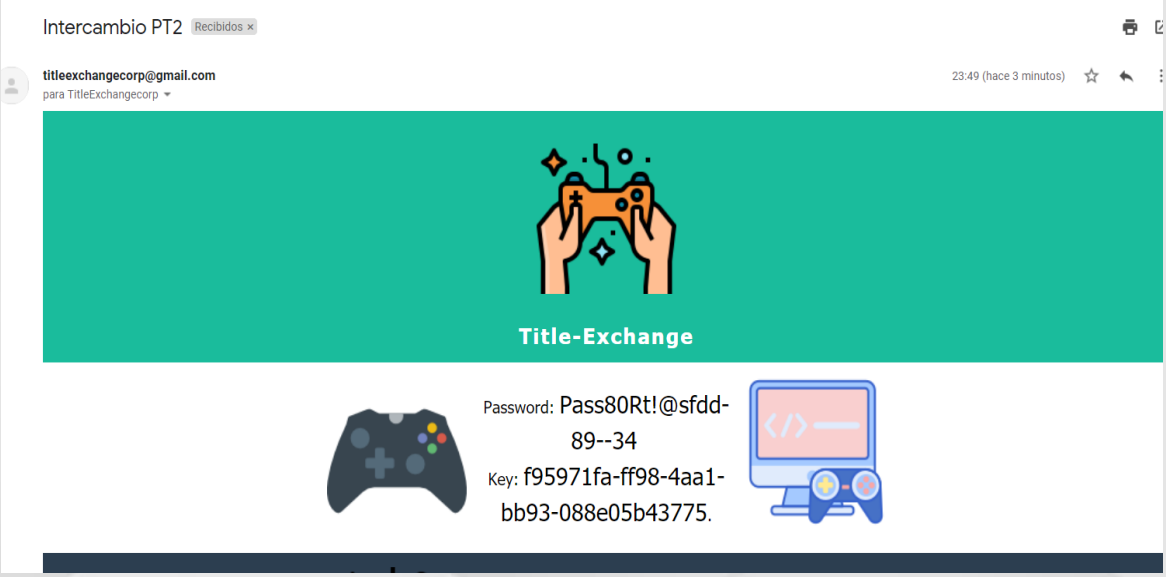


Figura 57correo de intercambio parte 2

Fuente: propia

Después de validar los títulos, se dispondrá a aceptar el intercambio. Para esto se dirigirá a un endpoint y llenar el request con los que llego por correo. Nota: las contraseñas solo las conocera este validador externo.

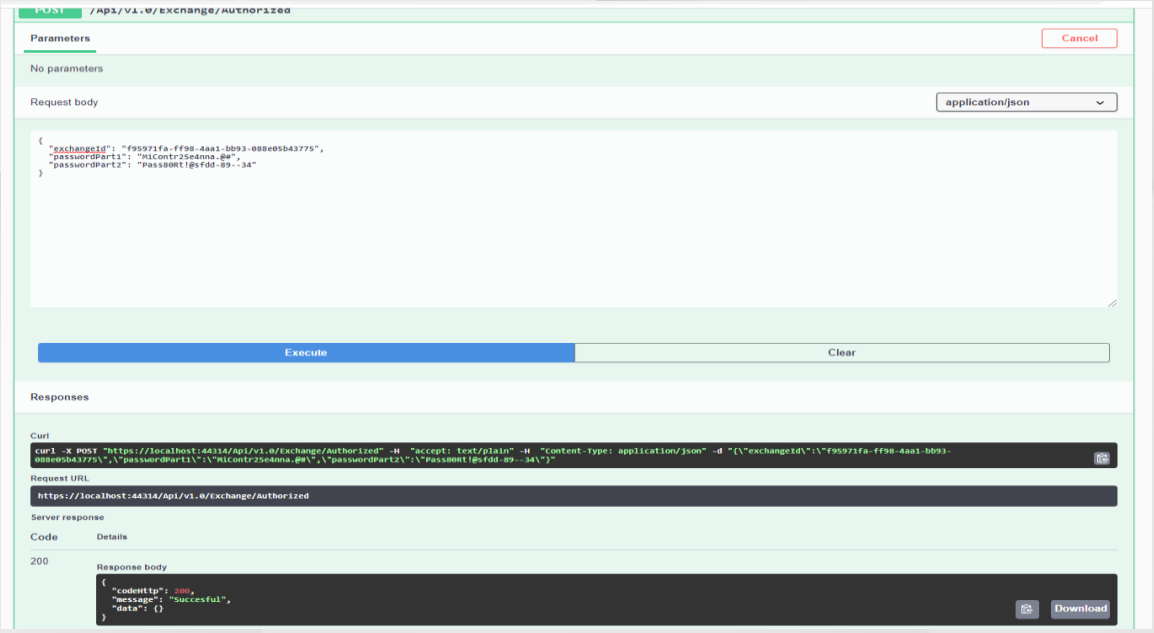


Figura 58 Back, endpoint de autorizar cambios

Fuente: propia

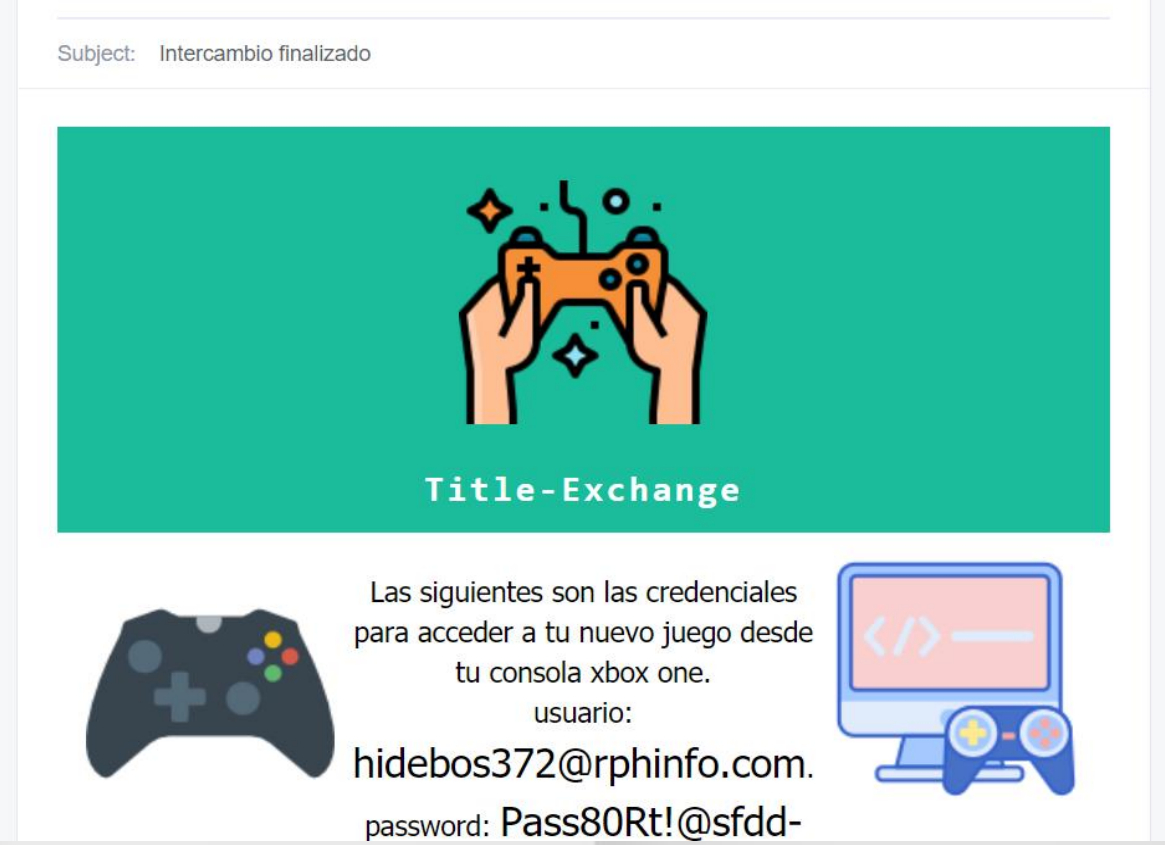


Figura 59 correo de confirmación de cambio

Fuente: propia

Capítulo 9

Instalación y Configuración.

IIS

El Internet information services es un servidor web que se usa para lanzar servicios de Windows y web sites, desde el IIS se puede configurar un sitio para en este caso desplegar el Backend y Frontend, en caso de un ambiente de desarrollo, se debe descargar las características en la maquina abrir el administrador del IIS y crear dos sitios dentro, después se configura el enrutamiento como los puertos de salida que estos deben tener.

SQL Server

Se debe ejecutar el query de inicialización para que cree todas las tablas, stored procedure necesarios para que la persistencia de datos funcione correctamente.

AppSetings

En este archivo se almacena la configuración de enrutamientos hacia el Backend

Publicaciones

Desde visual studio se puede configurar los proyectos para que se puedan publicar en un sitio especifico, en este caso se configura al IIS local, se configura desde el ambiente se acepta la publicación y ya estaría el sistema listo para usarse

Capítulo 10

Conclusiones.

Proyecto de grado

Kanban como metodología para el desarrollo de un proyecto pueden ser muy beneficioso en el sentido que se adapta a un solo usuario y/o muchos, esto es rentabiliza el tiempo invertido para aprender una metodología tan sencilla como esta.

El Domain Driven design es la mejor opción que un usuario pude tener para que su aplicación no sufra si debe escalar en el tiempo o por si en un caso extremo debe cambiar de tecnologías de desarrollo, esto hace que sea muy flexible

Las pruebas de penetración y seguridad son esenciales desde el inicio del proyecto, es indispensable que algo tan cotidiano como una inyección SQL no vulnere nuestro sistema de login. O una inyección de HTML muestre componentes que nos estén permitidos para el rol del usuario

Cumplir con los tiempos del cronograma es esencial, no se deben aplazar o se empezaran a crear atrasos en el proyecto y por ende un esfuerzo doble.

