UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 2

Sistema informático para la gestión de la contratación de alojamientos empleando arquitectura de microservicios.

Trabajo de diploma para optar por el título de ingeniero en ciencias informáticas.

Autor: Enit Martínez González.

Tutores:

Ing. Lianet Suárez Martínez.

Ing. Víctor Alejandro Roque Domínguez.

La Habana, mayo de 2019

Declaración de autoría

Se declara que soy el único autor de este trabajo titulado “Sistema informático para la gestión de la contratación de alojamientos empleando arquitectura de microservicios” y se autoriza a la Universidad de las Ciencias Informáticas (UCI) y a la Empresa de Tecnologías de la Información para la Defensa (XETID) para que hagan el uso que estimen pertinente.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____

Firma del autor

Enit Martínez González

_____

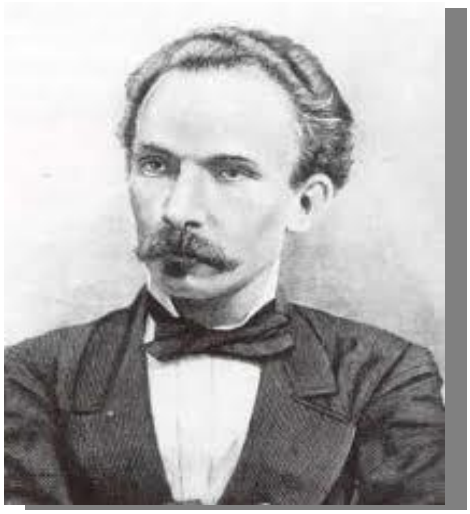
Firma de Tutor

Víctor Alejandro Roque Domínguez

_____

Firma de Tutor

Lianet Suárez Martínez



“Al igual que las piedras rodando por las colinas, las ideas justas alcanzan sus objetivos a pesar de todos los obstáculos y barreras. Puede ser posible para acelerar o dificultar, pero imposible de detener.”

José Martí.

Dedicatoria

Dedico la presente investigación a todos aquellos que confiaron en mí,

en especial a mis padres por ser regla y guía en mi vida personal.

A mi hermana por confiar en esta persona que, con mucho trabajo, aprendió los productos.

A mi amada familia por formar parte en todo momento de mi formación,

y por último a la mujer que compartió conmigo estos 5 años de mucho sacrificio.

A todos, gracias.

Agradecimientos

Ante todo, darle gracias a dios por darme la salud y abrirme las puertas en las elecciones tomadas en la vida. A mis padres por ese apoyo tan incondicional, lleno de amor y de consejos donde todo se los debo a ellos. A mi hermana Aylen que desde pequeño estuviste como ejemplo de excelente estudiante. A mi familia que en las buenas y en las malas siempre escucharon mis ideas, las cuales sin insulto o burla supieron indicarme el camino correcto. A mi tío Emildo por aconsejarme esta carrera y entregarme su tiempo para aprender cada día más. A mi novia que no importa el momento que fuera siempre estuvo a mi lado brindando amor. A mis hermanos en masonería por hacerme miembro de su eslabón en nuestra cadena fraternal en especial a Daniel Menéndez. A mis tutores Lianet y Víctor por su constante preocupación, apoyo y soporte en este y otros momentos de mi vida. Aquellos profesores, trabajadores y guías que participaron en mi formación. Por último a los compañeros de estudio que, desde la FICI hasta este quinto año, me han acompañado.

A todos, gracias.

Resumen

Con el paso del tiempo, se ha vuelto esencial que todos los negocios tengan una presencia en internet, ya sea a través de un sitio web, de una tienda de comercio electrónico, de una página en redes sociales o la combinación de las tres. Con ello los negocios logran reconocimiento por parte de los clientes, adquieren nuevos, fidelizan y conquistan el mercado. La gestión y promoción de alojamientos forma parte de esta gran cadena de negocios, que, en esta era digital, están en constante incremento a través de las tecnologías. Por otra parte, se fideliza la necesidad de apostar por la soberanía tecnológica como estrategia de Cuba para el desarrollo en todas las esferas.

El aumento del turismo ha permitido la necesidad de satisfacer la demanda exigida por clientes nacionales o internacionales en el aumento de alojamientos estatales o privados. La presente investigación tiene el objetivo de desarrollar una aplicación que permita la gestión de la contratación de los alojamientos y con ello la promoción de los mismos tanto para clientes nacionales como foráneos.

Para facilitar el proceso de desarrollo se utilizó la metodología Programación Extrema (XP), cumpliendo con cada una de sus fases, generando todos los artefactos. El trabajo incluye un estudio del estado del arte, las herramientas utilizadas, características del sistema, planificación, diseño, codificación y la estrategia de pruebas propuestas por la metodología seleccionada.

Palabras claves: aplicación web, contratación de alojamiento, gestión de contratación.

Índice

Índice de Figuras

Índice de Tablas

Introducción

Desde los años 70 las tecnologías de la información y las comunicaciones (TIC) han transformado el mercado turístico de manera sustancial, pero ha sido desde el año 2000, con los efectos de la llegada de Internet, cuando ha comenzado la verdadera transformación en especial en la comunicación y en la gestión de las relaciones con los públicos. El sector del turismo ha crecido de manera significativa durante los últimos años, permitiendo a cada país ofrecer y promocionar sus recursos para conseguir con ello, no sólo darse a conocer en todo el mundo sino, además, aumentar su número de visitantes, incentivando el consumo y mejorando así su situación en la economía. (Verdecia 2018)

La relación contractual que se celebra entre el turista y el proveedor del servicio turístico por Internet permite incorporar numerosas ventajas tanto para las empresas que intervienen en el sector turístico como para los turistas. (Márquez Lobillo 2011) De esta manera el turista puede conocer aún más sobre los servicios, comparar precios y ofertas, al igual, el proveedor de servicios turísticos conoce mejor las necesidades, gustos para ofrecer mucho más personalizadas las nuevas ideas y lograr por último una expansión comercial. En la actualidad, cada vez más, los consumidores actúan por si solos en la búsqueda a través de Internet de información sobre hoteles, lo que ha provocado un incremento de contrataciones de alojamientos por medio de esta vía de distribución. (Verdecia 2018)

Los turistas quieren conocer lugares diferentes al propio por lo cual, la imagen de un destino turístico difundida en Internet puede ser un estímulo para los potenciales viajeros. Esto no sólo despierta el interés y la curiosidad por el viaje, sino que también forma parte del proceso de decisión de un turista. A través de un simple clic llegan a conocer la historia, la cultura, los hábitos y las costumbres de los destinos nunca antes imaginados. La promoción de un lugar turístico no pasa sólo por la existencia de atractivos turísticos en Internet, también pasa por la forma como esos atributos son difundidos. Finalmente es importante tener en cuenta que quien no sabe comunicar su destino vía Internet corre el riesgo de ser superado por la competencia. (Verdecia 2018)

En Cuba el turismo ha devenido desde finales de la década de 1980 una actividad priorizada dentro de la estrategia de desarrollo diseñada por el Estado y Gobierno cubanos para la obtención a corto y mediano plazo de ingresos en divisas que contribuyan a la recuperación económica del país y posibiliten su reinserción, sobre bases radicalmente nuevas, en el mercado económico internacional. (Verdecia 2018)

En la actualidad la actividad turística ha evidenciado una evolución favorable. La isla cuenta actualmente con 66,547 habitaciones y para el año 2030 espera llegar a un total de 104,000. (Hosteltur 2017). El sitio

oficial de noticias Cubadebate publicó que el pasado año archivó la cifra récord de cuatro millones 689 mil 898 visitantes foráneos y un incremento del 16,2 por ciento, resultado notable cuando a nivel mundial el movimiento de turistas crece alrededor del cuatro por ciento (Cubadebate 2018), por tanto, ha provocado que el sector público y privado se incremente para satisfacer las demandas relacionadas al turismo y la búsqueda de nuevas maneras de remuneración. (Verdecia 2018)

El país ha desarrollado aplicaciones web para promocionar, contratar y brindar información para el conocimiento de clientes sobre centros turísticos haciendo uso de la soberanía tecnológica. Como parte de esta esfera está el alquiler de cuartos o viviendas que con ello es necesario utilizar sistemas de contratación con el que todos los dueños ganan porcientos del pago por ofrecer publicidad y reservaciones, a su vez obtener conocimiento por parte de los clientes sobre la disponibilidad, ofertas y condiciones de los propietarios.

A su vez, la isla ha apostado en el tema de la soberanía tecnológica, pues el desarrollo de software y hardware libres son condiciones necesarias, pero no suficientes en un entorno condicionado por el ciberespionaje y el control de las grandes transnacionales de la información y las comunicaciones. Uno de los casos más emblemáticos de este problema es el que sufre Cuba como parte del bloqueo impuesto en los años '60 como señala el sitio oficial del ministerio de comercio exterior del país. Actualmente el mapa mundial de cables submarinos muestra un diseño desalentador, en donde pocas y grandes corporaciones son las propietarias de dichos cables y en muchos de los casos tienen el poder y se exige el permiso de poder auditar, censurar o bloquear determinado tipo de contenidos o tráfico de determinados países o usuarios. («El bloqueo impide mayor y mejor acceso de Cuba a Internet» 2015)

Con el embargo comercial los turistas tanto nacionales o internacionales les hace difícil la navegación a sitios confiables y con carácter estatal en donde la información sea con carácter real. En la esfera del turismo ha asignado la responsabilidad a diferentes centros de desarrollo a implementar aplicaciones web para promocionar, contratar y brindar información para el conocimiento de clientes sobre centros turísticos poniendo en función el uso de la soberanía tecnológica.

Las Fuerzas Armadas Revolucionarias (FAR) posee una extensa variedad de centros de producción y de desarrollo entre ellos se encuentra la Empresa de Tecnologías de la Información para la Defensa (XETID) ubicada en la Universidad de las Ciencias Informáticas (UCI). Tras los beneficios encontrados en el incremento del turismo nacional e internacional, es necesario en el sector estatal sistemas cubanos que permitan la contrata y gestión de alojamientos tanto nacionales como internacionales, por lo cual la XETID desea como parte de las tareas del país para lograr la soberanía tecnológica un sistema que les permita

participar en el proceso de gestión de alojamientos que a su vez tiene vinculado la contratación en el sector estatal y privado.

Teniendo en cuenta la situación antes descrita se arriba al siguiente **problema de investigación:** ¿Cómo facilitar la contratación de alojamientos en el negocio de arrendamiento nacional e internacional dirigido por la XETID?

Objetivo general: Desarrollar un sistema informático empleando la arquitectura de microservicios que permita la contratación de alojamientos.

Por tanto, se centra el **objeto de estudio** en el proceso de gestión de alquileres de alojamientos definiendo como **campo de acción** la contratación de alojamientos para el sector estatal y privado.

Para dar cumplimiento al objetivo trazado anteriormente se plantean las siguientes **tareas de investigación**:

- Definición del marco teórico asociado a la contratación de locales y alojamientos.
- Análisis de los sistemas para la contratación de locales o alojamientos, así como sus conceptos asociados.
- Caracterización del proceso de desarrollo basado en microservicios.
- Selección de metodologías, tecnologías, lenguajes y herramientas para el desarrollo del sistema.
- Análisis y diseño de la propuesta de solución.
- Implementación de la propuesta solución.
- Diseño de pruebas de software y de validación de la solución implementada.

