Universidad de las Ciencias Informáticas

Facultad 4

“Módulo de chat para el portal de videojuegos Cosmox”

Trabajo Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor

Claudia de la Caridad Mercadet Vasallo

Tutores

Ing. Maydalis Hernández Pérez

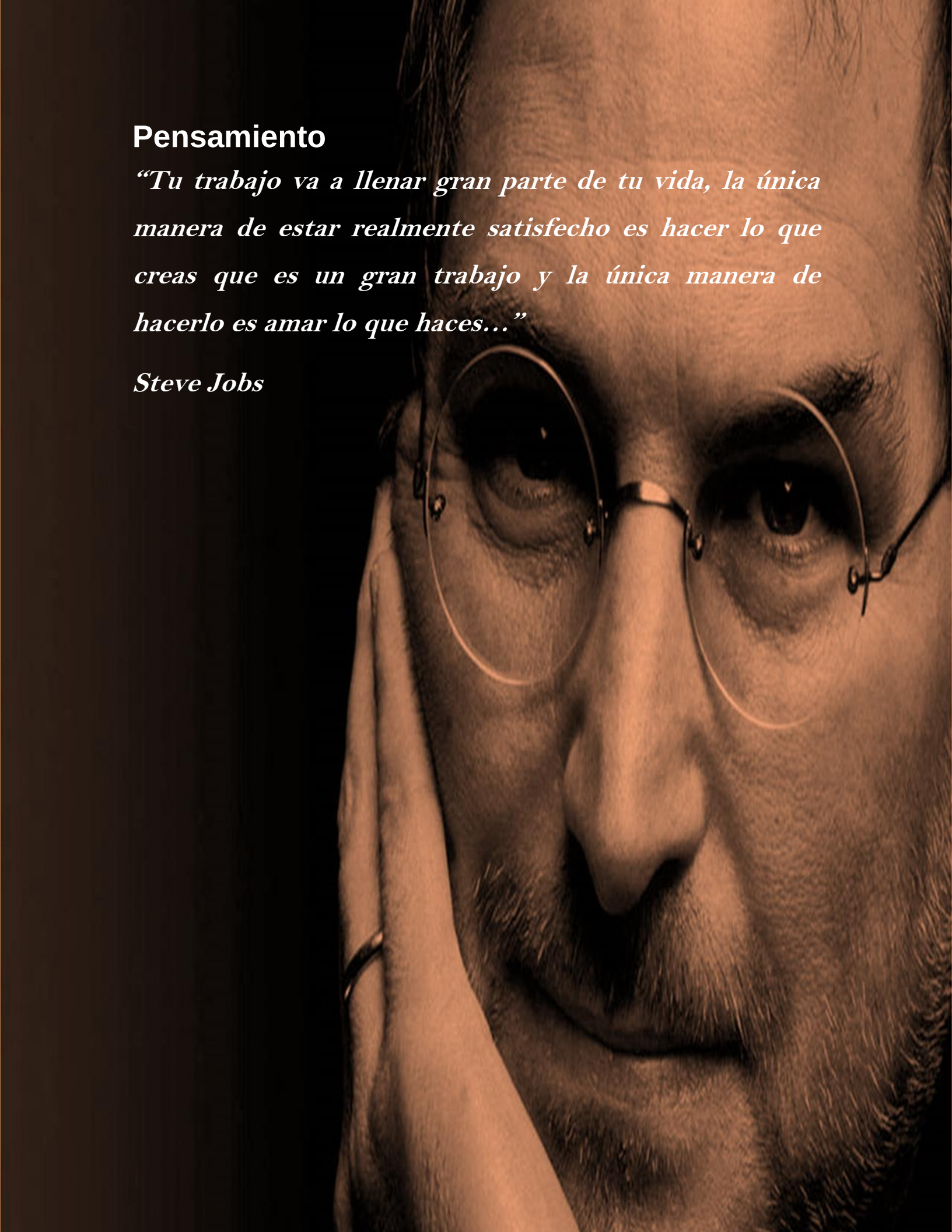
Ing. Luis Ángel Llull Céspedes

La Habana, diciembre de 2022

Pensamiento

“Tu trabajo va a llenar gran parte de tu vida, la única manera de estar realmente satisfecho es hacer lo que creas que es un gran trabajo y la única manera de hacerlo es amar lo que haces…”

Steve Jobs



Dedicatoria

Dedico este trabajo a mi familia, en especial a mi madre y padre que han sido mi apoyo durante estos duros años de sacrificio, gracias a ustedes soy cada día una mejor persona.

Agradecimientos

A mi madre, a mi padre, a mis hermanas y a Charlie por su amor, comprensión y apoyo incondicional.

A mi familia por su preocupación y apoyo.

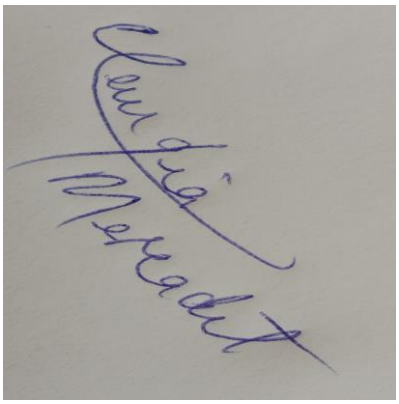
A mis tutores, amigos y todas aquellas personas que contribuyeron de manera positiva en esta etapa de mi vida.

Muchas gracias a todos

DECLARACIÓN DE AUTORÍA

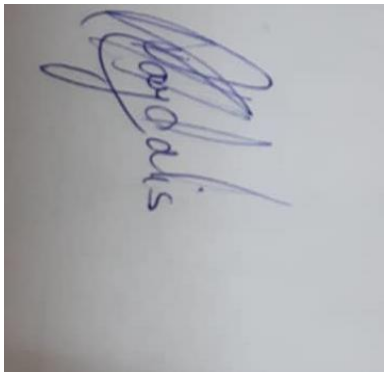
Declaramos ser autores de la presente tesis que tiene por título: y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los 7 días del mes de diciembre del año 2022

<Claudia de la C. Mercadet Vasallo>



Firma del Autor

< Maydalis Hernández Pérez **>**



Firma del Tutor

< Luis Ángel Llull Céspedes **>**

Firma del Tutor

Resumen

El Centro de Entornos Interactivos 3D VERTEX adscrito a la Facultad 4 de la Universidad de Ciencias Informáticas, desarrolla productos y servicios informáticos asociados a los Entornos Interactivos 3D con un alto valor agregado. COSMOX, es un portal videojuegos desarrollado por el centro que tiene como objetivo ofrecer un espacio virtual, que sirva como portal integrador de todo el conjunto de contenidos y servicios afines a los videojuegos cubanos. Entre los rasgos que presentan los videojuegos está la presencia de un chat global donde todos los jugadores comparten experiencias del videojuego. Hoy el portal Cosmox no cuenta con un chat que permita la participación e intercambio entre los usuarios de la comunidad. Por el motivo planteado el presente Trabajo de Diploma tuvo como objetivo desarrollar un módulo de chat de texto para del portal Cosmox que permitiera la comunicación sincrónica y asincrónica entre los jugadores a través de salas de chat. Para el desarrollo de este sistema, se empleó XP como metodología de desarrollo de software. Se utilizaron como herramientas y tecnologías a Django 4.1.1 como framework, Django Channels para el trabajo con websockets, PostgreSQL 14 como sistema gestor de bases de datos, Visual Paradigm en su versión 8.0 como herramienta de modelado y el lenguaje de programación Python 3.10 desde el lado del servidor y HTML, Java Script y CSS del lado del cliente.

Palabras clave: chat, comunicación, salas, portal, videojuegos.

Abstract

The VERTEX 3D Interactive Environments Center, attached to Faculty 4 of the University of Informatics Sciences, develops computer products and services associated with 3D Interactive Environments with high added value. COSMOX is a video game portal developed by the center that aims to offer a virtual space that serves as an integrating portal for the entire set of content and services related to Cuban video games. Among the features that video games present is the presence of a global chat where all players share video game experiences. Today the Cosmox portal does not have a chat that allows the participation and exchange between the users of the community. For the reason stated, this Diploma Work aimed to develop a text chat module for the Cosmox portal that would allow synchronous and asynchronous communication between players through chat rooms. For the development of this system, XP was used as software development methodology. Django 4.1.1 as a framework, Django Channels for working with websockets, PostgreSQL 14 as a database management system, Visual Paradigm in its version 8.0 as a modeling tool and the Python 3.10 programming language from the server side and HTML, Java Script and CSS on the client side.

Keywords: chat, communication, gamers, portal, video games.

Índice

Índice de tablas

Índice de Imágenes

Introducción

La comunicación ha marcado la evolución de la humanidad a través del tiempo. La creación, búsqueda y obtención de información son acciones esenciales a la naturaleza humana. Dado a eso los grandes saltos evolutivos de la humanidad tienen como hito la instauración de algún nuevo instrumento de comunicación. Es comprensivo entonces que la revolución tecnológica introducida en las comunicaciones influya en el comportamiento de los hombres. Las conductas de los grupos sociales y las actitudes de los mismos no se comprenden actualmente sin la existencia de los más modernos medios de comunicación, cuyo primer resultado ha sido una aproximación de los hombres del mundo.

Los medios de comunicación masiva son las herramientas para lograr que un solo emisor se comunique de forma idéntica con millones de receptores al mismo tiempo (Goya, 2012). De acuerdo con la tecnología o el formato utilizado, los medios masivos de comunicación se pueden clasificar en: Medios impresos, Radiodifusión, Cine, Televisión, Internet y servicio de red social.

Entre los medios de comunicación masivo con gran auge en la actualidad, y que constituyen una expansión de las formas de comunicación a través de tecnología están las redes sociales. El término “red social” es definido por la Real Academia Española (RAE) como “plataforma digital de comunicación global que pone en contacto a un gran número de usuarios”. Actualmente, es tal la importancia de las redes que llega a ser una obligación para sentirse integrado en la sociedad, para poder comunicarse, como si no existiese otra forma de hacerlo. Esto ocurre, en gran medida, en la población más joven (Pardo, 2018). Este tipo de espacios de intercambio han aumentado exponencialmente las oportunidades para compartir información, tanto social o lúdica como laboral.

Es claro, que la tecnología ha transformado la forma en que la humanidad vive, se comunica y aprende. Las décadas precedentes se han convertido de manera acelerada, en un lapso de creaciones y avances tecnológicos disruptivos, como la Inteligencia Artificial, el aprendizaje automático y el análisis de datos a gran

escala. De ahí que en los últimos años haya aumentado exponencialmente, entre otros, el uso de celulares, computadoras, videojuegos, televisores nombrados como inteligentes, y la web semántica, para el desarrollo cotidiano de actividades (Chibuque, 2022).

Continuamente se generan y mejoran dichas tecnologías en procesos de iteración constante; un claro ejemplo son los videojuegos que, desde sus orígenes, a finales de los años cincuenta y principio de los sesenta, evolucionaron gradualmente (Chibuque, 2022).

Los videojuegos son un tipo de juego especial, cuyo principal objetivo es la diversión. Requieren de un medio electrónico para ser ejecutados, donde los usuarios pueden interactuar tanto con el videojuego como con otros jugadores (Chibuque, 2022).

Los videojuegos dejaron de ser desde hace muchos años una simple opción de entretenimiento de niños y adolescentes para pasar a ser la industria más rentable y próspera del mundo. Gran parte de su reciente bonanza económica se debe a que pasaron de estar en una consola clásica, a ser multiplataforma, ahora los juegos se encuentran en tabletas, teléfonos inteligentes, redes sociales y otros dispositivos. La innovación tanto en hardware como en software han sido factores clave que impulsan el crecimiento de esta industria.

Los videojuegos están compuestos por diversos elementos y reglas específicos, por una narrativa, e implican la participación activa de uno o más jugadores. Recrean entornos y situaciones virtuales en los que el jugador puede controlar a uno o varios personajes (o cualquier otro elemento de dicho entorno), para conseguir uno o varios objetivos por medio de reglas determinadas. La finalidad principal de los videojuegos es el entretenimiento; y en el mercado actual hay una gran variedad, algunos de los más jugados son: Call of Dutty, Sartcraft, World of Warcfrat, Street Fighter, Mortal Kombat, Los Sims.

Los videojuegos cuentan con gran popularidad también Cuba y de igual manera en la isla existen entidades que se han dado a la tarea de crear videojuegos

autóctonos pues este tema se ha convertido un tema imprescindible de la sociedad actual, además de ser una línea fundamental de la política de informatización en el país, como son el caso los Joven Club de Computación, Cinesoft y los Estudios de Animación el Instituto Cubano de Arte e Industria Cinematográfica (ICAIC) en colaboración con la Universidad de Ciencias Informáticas (UCI).

El Centro de Entornos Interactivos 3D (VERTEX) adscrito a la Facultad 4 de la UCI, desarrolla productos y servicios informáticos asociados a los Entornos Interactivos 3D con un alto valor agregado, resultado de un ciclo completo de Investigación + Desarrollo + innovación (I+D+i).

La colaboración entre la UCI y el ICAIC permitió complementar la formación en áreas como el diseño, la animación de personajes y hasta la mercadotecnia. El resultado es que, de conjunto, hoy se ha aportado al país más de 12 títulos de videojuegos de muy buena factura y originalidad, por la historia, las costumbres y otros aspectos de la isla. Ejemplos son Aventuras en la manigua, Especies invasoras, Villa tesoro, Kuba Kart, y demás.

La Universidad de las Ciencias Informáticas lanzó el portal Cosmox, donde jugadores de todo el territorio podrán medir sus habilidades en videojuegos creados por la UCI y los Estudios de Animación ICAIC (Rebelde, 2020).

El portal de videojuegos Cosmox tiene como objetivo ofrecer un espacio virtual, que sirva como plataforma integradora de todo el conjunto de contenidos y servicios afines a los videojuegos cubanos, fundamentalmente desarrollados por la Alianza Estratégica entre la Universidad de las Ciencias Informáticas y los Estudios de Animación ICAIC (Rebelde, 2020).

Según su página web, se pretende crear un canal de comunicación interactivo, dinámico y comprometido con los contenidos cubanos. Adicionalmente a los enlaces de descarga de los videojuegos para diferentes plataformas, Cosmox permite a los usuarios interactuar con el portal desde los propios videojuegos, con

el objetivo de publicar o compartir con la comunidad el avance dentro de los juegos (Rebelde, 2020).

Los principales contenidos que se ofrecen son: Videojuegos: contiene una descripción detallada de los videojuegos, así como los enlaces de descarga para las diferentes plataformas. Adicionalmente, se puede descargar la música, el póster y los videos promocionales. Rankings*:* muestra la tabla de posiciones de los mejores jugadores por videojuegos o niveles de videojuegos. Noticias: contiene información de los temas más novedosos referente a los videojuegos en el ámbito nacional e internacional. Historial: muestra en orden cronológico los lanzamientos de los videojuegos. FAQ: facilita un conjunto de preguntas frecuentes y sus respuestas, con el objetivo de esclarecer a los usuarios en el uso del portal y sus servicios. Contacto*:* posibilita las vías de comunicación de los usuarios con los administradores del portal (Rebelde, 2020).

En el portal se encuentran varios de los principales videojuegos cubanos: La Chivichana, Súper Claria, Especies Invasoras, Aventuras en la Manigua, La Neurona 1 y 2, Villa Tesoro, Caos Numérico, Kuba Kart y Coliseum completan la lista hasta el momento (Rebede, 2020).

En un portal web, la presencia de un chat portara una serie de beneficios como: buena oportunidad de mostrar la personalidad del lugar, permitiría auxiliar a los usuarios y resolver sus dudas con más rapidez, facilitaría atender a varias personas a la vez, ofrecimiento de información adicional al instante y generaría más confianza en el consumidor. Otro ejemplo de los beneficios que aportan es la posibilidad de recopilar información de interés, preocupaciones o dudas de los usuarios para así ofrecer mejor servicio y experiencia contribuyendo de esta manera al mejoramiento del portal web.

En vista de lo anterior planteando y retomando el tema de los videojuegos es conveniente mencionar que entre las particularidades que presentan estos, está la presencia de un chat global donde todos los jugadores se conectan y comparten experiencias del videojuego en cuestión, ya sea en sitio web dedicado a este o en el mismo videojuego en caso de que sea online.

Actualmente los videojuegos con cuenta portal no cuenta con un mecanismo de comunicación donde la comunidad de Cosmox interactúe y comparta vivencias. Teniendo en cuenta la situación antes descrita se define como **problema de investigación:** ¿Cómo dotar al portal de videojuegos Cosmox de un chat que permita la comunicación sincrónica y asincrónica de los jugadores?
Para dar solución al problema de investigación planteado se define como **objetivo:** Desarrollar un módulo de chat de texto para los juegos del portal de videojuegos Cosmox que permita la comunicación sincrónica y asincrónica entre los jugadores.

A partir del problema planteado se identifica como **objeto de estudio**: Tecnologías que permiten la comunicación sincrónica y asincrónica utilizados en los chats.

Se delimita dentro del objeto de estudio de la investigación el **campo de acción**: Tecnologías que permitan la comunicación sincrónica y asincrónica utilizados en los chats desarrollados en Django.

Para dar cumplimiento al objetivo planteado es necesario realizar las siguientes **tareas de investigación**:

- Elaboración del marco teórico de la investigación mediante el estudio de Tecnologías que permitan la comunicación sincrónica y asincrónica.
- Definición de la metodología de desarrollo de software, herramientas y tecnologías a utilizar para el desarrollo del módulo.
- Planificación y diseño del módulo teniendo en cuenta lo establecido en la metodología de desarrollo de software seleccionada.
- Implementación de un módulo de chat que permita la comunicación entre la comunidad del portal Cosmox.
- Ejecución de pruebas al módulo para garantizar su correcto funcionamiento.

Para la realización de las tareas expuestas anteriormente se emplearon los siguientes **métodos de investigación:**

Métodos teóricos: constituyen el enfoque general para abordar los problemas científicos, permitiendo profundizar en las regularidades esenciales de los fenómenos.