Los objetivos específicos se deben verificar con la medición de un chequeo que demuestre si se cumplió o no dicho objetivo

Referencias

(de la Torre Llorente, Zorrilla Castro, Calvarro Nelson, & Ramos Barroso, 2011) de la Torre Llorente, C., Zorrilla Castro, U., Calvarro Nelson, J., & Ramos Barroso, M. Ángel. (2011). Guía de Arquitectura N-Capas Orientada al Dominio con .NET 4.0 (1.ª ed., pp. 9–20).

adegeo. *What's new in .NET Core 3.1*. https://docs.microsoft.com/en-us/dotnet/core/whats-new/dotnet-core-3-1. Consultado el 23 de septiembre de 2020.

"Blazor | Build client web apps with C# | .NET". *Microsoft*, https://dotnet.microsoft.com/apps/aspnet/web-apps/blazor. Consultado el 23 de septiembre de 2020.

"Comprar Halo: la colección Jefe Maestro: Microsoft Store es-CO". *Microsoft Store*, https://www.microsoft.com/es-co/p/halo-la-coleccion-jefe-maestro/9mt8ptgvhx2p. Consultado el 21 de septiembre de 2020.

Consola XBOX ONE S 1 Tera + 1 Control Inalámbrico+ Juego Digital Halo 5 Guardians + Game Pass 1 Mes+ Xbox Live Gold 14 días. Alkosto Tienda Online. http://www.alkosto.com/consola-xbox-one-s-1-tera-1-control-inalambrico. Consultado el 22 de septiembre de 2020.

Dinero. "Los deportes virtuales: un juego billonario que llega a Colombia". *Cuánto mueve la industria mundial de los videojuegos*, http://www.dinero.com/edicion-impresa/negocios/articulo/cuanto-mueve-la-industria-mundial-de-los-videojuegos/269818. Consultado el 21 de septiembre de 2020.

Disco Duro Externo 2 Teras Toshiba Original Usb 3.0 Estuche - $ 324.900. https://articulo.mercadolibre.com.co/MCO-470845732-disco-duro-externo-2-teras-toshiba-original-usb-30-estuche-_JM. Consultado el 21 de septiembre de 2020.

Fontes, Consultoria Web-Francisco de Brito. "Sobre el portal Intercambiojuegos - Portal Gamer Chile". *Intercambio Juegos*, https://www.intercambiojuegos.cl. Consultado el 22 de septiembre de 2020.

Gamers, Colombia. *COLOMBIA GAMERS, un estilo de vida*. https://www.colombiaGamer.com.co/quienes-somos. Consultado el 22 de septiembre de 2020.

Historia de los videojuegos. https://www.fib.upc.edu/retro-informatica/historia/videojocs.html. Consultado el 22 de septiembre de 2020.

Juego XBOX ONE Mortal Kombat Aftermath - LATAM Alkosto Tienda Online. http://www.alkosto.com/juego-xbox-one-mortal-kombat-aftermath-latam. Consultado el 21 de septiembre de 2020.

"Jugador de videojuegos". *Wikipedia, la enciclopedia libre*, el 31 de agosto de 2020, https://es.wikipedia.org/w/index.php?title=Jugador_de_videojuegos&oldid=128905242.

"La guía definitiva para entender las generaciones de consolas". *VIX*, https://www.vix.com/es/videojuegos/170697/la-guia-definitiva-para-entender-las-generaciones-de-consolas. Consultado el 23 de septiembre de 2020.

"Obtener Call of Duty®: Warzone: Microsoft Store es-CO". *Microsoft Store*, https://www.microsoft.com/es-co/p/call-of-duty-warzone/9mwwnmh6z0jh. Consultado el 21 de septiembre de 2020.

¿Qué es el impacto social y por qué es importante? https://wealthmanagement.bnpparibas/es/es/expert-voices/social-impact.html. Consultado el 22 de septiembre de 2020.

"Todo sobre .NET Core, ¿Qué es, ventajas, novedades? y mucho más". *Anexsoft*, https://anexsoft.com/todo-sobre-net-core-que-es-ventajas-novedades-y-mucho-mas. Consultado el 23 de septiembre de 2020.

"Videoconsola". *Wikipedia, la enciclopedia libre*, el 19 de septiembre de 2020, https://es.wikipedia.org/w/index.php?title=Videoconsola&oldid=129385276.

"Videojuego". *Wikipedia, la enciclopedia libre*, el 8 de mayo de 2020, https://es.wikipedia.org/w/index.php?title=Videojuego&oldid=125869024.

Anexos.

Lista de documentos

Los documentos listados a continuación ejemplifican y mejora la especificación técnica y funcional del software en cuestión, estos documentos se adjuntan en la entrega final del proyecto.

Anexos A. SAD (Software Architecture Document)

Anexos B. DCOE (Documento de chequeo para objetivos específicos)

VERSIÓN 1.0

25/03/2021

SOFTWARE ARQUITECTURE DOCUMENT(SAD)

WWW.CARATULASPARATRABAJOS.ORG

1. DESCRIPCIÓN DEL DOCUMENTO

1.1. Propósito y audiencia

En la industria de los videojuegos hay mucho por ganar para las compañías y mucho por gastar para los usuarios, hablando de temas monetarios es una industria que económicamente percibe más ingresos que las industrias de la música y cine juntas, con lo cual se puede deducir que hay un gran grupo de Gamers dispuestos a usar su dinero para adquirir el último título estrenado en el año, pero hay un grupo aún más grande que tiene los siguientes problemas:

1. Capacidad de almacenamiento en la consola
2. Desinterés que se adquiere por un título después de haberlo jugado mucho
3. Mas importante aún es el costo que estos llegan a tener.

El emprendimiento se centra en crear un sitio web para el intercambio de juegos virtuales para las consolas de XBOX ONE,

Se espera obtener, satisfacción, al saber que el poco o mucho conocimiento que tenga un ingeniero de sistemas(Gamer) puede ser puesto a disposición y ayuda de la comunidad Gamer a la cual va dirigida esta.

1.2. Organización del documento

Este documento se encuentra organizado en tres secciones principales, la primera incluye una descripción breve del problema a tratar y del contexto general de la organización. Más específicamente se definen los intereses particulares de los diferentes stakeholders y se realiza una introducción a las funcionalidades requeridas en el sistema.

En la segunda parte se analizan los motivadores de negocio que hacen parte fundamental del proyecto. Además, se extraen los atributos de calidad definidos por el cliente y estos se priorizan y definen en diferentes escenarios.

En la tercera parte se propone la arquitectura del sistema y se especifica en los diferentes puntos de vista a través de modelos gráficos.

1.3. Convenciones

En general en los modelos especificados en el documento se utiliza simbología que hace parte del estándar UML 2.0 y/o del uso de tecnologías como página web

1.4. Terminología y definiciones

Software: Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

Stakeholder: Se define como cualquier persona, entidad u organización que de una u otra forma posee interés en el sistema o negocio.

XBOX: Xbox es una marca de videojuegos creada por Microsoft que incluye una serie de videoconsolas desarrolladas por la misma compañía

VIDEOJUEGOS: Un videojuego es un software de interacción continua entre maquina (imágenes) y jugador, no llega a ser más complejo que definirlo como juego electrónico

UML: Lenguaje unificado de diagramación. Esencial para diagramar los casos de uso, de clases

1.5. Documentos relevantes

N/A

2. GENERALIDADES DEL PROYECTO

2.1. Problema a resolver

Generalmente una consola sale de fabrica con capacidad de almacenamiento entre 500GB y 1000GB esto suele ser engañoso ya que hay títulos como “Call of Duty: Warzone” O “Hallo Jefe Maestro” que llegan a pesar más de 100GB, Problema a la vista, quizás la solución sea adquirir un disco duro externo “Disco Duro Externo 2 Teras Toshiba Original USB 3.0, con estuche, precio: $324.900”. La experiencia de un Gamer se limita a cuánto tiempo se ha jugado a un título y sobre todo a que tan bueno se es en el mismo, la una va ligada a la otra,

pero en el momento en el que aparece el desinterés, el juego sale a un segundo plano y se pierde en el olvido al punto de ser desinstalado de la consola ¿y el dinero invertido?, expone el principal problema al que se exponen los Gamers, al alto costo que los títulos tienen el comprar uno tras otro se convierte en el talón de Aquiles de aquellos Gamers sin tanto presupuesto.

¿Si existen tiendas físicas para el intercambio de videojuegos físicos, porque no puede existir un WebSite para el intercambio de videojuegos digitales que solucione el talón de Aquiles de los Gamers?

2.2. Descripción general del sistema a desarrollar

Implementar un website para el intercambio de videojuegos digitales, que sea gratuito y genere confianza

Garantizar la veracidad de la información.

2.3. Objetivos

- Crear una estructura de aplicación de alto nivel empleando el paradigma de programación orientado a objetos que implemente en su diseño los principios para la alta cohesión, bajo acoplamiento, encapsulamiento, polimorfismo y herencia.
- Documentar, afianzar el concepto de las áreas del conocimiento y el correcto uso de las habilidades blandas.
- Sistematizar el ingreso de la información y almacenamiento de la misma en la base de dados.
- Digitalizar el intercambio de videojuegos de manera segura teniendo un validador que se encargue de saber si el título es válido o no.
- Generar el interés de los Gamers por títulos nuevos.

2.4. Stakeholders

En este caso para el primer nivel, como lo son los clientes; los Gamers son quien final del día van a aprobar, o dar el éxito del proyecto, el proveedor en nuestro caso de (Microsoft), de la razón de ser de este proyecto.