Los métodos de investigación empleados son:

Métodos teóricos

El método Analítico-Sintético se utilizó en la búsqueda y análisis de herramientas para la contratación de casas de renta. También en el análisis de la arquitectura de microservicios en los sistemas para la contratación de alojamientos analizando cómo se relacionan sus componentes entre sí.

El método Histórico-Lógico permitió una mayor comprensión del estado y tendencias actuales del proceso de contratación de alojamientos.

La Modelación permitió una mejor comprensión de los elementos analizados y diseñados para posterior implementación mediante el uso de diagramas y modelos más simples.

Métodos empíricos

La Entrevista se empleó en encuentros con profesores y especialistas de la XETID, la cual facilitó comprender el proceso de creación de la información; además, cómo se implementan los microservicios y que tecnologías utilizan.

Estructura del documento

La presente investigación está estructurada en 3 capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos que ampliarán la información que se aporta en la investigación. A continuación, se describe el contenido de cada capítulo.

Capítulo 1: "**Fundamentación teórica**". En este capítulo se definen conceptos importantes para la comprensión de la investigación. Se realiza un estudio y valoración sobre algunas de los sistemas para la contratación de alojamientos, al igual que la utilización de los microservicios y se explican las herramientas, metodologías y el lenguaje que serán utilizados en la construcción de la solución.

Capítulo 2: "**Análisis y diseño del sistema**". En este capítulo se presenta y analiza la solución que se propone, se brindan elementos de cómo está concebido el negocio y se realiza la selección de los requerimientos del sistema que se pretende implementar. Se modelan y describen los diagramas que representan las funcionalidades del sistema, aplicando los patrones de arquitectura y diseño seleccionados.

Capítulo 3: "**Implementación y prueba del sistema**"**.** En este capítulo mediante el apoyo en los diagramas de componentes y de despliegue, se describe la herramienta propuesta desde el enfoque de la programación. Además, se realizan casos de prueba para comprobar que el software cumple con los requerimientos establecidos, se describe la implementación y posterior validación realizada al producto obtenido como solución.

Capítulo I. Fundamentación teórica

En el presente capítulo se realiza la explicación de diferentes conceptos relacionados al tema de investigación como **gestión de alquileres y contratación**. Se explica el porqué de la selección de las herramientas, lenguajes, metodologías, IDEs de desarrollo y artefactos a utilizar. Además del análisis de aplicaciones ya existentes que gestionan la contratación de alojamientos sean estatales o privados.

1.1. Conceptos asociados

1.1.1. Gestión de alquileres

La gestión consiste en identificar y entender los procesos interrelacionados como un sistema, contribuye a la eficacia y eficiencia de una organización en el logro de sus objetivos («ISO 9000:2000 Sistemas de Gestión de la Calidad. Conceptos y Vocabulario.» 2000)

La gestión no es más que el proceso de organizar, evaluar, presentar, comparar los datos en un determinado contexto, controlando su calidad, de manera que esta sea veraz, oportuna, significativa, exacta y útil y que esta información esté disponible en el momento que se le necesite. Ella se encamina al manejo de la información, documentos, metodologías, informes, publicaciones, soportes y flujos en función de los objetivos estratégicos de una organización. (Vidal Ledo y Araña Pérez 2012)

El alquiler consiste en dar o tomar alguna cosa para usar de ella por tiempo determinado mediante el pago de una cantidad convenida. Entrar uno al servicio de otro por cierta remuneración. (Diccionario Ilustrado 2010)

El autor propone que la **gestión de alquileres** es el conjunto de operaciones que se realizan para dirigir y administrar procesos mediante el cual dos partes efectúan la cesión temporal de un bien o servicio a cambio de una contraprestación que generalmente es de tipo económico.

1.1.2. Contratación de alojamientos

El contrato electrónico se define como: “todo contrato celebrado sin la presencia física simultánea de las partes, prestando estas su consentimiento en origen y en destino por medio de equipos electrónicos de tratamiento y almacenaje de datos, conectados por medio de cable, radio, medios ópticos o cualquier otro medio electromagnético”. (Convelia 2011)

La contratación electrónica puede ser también la aceptación de un convenio de colaboración, la contratación de servicios o incluso la aceptación de una política de privacidad o las condiciones de uso de una red social. (Burgueño 2010)

El alojamiento es el lugar donde se hospeda o aposenta una persona (Diccionario Ilustrado 2010)

El autor propone que la **contratación de alojamientos** es el procedimiento a través del cual una persona o empresa contrata lugares que permiten alojar, en cambio de una remuneración monetaria y que quedará formalizado en un contrato que identificará obligaciones y derechos de cada parte.

1.2. Estudio de sistemas de contratación de alojamientos

Existen diferentes aplicaciones informáticas que realizan el proceso de contratación de locales o alojamientos, tales como **Homestay** y **Booking**. De las aplicaciones de Android se selecciona para su análisis Airbnb.apk para contratar al sector privado y estatal. Además, en el ámbito nacional existen alternativas que hacen uso de la contratación.

1.2.1. Internacionales

Homestay: Fundada en 2013 por los veteranos de la industria de viajes Tom Kennedy (cofundador de *Hostelworld.com*) y *Debbie Flynn* (*Irish Education Partners & A Star Hostels Group*), con la visión de tener una industria fuera de línea en línea y hacer de las casas de familia y habitaciones privadas. Homestay.com ahora ofrece a sus huéspedes una selección de más de 55,000 habitaciones en más de 160 países. (Homestay 2018) Entre otras características la misma posee información de Cuba como de otros países, no utiliza arquitectura de microservicios y no genera ingresos al país.

Booking: *Booking.com*, cuya misión es poner al mundo al alcance de todos, apuesta por la tecnología digital para que viajar sea más fácil. Booking conecta a los viajeros con la oferta más amplia de alojamientos, como apartamentos, casas, resorts de lujo, *bed & breakfasts* de gestión familiar, iglús y casas en los árboles. La web y apps de Booking.com están disponibles en más de 40 idiomas, y ofrecen 29.004.669 opciones de alojamiento en total, en 143.171 destinos de 230 países y territorios en todo el mundo. (Booking 2018) Entre otras características la misma posee información de Cuba como de otros países, no utiliza arquitectura de microservicios y no genera ingresos al país.

Airbnb.apk: Es una aplicación Android dedicada a la oferta de alojamientos a particulares y turísticos, en esta plataforma las personas pueden publicitar el arriendo de sus propiedades ya sea por periodos mínimos de tiempo y valores inferiores al resto de la oferta hotelera. Airbnb tiene una oferta de unas 2.000.000 propiedades en 192 países y 33.000 ciudades. (Airbnb 2018) Entre otras características la misma posee información de Cuba como de otros países, utiliza arquitectura de microservicios y no genera ingresos al país.

1.2.2. Nacionales

HospedajeCubano: Es un equipo de trabajo empeñado en que el visitante encuentre la información que necesita a la hora de realizar un viaje a Cuba. Cuenta con la experiencia de varios años en la comercialización de Cuba como destino turístico dentro del área del caribe. (HospedajeCubano 2018) Entre otras características la misma posee información de Cuba, pero no de otros países, no utiliza arquitectura de microservicios y no genera ingresos al país.

1.2.3. Resultados obtenidos por el estudio

Como resultado del estudio realizado se tuvo en cuenta las ventajas de los sistemas existentes concluyendo que ninguna de las aplicaciones analizadas realiza la correcta contratación de alojamientos. Se identificaron características y componentes reutilizables en el desarrollo de la solución, como el uso de los microservicios y carácter estatal. Por lo tanto, es necesario desarrollar una propuesta de solución que permita a través de una aplicación Web, realizar un correcto proceso de contratación de alojamientos.

1.3. Análisis de las tecnologías, metodologías, lenguajes y herramientas a utilizar en el desarrollo.

1.3.1. Metodologías de desarrollo

Existen metodologías que proporcionan el apoyo necesario para crear un software adaptado a las necesidades del usuario, incluyendo la gestión de riesgos y problemas; sin embargo, en ocasiones tal gestión no se aplica (Fernández Sanz y Bernad Silva 2014), poniendo en riesgo la integridad del producto. Es de vital importancia establecer parámetros que guíen a los desarrolladores y arquitectos a no cometer o repetir acciones equivocadas, y como resultado mejorar los procesos para la creación del software.

Por otra parte, considerando su filosofía de desarrollo, aquellas metodologías con mayor énfasis en la planificación y control del proyecto, en especificación precisa de requisitos y modelado, reciben el apelativo de Metodologías Tradicionales (o también denominadas Metodologías Pesadas o Peso Pesado). Otras metodologías, denominadas Metodologías Ágiles, están más orientadas a la generación de código con ciclos muy cortos de desarrollo, se dirigen a equipos de desarrollo pequeños, hacen especial hincapié en aspectos humanos asociados al trabajo en equipo e involucran activamente al cliente en el proceso. (Pressman y Contreras 2010) A continuación, se analizan algunas metodologías ágiles tras cumplir con la situación existente.

XP

A fin de ilustrar un proceso ágil con más detalle, daremos un panorama de la programación extrema (*XP*, por sus siglas en inglés), el enfoque más utilizado del desarrollo de software ágil. Aunque las primeras actividades con las ideas y los métodos asociados a XP ocurrieron al final de la década de 1980, el trabajo fundamental sobre la materia había sido escrito por Kent Beck. Una variante de esta metodología llamada XP industrial (IXP) se propuso en una época más reciente. IXP mejora la XP y tiene como objetivo el proceso ágil para ser usado específicamente en organizaciones grandes. (Pressman 2005)

La programación extrema (XP) es quizás el método ágil mejor conocido y más ampliamente usado. Debido a que el enfoque se desarrolló llevando a niveles “extremos” las prácticas reconocidas, como el desarrollo iterativo. Por ejemplo, en la XP muchas versiones actuales de un sistema pueden desarrollarse mediante diferentes programadores, integrarse y ponerse a prueba en un solo día. (Sommerville 2011)

En la programación extrema, los requerimientos se expresan como escenarios (llamados historias de usuario), que se implementan directamente como una serie de tareas. Los programadores trabajan en pares y antes de escribir el código desarrollan pruebas para cada tarea. Todas las pruebas deben ejecutarse con éxito una vez que el nuevo código se integre en el sistema. (Sommerville 2011)

SCRUM

Proviene de cierta jugada que tiene lugar durante un partido de rugby. Es un método de desarrollo ágil de software concebido por Jeff Sutherland y su equipo de desarrollo a principios de la década de 1990. En años recientes, Schwaber y Beedle han desarrollado más los métodos Scrum. Sus principios son congruentes con el manifiesto ágil y se utilizan para guiar actividades de desarrollo dentro de un proceso de análisis que incorpora las siguientes actividades estructurales: requerimientos, análisis, diseño, evolución y entrega. Dentro de cada actividad estructural, las tareas del trabajo ocurren con un patrón del proceso llamado sprint. El trabajo realizado dentro de un sprint se adapta al problema en cuestión y se define y con frecuencia se modifica en tiempo real por parte del equipo Scrum. Acentúa el uso de un conjunto de patrones de proceso del software que han demostrado ser eficaces para proyectos con plazos de entrega muy apretados, requerimientos cambiantes y negocios críticos. (Pressman y Contreras 2010)