- **Histórico-lógico:** este método se usó con el objetivo de realizar un estudio de los referentes teóricos, que sustentan el desarrollo y utilización de las tecnologías websocket que permiten la comunicación sincrónica y asincrónica.
- **Analítico-sintético:** permitió durante el proceso investigativo el estudio, análisis e identificación de los conceptos relacionados con la temática abordada. Posibilitó realizar el análisis teórico e identificar los principales conceptos a incluir en la fundamentación teórica y el análisis de la información. Facilitó la realización de un análisis de las herramientas similares que responden al objeto de estudio de la investigación, definiéndose un conjunto de indicadores para realizar una comparación entre las herramientas, con el objetivo de identificar características que tributen al desarrollo de la propuesta.
- **Modelado:** permitió crear abstracciones con el objetivo de explicar la realidad. Es empleado como herramienta para la comprensión del problema a resolver a partir de la creación de los prototipos de interfaz que permitan el diseño de la solución.

Métodos Empíricos: estudian los fenómenos, objetos y procesos observables, confirmados a través de la hipótesis y la teoría.

- **Entrevista:** este método de investigación fue utilizado para obtener información de forma directa con el cliente para determinar las herramientas que se utilizarían y el levantamiento de requisitos funcionales que presenta el sistema a través de una entrevista semiestructurada al arquitecto del proyecto de la Plataforma Cosmox, Juan Gabriel Valdés Días.
- **Observación:** posibilitó estudiar y observar el funcionamiento de sistemas similares, identificándose funcionalidades y características que se tuvieron en cuenta para el desarrollo de la solución.

- **Revisión documental:** permitió la obtención de información relacionada con la temática abordada mediante el asesoramiento a través de diferentes artículos y libros.

El presente trabajo cuenta con la siguiente estructura capitular:

Capítulo 1: Fundamentos y referentes teórico-metodológicos sobre el objeto de estudio.

En este capítulo se presentan los elementos teóricos que sirven de base a la investigación del problema planteado. Se describen los conceptos fundamentales asociados al dominio del problema; además, se proporciona una descripción de las herramientas, tecnologías y metodología de desarrollo de software a utilizar para dar solución al problema planteado

Capítulo 2: Planificación y diseño de la solución propuesta al problema científico.

En este capítulo se realiza la planificación y diseño de la propuesta de solución según los requisitos funcionales obtenidos. Se realiza una descripción del sistema que se pretende desarrollar, se elaboran las Historias de Usuarios para identificar las funcionalidades con las que debe cumplir la aplicación, además de presentar la arquitectura seleccionada para el desarrollo del sistema.

Capítulo 3: Implementación y prueba del módulo.

En este capítulo se presentan los elementos relacionados con la implementación del módulo. Se define la estrategia de pruebas a realizar para verificar el cumplimiento de los requisitos de software y se presenta el resultado del proceso de prueba y de la validación del módulo.

1 Capítulo1. Fundamentación Teórica

En este capítulo se sintetiza la búsqueda y análisis de la información relacionada con el dominio del problema. Son descritos los conceptos fundamentales relacionados con la investigación y analizadas algunas soluciones existentes referentes al mismo entorno. En el desarrollo del mismo se seleccionan las tecnologías lenguajes, marcos de trabajo y herramientas que se utilizarán en el desarrollo de la solución propuesta, así como la justificación de la metodología a emplear.

1.1 Conceptos asociados al dominio del problema:

El presente capítulo trata algunos de los conceptos asociados al dominio de la investigación, haciéndose necesario realizar un estudio de soluciones similares existentes y de las tecnologías idóneas para la implementación de la propuesta de solución, fundamentando de este modo las bases de la investigación.

Herramientas de Comunicación

El concepto de herramientas de comunicación ha sido expresado en diversos contextos, dado a que abarca un amplio terreno, por autores que han estudiado esta rama, algunas de las acepciones son:

Para Valverde, son “medios de comunicación interpersonal que a través de Internet adoptan dos formas: sincrónica, en la que los usuarios a través de una red telemática coinciden en el tiempo y se comunican entre sí mediante texto, audio y/o vídeo; y asincrónica donde los participantes utilizan el sistema de comunicación en tiempos diferentes” (MATHEUS, 2019).

Por otro lado, Martínez & Solano señalan que “las herramientas comunicativas, permiten las posibilidades de romper las barreras espacio temporales en las acciones formativas virtuales, y que son percibidas como ventajosas” (MATHEUS, 2019).

Rincón, establece que “las herramientas comunicativas son aquellas herramientas de comunicación que pueden ser sincrónicas o asincrónicas; es decir, que el emisor y el receptor se encuentren en diferentes espacios y tiempos en el acto de comunicar, permitiendo así la participación que cooperará a la construcción social del conocimiento, a partir de la interacción e interactividad adecuadas para tal fin” (MATHEUS, 2019).

Teniendo en cuenta los conceptos anteriores se ha definido en la presente investigación que se llama herramientas de comunicación a aquellas que utilizan las tecnologías de la información y comunicación como medio para desarrollar capacidades de diálogo, de discusión y debate, de interacción y comunicación y, en definitiva, de información, de manera sincrónica y asincrónica.

Comunicación sincrónica

Son las herramientas utilizadas para la comunicación que ocurren en el mismo tiempo sin importar la distancia que medie entre los usuarios, según la Real Academia de la Lengua Española (RAE), sincronía es la “coincidencia de hechos o fenómenos en el tiempo”. Las más conocidas de estas herramientas son: el chat, las video conferencias, y la pizarra compartida.

Por esta razón, las herramientas de comunicación sincrónicas son aquellas en las cuales se está hablando en tiempo real. Se refiere al acceso inmediato, en tiempo real de información u otros datos, este tipo de comunicación, suelen ser similares a la del diálogo mantenido cara a cara. Resulta dinámico, en donde una conversión evoluciona en tiempo real. Esta además intenta simular simbología paralingüística que refleja estados de ánimo y gestos como son el empleo de los denominados emoticonos o recursos expresivos como las exclamaciones que indica en ocasiones enfado o firmeza (MATHEUS, 2019).

Teniendo en cuenta los conceptos anteriores se define en la presente investigación que la comunicación síncrona es el proceso de comunicación que ocurre en tiempo real.

Comunicación asincrónica

Lo asincrónico es totalmente contrario a lo sincrónico y se define como proceso o efecto que no ocurre en completa correspondencia temporal con otro proceso u otra causa. Estas herramientas son: el foro y el correo electrónico (MATHEUS, 2019).

Para el autor Araujo, la comunicación asincrónica se define como un proceso o efecto que no ocurre en completa correspondencia temporal con otro proceso u otra causa. La comunicación se produce entre dos o más personas que pueden o no, encontrarse físicamente ubicadas en diferentes contextos; esta comunicación solo se desarrolla en formato escrito (Lay, y otros, 2019).

Según el autor Matías, las herramientas de comunicación asincrónicas “son aquéllas en las que la comunicación no se produce a tiempo real, es decir, en las que los participantes no están conectados en el mismo espacio de tiempo” (Lay, y otros, 2019).

Teniendo en cuenta los conceptos anteriores se define en la presente investigación que la comunicación asíncrona es aquella que se lleva a cabo entre personas de manera diferida en el tiempo, es decir, cuando no existe coincidencia temporal.

Mensajería instantánea

La mensajería instantánea se basa en la arquitectura cliente-servidor para enviar y recibir mensajes. El cliente está instalado en una computadora particular de un usuario final, la cual sería la interfaz que este utiliza para comunicarse con otros usuarios, pero el servidor es el que maneja toda la comunicación cliente/servidor, habilitando los permisos si corresponden, todo ello mantenido por un proveedor de dicho servicio. En este modelo, el servidor no sólo tiene la responsabilidad del envío de los mensajes, sino también de la autenticación de los usuarios, debido a que los clientes no se conectan directamente entre sí (López, 2005). Sus características más relevantes, entre otras, son: permitir saber en cada momento si una persona que es parte de una lista de contactos está en línea, con la cual se pueda tener una relación en tiempo real, propiciando el intercambio de información o chat; transferir archivos, conversar mediante el uso de micrófonos;

ejecutar sonidos; controlar o bloquear mensajes de determinadas personas; acceder a la lectura del correo electrónico personal y a Web links de interés, con noticias y sitios favoritos, entre otras características propias de cada producto (López, 2005).

Teniendo en cuanta la definición anterior se define en la presente investigación que la mensajería instantánea es una forma de comunicación en tiempo real entre dos o más personas basada en texto. El texto es enviado a través de dispositivos conectados ya sea a una red como Internet, (sin importar la distancia que exista entre los dos (o más) dispositivos conectados.

Chat

En general la noción de chat se usa para nombrar al intercambio de mensajes escritos de manera instantánea. Es decir, cuando un usuario escribe el mensaje y lo envía, el destinatario lo recibe al instante. Lo mismo ocurre si el usuario deja su mensaje en una sala pública (Merino, 2022).

Los chats se consideran como canales de conexión entre los diferentes ordenadores que posibilitan el dialogo colectivo entre sujetos que se encuentran separados en espacios físicos diferentes (Sáez, 2007).

Se utiliza el término conversación escrita para describir el género y la actividad comunicativa del chat, subrayando así su rasgo más peculiar: la combinación de formato escrito y concepción oral (Rodríguez, 2002).

Teniendo en cuenta los conceptos anteriores se define en la presente investigación que los chats son un servicio de comunicación en tiempo real que se realiza entre varios usuarios cuyas computadoras están conectadas a una red, generalmente Internet; los usuarios escriben mensajes en su teclado, y el texto aparece automáticamente y al instante en el monitor de todos los participantes.

Ventana de chat

Una ventana de chat es una ventana utilizada por un programa de chat para permitir que un usuario vea los mensajes que ha recibido, así como los mensajes

que ha enviado. Esta ventana generalmente se divide en dos áreas básicas: la sección de visualización que muestra los mensajes que ha enviado cada usuario y la sección de entrada que permite que un usuario vea el mensaje que está creando actualmente. Una ventana de chat puede ser parte de un programa de chat, como los programas de mensajería instantánea, o puede existir dentro de un marco más amplio, como un juego de computadora en línea o un sitio web de redes sociales (Spiegato, 2022).

Si bien una ventana de chat es a menudo una parte básica de cualquier programa de mensajería instantánea, también existen otras aplicaciones para las opciones de chat. Los videojuegos en línea, especialmente los juegos multijugador masivo en línea, a menudo incluyen ventanas de chat para la comunicación entre jugadores. Estas ventanas a menudo son personalizables y se pueden cambiar para recibir solo mensajes de ciertos jugadores o incluir varias pestañas para proporcionar información diferente en capas discretas de una sola ventana. Los sitios web de redes sociales también suelen incluir programas de mensajería instantánea para los usuarios que inician sesión en el sitio, que normalmente utilizan ventanas de chat conectadas al navegador web que se utiliza para acceder a dichos sitios (Spiegato, 2022).

Teniendo en cuenta los conceptos anteriores se define en la presente investigación que las ventanas de chats son usadas por programas de chat con el objetivo del intercambio de mensajes en tiempo real entre dos o más usuarios.

Salas de chat

A las salas de chateo se puede acceder por medio de una conexión a Internet con programas como iCq (I seek you) o irC (Internet Relay Chat) o cualquiera de sus versiones asociadas a los grandes servidores (Yahoo Messenger, msN). Al momento de ingresar a estas salas de chateo, los usuarios entablan un intercambio verbal entre sí en tiempo real a través de una escritura mediatizada por el teclado. Aún más, los avances en este campo incluyen ahora la posibilidad

de teleconferencias por medio de cámaras y micrófonos incorporados al computador (Farías, 2008).

Se entiende por “salas de chat” como aquellas plataformas virtuales enmarcadas en el contexto de un ambiente donde existan, al menos, más de dos usuarios que pretendan mantener conversaciones, inicialmente, públicas. Este concepto es más restringido que el del uso social del verbo “chatear” donde en espacios como el MSN o chat de Facebook se utiliza para las comunicaciones entre dos usuarios. Sin embargo, esta definición de sala de chat intenta referenciar a aquellos espacios donde la intencionalidad del sujeto es proceder a establecer contacto de manera abierta y pública con una cantidad de personas indeterminadas, excluyendo todos los demás (Pichel, 2012).

Teniendo en cuenta los conceptos anteriores se ha definido en la presente investigación que una sala de chat es un medio para establecer la comunicación síncrona entre dos o más usuarios conectados a una red.

1.2 Estudio de herramientas similares

Actualmente existen varios portales de videojuegos que al igual que Cosmox ofrecen una gran diversidad de servicios a sus clientes. Para la presente investigación se hizo necesario el estudio de estos para determinar si es común el uso de herramientas para la comunicación en este tipo de portales y como se llevan a cabo.

Steam Chat

Steam Chat, es una aplicación creada por Valve para unir las comunidades de Steam de las plataformas móviles y de PC a través del chat en los dispositivos móviles.

Las conversaciones no se limitan al texto. De hecho, en ellas será posible incluir emojis o GIFs, al más estilo WhatsApp, además de vídeos o tuits, entre otros. Además, Steam Chat cuenta con una lista de amigos en la que podrás ver qué contactos están conectados jugando de forma rápida además de poder invitarlos a

jugar a través de un enlace generado desde la propia app y enviado por texto o email (Macías, 2019).

Ubisoft Connect

Ubisoft Connect es el ecosistema de servicios para los jugadores de todos los juegos de Ubisoft en todas las plataformas. Su finalidad es ofrecer el mejor entorno para que todos los jugadores disfruten de sus juegos y estén en contacto entre ellos sea cual sea su dispositivo (UbisoftConnect, 2022).

Ubisoft asegura que su nuevo sistema de chat es mejor, más rápido, más seguro y estable que su predecesor, que fue instalado en Uplay. Además, la empresa también está preocupada por la toxicidad en el chat, afirma querer construir un entorno más seguro y amigable, cuando alguien está molestando, la empresa advierte a los usuarios que bloqueen el contacto y denuncien (Mascaretti, 2021).

Battle.net

En Battle.net se tiene acceso a prestigiosos juegos para PC aclamados por la crítica de franquicias como Warcraft, Overwatch, Diablo, StarCraft y Call of Duty. Se puede obtener los últimos juegos y expansiones. Tiene opciones obsequios digitales o permite completar los personajes del juego con mascotas, monturas y objetos cosméticos (Blizzard, 2022).

Con Battle.net, socializar es más sencillo gracias a sus elementos personalizables, como perfiles, chats, grupos, chat de voz y actualizaciones de estado en tiempo real (Blizzard, 2022).

GOG

GOG de sus iniciales Good Old Games, es una plataforma para comprar videojuegos de las muchas que existen hoy en día. Pertenece a CD Projekt conocidos por videojuegos como la saga The Witcher o Cyberpunk 2077, y que inicialmente se dedicaba, como su propio nombre indica, a distribuir videojuegos antiguos para hacerlos funcionar en equipos nuevos. GOG vende videojuegos de muchos desarrolladores y va añadiendo cada vez más contenido, actualmente

GOG también vende videojuegos actuales además de los antiguos. Todos estos juegos que se venden a través de GOG.com o su plataforma de juegos están libres de DRM, esto es que no están protegidos y no necesitan de algún lanzador especial para poder jugar a ellos (Delgado, 2020).

Origin

Origin es la nueva plataforma de venta directa al consumidor de EA que le permite al usuario jugar donde quiera y cuando quiera a los títulos más esperados. Ya disponible en www.origin.com, los usuarios pueden encontrar, comprar y descargar todos los títulos de EA. Más de 150 juegos ya están disponibles (Cabeza, 2011).

Características de Origin (Cabeza, 2011):

- Origin ofrecerá a los usuarios el modo más sencillo de comprar y jugar a los mejores juegos de EA mientras están conectados con sus amigos.
- **Exclusivas digitales**: ofrecerá descargas completas de los últimos juegos de PC. Se da acceso a las últimas betas online, así como a demos y pruebas de producto previas a la compra.
- **Conéctate, comparte y juega**: A través de la aplicación beta de Origin, será posible construir listas de amigos, importar contactos de Facebook y chatear sobre las experiencias en el juego con la comunidad de Origin. Conectar con amigos, ver quién está online y a qué juegos están jugando. Estas características estarán disponibles incluso durante el juego mediante un menú emergente.

Epic Games

El panel social de Epic Games es una función de la tienda de Epic Games donde los jugadores pueden conectarse con sus amigos durante el juego o mientras navegan por la tienda.

Con la nueva actualización social de Epic Games, los usuarios pueden habilitar el modo No molestar, conectarse con otros jugadores a través de amigos mutuos e incluso crear una lista de deseos, además de sistema de chat de grupo, una función que permite a los usuarios chatear por voz entre sí fuera del juego. Estas características facilitan la conexión y el juego con amigos en la plataforma de Epic Games (olg, 2021).

Itch.io

Itch.io se ha convertido en una plataforma ideal para adentrarse en el mundillo del desarrollo indie. Actualmente cuenta con más de 140.000 videojuegos para comprar, jugar o descargar de manera gratuita, desde los proyectos más experimentales hasta las obras independientes que más fama han cosechado en la industria. Cualquiera puede publicar su videojuego en Itch.io, aunque la plataforma, que ya cuenta con un equipo de personas dedicado a su gestión, realiza labores de selección y recomendación de aquellos nuevos proyectos que más interés suscitan (Zucarelli, 2022).

Rockstar Launcher

Rockstar Games, creadores de sagas tan importantes como Grand Theft Auto o Red Dead Redemption, ha presentado Rockstar Games Launcher, una nueva plataforma digital para PC similar a otros launchers de otras compañías como Steam o Epic Games Store con la que jugar a sus propios videojuegos, así como adquirirlos y coleccionarlos, todo bajo un mismo entorno creado y gestionado únicamente por Rockstar Games.

Así, Rockstar Games Launcher nace como una nueva tienda y plataforma digital de Rockstar Games para compatibles, disponible a través del siguiente enlace y con la que adquirir objetos digitales únicos como Tarjetas Tiburón; además, y para celebrar la llegada de esta nueva plataforma digital, Rockstar Games ofrece por tiempo limitado una copia digital de GTA San Andres de forma totalmente gratuita para todos aquellos que descarguen el propio launcher en su PC (Ciuraneta, 2019).