3. MOTIVADORES Y FUERZAS EXTERNAS

3.1. Motivadores de negocio

Aprovechar el gran número de juegos disponibles que hay al día de hoy para las consolas Xbox One, permitiendo a muchos Gamers poder jugar con títulos que no conocían hasta la fecha, al menos el 5% de intercambios efectivos, durante el primer año

Aumentar los intercambios de títulos en Colombia, expandiéndonos a una ciudad cada 15 meses, para lograr llegar a la mayoría de Gamers de Xbox one

3.2. Restricciones

El proyecto debe ser completado en un tiempo de 5 meses, por un solo ingeniero

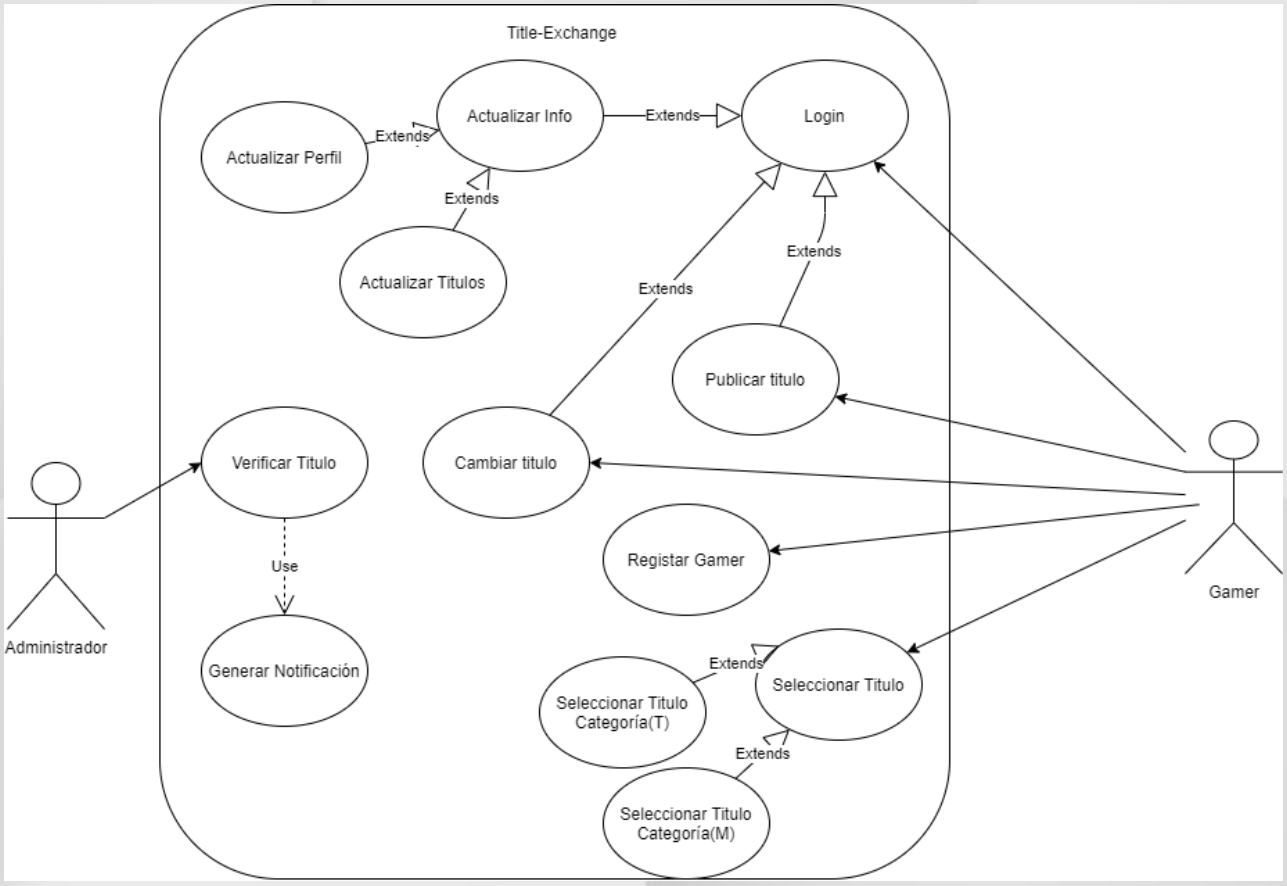
El website debe estar construido en net core

4. PUNTOS DE VISTA

A continuación, se presentarán los diagramas con la arquitectura propuesta enmarcada en los diferentes puntos de vista. Lo principal al momento de definir el modelo arquitectural fueron los escenarios de calidad cuya prioridad fuera alta. Específicamente se favoreció el desempeño visto desde sus dos grandes componentes, la latencia y la escalabilidad. Se emplearon tácticas tanto a nivel de software como en hardware en busca del favorecimiento del desempeño de la aplicación. Se omiten temporalmente los modelos que relacionan el flujo de la información dentro de la arquitectura propuesta.