AUP-UCI

El Proceso Unificado Ágil (AUP, por sus siglas en inglés) variante UCI (AUP-UCI) surge de la necesidad de converger a una única metodología de desarrollo que cubra las particularidades de los distintos centros productivos de la UCI. Al igual que AUP (que es una versión simplificada de RUP), describe de una manera

simple y fácil de comprender, la forma de desarrollar *software* utilizando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. (Rodriguez 2014)

AUP-UCI define 3 fases (inicio, ejecución, cierre), 8 disciplinas (modelado de negocio, requisitos, análisis y diseño, implementación, pruebas internas, pruebas de liberación, pruebas de aceptación, despliegue) y 11 roles (jefe de proyecto, planificador, analista, arquitecto de información, desarrollador, administrador de la configuración, *stakeholder* (cliente/proveedor de requisitos), administrador de calidad, probador, arquitecto de software, administrador de bases de datos). Los requisitos funcionales pueden ser encapsulados en historias de usuario, en descripción de requisitos por procesos o en casos de uso. (Rodriguez 2014)

Selección de la metodología de desarrollo

Se selecciona como metodología de desarrollo de software *extreme programming* (XP), pues se utiliza fundamentalmente para poca cantidad de miembros como es el caso y permite desarrollarse mediante diferentes programadores, integrarse y ponerse a prueba en un solo día. También indica trabajar sobre los constantes cambios de las funcionalidades y es la metodología propuesta por el cliente.

1.3.2. Lenguajes utilizados

HTML5

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. (Eguiluz Pérez 2006)

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. (Eguiluz Pérez 2006)

El lenguaje HTML es un estándar reconocido en todo el mundo y cuyas normas define un organismo sin ánimo de lucro llamado *World Wide Web Consortium* más conocido como W3C. Como se trata de un estándar reconocido por todas las empresas relacionadas con el mundo de Internet, una misma página HTML se visualiza de forma muy similar en cualquier navegador de cualquier sistema operativo. (Eguiluz Pérez 2006)

El propio W3C define el lenguaje HTML como "un lenguaje reconocido universalmente y que permite publicar información de forma global". Desde su creación, el lenguaje HTML ha pasado de ser un lenguaje utilizado exclusivamente para crear documentos electrónicos a ser un lenguaje que se utiliza en muchas aplicaciones electrónicas como buscadores, tiendas online y banca electrónica. (Eguiluz Pérez 2006)

JavaScript v1.8.5

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. (Eguiluz Pérez 2006)

CSS3

CSS 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. (Eguiluz Pérez 2006)

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. (Eguiluz Pérez 2006)

1.3.3. Entornos de desarrollo

Visual Studio Code v1.28.0: Visual Studio Code es un editor de código fuente ligero pero potente que se ejecuta en escritorio y está disponible para Windows, macOS y Linux. Viene con soporte incorporado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros idiomas (como C

++, C #, Java, Python, PHP, Go) y tiempos de ejecución (como .NET y Unity). Posee depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código y fragmentos.

Node.js v8.9.3: Concebido como un entorno de ejecución de JavaScript orientado a eventos asíncronos, Node está diseñado para construir aplicaciones en red escalables. Es de código abierto para la capa de servidor en una arquitectura basada en eventos y basado en el motor V8 de Google.

AJAX: Es el acrónimo para *Asynchronous* JavaScript + XML, en realidad no es una tecnología sino la combinación de muchas tecnologías como son: XHTML y CSS para crear una presentación basada en estándares, DOM (*Document Object Model*) para la interacción y manipulación dinámica de la presentación, XML, XSLT y JSON, para el intercambio y la manipulación de información, *XMLHttpRequest* para el intercambio asíncrono de información y JavaScript para unir todas las demás tecnologías.

Selección de entorno de desarrollo

Se utilizará como entornos de desarrollo **Visual Studio Code v1.28.0** pues permite el trabajo con HTML5, CSS3 y JavaScript. Es compatible con la mayoría de los sistemas operativos existentes además de que es un software libre. También se usará el **Node.js v8.9.3** por ser una aplicación de código abierto además que se necesita realizar trabajos en la capa del servidor. Además, el cliente define estos entornos de desarrollo tras ser utilizados en el centro (XETID).

1.3.4. Sistemas gestores de base de datos

Microsoft SQL Server: Es un sistema de gestión de bases de datos relacionales (RDBMS) de Microsoft que está diseñado para el entorno empresarial desarrollado por la empresa Microsoft. Utiliza un lenguaje de programación estándar e interactivo para la obtención de información desde una base de datos y para actualizarla. Aunque SQL es a la vez un ANSI y una norma ISO, muchos productos de bases de datos soportan SQL con extensiones propietarias al lenguaje estándar.

PostgreSQL: Es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD (Berkeley Software Distribution). Utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

MongoDB: MongoDB es un sistema de base de datos No SQL orientado a documentos, desarrollado bajo el concepto de código abierto. MongoDB forma parte de la nueva familia de sistemas de base de datos No SQL. También es una base de datos orientada a documentos JSON, salvo que está diseñada para ser una verdadera base de datos de objetos.

MySQL: es un sistema de gestión de bases de datos relacional (RDBMS). Este programa es capaz de almacenar una gran cantidad de datos y de distribuirlos para satisfacer las necesidades de cualquier tipo de organización. MySQL en sus inicios se consideraba la opción ideal para sitios web; pero actualmente incorpora muchas de las funciones necesarias para otros entornos sin perder su velocidad.

Selección de sistema gestor de base de datos

Se escoge como sistema gestor de base de datos el **PostgreSQL.** Permite que los fallos no afecten al resto del sistema en caso de la ocurrencia de los mismos. Además de la facilidad que nos brinda el PgAdmin para hacer mantenimiento de las tablas o respaldos, Hot-Standby permite que los usuarios puedan acceder a las tablas en modo lectura mientras que se realizan los procesos de backup o mantenimiento.

1.3.5. Herramientas CASE

Visual Paradigm v8.0

Visual Paradigm es una herramienta CASE: Ingeniería de Software Asistida por Computación. 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. Posee disponibilidad en varias plataformas, licencia gratuita y comercial, capacidades de ingeniería directa e inversa, soporta aplicaciones web y fácil de instalar.

1.3.6. Herramientas para pruebas de sistema

Para realizar las pruebas al software tras haber optado como metodología de software XP se utilizará Karma js para pruebas unitarias.

Karma js: Karma es esencialmente una herramienta que genera un servidor web que ejecuta código fuente contra código de prueba para cada uno de los navegadores conectados. Los resultados de cada prueba en cada navegador se examinan y se muestran mediante la línea de comandos al desarrollador para que puedan ver qué exploradores y pruebas pasaron o no. (Karma js 2019)

1.3.7. Conclusiones Parciales

Tras haberse realizado el estudio de los sistemas homólogos se logró establecer que ninguno de los sistemas existentes cumplía con la correcta gestión de la contratación de alojamientos, por lo que demostró la necesidad de desarrollar un sistema informático que permita la contratación de alojamientos tanto en el sector estatal como privado. Además, el estudio de las herramientas, tecnologías y metodologías permitió definir la base para el desarrollo de la propuesta de solución.

Capítulo II. Análisis y diseño del sistema

En el presente capítulo se realiza un análisis de relaciones, dominios y entidades a través de un modelo de dominio. Además, se presentan los distintos requisitos funcionales y no funcionales que presenta el sistema. Por otra parte, se estructura mediante las fases definidas por XP planeación, diseño, codificación y prueba. Para lograr el entendimiento del funcionamiento de la aplicación se describe a través de las historias de usuarios las distintas funcionalidades del sistema.

2.1. Modelo de dominio

Un modelo de dominio es una representación de las clases conceptuales del mundo real, no de componentes software. No se trata de un conjunto de diagramas que describen clases de software, u objetos software con responsabilidades. (Larman,2002)

Su utilidad radica en ser una forma de inspiración para el diseño de los objetos software, muestra las clases conceptuales significativas en un dominio del problema. Es entrada para muchos de los artefactos que se construyen en un proceso software, además, el artefacto clave del análisis orientado a objetos. En UML se utilizan los diagramas de clases para representar los modelos de dominio. En la ilustración 1 se presentan las relaciones entre clases

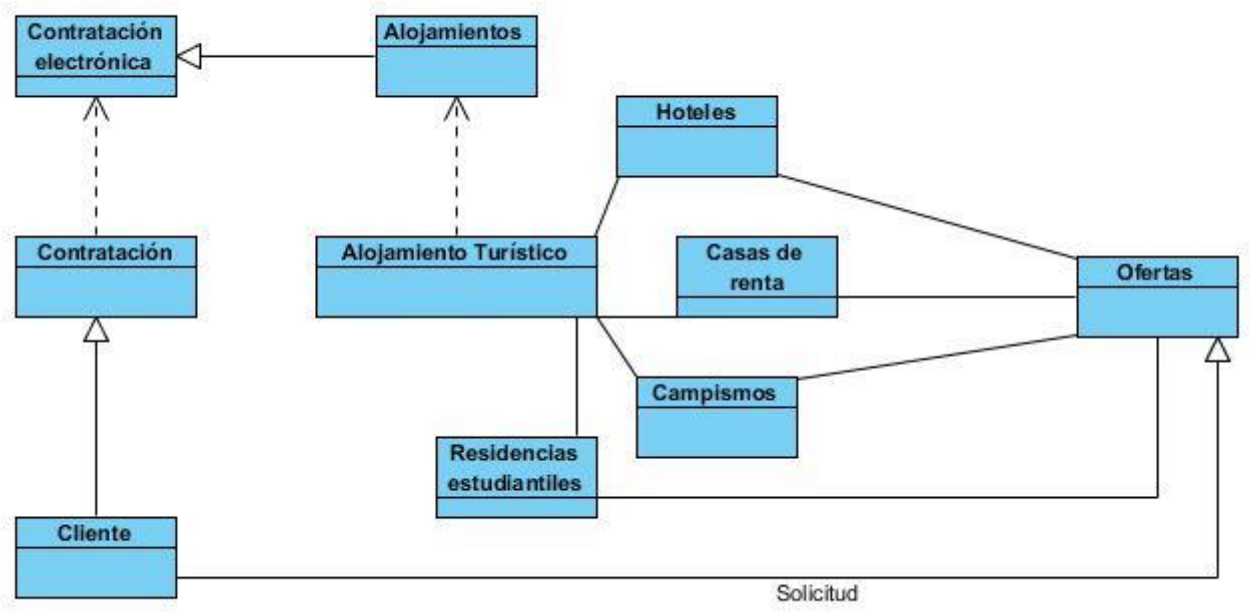


Figura 1 Diagrama de Clases del Modelo de Dominio

2.1.1. Descripción de clases del modelo de dominio

Se presenta el significado de las clases pertenecientes al modelo de dominio.