1.2.1 Análisis comparativo del estudio de herramientas similares

A continuación, se presentará una tabla con la comparación de los portales anteriormente descritos teniendo en cuenta los siguientes criterios: comunicación por chat/ fórum/ blog, la herramienta de comunicación que usan está dentro o fuera del portal, tipo de comunicación, utilizan chat grupal o salas de chat y tipo de comunicación.

Tabla 1Análisis comparativos de las herramientas similares

Homólogos	**Comunicación por chat/ fórum/ blog**	**La herramienta de comunicación está dentro o fuera del portal**	**Utilizan chat grupal o salas de chat**	**Tipo de comunicación**	**Chat: Texto/voz/video**
Steam	Chat, foro	Fuera de la plataforma/SteamChat	Sí	Sincrónica, Asincrónica	Texto, video, voz
Ubisoft Connect	Chat	Dentro de la plataforma	Sí	Sincrónica	Texto
Blizzard Battle.net	Chat	Dentro de la plataforma	Sí	Sincrónica	Texto, voz
GOG	Chat, Fórum, blog	Fuera de la plataforma/GOG Galaxy	Sí	Sincrónica, Asincrónica	Texto
Origin	Chat	Fuera de la plataforma	Sí	Sincrónica	Texto, voz
Epic Games	Chat	Dentro de la plataforma	Sí	Sincrónica	Texto, voz
Itch.io	No	No tiene chat	No tiene chat	No tiene chat	No tiene chat
Rockstar Launcher	No	No tiene chat	No tiene chat	No tiene chat	No tiene chat

Una vez realizado el estudio de las características que presentan los sistemas de comunicación en los portales de videojuegos se puede concluir que el uso del chat

en este tipo de portales aporta a sus clientes grandes beneficios y posibilidades A partir d esto se pudo estudiar el comportamiento y funcionamiento de los chats de estos portales para seleccionar algunas funcionalidades que pueden ser implementadas en el desarrollo de un chat para el portal de videojuegos Cosmox, como son:

- El establecimiento de comunicación a través de chat de texto.
- La comunicación sea llevada de manera sincrónica y asincrónica.
- Implementar un sistema de seguridad para bloquear y denunciar usuarios problemáticos.
- Permitir el envío de imágenes, emojis, gifs y audios.

1.1 Websocket

WebSocket permite establecer un canal de comunicaciones bidireccional. En contraste con los eventos enviados por el servidor (SSE), el protocolo WebSocket no está construido sobre HTTP. Sin embargo, define un comportamiento de saludo para cambiar la existente conexión HTTP para una conexión WebSocket de nivel más bajo. Este protocolo no trata de simular un canal de envío de información a partir del servidor sobre HTTP. Solamente define un protocolo de empaquetado sobre TCP. De esta manera WebSocket permite la comunicación en dos sentidos de forma nativa. El protocolo tiene dos partes: el saludo y la transferencia de datos. El saludo de apertura está orientado a ser compatible con software del lado del servidor e intermediarios basados en HTTP, de forma tal que un solo puerto puede ser usado por los clientes HTTP y por los clientes WebSocket comunicándose con el mismo servidor (Cruz, Troche y otros, 2015).

Después de recibir el encabezado de respuesta HTTP, la información será transmitida de acuerdo con el protocolo WebSocket. Esto significa en este punto que solamente serán transmitidos paquetes WebSocket por la red. Un paquete puede ser enviado en cualquier momento y en cualquier dirección (Cruz, Troche y otros, 2015).

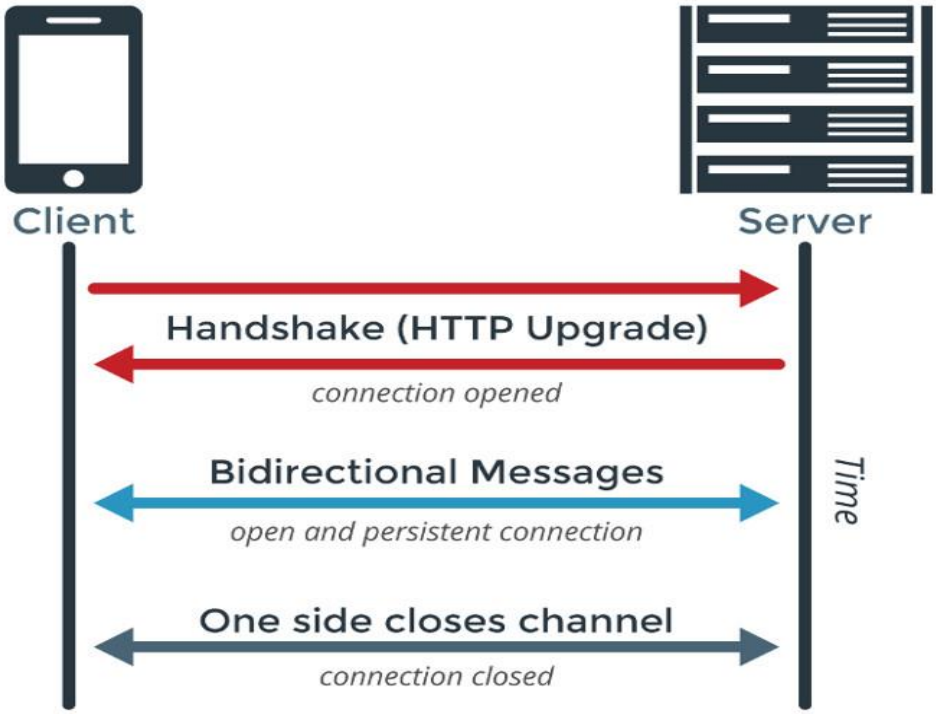


Ilustración 1 Esquema de funcionamiento de WebSocket

Para establecer una conexión WebSocket, el cliente y el servidor realizan un "upgrade" del protocolo HTTP al protocolo WebSocket durante el saludo inicial. Una vez establecido, los paquetes de datos WebSocket son enviados en ambos sentidos a través de un canal full-duplex a la misma vez (Cruz, 2015).

Ventajas (Cruz, 2015):

• Elimina el problema del sobre encabezado de los paquetes, transmitiéndose por la red solamente la información necesaria. En el protocolo WebSocket los mensajes tienen como encabezado entre 4 y 14 bytes, mientras que en HTTP oscilan aproximadamente entre 200 bytes y 2 Kb.

• El gasto involucrado en el manejo de WebSocket es mínimo. Protocolos Comet tales como Bayeux y BOSH, por ejemplo, usan algunos trucos para romper la barrera de HTTP de Petición-Respuesta. Esto fuerza a ambos protocolos a implementar un completo manejo de sesiones y conexiones. Debido al hecho de que WebSocket no está implementado sobre HTTP, nunca caerá en los problemas causados por las limitaciones de dicho protocolo.

- Los mensajes pueden viajar en ambos sentidos a la misma vez

- WebSockets está diseñado para funcionar sobre la infraestructura de red existente, por tanto, no tiene problemas con proxis intermediarios.

Utilizar WebSocket siempre que se necesite una conexión casi a tiempo real y de latencia baja entre el cliente y el servidor. Tomar en cuenta que esto podría significar tener que replantear cómo se ha desarrollado las aplicaciones de servidor, adoptando un nuevo enfoque en tecnologías como las colas de eventos (Gómez, 2018). Estos son algunos ejemplos de casos prácticos:

- Juegos online multijugadores.
- Aplicaciones de chat.
- Rotativos de información deportiva.
- Actualizaciones en tiempo real de las actividades

1.2 Tecnologías que hacen empleo de Websocket para establecer comunicación asincrónica y sincrónica usadas por Django

En esta sección se estudiarán tecnologías que hacen uso de websocket para establecer comunicación sincrónica y asincrónica usadas por Django. En el final de estudio se seleccionará cual se usará para el desarrollo del modula a implementar.

1.2.1 Django Channels

La mayoría de los frameworks de desarrollo Web, entre ellos Django, están construidos en torno al paradigma de la filosofía HTTP: cliente pide – servidor responde. Esto plantea un claro problema a la hora de desarrollar aplicaciones complejas que necesiten de una baja latencia o de una comunicación no dirigida por el cliente (Peña, 2016).

Channels es un módulo desarrollado para Django que trata de solucionar ese problema: provee un marco de trabajo en el que se pueden gestionar conexiones

con Websockets y HTTP2 (conexiones permanentes) así como el manejo de tareas asíncronas (Peña, 2016).

Esto se suma al manejo de HTTP que siempre ha ofrecido Django y cuyo comportamiento no se ve variado con la inclusión de Channels. Algunos de los conceptos que se manejan en Channels y que son clave para entender su implementación son:

Producer (productor): son eventos que deben ser escuchados y que transportan un mensaje. Estos eventos suelen ser una conexión desde un Websocket, un mensaje desde esa conexión o una desconexión (Peña, 2016).

Consumer (consumidor): se trata de un proceso o función en Python pensada para recibir un mensaje y ejecutarse de forma asíncrona, en segundo plano, en el sistema. Channel (canal): es en esencia una cola de tareas. Escucha mensajes enviados por los producers y delega su acción en el proceso llamado consumer. La capa de canales, o channel layer es el mecanismo de transporte que Channels usa para pasar los mensajes de los producers a los consumers, es decir, la que maneja los channels. Esta capa está pensada para ser usada con Redis como base de datos (Peña, 2016).

Group (grupo): es una clase introducida por Channels para manejar las respuestas múltiples. Los canales solo entregan mensajes a un único destinatario que está escuchando, por lo que intentar hacer broadcast implicaría enviar el mismo mensaje a numerosos destinatarios. Este problema se suple con el uso de grupos: un usuario se suscribe a un grupo y cuando se envíe un mensaje desde ese grupo, todos los usuarios suscritos lo recibirán. Lo que está haciendo Channels por debajo es simplemente evitar al desarrollador iterar sobre todos esos usuarios para enviar un mismo mensaje (Peña, 2016).

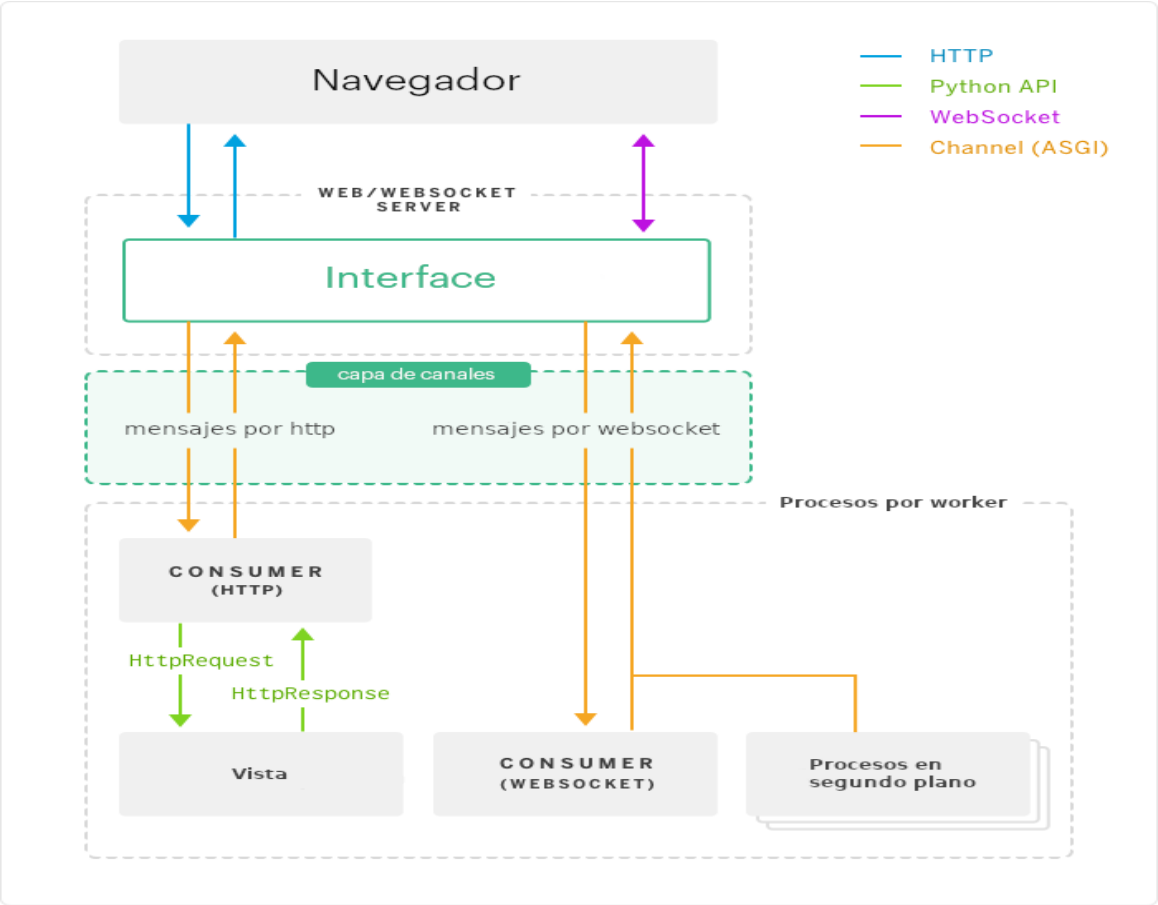


Ilustración 2Esquema de Funcionamiento de Django Channels

1.2.2 MQTT

MQTT es un protocolo de conectividad para el Internet de las Cosas, fue inventado en el año 1999 por el Dr. Andy Stanford-Clark de IBM y Arlen Nipper de Arcom (actualmente Eurotech). Fue diseñado para el transporte de mensajes extremadamente ligeros mediante la suscripción y publicación; es fácil de implementar y útil para conexiones en sitios remotos donde las redes posean poco ancho de banda y los dispositivos IoT trabajen con recursos limitados de memoria, batería y procesamiento. Además, intenta garantizar confiabilidad y seguridad en la entrega (MARTÍNEZ, 2019).

MQTT es un tipo de protocolo de mensajería de IoT liviano basado en el modelo de publicación/suscripción, que puede proporcionar un servicio de mensajería confiable y en tiempo real para dispositivos de IoT, solo usando muy poco código y

ancho de banda. Es adecuado para dispositivos con recursos de hardware limitados y el entorno de red con ancho de banda limitado. Por lo tanto, el protocolo MQTT se usa ampliamente en IoT, Internet móvil, IoV (Internet de los Vehículos), energía eléctrica y otras industrias (Tao, 2022).

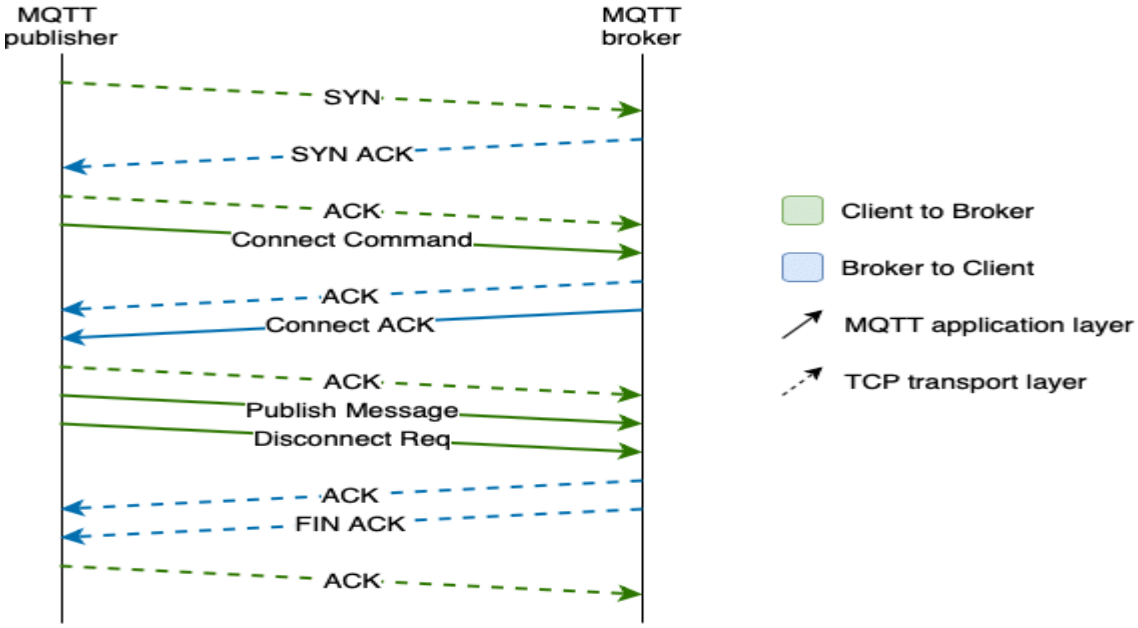


Ilustración 32Esquema de Funcionamiento de MQTT

MQTT y Python

La librería Paho-Mqtt de Python usada que permite conectarse como cliente a MQTT. Esto permite analizar los valores en los mensajes MQTT de los dispositivos para procesar, o activar algún proceso o evento con Python. La conexión requiere los parámetros de dirección IP del servidor, puerto, datos de usuario, contraseña y el tópico (Rosario, 2019).

Paho Python Client proporciona una clase de cliente compatible con MQTT v3.1 y v3.1.1 en Python 2.7 o 3.x. También proporciona algunas funciones de ayuda para que la publicación de mensajes únicos en un servidor MQTT sea muy sencilla (Tao, 2022).