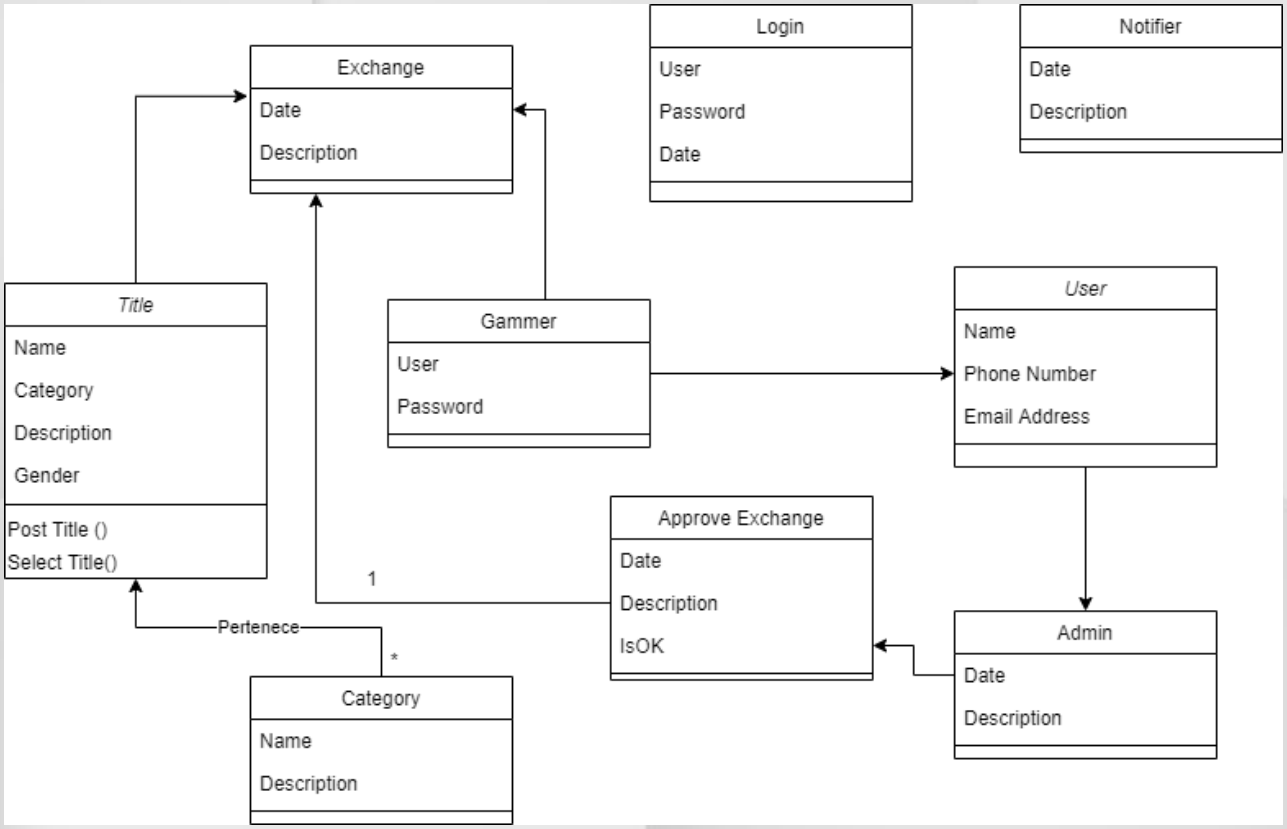
4.1. punto de vista Escenarios

4.1.1. Modelo de casos de uso



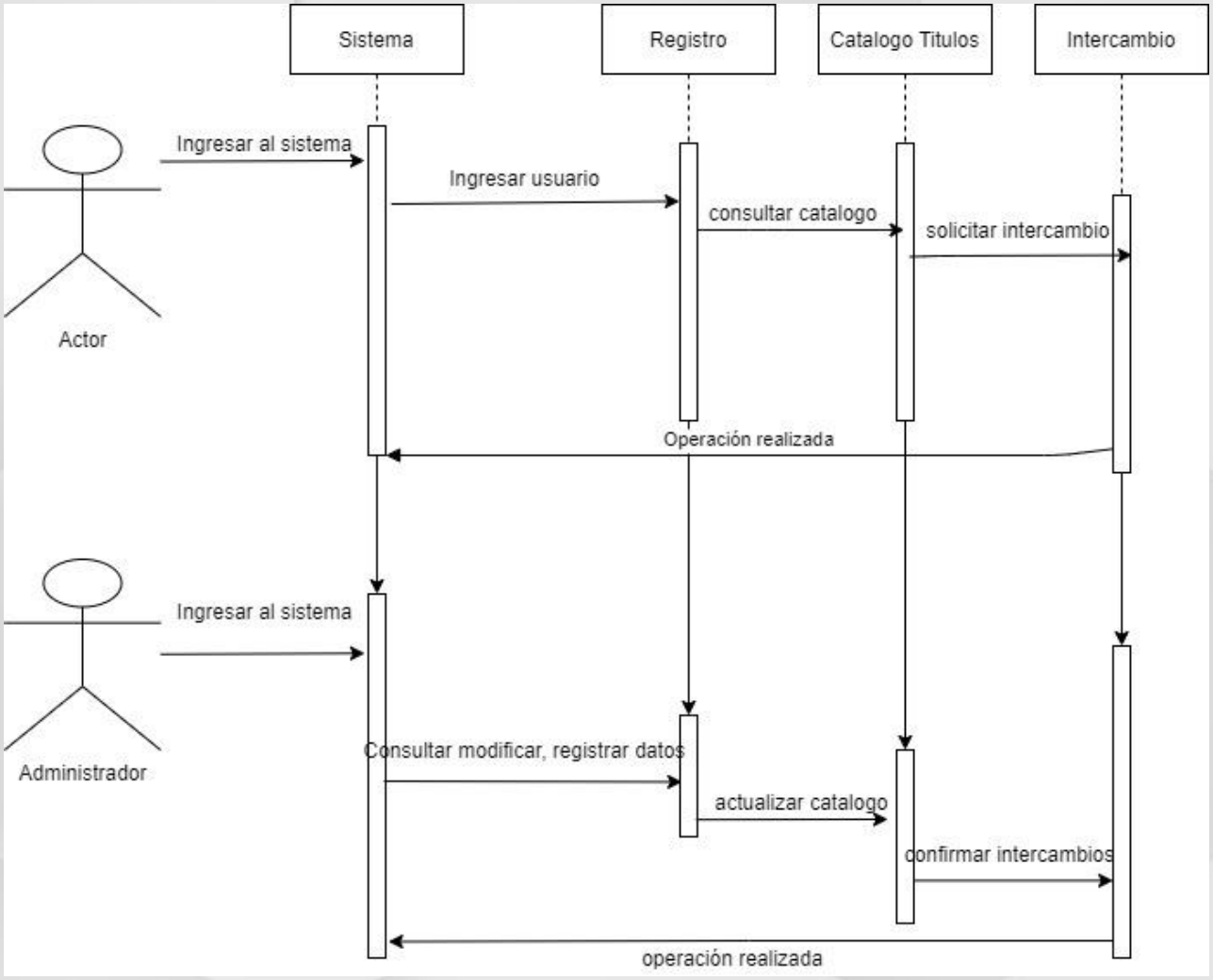
4.2. punto de vista Logíca

4.2.1. Modelo de clases



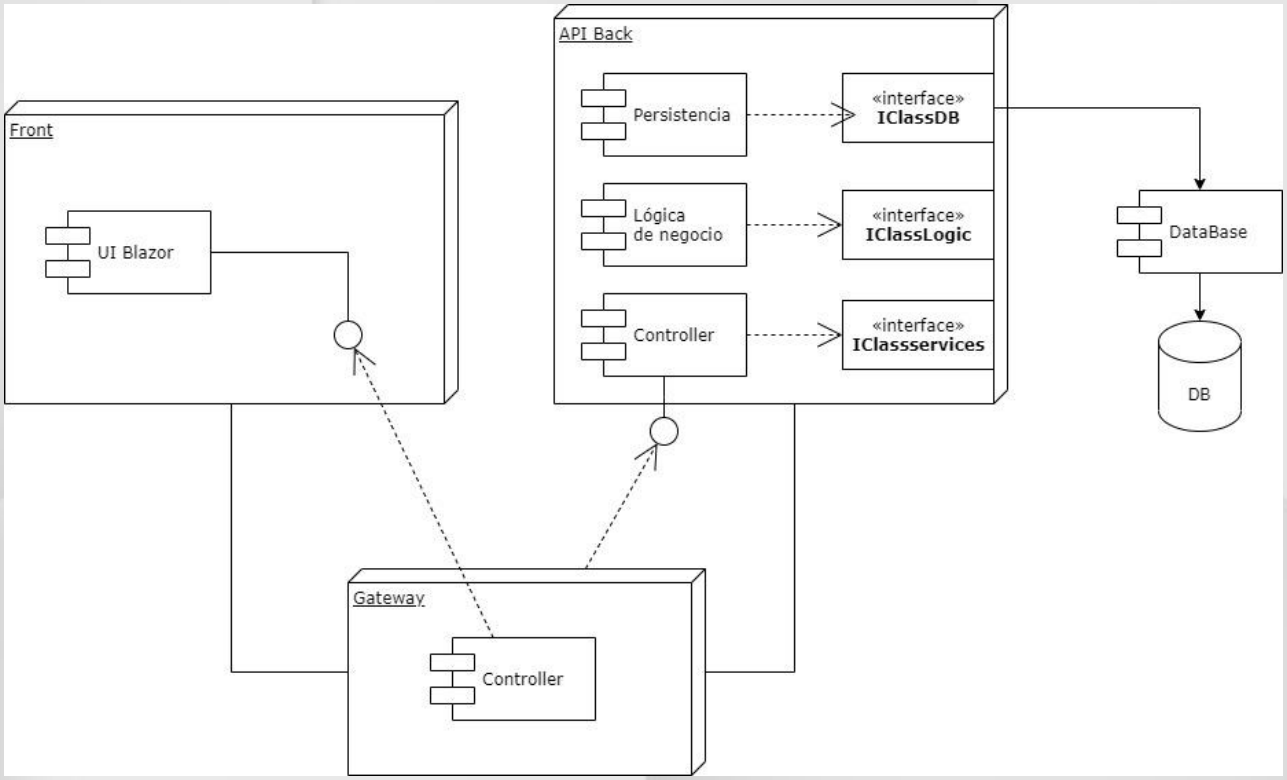
4.3. punto de vista de Proceso

4.3.1. Modelo de secuencia



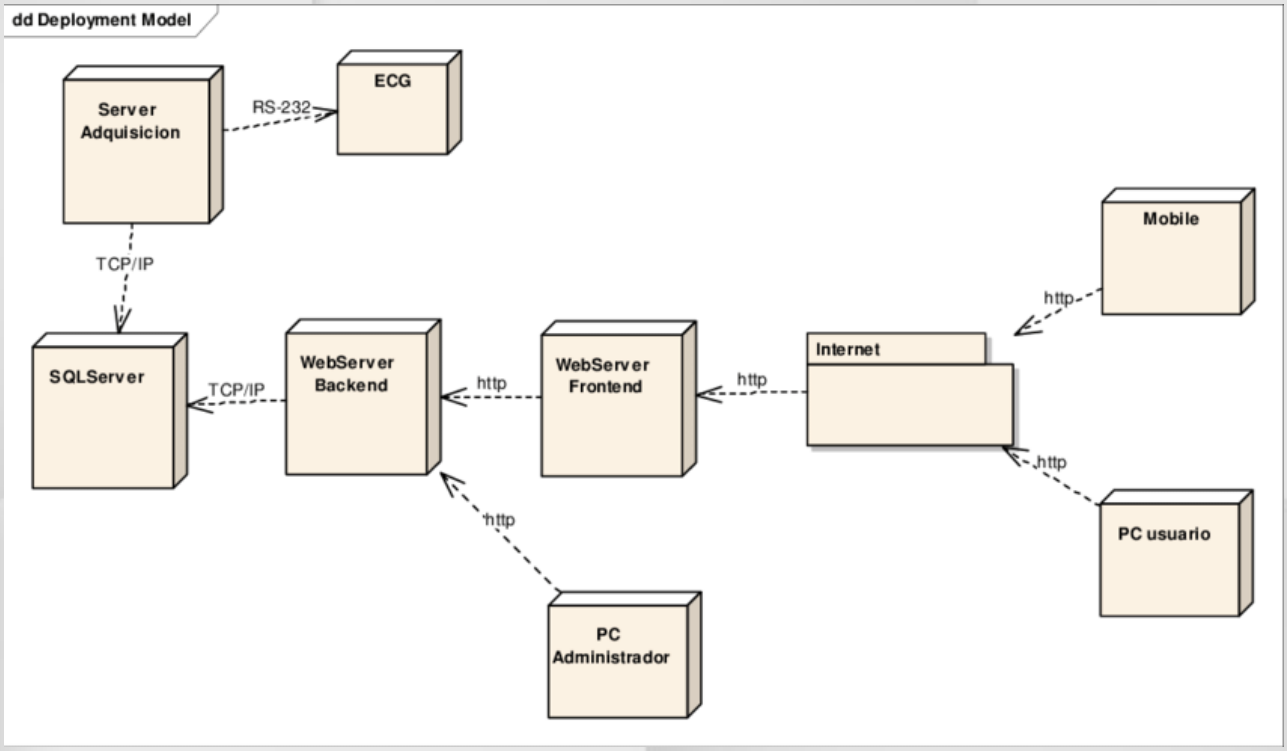
4.4. punto de vista desarrollo

4.4.1. Modelo de componentes



4.5. Punto de vista físico

4.5.1. Modelo de despliegue



VERSIÓN 1.0

15/05/2021

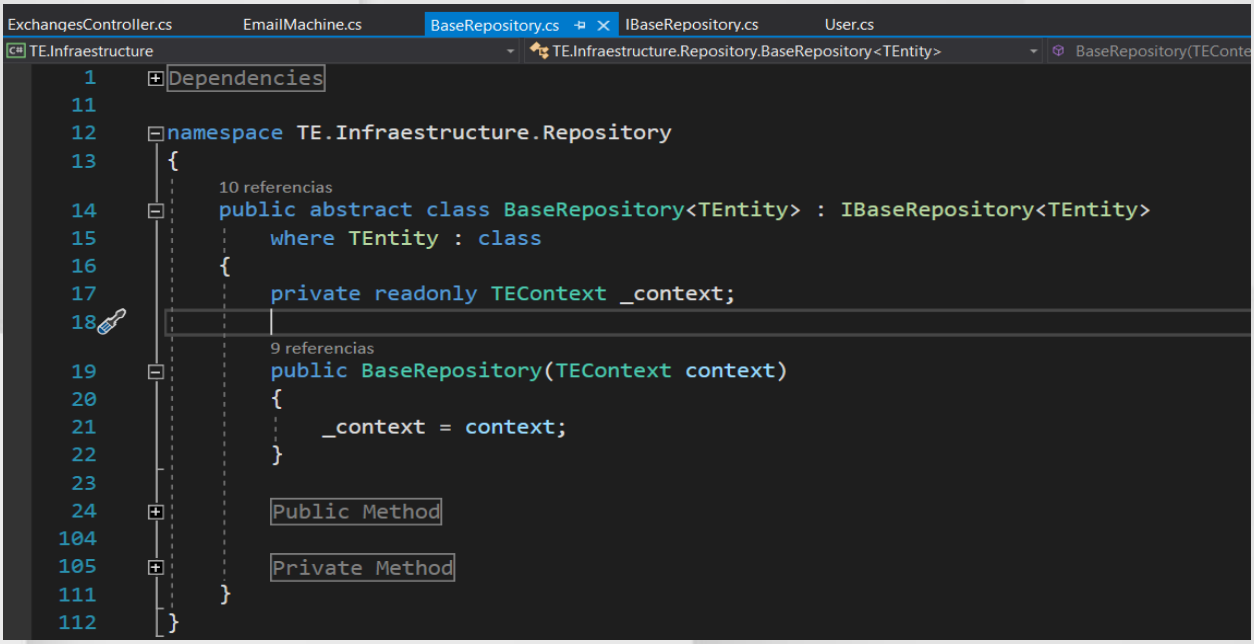
DOCUMENTO DE CHEQUEO PARA OBJETIVOS ESPECÍFICOS

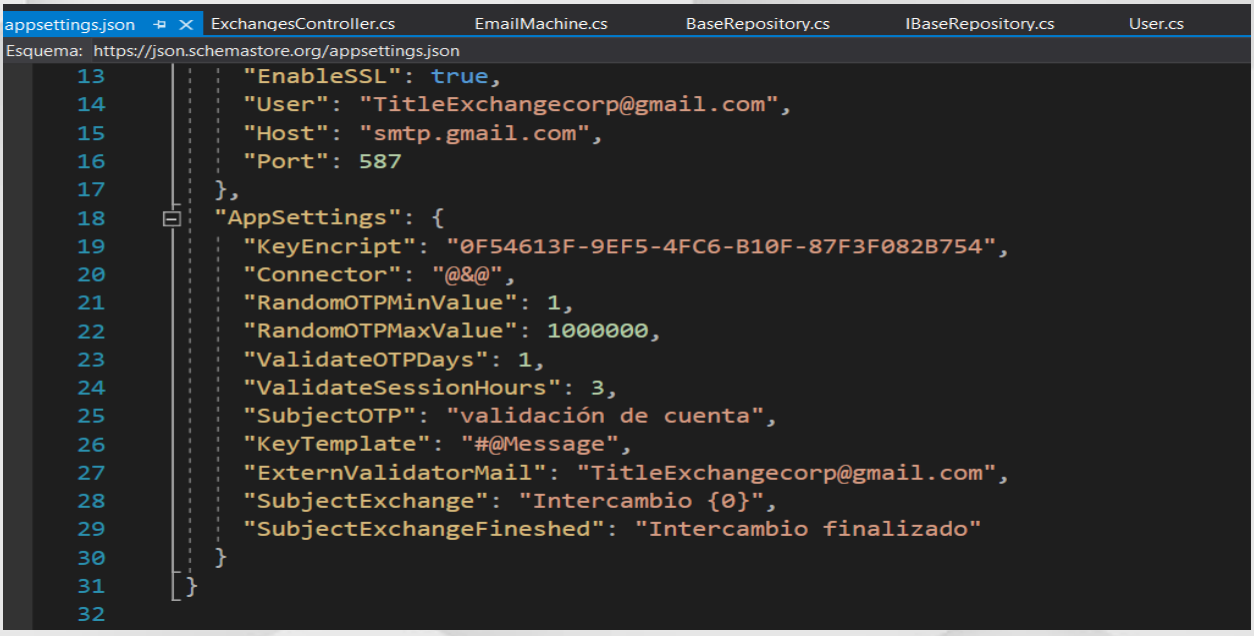
WWW.CARATULASPARATRABAJOS.ORG

Documento con las evidencias del cumplimiento de los objetivos específicos según corresponde su medición.

- Crear un software de calidad, lo que implica que se apliquen los siguientes ítems sobre el software; buenas prácticas de desarrollo, refactorización y escalabilidad.

Buenas prácticas aplicadas, en el código no hay variables quemadas, se usó del archivo AppSettings para guardar las variables de conexión necesaria, no hay código espagueti, y todo esta ordenado en regiones, para que la lectura y mantenibilidad sea sencilla.

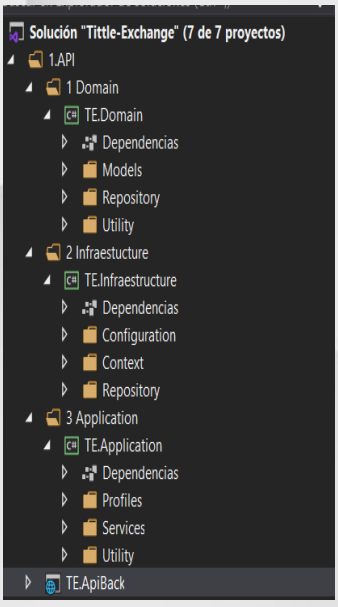


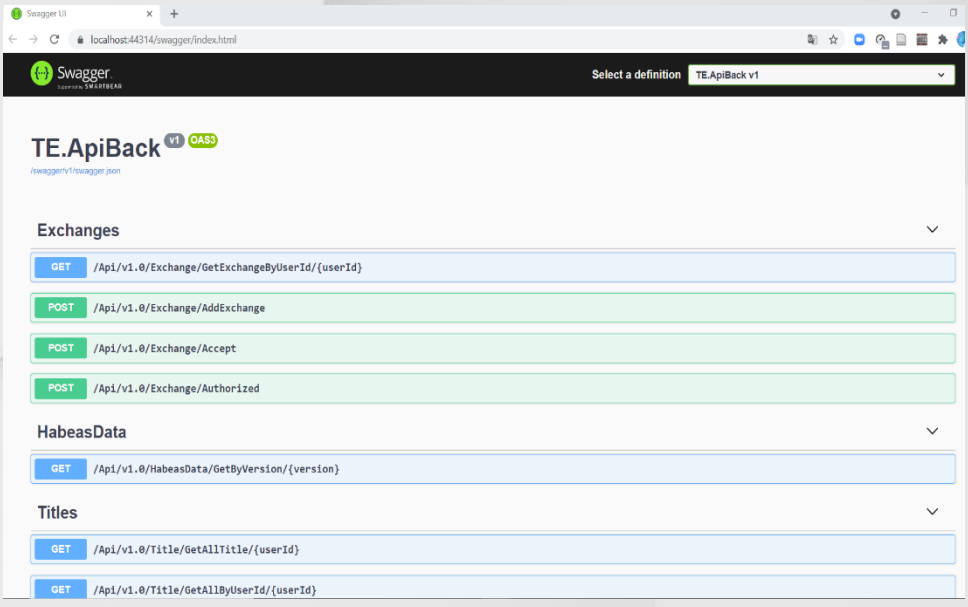


Refactorización

```
namespace TE.ApiBack.Controllers
{
    [ApiController]
    [Route(Routing.Users.Base)]
    1 referencia
    public class UsersController : Controller
    {
        private readonly IUserService _userService;

        0 referencias
        public UsersController(IUserService userService)
        {
            _userService = userService;
        }

        [HttpGet]
        [Route(Routing.Users.GetByUserId)]
        0 referencias
        public async Task<BaseResponse> GetByUserId(Guid userId) => await _userService.GetByUserIdAsync(userId);

        [HttpGet]
        [Route(Routing.Users.Authenticate)]
        0 referencias
        public async Task<BaseResponse> AuthenticateUser(string user, string password) => await _userService.Authe
```

Escalabilidad al ser un aplicativo que se define en comunicación REST, hace que sea adaptable y escalable por su diseño orientado a dominio, es mismo diseño de arquitectura permite esa flexibilidad que se necesita en una APP escalable, también se versionaron los endpoint con el fin de tener trazabilidad con la funcionabilidad nueva y la principal





- Implementar en el diseño los principios para la alta cohesión, bajo acoplamiento, encapsulamiento, polimorfismo y herencia.