Cliente: Solicitante de contratación y de información.

Contratación: Acción de solicitar un servicio o producto en cambio de remuneración.

Contratación electrónica: Acción de solicitar un servicio o producto en cambio de remuneración haciendo uso de las Tecnologías de la información y las comunicaciones.

Alojamientos: Referente al lugar donde se hospeda alguien.

Alojamientos Turísticos: Referente al lugar donde se hospeda el turista.

Hoteles: Tipo de alojamiento

Casas de renta: Tipo de alojamiento

Campismos: Tipo de alojamiento

Residencias Estudiantiles: Tipo de alojamiento

Ofertas: Conjunto de características que identifican los tipos de alojamientos que son solicitadas por el cliente y ofrecidas por los proveedores.

2.2. Roles y responsabilidades

A continuación, se muestran los roles y sus responsabilidades correspondiendo al sistema que se implementará.

1. **Administrador**

Responsable de mantener la disponibilidad de los datos y la correcta gestión del proceso, además de darle cierre a un evento y de administrar en el módulo, los niveles de acceso de los usuarios.

2. **Cliente**

Cuenta con las responsabilidades que le otorga el administrador.

2.3. Fase de Planeación

La actividad comienza desarrollando requerimientos que permite que los miembros técnicos que emplean la metodología XP entiendan el contexto del negocio para el software y adquieran la sensibilidad de la salida y características principales y funcionalidad que se requieren. Esto lleva a la creación de algunas historias del usuario que describen la salida necesaria, características y funcionalidad del software que se va a elaborar.

Cada historia es escrita por el cliente y colocada en una tarjeta indizada. El cliente asigna un valor a la historia con base en el valor general de la característica o función para el negocio. Después, los miembros del equipo XP evalúan cada historia y le asignan un costo, medido en semanas de desarrollo. Una vez que se llega a un compromiso sobre la entrega (acuerdo sobre las historias por incluir, la fecha de entrega y otros aspectos del proyecto), el equipo que emplea la metodología XP ordena las historias que serán desarrolladas, medida que avanza el trabajo, el cliente puede agregar historias, cambiar el valor de una ya existente, descomponerlas o eliminarlas.

2.3.1. Levantamiento de Requisitos

El proceso de captura de requisitos inicia con la interacción con el cliente. Durante este proceso se hace una relación de las necesidades que tiene el usuario y se define lo que debe hacer el sistema. Se detallan los usuarios que van a interactuar con el sistema y los niveles de acceso o permisos que tendrán en el sistema. Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Los requisitos no funcionales son requisitos que imponen restricciones en el diseño o la implementación. (Rodríguez, Kile y Romero 2013)

Requisitos funcionales

En XP se elimina la fase inicial de captura de requisitos y se permite que éstos se vayan definiendo de una forma ordenada durante el tiempo que dura el proyecto. El cliente puede cambiar de opinión sobre la marcha y a cambio debe encontrarse siempre disponible para resolver dudas del equipo de desarrollo y para detallar los requisitos especificados cuando sea necesario. El proceso de captura de requisitos de XP gira entorno a una lista de características que el cliente desea que existan en el sistema final. (Luis Calabria 2003) A continuación, se listan las funcionalidades del sistema, aunque se puedan observar en un sentido muy amplio, quedarán definidos de forma específica en las tareas de ingeniería. En la tabla 1 se identifican los requisitos funcionales.

Tabla 1 Casos de usos que responden requisitos funcionales

Código	Descripción de requisitos funcionales de la aplicación. (Parámetros: Tipo)	Prioridad
RF1	Autenticar usuario(usuario: string, contraseña: string)	Alta
RF2	Crear Usuario(id: number, nombre: string, apellidos: string, usuario: string, correo: string, país: string, dirección: string, sexo: string, contraseña: string, token: string)	Alta
RF3	Actualizar Usuario(id: number, nombre: string, apellidos: string, usuario: string, correo: string, país: string, dirección: string, sexo: string, contraseña: string, token: string)	Alta
RF4	Mostrar Usuario(id: number, nombre: string, apellidos: string, usuario: string, correo: string, país: string, dirección: string, sexo: string, contraseña: string, token: string)	Alta
RF5	Eliminar Usuario (Usuario: Usuario)	Alta
RF6	Crear Alojamiento(id: number, nombre: string, tipopropietario: string, tipoalojamiento: string, idioma: string, diasdecancelaciongratis: number, precio: number, moneda: string, descripción: string, wifi: string, baño: string, canthabitaciones: number, dirección: string, parqueo: string, imagen1: string, imagen2: string, imagen3: string, idUsuario: number)	Alta
RF7	Actualizar Alojamiento(id: number, nombre: string, tipopropietario: string, tipoalojamiento: string, idioma: string, diasdecancelaciongratis: number, precio: number, moneda: string, descripción: string, wifi: string, baño: string, canthabitaciones: number, dirección: string, parqueo: string, imagen1: string, imagen2: string, imagen3: string, idUsuario: number)	Alta
RF8	Mostrar Alojamiento(id: number, nombre: string, tipopropietario: string, tipoalojamiento: string, idioma: string, diasdecancelaciongratis: number, precio: number, moneda: string, descripción: string, wifi: string, baño: string, canthabitaciones: number, dirección: string, parqueo: string, imagen1: string, imagen2: string, imagen3: string, idUsuario: number)	Alta
RF9	Eliminar Alojamiento (Alojamiento: Alojamiento)	Alta
RF10	Buscar Alojamiento (nombre: string)	Media
RF11	Crear Imagen (Imagen: file)	Baja
RF12	Actualizar Imagen (Imagen: file)	Baja
RF13	Mostrar Imagen (Imagen: file)	Baja

RF14	Eliminar Imagen (Imagen: file)	Baja
RF15	Mostrar Ayuda	Baja

Requisitos no funcionales (RNF)

Son restricciones de los servicios o funciones ofrecidos por el sistema, incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los requerimientos no funcionales a menudo se aplican al sistema en su totalidad. Los requerimientos no funcionales, como su nombre sugieren, son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de éste como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento.

Las características no funcionales son rasgos que se desean del sistema, señalan una restricción del mismo y son fundamentales en el éxito del producto, son propiedades o cualidades que el producto debe tener y normalmente se vinculan a funcionalidades (Luis Calabria 2003).

Tabla 2 Requisitos no funcionales

Código	**Descripción de requisitos no funcionales de la aplicación.**
	Usabilidad
RNF1	El sistema debe presentar un acceso fácil e interfaces amigables con informaciones fáciles de entender, para facilitar el uso del mismo por usuarios con pocos conocimientos en el campo de la informática.
	Seguridad
RNF2	Se asignarán los permisos de acceso en dependencia del rol que desempeñe cada usuario del sistema.
RNF3	Se debe contar con un sistema de salvas externas para la información que maneja el sistema.
	Software
RNF4	Intérprete de aplicaciones Web (Mozilla Firefox 25.0, Internet Explorer 10, Google Chrome 32.0.1700.107).
RNF5	Sistema Gestor de Base Datos MySQL

Portabilidad	
RNF6	El sistema será multiplataforma, basado en sistemas operativos GNU/Linux y/o Windows
Implementación	
RNF7	Hojas de Estilos en Cascada 3 (CSS).
RNF8	SQL como lenguaje de base de datos.

2.3.2. Historias de usuario(HU)

Las HU son utilizadas por XP para definir los requisitos del software. En estas el cliente describe brevemente las características que debe poseer el sistema y además es el encargado de asignarles una prioridad, también se define el riesgo de desarrollo y el tiempo necesario para la implementación. El programador es el encargado de asignarle un costo de acuerdo al esfuerzo estimado. La metodología XP utiliza la técnica de las HU para sustituir a los documentos de especificación funcional y a los casos de uso. (Penadés 2010)

Estas HU son escritas por el cliente en su propio lenguaje como descripciones cortas de lo que el sistema debe realizar. El tratamiento de las HU es muy dinámico y flexible, permite que en cualquier momento se puedan romper, reemplazar por otras más específicas o generales, añadirse nuevas o ser modificadas. Para ser implementadas las HU, el cliente y los desarrolladores se reúnen para detallar las funcionalidades de cada una. El tiempo de desarrollo ideal para una HU varía entre una y tres semanas. (Penadés 2010) Las siguientes tablas muestran las Historias de Usuarios correspondientes.

Tabla 3 Historia de usuario: Autenticar usuario

Historia de usuario	
Número: HU_1	**Nombre Historia de Usuario:** Autenticar usuario
Programador: Enit Martínez González	**Iteración Asignada:**1
Prioridad: Alta	**Tiempo Estimado:**8 días
Riesgo en Desarrollo: Alta	**Tiempo Real:** 8
Descripción: muestra los campos correspondientes a cada tipo de contenido, los que pueden ser seleccionados por los usuarios a autenticar.	

Observaciones: el campo contenido esta seleccionado por defecto.

Tabla 4 Historia de usuario: Gestionar usuario

Historia de usuario	
Número: HU_2	**Nombre Historia de Usuario:** Gestionar Usuario
Programador: Enit Martínez González	**Iteración Asignada:**1
Prioridad: Alta	**Tiempo Estimado:**10 días
Riesgo en Desarrollo: Alta	**Tiempo Real:** 9
Descripción: muestra los campos correspondientes a insertar, eliminar, mostrar o actualizar los que pueden ser seleccionados por los usuarios.	
Observaciones: el campo contenido esta seleccionado por defecto.	

Tabla 5 Historia de usuario: Gestionar alojamiento

Historia de usuario	
Número: HU_3	**Nombre Historia de Usuario:** Gestionar alojamiento
Programador: Enit Martínez González	**Iteración Asignada:**1
Prioridad: Alta	**Tiempo Estimado:**10días
Riesgo en Desarrollo: Alta	**Tiempo Real:** 10
Descripción: muestra los campos correspondientes a insertar, eliminar, mostrar o actualizar los que pueden ser seleccionados por los usuarios autenticados para gestionar la información de su alojamiento.	
Observaciones: el campo contenido esta seleccionado por defecto.	

Tabla 6 Historia de usuario: Buscar alojamiento

Historia de usuario	
Número: HU_4	**Nombre Historia de Usuario:** Buscar alojamiento
Programador: Enit Martínez González	**Iteración Asignada:**2
Prioridad: Media	**Tiempo Estimado:**3 días
Riesgo en Desarrollo: Baja	**Tiempo Real:** 3
Descripción: muestra los campos correspondientes a cada tipo de contenido, los que pueden ser seleccionados por los usuarios autenticados para buscar alojamientos.	
Observaciones: el campo contenido esta seleccionado por defecto.	

Tabla 7 Historia de usuario: Gestionar Imagen de alojamiento

Historia de usuario	
Número: HU_5	**Nombre Historia de Usuario:** Gestionar Imagen de alojamiento
Programador: Enit Martínez González	**Iteración Asignada:**3
Prioridad: Baja	**Tiempo Estimado:**10 días
Riesgo en Desarrollo: Media	**Tiempo Real:** 8
Descripción: muestra los campos correspondientes a insertar, eliminar, mostrar o actualizar los que pueden ser seleccionados por los usuarios autenticados para gestionar la imagen de su alojamiento.	
Observaciones: el campo contenido esta seleccionado por defecto.	