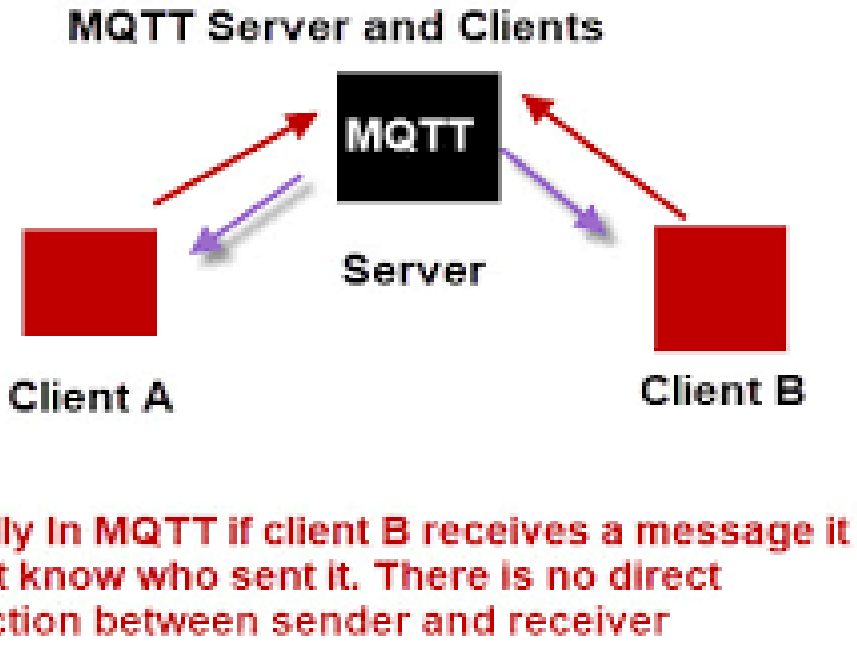


Ilustración 42Esquema de Funcionamiento de MQTT con Python

Después de terminado el estudio de las tecnologías que hacen uso de websocket para el establecimiento de comunicación sincrónica y asincrónica en Django, se concluyó que MQTT es más usado para envío de mensajes en proyectos basados Internet de las cosas (IOT) y Django Channels es más directo para el envío de mensajes en aplicaciones web, además de que fue creado específicamente para el manejo de websocket en Django. Por tanto, se definió utilizar como tecnología para el empleo de websockets para establecer comunicación sincrónica y asincrónica del proyecto a Django Channels.

1.3 Uso de los Archivos .env

Las variables de ambiento o archivos .env, son usadas en el desarrollo de destinos tipos de aplicaciones, tales como proyectos de Vue, PHP, Lavarel, Python. Para el desarrollo de la plataforma Cosmox, desarrollado con Python y Django, se hizo uso de los archivos .env, por tanto, es necesario que cuando el módulo propuesto se integre a la plataforma se requiera hacer el menor cambio posible. Para ello el módulo debe utilizar de igual manera los archivos .env para

que el sistema a desarrollar sea altamente configurable y todo lo que se use en la base de datos sea fácilmente modificable.

Generalmente mientras se crea un simple script en Python para importar datos a la base de datos, se necesita definir los parámetros de conexión. Normalmente los parámetros son diferentes en la base de datos de desarrollo y en la de producción. Según la metodología Twelve-Factor App, esto es una mala práctica, porque al guardar la configuración directamente en el código fuente se puede estar publicando información sensible a ojos indiscretos. Lo recomendado es ponerlo en un archivo de configuración utilizando los archivos .env, los cuales se pueden encontrar una librería o paquete casi en cualquier lenguaje (GB, 2019).

Un archivo **.env** no es más que un archivo de texto donde se definen una serie de variables de entorno a las cuales les asignamos un valor y que se agrega en el directorio raíz de un proyecto (Castro, 2021). En cada línea, se define una variable de entorno y le asigna un valor mediante el operador =, por ejemplo:

PG_USER=postgres
PG_PASSWORD=mi clave super secreta
PG_DBNAME=la base de datos
PG_HOST=el servidor
API_KEY=la key super secreta

Es necesario que cuando se vaya a realizar la integración del módulo al portal se requiera hacer el menor cambio posible. Se determinó entonces hacer uso de los archivos .env en el desarrollo de la configuración del módulo de chat en vista de que el portal Cosmox hace uso ellos.

1.4 Metodología de desarrollo del software

Para dar cumplimiento a uno de los objetivos de esta investigación es imprescindible el uso de una metodología de desarrollo de software que controle cada uno de los procesos que lleva a cabo el equipo de trabajo.

Teniendo en cuenta que el equipo es pequeño, integrado en su totalidad por dos personas (el desarrollador y el cliente, quien participa constantemente con el equipo de desarrollo), los requerimientos están sujetos a cambios, el proyecto es pequeño y de corto plazo, se selecciona como metodología XP para guiar el proceso de desarrollo de la solución.

PROGRAMACIÓN EXTREMA (EXTREME PROGRAMMING, XP)

Se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Se aplica de manera dinámica durante el ciclo de vida del software. Es capaz de adaptarse a los cambios de requisitos. Los individuos e interacciones son más importantes que los procesos y herramientas. Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas. La gente es el principal factor de éxito de un proyecto software. Es más importante construir un buen equipo que construir el entorno. Muchas veces se comete el error de construir primero el entorno y esperar que el equipo se adapte automáticamente. Es mejor crear el equipo y que éste configure su propio entorno de desarrollo en base a sus necesidades. La metodología XP determina que un Software que funcione es más importante que documentación exhaustiva por tanto se enfocan en desarrollar un software que funciona más que conseguir una buena documentación. La regla a seguir es no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante. Estos documentos deben ser cortos y centrarse en lo fundamental (Dayana Bustamante, 2018).

Las Historias de Usuario

Las historias de usuario son la técnica utilizada en XP para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible, en cualquier momento historias de usuario pueden romperse, reemplazarse por otras más específicas o generales, añadirse nuevas o

ser modificadas. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

No hay que preocuparse si en un principio no se identifican todas las historias de usuario. Al comienzo de cada iteración estarán registrados los cambios en las historias de usuario y según eso se planificará la siguiente iteración. Las historias de usuario son descompuestas en tareas de programación y asignadas a los programadores para ser implementadas durante una iteración. (Patricio Letelier, 2006)

Las **fases de la metodología de programación extrema o XP** se basan en 4 pasos, estos son:

Planificación: va de acuerdo con las historias de usuario, estas se priorizan y descomponen en mini versiones, luego la planificación se va revisando cada dos semanas aproximadamente, después de las iteraciones, para obtener un software útil, funcional, listo para las pruebas y su lanzamiento. Los artefactos usados en esta metodología para la fase de planificación son el plan de iteraciones, plan de entregas e Historias de Usuarios

Diseño: se trabaja con un código sencillo, realizando lo mínimo necesario para que funcione, se obtiene el prototipo. Luego para el diseño del software si va orientado a objetos se generan tarjetas CRC (Clase-Responsabilidad Colaboración).

Codificación: se hace a dos manos, es decir en parejas frente al mismo ordenador, en algunos casos se intercambian las parejas, para asegurar que el código sea más universal, de forma que cualquier otro trabajador pueda trabajar en él y entenderlo. Debe parecer que fue hecho por una sola persona, para que se obtenga una programación organizada y planificada.

Pruebas: deben ser automáticas y continuas, esto es clave para proyectos a corto plazo. Incluso el mismo cliente puede hacer pruebas, proponer pruebas nuevas y validar las mini versiones. Los artefactos usados en esta metodología para la fase de prueba son las pruebas unitarias y de aceptación.

1.5 Herramientas y Lenguajes Informáticos

En todo proyecto, la selección de las herramientas y tecnologías es un punto sumamente importante. Estas definen la naturaleza de la solución y son la base de la calidad de la misma. A continuación, se describen las herramientas y lenguajes informáticos seleccionados para desarrollar la solución.

Para el desarrollo del sistema se va utilizar el lenguaje de programación y las herramientas usadas en el desarrollo del portal Cosmox.

Lenguaje de programación del lado del servidor Python 3.10

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses “Monty Python”. Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible. Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos.

Lenguaje interpretado o de script.

Un lenguaje interpretado o de script es aquel que se ejecuta utilizando un programa intermedio llamado intérprete, en lugar de compilar el código a lenguaje máquina que pueda comprender y ejecutar directamente una computadora (lenguajes compilados).

La ventaja de los lenguajes compilados es que su ejecución es más rápida. Sin embargo, los lenguajes interpretados son más flexibles y más portables. Python tiene, no obstante, muchas de las características de los lenguajes compilados, por lo que se podría decir que es semi interpretado. En Python, como en Java y muchos otros lenguajes, el código fuente se traduce a un pseudo código máquina intermedio llamado bytecode la primera vez que se ejecuta, generando archivos .pyc o .pyo (bytecode optimizado), que son los que se ejecutarán en sucesivas ocasiones (Duque, 2011).

Tipado dinámico

La característica de tipado dinámico se refiere a que no es necesario declarar el tipo de dato que va a contener una determinada variable, sino que su tipo se

determinará en tiempo de ejecución según el tipo del valor al que se asigne, y el tipo de esta variable puede cambiar si se le asigna un valor de otro tipo (Duque, 2011).

Fuertemente tipado

No se permite tratar a una variable como si fuera de un tipo distinto al que tiene, es necesario convertir de forma explícita dicha variable al nuevo tipo previamente. Por ejemplo, si tenemos una variable que contiene un texto (variable de tipo cadena o string) no podremos tratarla como un número (sumar la cadena “9” y el número 8). En otros lenguajes el tipo de la variable cambiaría para adaptarse al comportamiento esperado, aunque esto es más propenso a errores (Duque, 2011).

Multiplataforma

El intérprete de Python está disponible en multitud de plataformas (UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS, etc.) por lo que si no utilizamos librerías específicas de cada plataforma nuestro programa podrá correr en todos estos sistemas sin grandes cambios (Duque, 2011).

Orientado a objetos

La orientación a objetos es un paradigma de programación en el que los conceptos del mundo real relevantes para nuestro problema se trasladan a clases y objetos en nuestro programa. La ejecución del programa consiste en una serie de interacciones entre los objetos. Python también permite la programación imperativa, programación funcional y programación orientada a aspectos (Duque, 2011).

JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los

programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java (Pérez, 2008).

Lenguaje de programación del lado del cliente HTML5

HTML es un lenguaje muy sencillo que permite describir hipertexto, es decir, texto Definiéndolo de forma sencilla, "HTML es lo que se utiliza para crear todas las páginas web de Internet". Más concretamente, HTML es el lenguaje con el que se "escriben" la mayoría de páginas web. Los diseñadores utilizan el lenguaje HTML para crear sus páginas web, los programas que utilizan los diseñadores generan páginas escritas en HTML y los navegadores que utilizamos los usuarios muestran las páginas web después de leer su contenido HTML. Aunque HTML es un lenguaje que utilizan los ordenadores y los programas de diseño, es muy fácil de aprender y escribir por parte de las personas. En realidad, HTML son las siglas de HyperText Markup Language y más adelante se verá el significado de cada una de estas palabras (Pérez, 2008).

Hojas de estilo en cascada CSS3

SS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas.

Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien definidos y con significado completo (también llamados "documentos semánticos"). Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para marcar los contenidos, es decir, para designar la función de cada elemento dentro

de la página: párrafo, titular, texto destacado, tabla, lista de elementos, etc. Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, etc. (Pérez, 2008).

Herramienta de Modelado Visual Paradigm

Visual Paradigm Visual Paradigm es una herramienta CASE[1]. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación.

Visual Paradigm ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Constituye una herramienta privada disponible en varias ediciones, cada una destinada a satisfacer diferentes necesidades: Enterprise, Professional, Community, Standard, Modeler y Personal. Existe una alternativa libre y gratuita de este software, la versión Visual Paradigm UML 6.4 Community Edition (Community Edition, ya que existe la Enterprise, Professional, etc). Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de software de forma fiable a través de la utilización de un enfoque Orientado a Objetos (Pressman, 2002).

Sistema Gestor de Base de Datos PostgresSQL14

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) basado en Postgres, pionero de muchos conceptos que sólo estuvo disponible en algunos sistemas de bases de datos comerciales.

El Sistema Gestor de Bases de Datos Relacionales Orientadas a Objetos conocido como PostgreSQL (y brevemente llamado Postgres95) está derivado del paquete Postgres escrito en Berkeley. Con cerca de una década de desarrollo tras él,

[1] CASE: Ingeniería de Software Asistida por Computación

PostgreSQL es el gestor de bases de datos de código abierto más avanzado hoy en día, ofreciendo control de concurrencia multi-versión, soportando casi toda la sintaxis SQL (incluyendo subconsultas, transacciones, y tipos y funciones definidas por el usuario), contando también con un amplio conjunto de enlaces con lenguajes de programación (incluyendo C, C++, Java, perl, tcl y python). Postgres, desarrollada originalmente en el Departamento de Ciencias de la Computación de la Universidad de California en Berkeley, fue pionera en muchos de los conceptos de bases de datos relacionales orientadas a objetos que ahora empiezan a estar disponibles en algunas bases de datos comerciales. Ofrece suporte al lenguaje SQL: 2003, integridad de transacciones, y extensibilidad de tipos de datos. PostgreSQL es un descendiente de dominio público y código abierto del código original de Berkeley (Patricia, 2016).

Editor de código fuente independiente Visual Studio Code

Visual Studio Code (VS Code) es un editor de código fuente desarrollado por Microsoft. Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS. VS Code tiene una buena integración con Git, cuenta con soporte para depuración de código, y dispone de un sinnúmero de extensiones, que básicamente te da la posibilidad de escribir y ejecutar código en cualquier lenguaje de programación.

VS Code tiene una gran variedad de características útiles para agilizar el trabajo, que lo hacen el editor preferido por muchos (me incluyo) para trabajar los proyectos.

Multiplataforma: Es una característica importante en cualquier aplicación y más si trata de desarrollo. Visual Studio Code está disponible para Windows, GNU/Linux y macOS.

IntelliSense: Esta característica está relacionada con la edición de código, autocompletado y resaltado de sintaxis, lo que permite ser más ágil a la hora de escribir código. Como su nombre lo indica, proporciona sugerencias de código y terminaciones inteligentes en base a los tipos de variables, funciones, etc. Con la

ayuda de extensiones se puede personalizar y conseguir un IntelliSense más completo para cualquier lenguaje.

Depuración: Visual Studio Code incluye la función de depuración que ayuda a detectar errores en el código. De esta manera, nos evitamos tener que revisar línea por línea a puro ojo humano para encontrar errores. VS Code también es capaz de detectar pequeños errores de forma automática antes de ejecutar el código o la depuración como tal.

Uso del control de versiones: Visual Studio Code tiene compatibilidad con Git, por lo que puedes revisar diferencias o lo que conocemos con git diff, organizar archivos, realizar commits desde el editor, y hacer push y pull desde cualquier servicio de gestión de código fuente (SMC). Los demás SMC están disponible por medio de extensiones.

Extensiones: Hasta ahora, he mencionado varias veces el término *extensiones* porque es uno de los puntos fuertes. Visual Studio Code es un editor potente y en gran parte por las extensiones. Las extensiones nos permiten personalizar y agregar funcionalidad adicional de forma modular y aislada. Por ejemplo, para programar en diferentes lenguajes, agregar nuevos temas al editor, y conectar con otros servicios. Realmente las extensiones nos permiten tener una mejor experiencia, y lo más importante, no afectan en el rendimiento del editor, ya que se ejecutan en procesos independientes.

Django Rest Framework (Django4.1.1)

Para el desarrollo del portal Cosmos, el centro Vertex uso Django como framework de desarrollo.

Django es un framework web de alto nivel, escrito en Python, que ayuda al desarrollo rápido y a un diseño limpio y pragmático. Construido por desarrolladores experimentados, resuelve una buena parte de los problemas del desarrollo web de tal manera que uno se pueda enfocar en escribir su app sin necesidad de reinventar la rueda. Es gratis y de código abierto (Foundation, 2015).

Permite construir proyectos software bajo la arquitectura REST, incluye gran cantidad de código para reutilizar (Views, Resources, etc.) y una interfaz administrativa desde la cual es posible realizar pruebas sobre las operaciones HTTP como lo son: POST y GET. (Bernal, 2022)

Algunas de las características de Django son: (Bedu, 2020)

- Seguridad: Django tiene activados mecanismos incluidos para proteger tu base de datos, formularios y JavaScript.
- Escalabilidad: puedes utilizar el *framework* para un desarrollo sencillo, hasta uno mucho más complejo, ambos casos funcionarán de manera estable y con rapidez.
- Interfaz: Su interfaz para acceso a la base de datos y hacer consultas es sumamente buena.
- Portable: Al estar escrito en Python, se puede ejecutar en muchas plataformas como Windows, OS X, entre otras, dándole muchísima libertad al programador al momento de ejecutar las aplicaciones.

1.6 Conclusiones parciales del capítulo 1

Al finalizar el presente capitulo se arribó a las siguientes conclusiones:

- La investigación de los principales conceptos relacionados a la búsqueda de la solución del trabajo de investigación y el estudio de los portales similares permitió identificar las funcionalidades que formarían parte del chat de texto a desarrollar para el portal Cosmox con base en el funcionamiento de salas de chat.

- Se definió XP como metodología para guiar el proceso de desarrollo, pues es una perfecta metodología para equipos d trabajos pequeños de hasta una persona

- Se estableció como lenguajes de programación Python y JavaScript, para el diseño del sistema se usura HTML y CSS.

- Se hará uso de la tecnología WebSockets (implementado a través de Django Channels) para administrar la comunicación entre el cliente y el servidor.
- Además, se usarán las herramientas Visual Studio Code para llevar a cabo la implementación del sistema, DjangoRest Framework como framework para el desarrollo de la API, Visual Paradigm como herramienta CASE de modelado y PostgresSQL14 como Sistema Gestor de Base de Datos.
- También se determinó que el sistema deberá usar los estándares de los archivos .env

2 Capítulo2. Planificación y diseño de la solución propuesta al problema científico

Luego de haber realizado un estudio a las soluciones existentes y haber seleccionado las herramientas, metodología y lenguaje de programación que se utilizará en la presente investigación, en el presente capítulo se presenta la propuesta de solución y los artefactos obtenidos en las fases de Planificación y Diseño correspondientes a la metodología de desarrollo seleccionada (XP) la cual servirá de guía para tener mayor organización en la distribución del tiempo, actividades y artefactos a desarrollar, confeccionándose las HU que proporcionan un mayor entendimiento y comprensión del sistema.