Aquí está el claro ejemplo de la herencia, encapsulamiento, polimorfismo.

```
15    namespace TE.Application.Services.Implementations
16    {
          2 referencias
17        public class UserService : IUserService
18        {
19            private readonly ParametersMailRepository _parametersMailRepository;
20            private readonly TemplateMailRepository _templateMailRepository;
21            private readonly ActivateAccountRepository _accountRepository;
22            private readonly UserSessionRepository _userSessionRepository;
23            private readonly UserRepository _userRepository;
24            private readonly AppSettings _appSettings;
25
26            private readonly IEmailMachine _emailMachine;
27            private readonly IMapper _mapper;
28
              0 referencias
29            public UserService(IEmailMachine emailMachine
30                , IMapper mapper
31                , IOptions<AppSettings> appSettings
32                , UserRepository userRepository
33                , ActivateAccountRepository accountRepository
34                , UserSessionRepository userSessionRepository
35                , TemplateMailRepository templateMailRepository
36                , ParametersMailRepository parametersMailRepository)...
47
```

```
2 referencias
public class ExchangeService : BaseService, IExchangeService
{
    0 referencias
    public ExchangeService(IOptions<Urls> urls) : base(urls)
    {
    }

    2 referencias
    public async Task<BaseResponse> AcceptAsync(AcceptExchangeDTO model)
    {
        BaseResponse response = await _client.AcceptAsync(model);
        GetDataBaseReponse<AcceptExchangeDTO>(response);
        return response;
    }

    2 referencias
    public async Task<BaseResponse> AddExchangeAsync(AddExchangeDTO model)
    {
        BaseResponse response = await _client.AddExchangeAsync(model);
        GetDataBaseReponse<Transverse.ModelsDTO.ExchangeDTO>(response);
        return response;
```

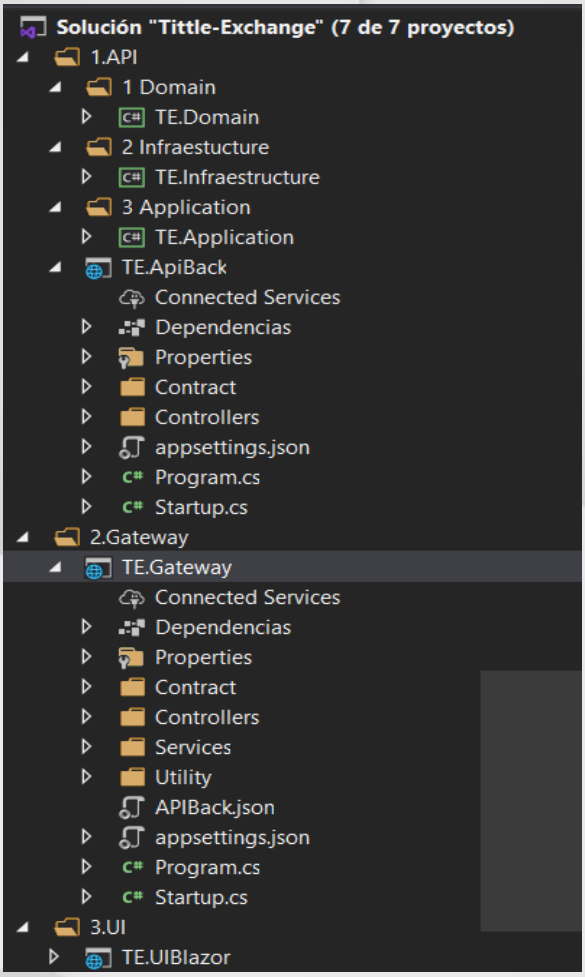
```
namespace TE.Gateway.Services.Implementations
{
    5 referencias
    public class BaseService
    {
        public readonly APIBackClient _client;
        private readonly Urls _urls;

        4 referencias
        public BaseService(IOptions<Urls> urls)
        {
            _urls = urls.Value;
            _client = new APIBackClient(_urls.BaseUrlBack, new HttpClient());
        }

        16 referencias
        public static void GetDataBaseReponse<T>(BaseResponse response)
        {
            if (response.Data != null)
                response.Data = JsonConvert.DeserializeObject<T>(response.Data.ToString());
        }
    }
}
```

TitleService.cs | ExchangeService.cs | UsersController.cs | appsettings.json | ExchangesController.cs | EmailMachine.cs

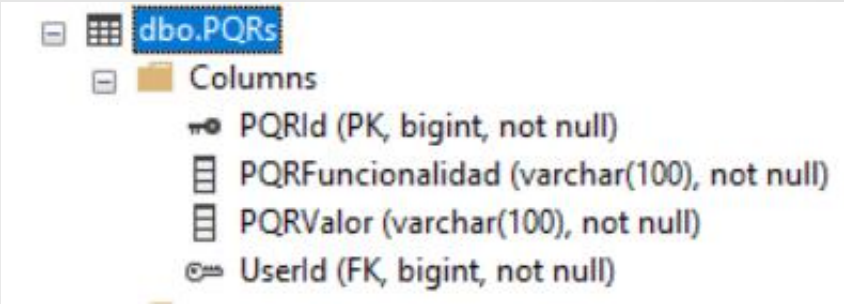
TE.Application.Services.Implementations.EmailMachine | _sMTPSettings

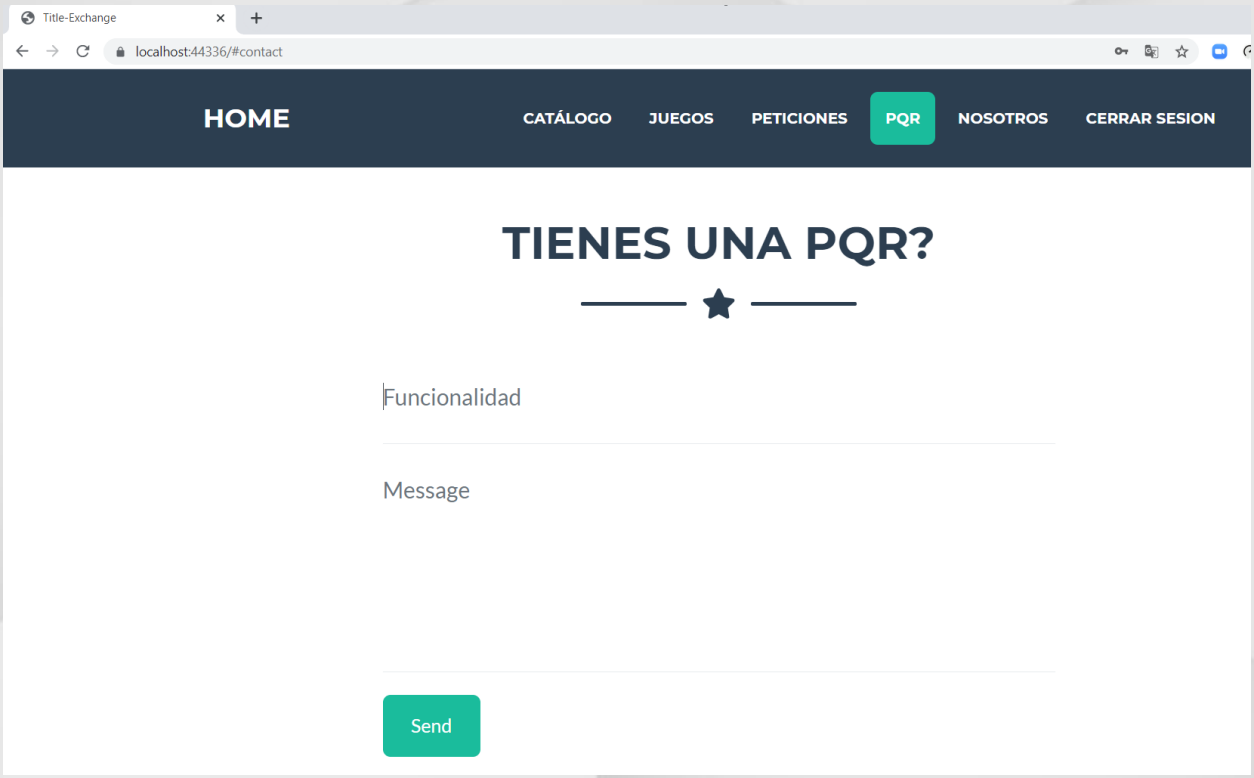
```
        {
            _sMTPSettings = sMTPSettings.Value;
            _cliente = new SmtpClient(_sMTPSettings.Host, _sMTPSettings.Port)
            {
                EnableSsl = _sMTPSettings.EnableSSL,
                DeliveryMethod = SmtpDeliveryMethod.Network,
                UseDefaultCredentials = false,
                Credentials = new NetworkCredential(_sMTPSettings.User, _sMTPSettings.Password)
            };
        }

        4 referencias
        public async Task EnviarCorreo(string destinatario, string asunto, string mensaje, bool esHtlm = false)
        {
            MailMessage message = new MailMessage(_sMTPSettings.User, destinatario, asunto, mensaje);
            message.IsBodyHtml = esHtlm;
            await _cliente.SendMailAsync(message);
        }

        0 referencias
        private async Task EnviarCorreoAsync(MailMessage message)
        {
            await _cliente.SendMailAsync(message);
        }
```

Cada componente se encarga una tarea en específico y sus miembros datos están encapsulados



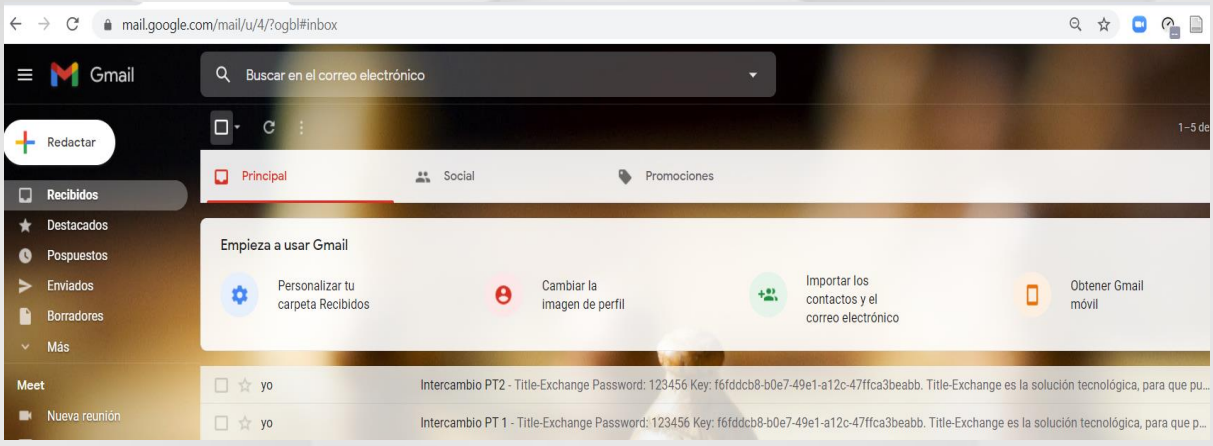
- Implementar módulo de PQR para que los usuarios puedan poner sus peticiones y/o sugerencias sobre el sitio web.