Tabla 8 Historia de usuario: Mostrar ayuda

Historia de usuario	
Número: HU_6	**Nombre Historia de Usuario:** Mostrar Ayuda
Programador: Enit Martínez González	**Iteración Asignada:**3

Prioridad: Baja	**Tiempo Estimado:**4 días
Riesgo en Desarrollo: Baja	**Tiempo Real:** 3
Descripción: muestra los campos correspondientes a cada tipo de contenido, los que se muestran con carácter informativo.	
Observaciones: el campo contenido esta seleccionado por defecto.	

2.4. Fase de Diseño

El diseño XP estimula el uso de las tarjetas CRC como un mecanismo eficaz para pensar en el software en un contexto orientado a objetos. Las tarjetas CRC (clase-responsabilidad-colaborador) identifican y organizan las clases orientadas a objetos que son relevantes para el incremento actual de software. Las tarjetas CRC son el único producto del trabajo de diseño que se genera como parte del proceso XP. Si en el diseño de una historia se encuentra un problema de diseño difícil, XP recomienda la creación inmediata de un prototipo operativo de esa porción del diseño. Entonces, se implementa y evalúa el prototipo del diseño, llamado solución en punta. El objetivo es disminuir el riesgo cuando comience la implementación verdadera y validar las estimaciones originales para la historia que contiene el problema de diseño.

2.4.1. Tarjetas CRC

Como parte de la fase de diseño en la metodología XP es el uso de las tarjetas CRC como un mecanismo eficaz para pensar en el software en un contexto orientado a objetos. Las tarjetas CRC son el único producto del trabajo de diseño que se genera como parte del proceso XP. Las tarjetas se dividen en tres secciones. En la parte superior de la tarjeta se escribe el nombre de la clase, en la parte izquierda del cuerpo se enlistan las responsabilidades de la clase y en la derecha, los colaboradores.

Tabla 9 Tarjeta CRC: AlojService

Clase : AlojService	
Descripción: Se encarga de gestionar el uso del microservicio alojamientos.	
Responsabilidades	Colaboradores

Mostrar las casas guardadas en la base de datos	casa.modelo.ts
Mostrar las casas pertenecientes al usuario autenticado	casa.modelo.ts, usuario.ts
Actualizar los datos de una casa	casa.modelo.ts, usuario.ts
Eliminar casa	casa.modelo.ts, usuario.ts
Crear casa	casa.modelo.ts, usuario.ts

Tabla 10 Tarjeta CRC: RegistrousuarioService

Clase : RegistrousuarioService	
Descripción: Se encarga de gestionar el uso del microservicio usuario.	
Responsabilidades	Colaboradores
Mostrar los usuarios guardados en la base de datos	usuario.modelo.ts
Eliminar usuario	usuario.ts y user.ts
Actualizar los datos de un usuario	usuario.ts y user.ts
Crear usuario	usuario.ts

2.4.2. Arquitectura de software

La arquitectura del software de un programa o sistema de cómputo es la estructura o estructuras del sistema, lo que comprende a los componentes del software, sus propiedades externas visibles y las relaciones entre ellos. (Pressman 2010)

La arquitectura no es el software operativo. Es una representación que permite analizar la efectividad del diseño para cumplir los requerimientos establecidos, considerar alternativas arquitectónicas en una etapa en la que hacer cambios al diseño todavía es relativamente fácil y reducir los riesgos asociados con la

construcción del software. Esta definición pone el énfasis en el papel de los “componentes del software” en cualquier representación arquitectónica.

En el contexto del diseño de la arquitectura, un componente del software puede ser algo tan simple como un módulo de programa o una clase orientada a objeto, pero también puede ampliarse para que incluya bases de datos y “middleware” que permitan la configuración de una red de clientes y servidores. Las propiedades de los componentes son aquellas características necesarias para entender cómo interactúan unos componentes con otros.

En el nivel arquitectónico, no se especifican las propiedades internas (por ejemplo, detalles de un algoritmo). Las relaciones entre los componentes pueden ser tan simples como una invocación de procedimiento de un módulo a otro o tan complejos como un protocolo de acceso a una base de datos. (Pressman 2010)

Arquitecturas empleadas

Angular es una plataforma y un marco para crear aplicaciones de cliente en HTML y TypeScript. Implementa la funcionalidad central y opcional como un conjunto de bibliotecas de TypeScript que importa a sus aplicaciones.

Los bloques de construcción básicos de una aplicación Angular son NgModules, que proporcionan un contexto de compilación para los componentes. NgModules recopila códigos relacionados en conjuntos funcionales; una aplicación Angular se define por un conjunto de NgModules.

Una aplicación siempre tiene al menos un módulo raíz que habilita el arranque, y generalmente tiene muchos más módulos de funciones. Los componentes definen vistas, que son conjuntos de elementos de pantalla que Angular puede elegir y modificar según la lógica y los datos de su programa. Los componentes utilizan servicios, que proporcionan una funcionalidad específica que no está directamente relacionada con las vistas. Los proveedores de servicios se pueden inyectar en los componentes como dependencias, haciendo que su código sea modular, reutilizable y eficiente.

Tanto los componentes como los microservicios son simples clases, con decoradores que marcan su tipo y proporcionan metadatos que le indican a Angular cómo usarlos. Los metadatos de una clase de componente lo asocian con una plantilla que define una vista. Una plantilla combina HTML ordinario con directivas angulares y un marcado de enlace que permite a Angular modificar el HTML antes de representarlo. Los metadatos para una clase de servicio proporcionan la información que Angular necesita para que esté disponible para los componentes a través de la inyección de dependencia(DI).

Los componentes de una aplicación típicamente definen muchas vistas, ordenadas jerárquicamente. Angular proporciona el servicio de enrutador para ayudarlo a definir rutas de navegación entre las vistas. El enrutador proporciona sofisticadas capacidades de navegación en el navegador. El siguiente diagrama muestra cómo se relacionan estas piezas básicas.

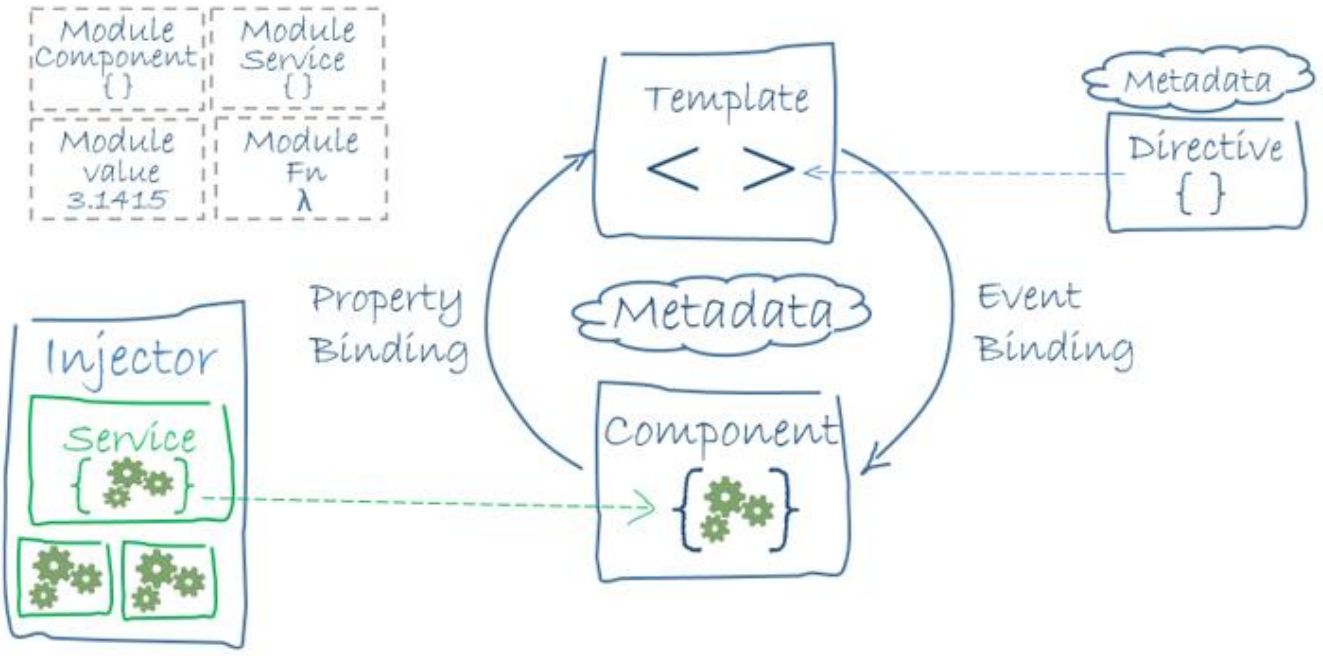


Figura 2 Arquitectura de Angular

Por otra parte, se utiliza otro tipo de arquitectura llamada **arquitectura de microservicios** el cual es un enfoque para el desarrollo de una aplicación única como un conjunto de pequeños servicios, cada uno ejecutándose en su propio proceso y mecanismos ligeros de comunicación, a menudo un recurso de una interfaz de programación de aplicaciones (API) sobre protocolo de transferencia de hipertexto (HTTP). (López 2017)

Estos servicios están construidos alrededor de las capacidades del negocio y con independencia de despliegue e implementación totalmente automatizada. Existe un mínimo de gestión centralizada de estos servicios, los que pueden estar escritos en lenguajes de programación diferentes y utilizar diferentes tecnologías de almacenamiento de datos. No existe una definición en concreto para microservicios, sin embargo, una aproximación que la realiza lo define como: “Pequeños servicios autónomos que trabajan juntos”. Una arquitectura de microservicios promueve el desarrollo y despliegue de aplicaciones

compuestas por unidades independientes, autónomas, modulares y autocontenidas, lo cual difiere de la forma tradicional o monolítico. (López 2017)

Para explicar la arquitectura de microservicios la figura 3 en la cual se define el acceso a los microservicios por parte del cliente, cada uno de los elementos de esta arquitectura se subdivide en pequeños servicios que le permiten al usuario interactuar con las distintas funcionalidades o peticiones, estos realizan conexiones con la base de datos para poder devolver el resultado de las distintas consultas.