2.1 Descripción de la propuesta de solución

Para darle respuesta la situación problemática antes descrita se propone desarrollar un módulo de chat para la comunicación sincrónica y asincrónica en el portal de videojuegos Cosmox a través de salas de chat**.** Este tiene como objetivo responder a la necesidad de establecer un mecanismo de comunicación para los jugadores de la plataforma mediante el uso de salas de chat. Según la información de las entrevistas realizadas se definió que se debe crear una sala de chat por cada juego, donde el usuario que se conecte vea o escriba mensajes, además de que la comunicación se pueda establecer de manera asincrónica.

Para un mejor entendimiento en el desarrollo del capítulo se describen los artefactos que permiten conceptualizar el funcionamiento del sistema.

Las actividades que se quieren automatizar son las acciones que se realizan para el envío de mensajes de forma sincrónica y asíncrona.

Chatear

El usuario podrá unirse a una sala de chat a través del botón unirse situados de bajo de los nombres de cada una de las salas listadas. Al unirse a una sala seleccionar el usuario podrá ver en pantalla una ventana de chat en la que podrá mantener una comunicación en tiempo real correspondiente a dicha sala la cual

estará compuesta por una sección de visualización que muestra los mensajes que se han enviado en el chat, la sección de entrada que permite que un usuario vea el mensaje que está creando actualmente y un botón de enviar para remitir el mensaje que se escribió, además de un botón Desconectarse para que el usuario abandone la sala donde se encuentra si lo desea, opción que lo reenviara a la lista de salas. En el caso de que un usuario desee visualizar las conversaciones realizadas en una sala de chat al unirse podrá ver los mensajes anteriores de la sala a la que se unió.

2.2 Fase de Planificación

La planificación es la fase donde se llega a un acuerdo entre el cliente y los programadores en las cuáles con las historias de usuario a ser implementada en cada iteración. Se define el orden y la prioridad de cada historia de usuario en un plan de iteraciones. Además, se realiza un plan de duración de iteración por semana para poder estimar el tiempo de desarrollo.

2.2.1 Historias de usuarios

Las HU son técnica utilizadas en XP para especificar los requisitos del software. Se trata de una tarjeta en la cual el cliente describe brevemente las características que el sistema debe poseer. El tratamiento de las HU es muy dinámico y flexible, en cualquier momento pueden reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas.

Fueron descritas por parte del cliente un total de 8 HU, en este epígrafe se muestra la HU3 confeccionada para el desarrollo del sistema. Las restantes se encuentran en el Anexo 7.1 del documento.

Tabla 2Historia de Usuario3

Historia de usuario	
Número: 3	**Nombre:** Enviar mensaje en sala de chat
Usuario: Usuario del Sistema	

Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Estimación: 10días	**Iteración asignada:** 2

Descripción: El sistema debe permitir que todos los usuarios podrán enviar mensajes en una sala de chat.

Objetivo: Permitir al usuario enviar mensajes en cualquier sala de chat en que se encuentre.

Acciones para lograr el objetivo (precondiciones y datos):

Para enviar mensajes a una sala de chat hay que:

- Estar autenticado en el sistema.

Flujo de la acción a realizar:

- Cuando el usuario se une a una sala de chat, este podrá escribir mensajes en la ventana de chat en un campo de tipo texto de dicha sala y selecciona la opción Enviar. Una vez seleccionada la opción Enviar el usuario podrá visualizar los mensajes que el resto de los usuarios.

Prototipo de Interfaz:

Tabla 3Historia de Usuario #1

Historia de usuario

Número: 1	**Nombre:** Ingresar a la sala chat
Usuario: Usuario del Sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Estimación: 10 días	**Iteración asignada:** 1

Descripción: El sistema debe permitir que los usuarios rol jugador podrá ingresar a la sala de chat existentes en el sistema

Objetivo: Permitir acceder a la sala de chat que seleccione del listado de salas de chat existentes

Acciones para lograr el objetivo (precondiciones y datos):

Para ingresar a una sala de chat hay que:

- Estar autenticado en el sistema.

Flujo de la acción a realizar:

- El sistema debe permitir seleccionar cualquier sala de chat del listado para que el usuario autenticado pueda entrar en esta
- Cuando el usuario decide la sala a la que desea entrar selecciona la opción Unirse que esta debajo del nombre de dicha sala que lo llevara a la ventana de chat perteneciente a la sala seleccionada.

Prototipo de Interfaz:

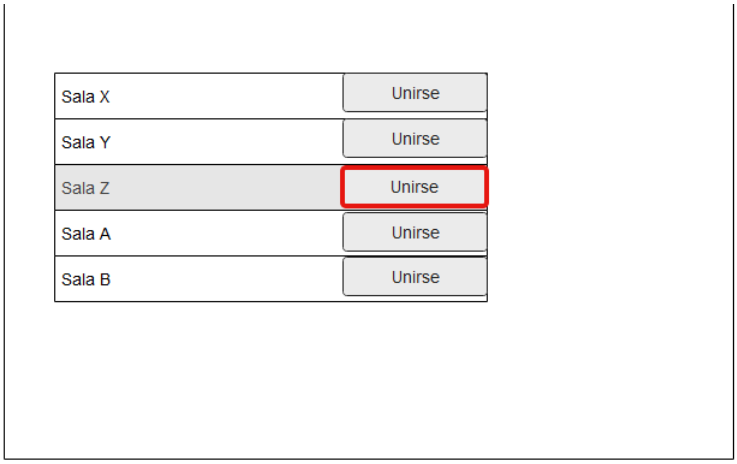


Tabla 4Historia de Usuario #2

Historia de usuario

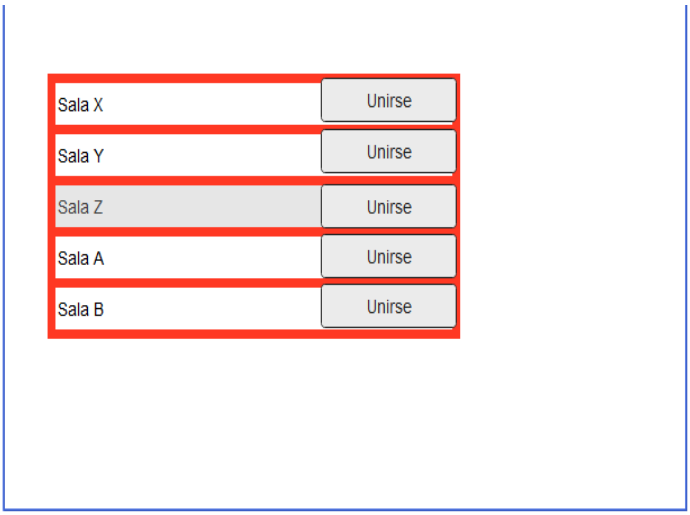
Número: 2	**Nombre:** Mostrar salas de chats.
Usuario: Usuario del Sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Estimación: 10 días	**Iteración asignada:** 1
Descripción: muestra al usuario un listado de las salas de chat existentes. **Descripción:** El sistema muestra al usuario un listado de las salas de chat existentes **Objetivo:** Listar las salas de chat existentes **Acciones para lograr el objetivo (precondiciones y datos):** Para el sistema liste las salas chat hay que: - Estar autenticado en el sistema. **Flujo de la acción a realizar:** - El sistema debe listar todas las salas de chat existentes en el sistema - Cuando el usuario se autentica el sistema le muestra el listado de todas las salas de chat existentes en el sistema.	
Prototipo de Interfaz:  	

Tabla 5Historia de Usuario #4

Historia de usuario	
Número: 4	**Nombre:** Mostrar historial de conversación
Usuario: Usuario del Sistema	

Prioridad en negocio: Alta	**Riesgo en desarrollo**: Alto
Estimación: 10 días	**Iteración asignada:** 2

Descripción: muestra al usuario todas las conversaciones después de su entrada en una sala de chat.

Objetivo: Permitir al usuario ver el historial de conversaciones de una sala de chat en la ventana de chat correspondiente a dicha sala

Acciones para lograr el objetivo (precondiciones y datos):

Para ver el historial de una sala de chat hay que:

- Estar autenticado en el sistema con el rol jefe de grupo

-Haberse unido a una sala de chat

Flujo de la acción a realizar:

- El sistema debe permitir al usuario ver el historial de conversaciones de una sala de chat determinada.

- Cuando el usuario se una sala de chat es enviada a una ventana de chat y este podrá ver en la sección de visualización de la ventana los mensajes que ya han sido enviados por los usuarios del programa de chat, o muestra el historial en blanco si ningún usuario ha accedido o escribida en esa sala

-Los mensajes aparecen en el orden en que son recibidos, con los mensajes anteriores en la parte superior de la vista.

Prototipo de Interfaz:

2.2.2 Plan de Iteraciones

En la metodología XP, la creación del sistema se divide en iteraciones. La duración ideal de una iteración está entre una y tres semanas. Para cada una de las iteraciones el cliente establece un conjunto de HU que serán implementadas en cada iteración del sistema. Al final de cada iteración se realizan las pruebas de aceptación y la aplicación tendrá implementadas funcionalidades para dar cumplimiento a los objetivos propuestos

Tabla 6Plan de Iteraciones

Iteración	Historias de Usuario	Duración total de las iteraciones
1	Iniciar sesión	5 semanas
	Cerrar sesión	
	Registrar usuario en el sistema	
	Mostrar lista de salas de chat	
2	Ingresar a sala de chat	5 semanas
	Enviar mensaje en la sala de chat	
	Mostrar historial de conversación	
	Abandonar sala de chat	

2.2.3 Plan de entregas

En el plan de entregas se realiza un cronograma de entregas donde el cliente establece la prioridad de cada HU, cuáles serán agrupados para conformar una entrega y el orden de las mismas. En correspondencia, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas**.**

Tabla 7Plan de Entregas

Entregable	1ra Iteración	2da Iteración
Versión	Versión 1.0	Versión 1.2
Fecha	(24/8/22-28/9/22)	(29/8/22-4/11/22)

2.3 Fase de Diseño

La Metodología XP hace especial énfasis en los diseños simples y claros (MELÉNDEZ, 2016). El proceso de diseño establece diseños simples y sencillos pero que sea funcional para poder cumplir con el tiempo de entrega y facilitar el desarrollo en cada una de las etapas del proyecto, para lo cual se debe elaborar un glosario de términos, optimización de código, métodos y clases organizadas correctamente con el fin de que se pueda modificar cualquier estructura del código con mayor facilidad (CÁRDENAS, 2017).

2.3.1 Tarjetas CRC

Las Tarjetas CRC (Clase-Responsabilidades-Colaboradores), permiten conocer que clases componen el sistema y cuales interactúan entre sí. Se dividen en tres secciones: Nombre de la Clase, Responsabilidades y Colaboradores.

Tabla 8Tarjeta CRC #1

Tarjetas CRC	
Nombre de la clase: chat.view	
Responsabilidades: Proveer al sistema los métodos correspondientes. def room() def rooms()	**Colaboradores:** • chat.models • chat.urls • core.urls

Fueron descritas un total de 4 tarjetas CRC, en este epígrafe se muestra 1. Las restantes se encuentran en el Anexo 7.2 de la investiga

2.4 Patrón arquitectónico

Los patrones arquitectónicos se abocan a un problema de aplicación específica dentro de un contexto dado y sujeto a limitaciones y restricciones. El patrón propone una solución arquitectónica que sirve como base para el diseño de la arquitectura (Pressman, 2010).

Django se basa en la arquitectura **MVT(Model-View-Template)**. MVT es un patrón de diseño de software para desarrollar una aplicación web.

La estructura MVT tiene las siguientes tres partes:

Modelo: el modelo actuará como la interfaz de sus datos. Es responsable de mantener los datos. Es la estructura de datos lógica detrás de toda la aplicación y está representada por una base de datos (generalmente bases de datos relacionales como MySql, Postgres).

Vista: La vista es la interfaz de usuario, lo que ve en su navegador cuando representa un sitio web. Está representado por archivos HTML / CSS / Javascript

Plantilla: una plantilla consta de partes estáticas de la salida HTML deseada, así como una sintaxis especial que describe cómo se insertará el contenido dinámico. (acervolima.com, 2022)

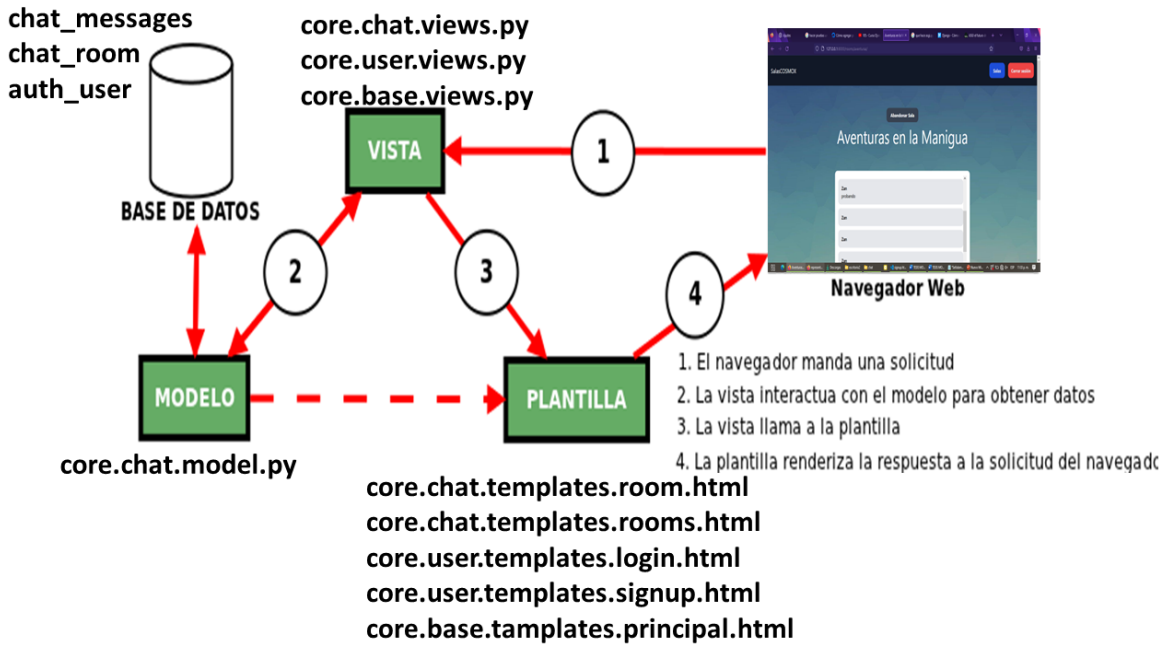


Ilustración 5Modelo Vista Plantilla del Módulo de Chat

2.5 Patrones de Diseño:

Un patrón de diseño puede ser caracterizado como una regla de tres partes, la cual expresa una relación entre cierto contexto, un problema y una solución. Para el diseño de software, el contexto permite a quien usa el patrón entender el entorno en el cual reside el problema y qué solución pudiera ser apropiada dentro de ese entorno. Un conjunto de requisitos, incluyendo limitaciones y restricciones, actúan como un sistema de fuerzas que influencia en cómo el problema puede ser interpretado dentro de su contexto y cómo la solución puede ser aplicada efectivamente (Pressman, 2010).

Patrones GRASP

Un patrón de diseño se caracteriza como “una regla de tres partes que expresa una relación entre cierto contexto, un problema y una solución”. Para el diseño de software, el con texto permite al lector entender el ambiente en el que reside el problema y qué solución sería apropiada en dicho ambiente. Un conjunto de requerimientos, incluidas limitaciones y restricciones, actúan como sistema de fuerzas que influyen en la manera en la que puede interpretarse el problema en

este contexto y en cómo podría aplicarse con eficacia la solución (Pressman, 2010).

Patrón Controlador: Está representado por una clase a la cual se le asigna la responsabilidad de las operaciones del sistema, ofrece también una guía para tomar decisiones apropiadas.

Por ejemplo, este patrón se refleja en la clase **chat.views.py** y **user.views.py** que es la encargada de controlar las acciones que realiza el usuario con la interfaz de la aplicación y dar respuesta a las peticiones realizadas.

Patrón Experto: Este patrón define que la responsabilidad de ciertas labores solo debe ser realizadas por las clases que contengan la información necesaria.
Por ejemplo, en la función rooms en la vista de chat.

```
#Vista donde está la plantilla room.html que se definió
@login_required
def room(request, slug):
  room = Room.objects.get(slug=slug)
  messages = Message.objects.filter(room=room)
  return render(request, 'room.html', {'room': room, 'messages': messages})
```

Patrón Alta Cohesión: Se aplica en la mayoría las clases del diseño, ya que en cada una solo se implementan las funcionalidades que le corresponden. El patrón alta cohesión se puede evidenciar en las plantillas encargadas de mostrar los datos al usuario y en las clases manejadoras de los formularios.

Por ejemplo, como se puede observar a continuación:

```
class SignUpForm(UserCreationForm):
  class Meta:
  model = User
  fields = ['username', 'password1', 'password2']
```

con la clase SignUpForm, se maneja el formulario para registrar usuarios en el sistema.

```
<div class="mb-5">
```

```
  <label class="text-white">Nombre de usuario:</label>
  <input type="text" name="username" class="w-full mt-2 px-4 py-2 rounded-
xl">
  </div>

  <div class="mb-5">
  <label class="text-white">Contraseña:</label>
  <input type="password" name="password1" class="w-full mt-2 px-4 py-2
rounded-xl">
  </div>

  <div class="mb-5">
  <label class="text-white">Repetir contraseña:</label>
  <input type="password" name="password2" class="w-full mt-2 px-4 py-2
rounded-xl">
  </div>
```

Aquí se puede ver este fragmento del html donde se forma el formulario para registrar los usuarios al sistema.

Patrones GOF

Patrones Gof. Los patrones GoF se descubren como una forma indispensable de enfrentarse a la programación a raíz del libro “Design Patterns—Elements of Reusable Software” de Erich Gamma, Richard Helm, Ralph Jonson y John Vlissides, a partir de entonces estos patrones son conocidos como los patrones de la pandilla de los cuatro (GoF, gang of four) (Pérez Mariñán, 2003).