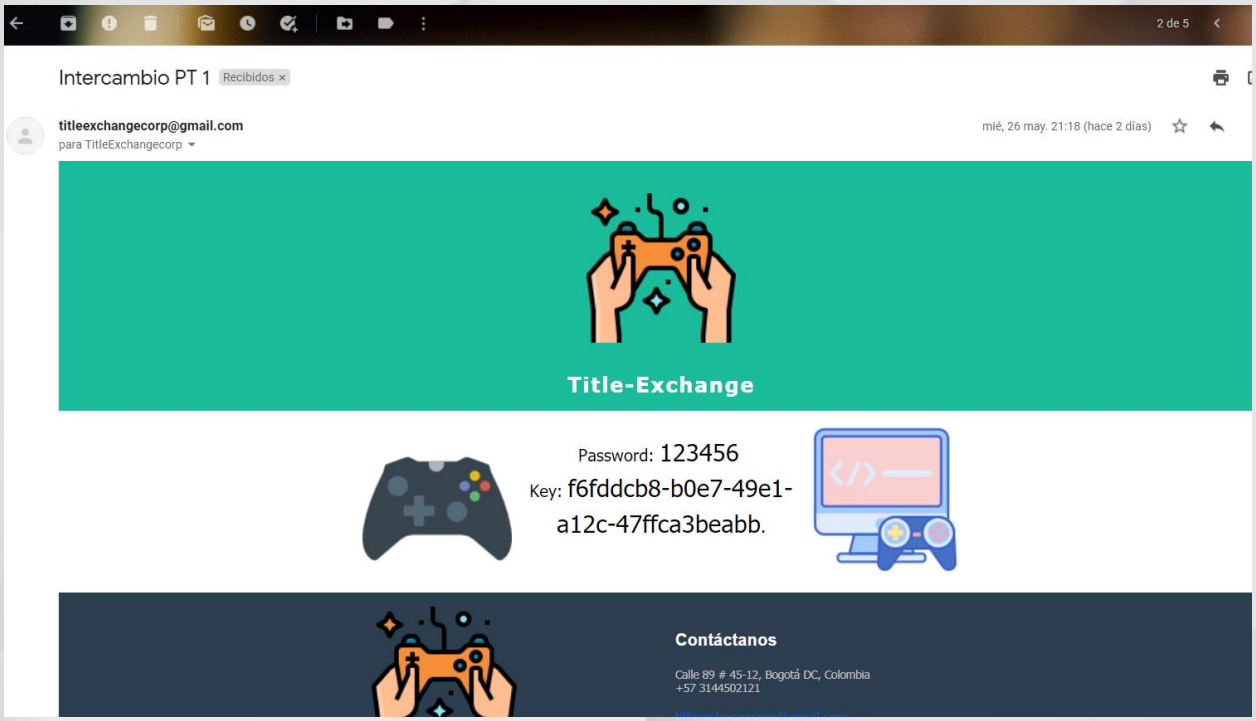


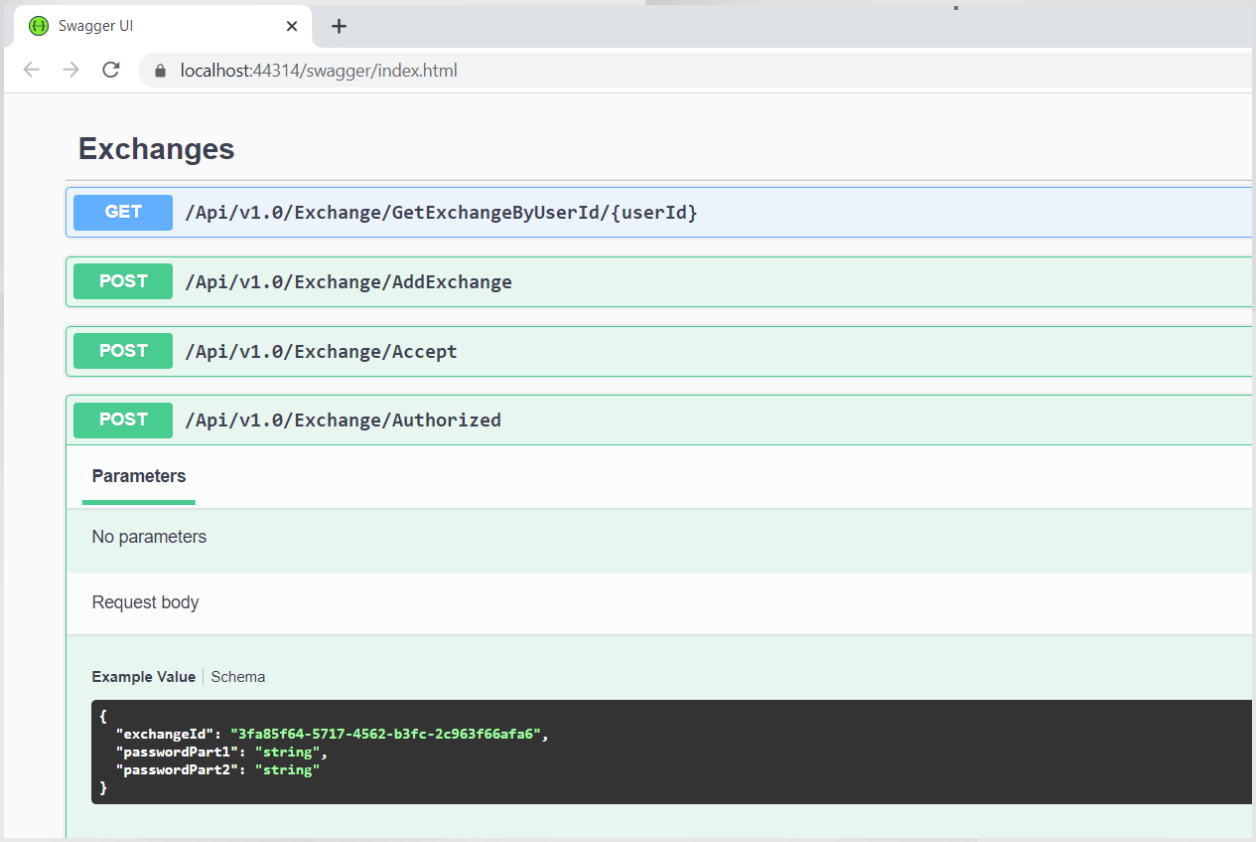


- Digitalizar el intercambio de videojuegos de manera segura teniendo un validador (Externo async) que se encargue de la validez del título.

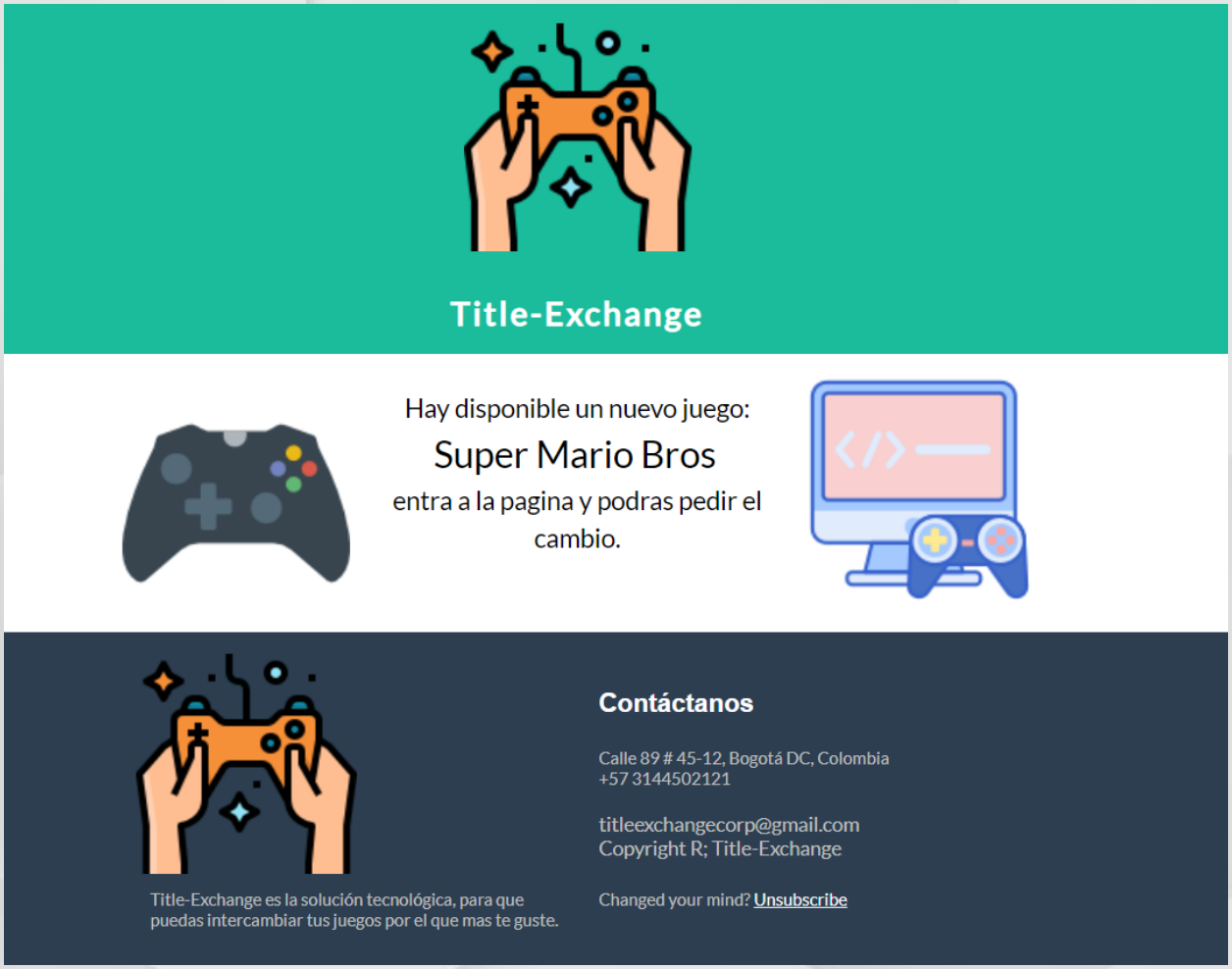
Al validador se le notifica mediante correo electrónico las peticiones de intercambio de un video juego y este realiza la aprobación desde el back