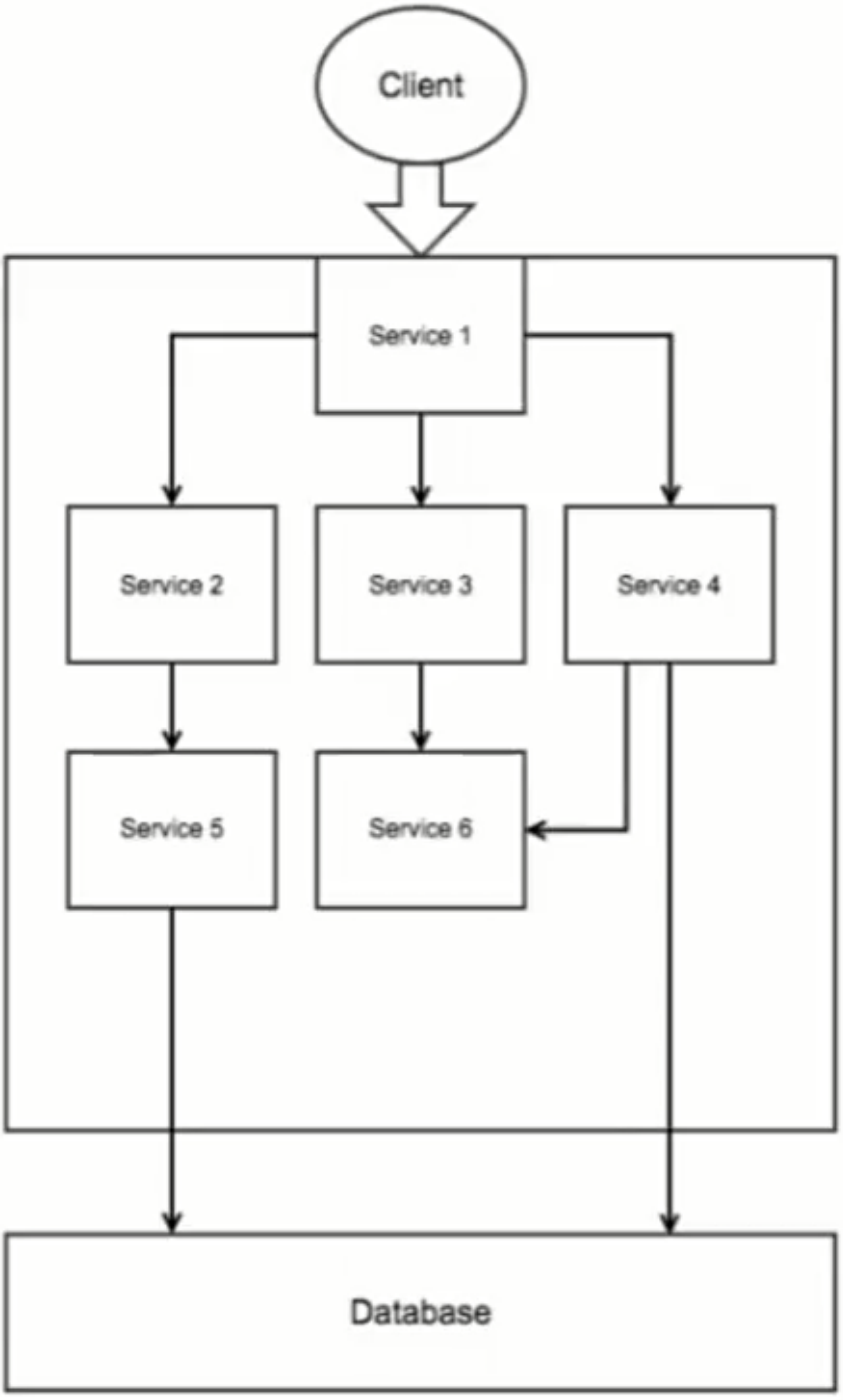


Figura 3 Arquitectura de Microservicios

2.5. Modelo de datos

Un modelo de datos es la representación abstracta de los datos en un sistema gestor de base de datos. Básicamente el modelo de datos está formado por tres elementos fundamentales que son: objetos (entidades que existen y se manipulan), atributos (Características básicas de estos objetos), relaciones (forma en que se enlazan los distintos objetos entre sí). A continuación, se muestra el modelo de datos que se obtuvo a partir de las clases generadas en las tarjetas CRC:

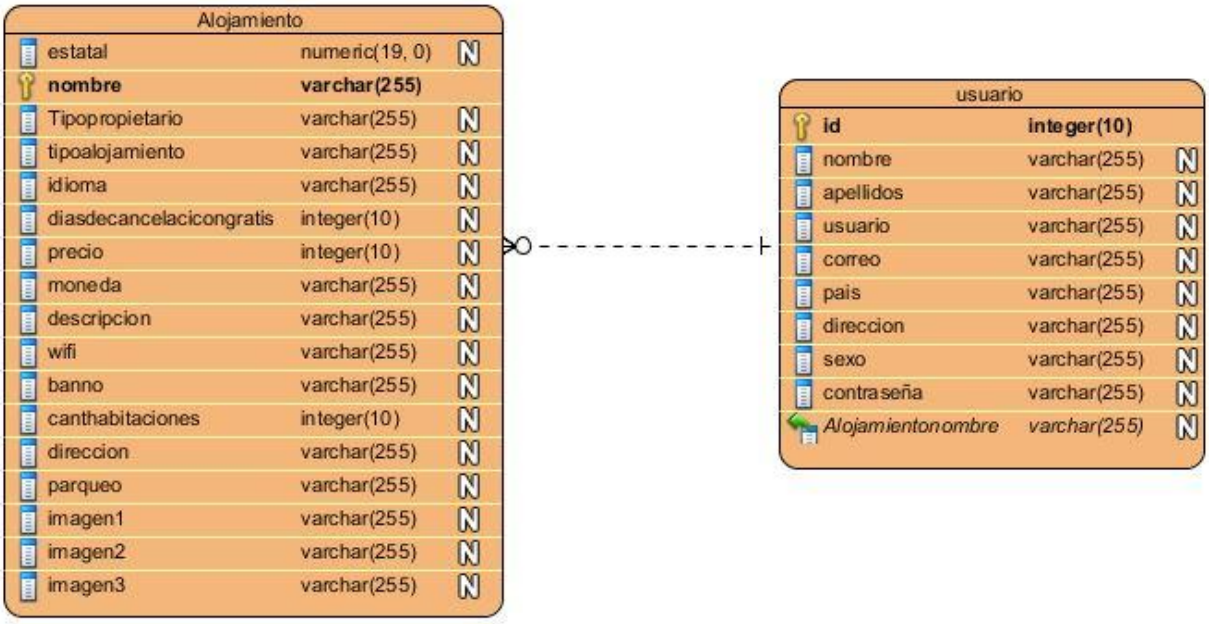


Figura 4 Modelo de Datos

2.6. Patrones de diseños empleados

Los patrones de diseño son considerados soluciones ya probadas a problemas de desarrollo de software sujeto a contextos similares. No se utilizan arbitrariamente, se debe tener en cuenta el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y los beneficios.

2.6.1. Patrones GRASP

En diseño orientado a objetos, GRASP son patrones generales de software para asignación de responsa bilidades, es el acrónimo de "General Responsibility Assignment Software Patterns". Aunque se consider a que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomen dable en el diseño de software. El objetivo fundamental de su utilización fue definir la clase con mayor

jerarquía e implementar en ella los métodos necesarios, además de asignar responsabilidades a las clases dependientes.

Experto: Este patrón se evidencia en las clases modelos del sistema, debido a que estas son las encargadas de realizar la abstracción de datos y consultas a base de datos, además de presentar todos los atributos necesarios para modelar el comportamiento de las entidades.

Creador: Se utilizó este patrón en las clases controladoras para trabajar con objetos instanciados en las clases que representan las entidades. Por ejemplo, en el componente registro.component.ts se implementa el método CreateForm para la creación de nuevos formularios, siendo el creador createForm.

Alta Cohesión: El uso de este patrón permitió que las tareas trabajen sobre los mismos objetos pero que tengan su codificación independiente y sus propias responsabilidades. Por ejemplo, unicacasa.component.ts y vistacasa.component.ts utilizan el objeto Casa, sin embargo, cada una tiene sus propias funciones e implementaciones, lo que permite que la complejidad sea lo más manejable posible y que se adapte mejor en casos de reutilización.

Controlador: Al utilizar Angular, se extiende el uso del patrón controlador, todas las clases que separan la lógica de negocio de la capa de presentación emplean este patrón, ejemplo de esta es app-routing.module.ts.

2.6.2. Patrones GoF

Los patrones GOF se clasifican en tres categorías: de **creación**, **estructurales** y de **comportamiento**.

1. Patrones **creacionales**: Abstraen el proceso de creación de instancias. Se observan en la inicialización y configuración de objetos. (Pavón 2011)

2. Patrones **estructurales**: Separan la interfaz de la implementación. Se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes. (Pavón 2011)

3. Patrones de **comportamiento**: Más que describir objetos o clases, describen la comunicación entre ellos y la asignación de responsabilidades. (Pavón 2011)

Algunos de los patrones de diseño GoF utilizados son los que se muestran a continuación:

Patrón command: Este patrón se aplica en la clase Routing, que está desactivada por defecto y procede según las necesidades del administrador del sistema donde se aplique el framework. En este método es

analizada la URL con el objetivo de precisar los parámetros de la misma y de esta forma saber la ruta que debe responder a la petición.

Patrón registry: Este patrón es muy útil para los desarrolladores en la Programación Orientada a Objetos. Es un medio sencillo y eficiente de compartir datos y objetos en la aplicación sin la necesidad de preocuparse por conservar numerosos parámetros o hacer uso de variables globales. Este patrón se aplica en la clase Searchalojamiento, que es la encargada de acumular todas las variables de uso global en el sistema.

2.7. Conclusiones Parciales

Con la elaboración de los artefactos que define la metodología XP, se garantizó una organización en el desarrollo del módulo y se logró simplificar el trabajo del implementador. Además, se describieron los patrones de diseño y estándares de codificación para realizar el correcto diseño del sistema. De esta manera quedaron definidas las pautas para la futura implementación.

Capítulo III. Implementación y prueba del sistema

El presente capítulo describe la implementación y prueba del sistema, lo cual forma parte de la fase de construcción del mismo para su futura entrega y correcto funcionamiento. La metodología XP tiene este proceso después de la fase de diseño, cuyo objetivo es el de efectuar análisis y exámenes en todo momento del desarrollo. Se realizan pruebas unitarias a la aplicación para presenciar su correcto funcionamiento y pruebas de aceptación para no desviarse del producto final esperado por el cliente.

3.1. Tareas de ingeniería

La metodología XP, define que es muy importante y necesario mostrar al cliente las especificaciones solicitadas, para esto se toman las historias de usuarios y se transforman en tareas de ingeniería para su fácil comprensión. Se muestran sus respectivos campos:

Número de la tarea: Los números deben ser consecutivos.

Número de HU: Número de la historia de usuario a la que pertenece la tarea.

Nombre Tarea: Nombre que identifica a la tarea.

Tipo de Tarea: Las tareas pueden ser de: Desarrollo, Corrección, Mejora, Otra (Especificar).

Puntos Estimados: Tiempo estimado en días que se le asignará a su desarrollo.

Fecha Inicio: Fecha en que se inicia el desarrollo de la tarea.

Fecha Fin: Fecha en que finaliza el desarrollo de la tarea.

Programador Responsable: Nombre y apellidos del programador.

Descripción: Breve descripción de la tarea.