Front controller:

Front controller es el primer punto de entrada para la solicitud. Maneja el envío general al controlador y se ocupa de las necesidades comunes tales como, autenticación, gestión de sesiones, seguridad, redirección. Contiene la lógica que todas las solicitudes deben considerar, reduciendo la duplicación de código. También agrega datos al contexto de ejecución.

Por ejemplo, en la siguiente función:

```
def signup(request):
  if request.method == 'POST':
  form = SignUpForm(request.POST)

  if form.is_valid():
    user = form.save()
    login(request, user)
    return redirect('rooms')
  else:
  form = SignUpForm()

  return render (request, 'signup.html', {'form': form})
```

se puede observar la autenticación, gestión de sesiones, seguridad, redirección que define el patrón.

```
if form.is_valid():
```

Comprueba la validez de los datos que se introducen en el formulario.

```
    return redirect('rooms')
```

Aquí si los valores se son válidos, redirecciona al usuario a la lista de salas

```
  return render (request, 'signup.html', {'form': form})
```

Aquí si los valores no son válidos, redirecciona al usuario a un formulario vacío para que vuelva a introducir los datos que se le piden

Patrón Decorator:

Añade responsabilidades adicionales a un objeto dinámicamente, proporcionando una alternativa flexible a la especialización mediante herencia, cuando se trata de añadir funcionalidades.

Por ejemplo, en la siguiente función:

```
@login_required
def rooms(request):
```

```
  rooms = Room.objects.all()
  return render(request, 'rooms.html', {'rooms': rooms})
```

@login_required, decorador de la función, indica que solo pueden ver la lista de salas los usuarios que estén autenticados en el sistema.

Patrón Madiator:

Encargado de manejar la interacción entre los diferentes subsistemas. Realiza el mapeo objeto–relacional (ORM) a cargo del motor de Django. Simplifica y facilita los resultados de las consultas realizadas a la base de datos, así como brinda varias funcionalidades adicionales, que consultas SQL tradicionales no proveen.

Por ejemplo, como se ve a continuación:

```
core.chat.views.py
#Vista donde está la plantilla room.html que se definió
@login_required
def room(request, slug):
  room = Room.objects.get(slug=slug)
  messages = Message.objects.filter(room=room)
  return render(request, 'room.html', {'room': room, 'messages': messages})

core.chat.consumers.py
#Sección del código del consumers.py que guarda los mensajes
  @sync_to_async
  def save_message(self, username, room, message):
  user = User.objects.get(username=username)
  room = Room.objects.get(slug=room)
  Message.objects.create(user=user, room=room, content=message)

core.chat.admin.py
#Usa el modelo Room para construir automáticamente un área dentro del sitio
que el administrador puede usar para crear, consultar, actualizar y borrar
salas

admin.site.register(Room)
```

Los componentes se comunican con un mediador utilizando la interfaz mediadora. Gracias a ello, se puede utilizar los mismos componentes en otros contextos vinculándolos con diferentes objetos mediadores.

Aquí se puede ver en el uso de los modelos Meesages, Room y User que guardan los datos los mensajes, salas y usuarios, y son utilizados en distintos componentes.

2.6 Conclusiones del capítulo 2

Durante el desarrollo del capítulo se dejó plasmado los artefactos generados por las fases planificación y diseño de la metodología XP.

- La caracterización de la propuesta de solución permitió alcanzar un mayor conocimiento del funcionamiento y aspectos claves de la herramienta a desarrollar.
- La definición de los requisitos funcionales y la declaración de las historias de usuarios contribuyeron a la identificación de las necesidades del cliente, dando lugar a un total de 8 historias de usuario.
- La realización de los planes de iteración y entrega permitieron planificar y organizar los procesos a ejecutarse en un plazo de 3 meses.
- La definición de los patrones de diseño, el patrón arquitectónico y las tarjetas CRC sentaron las bases para el desarrollo de un sistema sencillo y de fácil implementación.

3 Capítulo3. Implementación y prueba del módulo.

En el presente capítulo se detallan las iteraciones realizadas durante la etapa de construcción del módulo propuesto, además se exponen las tareas de ingeniería generadas para cada HU que fueron definidas, así como las pruebas de aceptación planificadas para el sistema. De esta forma es obtenido un producto funcional probado y listo para entregar al cliente al final de cada iteración como propone XP

3.1 Fase de Codificación

En esta fase, XP plantea que las HU seleccionadas para ser implementadas se realizan durante el transcurso de la iteración a la que pertenecen. Por estas razones, se lleva a cabo una revisión del plan de iteraciones y se modifican en caso de ser necesario. Como parte de este plan se descomponen las HU en tareas de ingeniería.

3.2 Descripción de la Solución

Para el desarrollo de módulo propuesto se tuvieron en cuenta las siguientes características con el objetivo de lograr la funcionalidad principal del sistema que es el envío de mensajes.

- **Integración completa con Django y Websockets**, por medio de Channels.
- **Respuestas asíncronas** con el objetivo de mejorar el rendimiento.
- **Salas grupales** donde todos reciben los mensajes nuevos en tiempo real.

Por consiguiente, se describen como se realizó el chat:

Para el enviar/recibir por Webscokets se monta una puerta de entrada para que el cliente de Websocket pueda conectarse y esperar nuevo contenido, para ser alimentado con información asíncrona hasta que se sacie o se desconecte. A este

concepto se le denomina Consumidor (Consumer). Para aplicarlo dentro de Django se creó un archivo llamado consumers.py con el cual se realizaron las siguientes acciones: recoger el nombre de la sala, salir del grupo de la sala, enviar el mensaje a la sala, enviar mensaje al websocket y guardar los mensajes en las salas.

Lo siguiente que se realizó fue definir las rutas. Estas permiten combinar y apilar consumidores. Para esto se creó un archivo routing.py en la aplicación de chat.

routing.py

```
from django.urls import path
from . import consumers

websocket_urlpatterns = [
  path('ws/<str:room_name>/', consumers.ChatConsumer.as_asgi()),
]
```

Como Django no permite trabajar con aplicaciones asincrónicas con su servidor WSGI, se debe dar el salto a ASGI (Asynchronous Server Gateway Interface) para el desarrollo del módulo, servidor creado con la finalidad de otorgar una interfaz estándar entre servidores web, marcos y aplicaciones Python con capacidad asíncrona. Se creó entonces el archivo asgi.py para el trabajo con servidor ASGI para otorgarle al módulo de chat la capacidad de establecer comunicación asincrónica.

En settings.py (archivo de configuración de Django que contiene toda la configuración de la instalación Django) se añade la configuración para que Django cambie a ASGI de la siguiente manera:

settings.py

```
WSGI_APPLICATION = 'core.wsgi.application'
ASGI_APPLICATION = 'core.asgi.application'
```

A continuación, se creó una interfaz web para poder enviar y recibir todos los mensajes que se envíen sin limitación de usuarios, a partir de usar como base el

sistema de plantillas que incorpora Django, donde se incluyó todo el HTML y JavaScript necesario.

Dentro de la vista chat.views.py se indicó la función donde está la plantilla definida para la ventana de chat donde los usuarios participarán. Posteriormente se define entonces la ruta donde será mostrada la vista chat.urls.py.

3.2.1 Prototipos finales del chat desarrollado.

Ilustración 6Listado de salas de chat

Ilustración 7Ventana de chat.

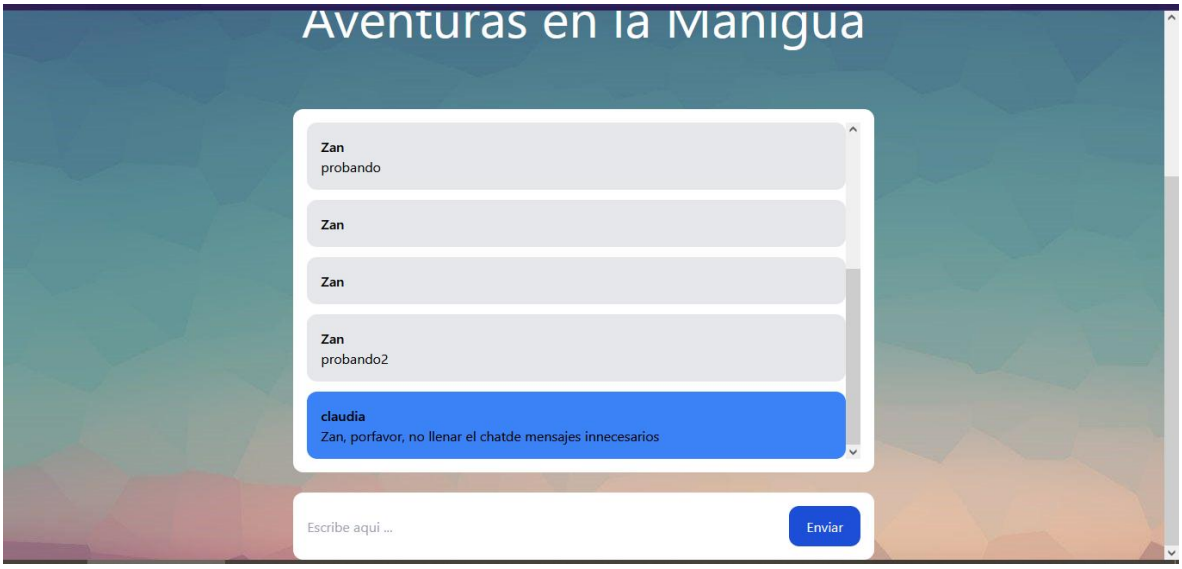


Ilustración 8Ventana de chat

3.3 Estándares de codificación

Los estándares de código, son parte de las llamadas buenas prácticas o mejores prácticas, estas son un conjunto no formal de reglas, que han ido surgiendo en las distintas comunidades de desarrolladores con el paso del tiempo y las cuales, bien aplicadas pueden incrementar la calidad del código, notablemente.

El objetivo de los estándares de codificación de software es inculcar prácticas de programación probadas que conduzcan a un código seguro, confiable, comprobable y mantenible. Por lo general, esto significa evitar las prácticas de codificación inseguras conocidas o el código que puede causar un comportamiento impredecible (Hiken, 2020).

Diseño de Código (programmerclick.com, 2022):

- Las funciones o clases deben estar separadas por 2 líneas en blanco y se deben usar diferentes funciones dentro de la clase**1 línea en blanco** Separa

```
from django.db import models
from django.contrib.auth.models import User

class Room(models.Model):
  name = models.CharField(max_length=255)
```

```
    slug = models.SlugField(unique=True)


class Message(models.Model ):
    room      =      models.ForeignKey(Room,      related_name='messages',
on_delete=models.CASCADE)
    user      =      models.ForeignKey(User,      related_name='messages',
on_delete=models.CASCADE)
    content = models.TextField()
    date_added = models.DateTimeField(auto_now_add=True)
    image=models.ImageField( blank=True)
```

- Cada línea de código se controla en **80.** Dentro de los caracteres, si es demasiado largo, use barras, corchetes, corchetes o llaves para doblar.

```
10      room = models.ForeignKey(Room, related_name='messages', on_delete=models.CASCADE)
11      user = models.ForeignKey(User, related_name='messages', on_delete=models.CASCADE)
```

Ilustración 9Ejemplo de Estándares de Codificación #2

Convención de nomenclatura (programmerclick.com, 2022):

- Las variables, funciones, paquetes y módulos llevan el nombre de**lower_with_under**(Minúsculas y subrayado)

```
def rooms(request):
```

- Las clases y excepciones llevan el nombre de**CapWords**(Caso camel, primera letra en mayúscula) formulario.

```
class Room(models.Model):
```

Estilo de Importación (programmerclick.com, 2022):

- El módulo importado debe estar en la parte superior del archivo, después del comentario del módulo y la cadena de documentos, y antes de la variable o constante global. El orden de importación debe ordenarse de acuerdo con la generalidad del paquete de la biblioteca, como se muestra a continuación:
  1. Importación de biblioteca estándar
  2. Importación de biblioteca de terceros

3. Importación de la biblioteca de aplicaciones locales

```
from django.db import models

from django.contrib.auth.models import User
```

Comentario (programmerclick.com, 2022):

- Comentario de línea Agregue un comentario después de una oración de código, pero si el significado del código es obvio, no se requieren comentarios en línea.

```
#recoge el nombre de la sala

 self.room_name = self.scope['url_route']['kwargs']['room_name']

 self.room_group_name = 'chat_%s' % self.room_name
```

Comillas (programmerclick.com, 2022):

En pocas palabras, el lenguaje natural usa comillas dobles y las etiquetas de máquina usan comillas simples, por lo que la mayor parte del código debe usar comillas simples

- El lenguaje natural usa comillas dobles"..."
- ID de máquina Utilice comillas simples'...' Por ejemplo, la clave en el dict
- La cadena de documentos usa tres comillas dobles"""....."""

```
'''Cliente se conecta'''



 #recoge el nombre de la sala

 self.room_name = self.scope['url_route']['kwargs']['room_name']

 self.room_group_name = 'chat_%s' % self.room_name
```

3.4 Tareas de la ingeniería

Una Historias de Usuario se descompone en varias tareas de ingeniería, las cuales describen las actividades que se realizarán en cada historia de usuario, así

mismo las tareas de ingeniería se vinculan más al desarrollador, ya que permite tener un acercamiento con el código.

3.4.1 Tareas de la ingeniería para la Iteración I

Tabla 9Tarea de la Ingeniería # 1

Tarea de Ingeniería 1	
Número de HU: 5	
Nombre: Iniciar sesión	
Tipo de tarea: Desarrollo	**Estimación:** 10 días
Fecha inicio: (24/8/22).	**Fecha de fin:** (2/9/22).
Programador responsable: Claudia de la Caridad Mercadet Vasallo	
Descripción: Crear plantilla y vista para autenticar usuarios. Manejar mensajes en caso satisfactorio o de autenticación fallida.	

3.4.2 Tareas de la ingeniería para la Iteración II

Tabla 10Tarea de la Ingeniería # 5

Tarea de Ingeniería5	
Número de HU: 1	
Nombre: Ingresar a sala de chat	
Tipo de tarea: Desarrollo	**Estimación:** 10 días
Fecha inicio: (29/9/22).	**Fecha de fin:** (8/10/22).
Programador responsable: Claudia de la Caridad Mercadet Vasallo	
Descripción: La función recibe como parámetros un nombre y slug de la sala seleccionada y redirecciona al usuario a la vista room.html	

Las 8 HU fueron desglosadas en un total de 8 tareas de ingeniería, en este epígrafe solo se muestran 1 por iteración, las restantes se encuentran en el Anexo 7.3 del documento.

3.5 Fase de Prueba

Uno de los pilares de XP es el proceso de pruebas, la cual anima a probar constantemente tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones

Los principales aspectos a ser evaluados en un producto software son la Fiabilidad (resistente a fallos) la Funcionalidad (hace lo que debe) y el Rendimiento (lleva a cabo su trabajo de manera efectiva). Las pruebas pueden hacerse a diferentes niveles dependiendo del objetivo de los mismos: pruebas de unidad (se prueban las unidades mínimas por separado, y normalmente se hacen durante la implementación misma), de integración (varias unidades juntas), de sistema (sobre la aplicaci6n o sistema completo) y de aceptaci6n (realizado sobre el sistema global por las usuarios o terceros). Cada Una de ellas se realizan en determinados momentos del ciclo de Vida del software. Las técnicas y herramientas para encontrar problemas en un programa son extensamente variadas (Fernández, Cúe, y otros, 2007).

3.5.1 Estrategia de prueba

Una estrategia de prueba de software proporciona una guía que describe los pasos que deben realizarse como parte de la prueba, cuándo se planean y se llevan a cabo dichos pasos, y cuánto esfuerzo, tiempo y recursos se requerirán. Por tanto, cualquier estrategia de prueba debe incorporar la planificación de la prueba, el diseño de casos de prueba, la ejecución de la prueba y la recolección y evaluación de los resultados (Pressman, 2010).

Una estrategia de prueba de software debe ser suficientemente flexible para promover un uso personalizado de la prueba. Al mismo tiempo, debe ser suficientemente rígida para alentar la planificación razonable y el seguimiento de la gestión conforme avanza el proyecto (Pressman, 2010).

Teniendo presente los niveles de prueba unidad, sistema y aceptación se definen diferentes tipos de pruebas a realizar, en el que se va a tener en cuenta las dos iteraciones definidas en el proyecto. En el primer nivel de pruebas, por cada uno de los métodos que se creen en la aplicación que responden a los RF se les desarrollará pruebas unitarias para comprobar que el mismo funciona de forma correcta y las salidas de los métodos son las esperadas. Como una característica de XP, Estas pruebas son definidas antes de realizar el código. En caso de que ocurran no conformidades las no conformidades pendientes en esta iteración pasan a la siguiente para su posterior reevaluación a través pruebas de confirmación y regresión para verificar que las no conformidades fueron solucionadas y que las modificaciones realizadas no afecten a la solución. Una vez que ya, el sistema se encuentre desarrollado se va a realizar pruebas funcionales, para el desarrollo de las mismas se utilizará el método de caja negra utilizando la técnica de partición de equivalencia. Finalmente, para verificar que se cumple con los requisitos funcionales se realizan pruebas de aceptación.

3.5.2 Pruebas Unitarias

Todos los módulos deben de pasar las pruebas unitarias antes de ser liberados o publicados. Por otra parte, como se mencionó anteriormente, las pruebas deben ser definidas antes de realizar el código (“Test-Driven Programmming”). Que todo código liberado pase correctamente las pruebas unitarias, es lo que habilita que funcione la propiedad colectiva del código (MELÉNDEZ, 2016).

Las Pruebas Unitarias o Unit Testing son Se trata de un método para determinar si un módulo o un conjunto de módulos de código funciona correctamente. El concepto de Unit testing no se limita a ningún lenguaje específico, sino que es una herramienta de la programación en general. Las pruebas unitarias se implementan

a la par con el desarrollo de un módulo o proyecto, y se ejecutan cuando este último sufre modificaciones para garantizar su funcionamiento. Si bien el código mismo de la prueba unitaria puede contener errores, la clave está en la separación del código de un módulo de su respectiva prueba unitaria, de modo que puedan correr independientemente (Python, 2016).