2 de 5
Intercambio PT 1
Recibidos
titleexchangecorp@gmail.com
para TitleExchangecorp
mié, 26 may. 21:18 (hace 2 días)
Title-Exchange
Password: 123456
Key: f6fddcb8-b0e7-49e1-a12c-47ffca3beabb.
Contáctanos
Calle 89 # 45-12, Bogotá DC, Colombia
+57 3144502121


Swagger UI
localhost:44314/swagger/index.html
Exchanges
GET /Api/v1.0/Exchange/GetExchangeByUserId/{userId}
POST /Api/v1.0/Exchange/AddExchange
POST /Api/v1.0/Exchange/Accept
POST /Api/v1.0/Exchange/Authorized
Parameters
No parameters
Request body
Example Value | Schema
{
"exchangeId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
"passwordPart1": "string",
"passwordPart2": "string"
}

- Generar el interés de los Gamers por títulos nuevos, a través de notificaciones de correo electrónico según sus gustos.



- Crear un software de calidad, lo que implica que se apliquen los siguientes ítems sobre el software; buenas prácticas de desarrollo, refactorización y escalabilidad

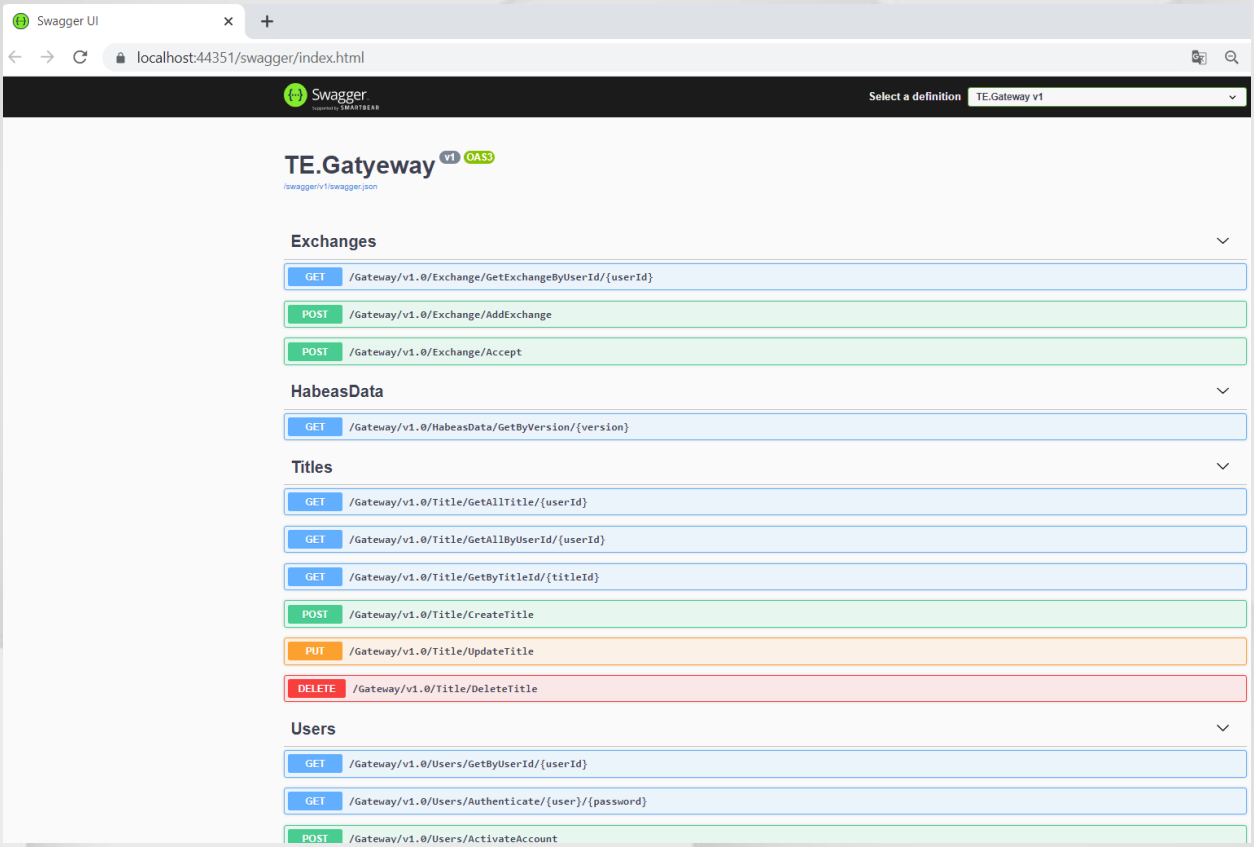
Se implantó la simplicidad en el código, las líneas necesarias para lograr el objetivo de la función, cero líneas con código quemado, inyección de dependencias, swagger en la documentación y testeabilidad, y lograr la escalabilidad no es complicado en el sentido de que todo está separado por modelos y versionados

```
[ApiController]
[Route(Routing.Title.Base)]
1 referencia
public class TitlesController : Controller
{
    private readonly ITitleService _titleService;

    0 referencias
    public TitlesController(ITitleService titleService)...

    [HttpGet]
    [Route(Routing.Title.GetAll)]
    0 referencias
    public async Task<BaseResponse> GetAll(Guid userId) => await _titleService.GetAll(userId);

    [HttpGet]
    [Route(Routing.Title.GetAllByUserId)]
    0 referencias
    public async Task<BaseResponse> GetAllByUserId(Guid userId) => await _titleService.GetAllByUserId(userId);

    [HttpGet]
    [Route(Routing.Title.GetByTitleId)]
    0 referencias
    public async Task<BaseResponse> GetByTitleId(Guid titleId) => await _titleService.GetByTitleId(titleId);

    [HttpPost]
    [Route(Routing.Title.Create)]
    0 referencias
    public async Task<BaseResponse> CreateTitle([FromBody] AddTitleDTO model) => await _titleService.AddAsync(model)
```

UserService.cs | TitleService.cs | ExchangeService.cs | UsersController.cs | appsettings.json | ExchangesController.cs | EmailMachine.cs

TE.Application | TE.Application.Services.Implementations.EmailMachine | EnviarCorreo(string destinatario, string asunto, string mensaje, bo

```
1   using ...
7
8   namespace TE.Application.Services.Implementations
9   {
        2 referencias
10      public class EmailMachine : IEmailMachine
11      {
12          private readonly SMTPSettings _sMTPSettings;
13          private readonly SmtpClient _cliente;
14
            0 referencias
15          public EmailMachine(IOptions<SMTPSettings> sMTPSettings)
16          {
17              _sMTPSettings = sMTPSettings.Value;
18              _cliente = new SmtpClient(_sMTPSettings.Host, _sMTPSettings.Port)
19              {
20                  EnableSsl = _sMTPSettings.EnableSSL,
21                  DeliveryMethod = SmtpDeliveryMethod.Network,
22                  UseDefaultCredentials = false,
23                  Credentials = new NetworkCredential(_sMTPSettings.User, _sMTPSettings.Password)
24              };
25          }
26
            4 referencias
27          public async Task EnviarCorreo(string destinatario, string asunto, string mensaje, bool esHtlm = false)
28          {
29              MailMessage message = new MailMessage(_sMTPSettings.User, destinatario, asunto, mensaje);
30              message.IsBodyHtml = esHtlm;
31              await _cliente.SendMailAsync(message);
32          }
33
```


Swagger UI
localhost:44351/swagger/index.html
Swagger
Select a definition
TE.Gateway v1
TE.Gatyeway
v1
OAS3
/swagger/v1/swagger.json
Exchanges
GET /Gateway/v1.0/Exchange/GetExchangeByUserId/{userId}
POST /Gateway/v1.0/Exchange/AddExchange
POST /Gateway/v1.0/Exchange/Accept
HabeasData
GET /Gateway/v1.0/HabeasData/GetByVersion/{version}
Titles
GET /Gateway/v1.0/Title/GetAllTitle/{userId}
GET /Gateway/v1.0/Title/GetAllByUserId/{userId}
GET /Gateway/v1.0/Title/GetByTitleId/{titleId}
POST /Gateway/v1.0/Title/CreateTitle
PUT /Gateway/v1.0/Title/UpdateTitle
DELETE /Gateway/v1.0/Title/DeleteTitle
Users
GET /Gateway/v1.0/Users/GetByUserId/{userId}
GET /Gateway/v1.0/Users/Authenticate/{user}/{password}
POST /Gateway/v1.0/Users/ActivateAccount