En otras palabras, es una forma de comprobar que un conjunto de funciones o clases (tantas como queramos) funcionan como esperamos. Lógicamente, las pruebas unitarias nunca pueden garantizar completamente el correcto funcionamiento de una porción de código. No obstante, ello, serán capaces de detectar gran cantidad de anomalías y de ahorrarnos tiempo de depuración (Python, 2016).

Pruebas unitarias en Python: Unittest

La infraestructura de tests unitarios unittest se inspiró en primera instancia en JUnit y ofrece aspectos similares a las principales estructuras de tests unitarios más importantes de otros lenguajes. Da soporte a automatización de tests, inicialización compartida, código de cierre de los tests, agregación de los tests en colecciones e independencia de los tests de la infraestructura que los reporta (Foundation, 2022).

Para conseguir esto, unittest da soporte a ciertos conceptos importantes de una forma orientada a objetos (Foundation, 2022) tal es el caso de la aplicación de **caso de prueba o Test Case.** Un test case es la unidad mínima de prueba. Verifica la respuesta específica a un juego particular de entradas. Unittest proporciona una clase base, TestCase, que se puede utilizar para crear nuevos casos de uso.

Para la realización de las pruebas unitarias del módulo de chat se utilizó las pruebas unitarias de caso de prueba o Test Case. A continuación, se muestra un ejemplo de una de las pruebas realizadas en la Iteración 1.

```
from unittest import TestCase
```

```
from django.contrib.auth import get_user_model
# Create your tests here.
User= get_user_model()
class UserTestCase(TestCase):
  def setUp(self):
  self.user_a= User.objects.create(username='jorgitos',
password='jor*123jortos')
  self.user_b= User.objects.create(username='mariachan',
password='jor*123jorchan')
  self.user_c= User.objects.create(username='zurynam',
password='jor*123jorzan')


  def test_user_count(self):
  qs = User.objects.all()
  self.assertEqual(qs.count(),16)
```

En esta prueba se confirma que se registren usuario de manera correcta. Si se introducen datos no validos la prueba arroja un cartel d error especificando los campos llenados con datos erróneos. Si se introducen datos validos la prueba deberá mostrar en un mensaje ´´OK´´ en representación de que fue exitosa. Si la prueba realizada manda un mensaje distinto a una prueba exitosa o especificando que datos no fueron válidos, representa que la prueba fue mal realizada o hay problemas en la implementación de la función que se confirma su efectividad fue mal implementada.

```
TERMINAL  JUPYTER  DEBUG CONSOLE  PROBLEMS 27  OUTPUT
(salaenv) PS C:\Users\Claudia\Desktop\SalasCosmox\core> python manage.py test user
Found 1 test(s).
System check identified no issues (0 silenced).
.
----------------------------------------------------------------------
Ran 1 test in 0.589s

OK
(salaenv) PS C:\Users\Claudia\Desktop\SalasCosmox\core>
```

Ilustración 10Resultado de Prueba unitaria Registrar usuario

Las pruebas unitarias fueron realizadas en dos iteraciones. A continuación, se muestran resultados obtenidos en cada iteración y la corrección de errores realizados.

Tabla 11Tabla de No conformidades de las Pruebas Unitarias

No conformidades	**Iteración1**	**Iteración2**
Detectadas	0	1
Pendientes	0	0
Resueltas	0	1

3.5.3 Pruebas Funcionales

A través del método de caja negra se examinó si la herramienta funciona y cumple con su objetivo de satisfacer al cliente. Partición de equivalencia fue la técnica empleada para realizar esta prueba con el objetivo de detectar errores en las funcionalidades y validar los requisitos de la propuesta solución. A continuación, la tabla 12 muestra el caso de prueba “Enviar mensaje”.

Tabla 12Prueba funcional de Caso de Prueba #6

Escenario	**Descripción**	**Requisito**	**Respuesta del sistema**	**Resultado de la prueba**
EC 1.1 Enviar mensaje	Toma el texto escrito en la sección de entrada de la ventana de chat y lo muestra en la sección de visualización al presionar el botón “Enviar”	Válido El sistema toma el mensaje introducido, lo muestra y lo guarda en la base de datos	Se muestra un nuevo mensaje en la sección de visualización de la ventana de chat	Satisfactoria

EC 1.2 Campos vacíos	Presiona el botón “Enviar’’ estando la sección de entrada de mensajes vacío	Inválido	El sistema muestra un mensaje “El mensaje está vacío”	Satisfactorio a el sistema alerto al usuario del campo vacío

Con el fin de comprobar que las funcionalidades del módulo se realizaron de manera correcta y responden a las necesidades del cliente, se realizaron pruebas funcionales a lo lago de las dos iteraciones definidas en la estrategia de prueba. En la primera se detectó una No Conformidades de tipo de Interfaz. Al finalizar la iteración la NC encontrada fue resuelta satisfactoriamente.

En la segunda iteración se encontró una NC de tipo Funcional. Al finalizar la iteración la NC encontrada fue resuelta satisfactoriamente.

La siguiente tabla muestra los resultados de estas pruebas, teniendo en cuenta las NC encontradas.

Tabla 13No Conformidades de las Pruebas Funcionales

No	No Conformidades	Tipo	Estado
1.	No muestra la lista de salas de chat	Interfaz	Resuelto
2.	No se ingresa a la sala de chat seleccionada.	Funcional	Resuelto

3.5.4 Pruebas de Aceptación

Son creadas en base a las historias de usuarios, en cada ciclo de la iteración del desarrollo. El Cliente debe especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada. Asimismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad de

resolución. Una historia de usuario no se puede considerar terminada hasta que pase correctamente todas las pruebas de aceptación (MELÉNDEZ, 2016).

Pruebas de Aceptación para la Iteración 1

Tabla 14Caso de Prueba #1

Caso de Prueba	
Numero: 1	**Historia de usuario:** 5
Nombre: Iniciar sesión	
Condiciones de ejecución: el sistema debe haber iniciado sin problemas	
Entrada/Pasos de ejecución: una vez el usuario entra en la página por primera vez esta le proporcionará la opción de crease una cuenta en el sistema y posteriormente le facilitará la opción de autenticarse para ingresar en el sistema	
Resultado esperado: se muestra una página donde el usuario se autenticará o registrara en el sistema si nunca ha entrado a este.	
Evaluación de la Prueba: Satisfactoria	

Pruebas de Aceptación para la Iteración 2

Tabla 15Caso de Prueba #5

Caso de Prueba	
Numero: 6	**Historia de usuario:** 1
Nombre: Ingresar a sala de chat	
Condiciones de ejecución: Tiene que existir un usuario registrado en el sistema	
Entrada/Pasos de ejecución: una vez el usuario este en el sistema podrá a través del botón “Unirse” ingresar en la sala de chat que desee.	
Resultado esperado: se ingresa a una sala.	
Evaluación de la Prueba: Satisfactoria	

Fueron elaboradas un total 8 casos de prueba, para realizar las pruebas de aceptación. En el presente epígrafe se muestran 1 casos de prueba por Iteración, las restantes se encuentran en el Anexo 7.4 del documento.

Análisis de los resultados de la prueba aceptación.

Se realizaron las pruebas al sistema a partir de los siguientes datos:

- Fueron definidas 8 Historias de Usuario
- Se elaboraron 8 Tareas de Programación
- Se diseñaron 8 Casos de prueba.

En una primera iteración se realizaron 4 pruebas de aceptación, en la que no salieron no conformidades, por lo que las pruebas en el la iteración 1 resultaron satisfactorias. No se realizaron peticiones de cambio.

En la segunda Iteración se realizaron 4 pruebas de aceptación, en la que se encontró una no conformidad en el caso de prueba 8, el cual fue resuelto con rapidez, y después de una prueba de regresión salió con resultados satisfactorios. Durante esta entrega el cliente pudo realizar pruebas a un sistema más completo. No se realizaron peticiones de cambio y se solucionó la no conformidad encontrada en el tiempo requerido.

Al final del proceso de ejecución de las pruebas de aceptación arrojó que las 8 HU implementadas cumplen con los criterios de aceptación del cliente.

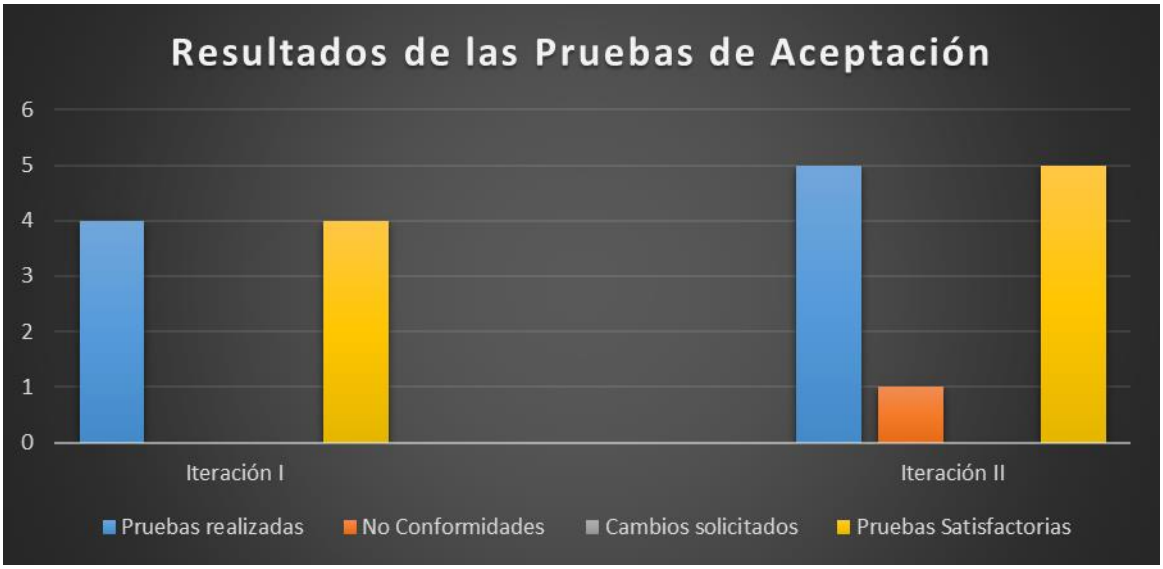


Ilustración 11Análisis de los resultados de las Pruebas de Aceptación

3.6 Conclusiones parciales.

Durante el desarrollo del capítulo se dejó plasmado los artefactos generados por las fases codificación y prueba de la metodología XP.

- Se especificó el proceso de implementación del sistema a partir del desglose de las HU en tareas de ingeniería, lo que permitió especificar los procedimientos necesarios para dar cumplimiento a cada una de ellas.

- Además, se definieron y aplicaron pruebas unitarias, funcionales y pruebas de aceptación a las HU y las funcionalidades del mismo respectivamente. Estas pruebas permitieron detectar fallas en el sistema y corregirla, lo que posibilitó mejorar la operabilidad del mismo.

4 Conclusiones Generales

Mediante la investigación realizada y el desarrollo del Módulo de chat para el portal de videojuegos Cosmox, en el presente trabajo de diploma se cumplen los objetivos trazados, destacando los siguientes aspectos:

- Con la elaboración del marco teórico del sistema se estableció que ninguno de los sistemas homólogos estudiados responde al objetivo de la investigación por lo que se evidencia la necesidad de la realización del módulo.
- La metodología software XP permitió guiar el proceso de desarrollo de software generando artefactos que permitieron comprender el diseño del módulo.
- Se desarrollo un módulo de chat de texto que permitirá la comunicación sincrónica y asincrónica de los jugadores de al portal Cosmox.
- La ejecución de las pruebas de unitarias y de aceptación permitió comprobar el correcto funcionamiento de la propuesta evidenciando el cumpliendo del objetivo planteado.
- Si bien el sistema no alcanzó a integrarse al portal, se utilizó el estándar de usar archivos .env para dejar el sistema desarrollado altamente configurable y todo lo utilizado en la gestión de base de datos fácilmente modificable para su integración posterior final al portal.

Por lo antes expuesto, se concluye que se desarrolló satisfactoriamente una solución informática que permite la comunicación sincrónica y asincrónica de los usuarios del portal Cosmox.

5 Recomendaciones

En aras de mejorar el funcionamiento del sistema se recomienda continuar con el desarrollo del módulo propuesto, agregando nuevas funcionalidades, como, por ejemplo:

- Adicionar funcionalidades de seguridad que permita a los usuarios denunciar a otros usuarios que creen un ambiente alejado a los objetivos del sistema en los chats y que el administrador pueda enviarles advertencias y bloquear la cuenta de esos usuarios.
- Agregar la opción de envió de imágenes y emojis para una mayor diversidad de opciones e la hora de comunicarse en las salas de chat.

6 Referencias Bibliográficas

1. acervolima.com. 2022. Estructura MVT del proyecto Django. *estructura mvt del proyecto django.* [En línea] 2022. https://es.acervolima.com/estructura-mvt-del-proyecto-django/.
2. Dayana Bustamante, J. C. (2018). *Metodología Actual. Metología XP.* Barinas: UNIVERSIDAD NACIONAL EXPERIMENTAL DE LOS LLANOS OCCIDENTALES EZEQUIEL ZAMORA.
3. **Andalucía., Junta de.** Chat y mensajería. *Educar para proteger.* [En línea] http://www.andaluciaesdigital.es/educarparaproteger/adolescentes/capitulos/perfilestics/chat-y-mensajeria.html. 4.
4. **Bedu. 2020.** 3 razones para usar el framework Django. *bedu.org.* [En línea] 2020. https://bedu.org/blog/tecnologia/3-razones-para-usar-el-framework-django.
5. **Bernal, Juan Pablo Romero. 2022.** Introducción a Django Rest Framework. *axiacore.com.* [En línea] 2022. https://axiacore.com/blog/introduccion-a-django-rest-framework-460/.
6. **BR. SINTYA MILENA MELÉNDEZ VALLADAREZ, BR. MARIA ELIZABETH GAITAN, BR. NELDIN NOEL PÉREZ REYES. 2016.** *METODOLOGIA ÁGIL DE DESARROLLO DE SOFTWARE PROGRAMACION.* NICARAGUA, MANAGUA : s.n., 2016.
7. **Castro, David. 2021.** Archivos .env en Python. *davidcasr.medium.com.* [En línea] mayo 12, 2021. https://davidcasr.medium.com/archivos-env-en-python-c80ec95cb991.
8. **CHRISTIAN JOHN CÁRDENAS TUTILLO, EDGAR FERNANDO QUIMBITA QUINGALUISA. 2017.** *ANÁLISIS, DISEÑO Y CONSTRUCCIÓN DE UN PROTOTIPO DE UNA.* Quito : s.n., 2017.
9. **Duque, Raúl González.** *Python para todos.* España : Creative Commons Reconocimiento 2.5 España.

10. **Farías, Miguel Farías. 2008.** El ciberlecto de las salas de chateo: ¿conversación escrita o escritura conversada? Chile : Universidad de Santiago de Chile, 2008.
11. **Foundation, Django Software. 2015.** Django Documentation. 2015.
12. **GB, Tony. 2019.** Variables de entorno con Dotenv en Python. *medium.com/.* [En línea] octubre 6, 2019. https://medium.com/@tonybolanyo/variables-de-entorno-con-dotenv-en-python-4697540bf7e4.
13. **Gómez, Marcos Leva. 2018.** WebSockets y WebServices. Mazatlán, Sinaloa : s.n., 2018.
14. **Goya, Emelia Domínguez. 2012.** *MEDIOS DE COMUNICACIÓN MASIVA.* s.l. : ED TERCER MILENIO S.C., 2012.
15. **Hiken, Arthur. 2020.** Una onza de prevención: seguridad y protección a través de estándares de codificación de software. *es.parasoft.com.* [En línea] abril 24, 2020. https://es.parasoft.com/blog/an-ounce-of-prevention-software-safety-security-through-coding-standards/.
16. **Ing. Liester Cruz Castro, Ing. Miosotis Aida Troche Rodríguez, Ing. Javier Rodríguez Ramírez, Ing. Arianna Guerra Medrano. 2015.** Protocolos y tecnologías para la transmisión de datos en tiempo real por la red. *Sociedadelainformacion.com.* [En línea] 2015. http://www.sociedadelainformacion.com/53/protocolos.pdf.
17. **Jessika Natalia Chibuque Molano, Wilder Banoy Suárezb. 2022.** Los videojuegos y su contribución al desarrollo cognitivo y social en laniñez intermedia de Iberoamérica. 2022, Vol. 15.
18. **Lay, Nelson, y otros. 2019.** Uso de las herramientas de comunicación asincrónicas y sincrónicas en la banca privada del municipio Maracaibo(Venezuela) . 2019. Vol. 40.
19. **López, Patricia Alejandra. 2005.** MensajerÌa Instantanea en Internet. 2005.
20. Los videojuegos y su contribución al desarrollo cognitivo y social en la niñez intermedia de Iberoamérica. **Jessika Natalia Chibuque Molano, Wilder Banoy Suárezb. 2022.** 2022, Vol. 15.

21. **Macías, Brenda. 2019.** Steam Chat, la nueva app de Valve, ya está disponible en iOS y Android . *xatakamovil.com.* [En línea] mayo 22, 2019. https://www.xatakamovil.com/aplicaciones/steam-chat-nueva-app-valve-esta-disponible-ios-android.
22. **MARTÍNEZ, SIAMARA ADRIANA ÑAMO. 2019.** DESARROLLO DE UN PROTOTIPO PARA LA AUTOMATIZACIÓN DE UN SISTEMA DE RIEGO DE AGUA Y CONTROL REMOTO MEDIANTE LA PLATAFORMA ZOLERTIA REMOTE. Quito : s.n., 2019.
23. **MATHEUS, Hender Alexander VILORIA. 2019.** *Uso de las herramientas comunicativas en los entornos virtuales de aprendizaje, Javier HAMBURGER.* Ecuador : Chasqui. Revista Latinoamericana de Comunicación, 2019.
24. **Maximiliano Pichel Luck, José María Lezcano. 2012.** *Las salas de chat y el moderador.* s.l. : Facultad de Ciencias Jurídicas y Sociales –UNLP, 2012.
25. mensajeria-instantanea-tipos-para-que-sirve. *mensajeria-instantanea.* [En línea] https://www.b2chat.io/blog/mensajeria-instantanea/mensajeria-instantanea-tipos-para-que-sirve.
26. **Merino, J. Perez Porto y M. 2022.** Definición de Chat-Qué es, Significado y Concepto. *definicion.de.* [En línea] noviembre 25, 2022. https://definicion.de/chat/.
27. **Pardo, Raquel Molina. 2018.** Las redes sociales en la actualidad. 2018.
28. **Patricia, López Herrera. 2016.** Comparación del desempeño de los Sistemas Gestores de Bases de Datos MySQL y PostgreSQL. México : s.n., 2016.
29. **Patricio Letelier, M. Carmen Penadés. 2006.** Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). *Técnica Administrativa, Buenos Aires.* 2006, Vol. 05.
30. **Peña, Jorge Rábanos. 2016.** Análisis y Desarrollo de un Sistema de. Madrid : UNIVERSIDAD POLITÉCNICA DE MADRID, 2016.
31. **Pérez Mariñán, P. "Patrones de Diseño (Design Patterns).** Patrones Gof. [En línea] https://www.ecured.cu/Patrones_Gof.

32. **Pérez, Javier Eguíluz. 2008.** *Introducción a CSS.* 2008.
33. **—. 2008.** Introduccion a javascript. 2008.
34. **—. 2008.** *ntroducción a XHTML.* s.l. : Creative Commons Reconocimiento -, 2008.
35. **Pressman, Roger S. 2002.** Visual Paradigm. *Ecured.* [En línea] 2002. https://www.ecured.cu/Visual_Paradigm.
36. **Pressman, Roger S. 2010.** *Ingenieria de Software. Un enfoque práctico.* 2010. Séptima Edición.
37. **programmerclick.com. 2022.** Estándares de codificación de Django. [En línea] 2022. https://programmerclick.com/article/88171495957/.
38. **Rebelde, Juventud. 2020.** COSMOX: nuevo portal de la UCI para videojuegos online. *Juventudrebelde.cu.* [En línea] febrero 20, 2020. https://www.juventudrebelde.cu/ciencia-tecnica/2020-02-20/cosmox-nuevo-portal-de-la-uci-para-videojuegos-online.
39. **Rodríguez, María José Blanco. 2002.** *El chat: la conversación escrita.* s.l. : Universidad de Alicnate, 2002.
40. **Rosario, Edison Del. 2019.** MQTT – Conexión desde Python. *blog.espol.edu.ec.* [En línea] 2019. http://blog.espol.edu.ec/girni/mqtt-conexion-desde-python/.
41. **Sáez, Julia Sanmartín. 2007.** *El chat. La conversación tecnologica.* Madrid : LAVEL. S.A, 2007.
42. **SINTYA MILENA MELÉNDEZ VALLADAREZ, MARIA ELIZABETH GAITAN, NELDIN NOEL PÉREZ REYES. 2016.** METODOLOGIA ÁGIL DE DESARROLLO DE SOFTWARE PROGRAMACION. Managua, nicaragua : s.n., 2016.
43. **Spiegato. 2022.** ¿Qué es una ventana de chat? *spiegato.com.* [En línea] 2022. https://spiegato.com/es/que-es-una-ventana-de-chat.
44. **Tao, Dekun. 2022.** How to use MQTT in Python (Paho). *emqx.com/.* [En línea] 2022. https://www.emqx.com/en/blog/how-to-use-mqtt-in-python.

45. Uso de las herramientas de comunicación asincrónicas y sincrónicas en la banca privada del municipio Maracaibo (Venezuela). **LAY, Nelson 1, y otros. 2019.** Venezuela : Revista ESPACIOS, 2019, Vol. Vol. 40 .
46. Blizzard. (2022). *Aplicación de escritorio de Blizzard*. Obtenido de blizzard.com: https://www.blizzard.com/es-es/apps/battle.net/desktop
47. Cabeza, M. (2011, junio 3). *EA anuncia Origin*. Obtenido de ea.com: https://www.ea.com/es-es/news/ea-anuncia-origin
48. Ciuraneta, C. (2019, septiembre 17). *Rockstar lanza su propio launcher para PC con GTA San Andreas gratis*. Obtenido de as.com/: https://as.com/meristation/2019/09/17/noticias/1568743049_438646.html
49. Delgado, A. (2020, noviembre 17). *¿Qué es GOG y para qué sirve?* Obtenido de geeknetic.es: https://www.geeknetic.es/GOG/que-es-y-para-que-sirve
50. Mascaretti, N. (2021, junio 6). Obtenido de hd-tecnologia.com: https://www.hd-tecnologia.com/nvidia-rtx-dlss-ray-tracing-o-reflex-llegan-a-red-dead-redemption-2-doom-eternal-rainbow-six-siege-y-a-otros-juegos/
51. olg, O. (2021, junio 28). *Cómo usar las nuevas funciones del panel social en la tienda de Epic Games*. Obtenido de clasesordenador.com: https://www.clasesordenador.com/como-usar-las-nuevas-funciones-del-panel-social-en-la-tienda-de-epic-games/
52. UbisoftConnect. (2022). Obtenido de ubisoftconnect.com: https://ubisoftconnect.com/es-ES/
53. Zucarelli, S. (2022, marzo 9). *Cómo es Itch.io, la página de videojuegos gratis y el paraíso de los desarrolladores independientes*. Obtenido de nfobae.com: https://www.infobae.com/latinpower/gaming/2022/03/09/como-es-itchio-la-pagina-de-videojuegos-gratis-y-el-paraiso-de-los-desarrolladores-independientes/

7 Anexos

7.1 Historias de usuario

Tabla 16Historia de Usuario #5

Historia de usuario	
Número: 5	**Nombre:** Iniciar sesión
Usuario: Usuario del Sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:**
Estimación: 10 días	**Iteración asignada:** 1
Descripción: *El sistema debe permitir que los usuarios* poder autenticarse en el sistema **Objetivo:** Permitir a los usuarios autenticarse al sistema. **Comportamientos válidos y no válidos (flujo central y alternos):** -Si el usuario introduce un nombre de usuario que no está registrado en el sistema este lanzara un cartel de error -Si el usuario introduce la contraseña que no corresponde con nombre de usuario que se puso el sistema este lanzara un cartel de error **Flujo de la acción a realizar:** - El sistema debe permitir a los usuarios autenticarse en el sistema. - Cuando el usuario rellena los campos “usuario” y “contraseña” correctamente el sistema lo redirecciona a la vista rooms.html -Si el usuario introduce datos erróneos el sistema permanecerá en la página de autenticación	
Prototipo de Interfaz:	

Tabla 17Historia de Usuario #6

Historia de usuario	
Número: 6	**Nombre:** Cerrar sesión
Usuario: Usuario del Sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Estimación: 5 días	**Iteración asignada:** 1
Descripción: el sistema permitirá al usuario cerrar la sesión activa en el sistema **Descripción:** el sistema permitirá al usuario cerrar la sesión activa en el sistema. **Objetivo:** Permitir a los usuarios cerrar la sesión. **Acciones para lograr el objetivo (precondiciones y datos):** -el usuario deberá estar autenticado. **Flujo de la acción a realizar:** - El sistema debe permitir a los usuarios cerrar la sesión iniciada - Cuando el usuario decida terminar la sesión solo deberá ir a la opción “Cerrar Sesión” en el menú y este lo redireccionara a la pantalla principal del sistema con las opciones “Autenticarse” y “Registrarse” en el menú.	
Prototipo de Interfaz:	

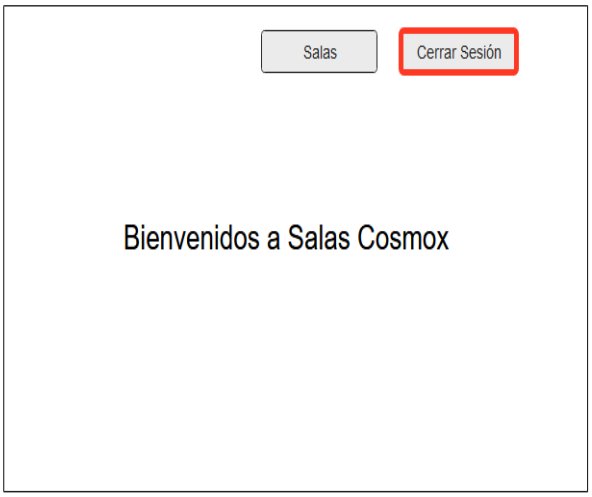


Tabla 18Historia de Usuario #7

Historia de usuario	
Número: 7	**Nombre:** Registrarse en el sistema
Usuario: Usuario del Sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Estimación: 10 día	**Iteración asignada:** 1
Descripción: El sistema debe permitir que los usuarios poder registrarse dentro del sistema ingresando un nombre o Nick y una contraseña. **Objetivo:** Permitir a los usuarios registrarse al sistema. **Acciones para lograr el objetivo (precondiciones y datos):** No tiene **Comportamientos válidos y no válidos (flujo central y alternos):** -Si el usuario introduce un nombre de usuario que ya está registrado en el sistema este lanzara un cartel de error “Ya existe un usuario con ese nombre de usuario. ” -Si el usuario introduce la contraseña que no cumple con las normas de seguridad el sistema lanzará uno o varios de los siguientes carteles de error “La contraseña es muy similar al nombre de usuario”, “Esta contraseña es muy corta”, “La contraseña debe contener más de 8 caracteres” o “Esta contraseña es muy común.” **Flujo de la acción a realizar:** - El sistema debe permitir a los usuarios autenticarse en el sistema.	

- Cuando el usuario rellena los campos “Usuario”, “Contraseña” y” “Repetir contraseña” correctamente el sistema lo redirecciona a la vista rooms.html -Si el usuario introduce datos erróneos el sistema lanzara los mensajes de error correspondientes.
Prototipo de Interfaz: 

Tabla 19Historia de Usuario #8

Historia de usuario	
Número: 8	**Nombre:** Abandonar sala de chat
Usuario: Usuario del Sistema	
Prioridad en negocio: Alto	**Riesgo en desarrollo:** Alto
Estimación: 5días	**Iteración asignada:** 2
Descripción: permite al usuario salir de una sala de chat determinada. **Objetivo:** Permitir a los usuarios salir de la sala de chat en la que se encuentren. **Acciones para lograr el objetivo (precondiciones y datos):** Para enviar abandonar una sala de chat hay que: -Estar autenticado en el sistema -Haberse unido a una sala de chat **Flujo de la acción a realizar:** - El sistema debe permitir a los usuarios abandonar la sala de chat donde se encuentre.	

- Cuando se une a una sala de chat este lo envía a una ventana de chat correspondiente a dicha sala donde estará la opción “Desconectarse” a través de la cual al presionar ese botón lo mandará a la vista rooms.html. el usuario también podrá abandonar una sala accediendo a cualquiera de las opciones presentes en el menú del sistema.
Prototipo de Interfaz: 

7.2 Tarjetas CRC

Tabla 20Tarjetas CRC #2

Tarjetas CRC	
Nombre de la clase: chat.model	
Responsabilidades: Almacenar la información referente a los chats realizados por los clientes.	**Colaboradores:** • chat.views

Tabla 21Tarjetas CRC #3

Tarjetas CRC

Nombre de la clase: user.views	
Responsabilidades: Pedirá información del modelo y se la pasará a la plantilla. def signup(): para registrar a los usuarios en el sistema	**Colaboradores:** • chat.model • chat.urls • core.urls

Tabla 22Tarjetas CRC #4

Tarjetas CRC	
Nombre de la clase: user.forms	
Responsabilidades: Generar formularios para el registro de usuarios.	**Colaboradores:** • chat.model • chat.urls • core.urls

7.3 Tareas de Ingeniería

Tabla 23Tarea de la Ingeniería #2

Tarea de Ingeniería2	
Número de HU: 7	
Nombre: Registrarse en el sistema	
Tipo de tarea: Desarrollo	**Estimación:** 10 días
Fecha inicio: (3/9/22).	**Fecha de Fin:** (12/9/22).
Programador responsable: Claudia de la Caridad Mercadet Vasallo	
Descripción: Crear plantilla y vista para registrar usuarios. Manejar mensajes en	

caso satisfactorio o de registro fallido

Tabla 24Tarea de la Ingeniería #3

Tarea de Ingeniería3	
Número de HU: 6	
Nombre: Cerrar sesión	
Tipo de tarea: Desarrollo	**Estimación:** 5 días
Fecha inicio: (13/9/22).	**Fecha de Fin:** (17/9/22).
Programador responsable: Claudia de la Caridad Mercadet Vasallo	
Descripción: El usuario al darle clic a la opción Cerrar Sesión se redireccionado a la vista login.html	

Tabla 25Tarea de la Ingeniería #4

Tarea de Ingeniería5	
Número de HU: 2	
Nombre: Mostrar lista de salas de chat	
Tipo de tarea: Desarrollo	**Estimación:** 10 días
Fecha inicio: (18/9/22).	**Fecha de Fin:** (29/9/22).
Programador responsable: Claudia de la Caridad Mercadet Vasallo	
Descripción: La función recibe como parámetros una lista de nombres de las salas existentes en la base de datos. Crear plantilla, vista y url para listar salas y las muestra en la vista rooms.html	

Tabla 26Tarea de la Ingeniería #6

Tarea de Ingeniería7	
Número de HU: 3	
Nombre: Enviar mensaje en la sala de chat	
Tipo de tarea: Desarrollo	**Estimación:** 10 días
Fecha inicio: (11/10/22).	**Fecha de Fin:** (20/10/22).
Programador responsable: Claudia de la Caridad Mercadet Vasallo	
Descripción: permite el envío de mensajes dentro de una sala de chat.	

Tabla 27Tarea de la Ingeniería #7

Tarea de Ingeniería8	
Número de HU: 4	
Nombre: Mostrar historial de conversación	
Tipo de tarea: Desarrollo	**Estimación:** 10 días
Fecha inicio: (21/10/22).	**Fecha de Fin:** (30/10/22).
Programador responsable: Claudia de la Caridad Mercadet Vasallo	
Descripción: el usuario podrá ver las conversaciones de sus chats y cualquier sala abiertas en cualquier momento.	

Tabla 28Tarea de la Ingeniería #8

Tarea de Ingeniería9	
Número de HU: 8	
Nombre: Abandonar sala de chat	
Tipo de tarea: Desarrollo	**Estimación:** 5 días

Fecha inicio: (31/10/22).	**Fecha de Fin:** (4/11/22).
Programador responsable: Claudia de la Caridad Mercadet Vasallo	
Descripción: permite a un usuario salir de cualquier sala de chat enviándolo a la lista de salas.	

7.4 Casos de pruebas

Tabla 29Caso de Prueba #2

Caso de Prueba	
Numero: 2	**Historia de usuario:** 6
Nombre: Cerrar sesión	
Condiciones de ejecución: Tiene que existir un usuario registrado en el sistema	
Entrada/Pasos de ejecución: una vez el usuario está en el sistema podrá acceder cerrar sesión a través del botón ´´Cerrar sesión´´ en el menú de la vista principal	
Resultado esperado: se muestra un listado con las salas de chat existentes en el momento en el sistema.	
Evaluación de la Prueba: Satisfactoria	

Tabla 30Caso de Prueba #3

Caso de Prueba	
Numero: 3	**Historia de usuario:** 7
Nombre: Registrarse en el sistema	
Condiciones de ejecución: tiene que estar en la vista Signup	
Entrada/Pasos de ejecución: una vez el usuario está en esta vista podrá registrarse en el sistema una vez haya completado las opciones del formulario, ´´nombre de usuario´´, ´´Contraseña´´ y ´´Repetir Contraseña´´	
Resultado esperado: se muestra un listado con las salas de chat existentes en el momento en el sistema.	
Evaluación de la Prueba: Satisfactoria	

Tabla 31Caso de Prueba #5

Caso de Prueba	
Numero: 5	**Historia de usuario:** 2
Nombre: Mostrar lista de salas de chat	
Condiciones de ejecución: Tiene que existir un usuario registrado en el sistema	
Entrada/Pasos de ejecución: una vez el usuario está en el sistema podrá acceder a un menú en el que podrá seleccionar una sala de chat en la que entrar e interactuar.	
Resultado esperado: se muestra un listado con las salas de chat existentes en el momento en el sistema.	
Evaluación de la Prueba: Satisfactoria	

Tabla 32Caso de Prueba #6

Caso de Prueba	
Numero: 6	**Historia de usuario:** 3
Nombre: Enviar mensaje en la sala de chat.	
Condiciones de ejecución: Tiene que existir un o más usuarios conectados a una sala.	
Entrada/Pasos de ejecución: una vez el usuario este dentro de una sala el sistema deberá permitirle escribir un mensaje y posteriormente enviarlo con el botón “Enviar”	
Resultado esperado: se muestran los mensajes del usuario en la sala donde lo haya enviado.	
Evaluación de la Prueba: Satisfactoria	

Tabla 33Caso de Prueba #7

Caso de Prueba	
Numero: 7	**Historia de usuario:** 4
Nombre: Mostrar historial de conversación	
Condiciones de ejecución: El usuario debe estar registrado en el sistema y estar dentro de una sala.	
Entrada/Pasos de ejecución: una vez el usuario este en la sala podrá ver mensajes anteriores a su unión a dicha sala.	

Resultado esperado: se muestra una las conversaciones existentes en una sala.
Evaluación de la Prueba: Satisfactoria

Tabla 34Caso de Prueba #8

Caso de Prueba	
Numero: 8	**Historia de usuario:** 8
Nombre: Abandonar sala de chat	
Condiciones de ejecución: el usuario debe estar autenticado en el sistema y estar dentro de una sala.	
Entrada/Pasos de ejecución: una vez el usuario este en la sala podrá salir de esta y se le mostrará la lista de salas.	
Resultado esperado: el usuario abandona la sala de chat donde se encuentra y ve el listado de salas	
Evaluación de la Prueba: Satisfactoria	