Con estos campos se muestran las siguientes tareas de ingeniería:

Tabla 11 Tarea de ingeniería: Autenticación de usuario

Tarea de ingeniería	
Numero de tarea:1	Numero de Historia de Usuario:1

Nombre de la Tarea: Autenticación de usuario	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:13/12/2018	Fecha Fin:21/12/2018
Descripción: Cuando el usuario desea poseer permisos como usuario del sistema el mismo necesita a través del botón login, comenzar a llenar los campos usuario y contraseña y si los mismos están bien es enviado a sus alojamientos.	

Tabla 12 Tarea de ingeniería: Crear usuario

Tarea de ingeniería	
Numero de tarea:2	Numero de Historia de Usuario:2
Nombre de la Tarea: Crear usuario	
Tipo de Tarea : Desarrollo	Puntos estimados:0.1
Fecha Inicio:22/12/2018	Fecha Fin:24/12/2018
Descripción: Cuando el usuario desea poseer permisos como usuario del sistema el mismo si no posee cuenta a través del botón registrarse, puede comenzar a llenar los campos solicitados y si los mismos están bien es enviado a sus alojamientos.	

Tabla 13 Tarea de ingeniería: Actualizar usuario

Tarea de ingeniería	
Numero de tarea:3	Numero de Historia de Usuario:3
Nombre de la Tarea: Actualizar usuario	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2

Fecha Inicio:25/12/2018	Fecha Fin:30/12/2018
Descripción: Cuando el usuario autenticado desea cambiar datos de su perfil a través de mi cuenta, puede comenzar a llenar los campos solicitados y si los mismos están bien es enviado a sus alojamientos.	

Tabla 14 Tarea de ingeniería: Eliminar usuario

Tarea de ingeniería	
Numero de tarea:4	Numero de Historia de Usuario:4
Nombre de la Tarea: eliminar usuario	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:31/12/2018	Fecha Fin:1/1/2019
Descripción: Cuando el usuario administrador desea eliminar el perfil de un usuario, en la lista de usuarios del sistema lo podrá hacer través del botón eliminar.	

Tabla 15 Tarea de ingeniería: Leer usuario

Tarea de ingeniería	
Numero de tarea:5	Numero de Historia de Usuario:5
Nombre de la Tarea: Leer usuario	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:2/1/2019	Fecha Fin:3/1/2019
Descripción: Cuando el usuario administrador o el usuario autenticado desea ver su perfil, debe acceder a la opción mi cuenta .	

Tabla 16 Tarea de ingeniería: Crear alojamiento

Tarea de ingeniería	
Numero de tarea:6	Numero de Historia de Usuario:5
Nombre de la Tarea: Crear alojamiento	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:4/1/2019	Fecha Fin:7/1/2019
Descripción: Cuando el usuario administrador o el usuario autenticado desea insertar un alojamiento, debe acceder a la ruta misalojamientos y presionar el botón adicionar alojamiento, que a su vez es direccionado a la ruta adicionaralojamientos, completar todos los espacios de textos y por ultimo si todos los parámetros están correctos presionar registrar alojamiento .	

Tabla 17 Tarea de ingeniería: Actualizar alojamiento

Tarea de ingeniería	
Numero de tarea:7	Numero de Historia de Usuario:3
Nombre de la Tarea: Actualizar alojamiento	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:8/1/2019	Fecha Fin:12/1/2019
Descripción: Cuando el usuario autenticado desea cambiar datos de su alojamiento, puede comenzar a llenar los campos solicitados, oprimiendo el botón editar, completar los espacios de textos y presionar el botón actualizar.	

Tabla 18 Tarea de ingeniería: Eliminar alojamiento

Tarea de ingeniería	
Numero de tarea:8	Numero de Historia de Usuario:4
Nombre de la Tarea: eliminar alojamiento	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:13/1/2019	Fecha Fin:15/1/2019
Descripción: Cuando el usuario autenticado desea eliminar su alojamiento, en la lista de sus alojamientos lo podrá hacer través del botón eliminar.	

Tabla 19 Tarea de ingeniería: Leer alojamiento

Tarea de ingeniería	
Numero de tarea:9	Numero de Historia de Usuario:5
Nombre de la Tarea: Leer alojamiento	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:15/1/2019	Fecha Fin:16/1/2019
Descripción: Cuando el usuario autenticado desea ver sus alojamientos, debe acceder a la ruta /misalojamientos.	

Tabla 20 Tarea de ingeniería: Buscar alojamiento

Tarea de ingeniería	
Numero de tarea:10	Numero de Historia de Usuario:5
Nombre de la Tarea: Buscar alojamiento	

Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:17/1/2019	Fecha Fin:20/1/2019
Descripción: Cuando el usuario desee buscar algún alojamiento lo podrá realizar solo llenando el campo que se encuentra en la parte superior que se titula buscar alojamiento.	

Tabla 21 Tarea de ingeniería: Insertar imagen

Tarea de ingeniería	
Numero de tarea:11	Numero de Historia de Usuario:5
Nombre de la Tarea: Insertar Imagen	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:21/1/2019	Fecha Fin:23/1/2019
Descripción: Cuando el usuario autenticado desea insertar una imagen de su alojamiento, debe acceder a la ruta /adicionaralojamientos y en el botón insertar imagen buscar su imagen, completar todos los espacios de textos y por ultimo si todos los parámetros están correctos presionar registrar alojamiento .	

Tabla 22 Tarea de ingeniería: Actualizar imagen

Tarea de ingeniería	
Numero de tarea:12	Numero de Historia de Usuario:5
Nombre de la Tarea: Actualizar Imagen	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:24/1/2019	Fecha Fin:26/1/2019

Descripción: Cuando el usuario autenticado desea actualizar una imagen de su alojamiento, debe acceder a la ruta /misalojamientos y en el botón editar alojamiento buscar su imagen, completar todos los espacios de textos y por ultimo si todos los parámetros están correctos presionar guardar cambios .

Tabla 23 Tarea de ingeniería: Eliminar imagen

Tarea de ingeniería	
Numero de tarea:13	Numero de Historia de Usuario:5
Nombre de la Tarea: Eliminar Imagen	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:27/1/2019	Fecha Fin:29/1/2019
Descripción: Cuando el usuario autenticado desea eliminar una imagen de su alojamiento, debe acceder a la ruta /misalojamientos y en el botón editar alojamiento no buscar su imagen, completar todos los espacios de textos y por ultimo si todos los parámetros están correctos presionar guardar cambios .	

Tabla 24 Tarea de ingeniería: Leer imagen

Tarea de ingeniería	
Numero de tarea:14	Numero de Historia de Usuario:5
Nombre de la Tarea: Leer Imagen	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:3/2/2019	Fecha Fin:4/2/2019
Descripción: Cuando el usuario autenticado desea ver imágenes de su alojamiento, debe acceder a la ruta /misalojamientos y en el botón detalles accede a todas las imágenes del alojamiento .	

Tabla 25 Tarea de ingeniería: Mostrar ayuda

Tarea de ingeniería	
Numero de tarea:14	Numero de Historia de Usuario:5
Nombre de la Tarea: Mostrar Ayuda	
Tipo de Tarea : Desarrollo	Puntos estimados:0.2
Fecha Inicio:4/2/2019	Fecha Fin:7/2/2019
Descripción: Cuando el usuario autenticado desea aprender o entender de su alojamiento, debe acceder a la ruta /misalojamientos y en el botón detalles accede a todas las imágenes del alojamiento .	

3.2. Estrategia de Pruebas

Las pruebas de software son procesos que permiten verificar y revelar la calidad de un producto. Son utilizadas para identificar posibles fallos de implementación, calidad, o usabilidad de un programa. (Pressman 2010).

La creación de pruebas unitarias antes de que comience la codificación es un elemento clave del enfoque de XP. Las pruebas unitarias que se crean deben implementarse con el uso de una estructura que permita automatizarlas (de modo que puedan ejecutarse en repetidas veces y con facilidad). (Pressman 2010).

Las pruebas de aceptación XP, también llamadas pruebas del cliente, son especificadas por el cliente y se centran en las características y funcionalidad generales del sistema que son visibles y revisables por parte del cliente. Las pruebas de aceptación se derivan de las historias de los usuarios que se han implementado.

3.2.1. Pruebas Unitarias

La creación de pruebas unitarias antes de que comience la codificación es un elemento clave del enfoque de XP. Las pruebas unitarias que se crean deben implementarse con el uso de una estructura que permita automatizarlas (de modo que puedan ejecutarse en repetidas veces y con facilidad). (Pressman 2010)

Las pruebas unitarias son realizadas a pequeñas porciones de código, por separados, para verificar su correcta funcionalidad, las mismas se pueden ir efectuando desde el comienzo de la implementación, no

necesariamente se tiene que esperar al finalizar el software. En las siguientes imágenes se muestran las pruebas unitarias por al finalizar cada iteración.

Primera Iteración

La siguiente imagen muestra una prueba unitaria basada en el test de angular, la cual muestra errores con la propiedad map del servicio alojamientos:

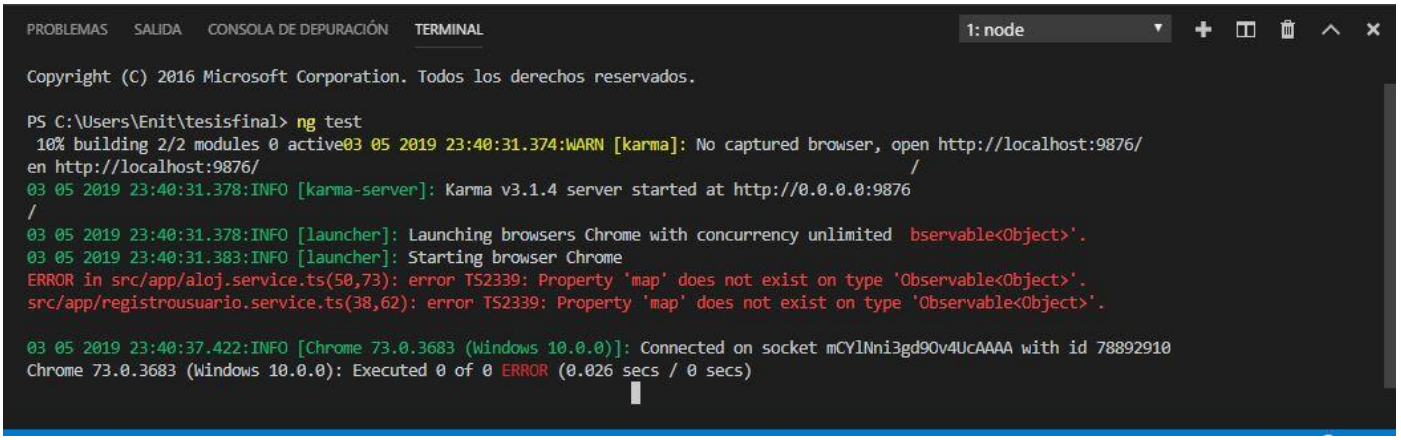


Figura 5 Prueba Unitaria 1ra Iteración

Después de haberse realizado la rectificación de errores observamos los resultados:

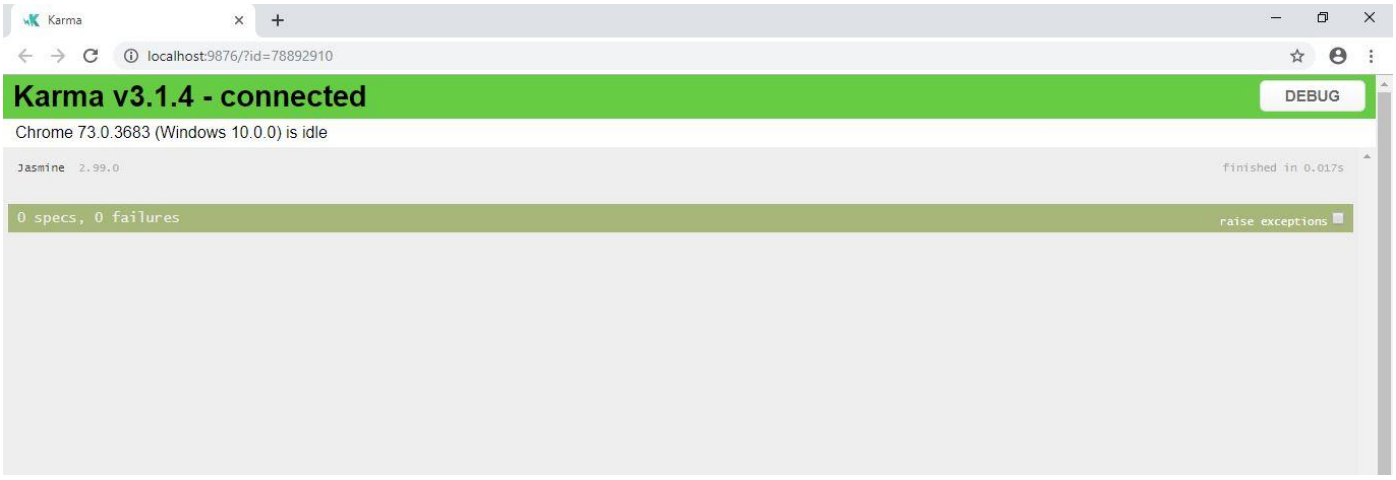


Figura 6 Prueba Unitaria 1ra Iteración

Segunda Iteración:

La siguiente imagen muestra una prueba unitaria basada en el test de angular, la cual no muestra errores:

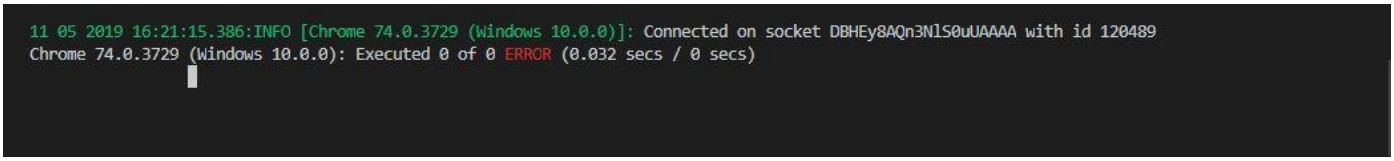


Figura 7 Prueba Unitaria 2da Iteración

Se muestran resultados de Karma:

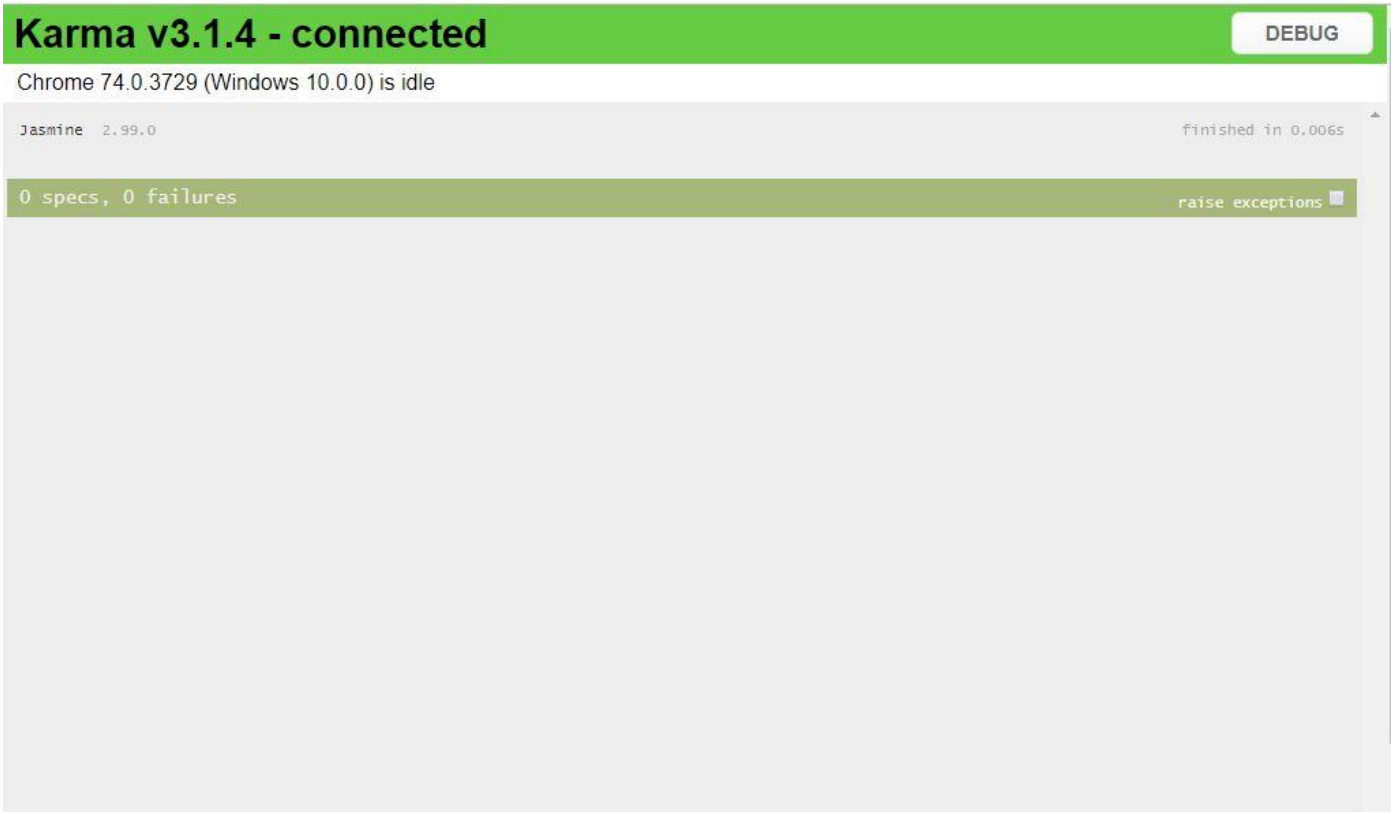


Figura 8 Prueba Unitaria 2da Iteración

Tercera Iteración:

La siguiente imagen muestra una prueba unitaria basada en el test de angular, la cual muestra errores tras intentar relacionar el servicio de alojamientos con el de las imágenes en el *frontend*:

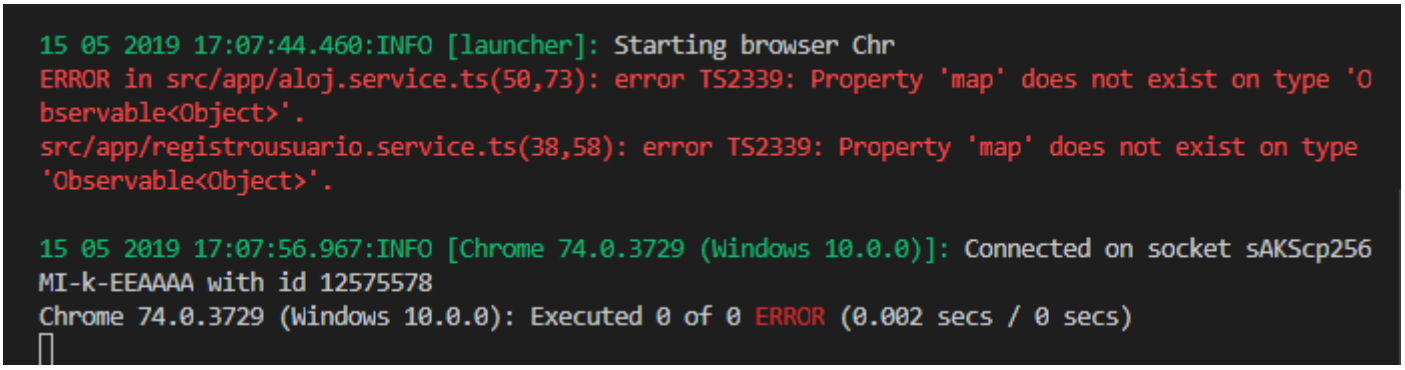


Figura 9 Prueba Unitaria 3ra Iteración

Se muestran resultados luego arreglado de Karma:

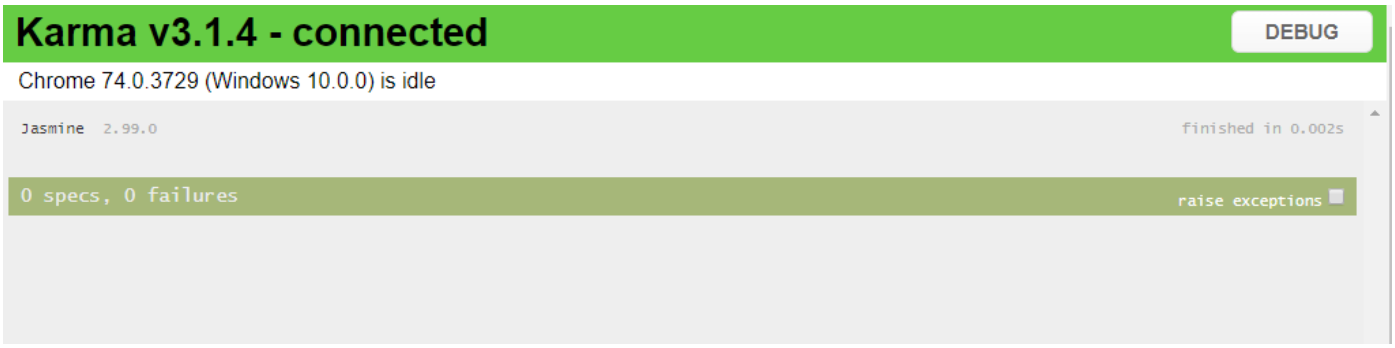


Figura 10 Prueba Unitaria 3ra Iteración

3.2.2. Pruebas de aceptación

Las pruebas de aceptación son creadas en base a las Historias de Usuarios, en cada ciclo de la iteración del desarrollo. Las pruebas de aceptación XP, también llamadas pruebas del cliente, son especificadas por el cliente y se centran en las características y funcionalidad generales del sistema que son visibles y revisables por parte del mismo. Por lo tanto, son los clientes los responsables de verificar que los datos de estas pruebas sean correctos.

Tabla 26 Prueba de Aceptación: Autenticar usuario

Caso de prueba de Aceptación	
Código: HU1_CP1	**Historia de Usuario: 1**
Nombre: Autenticar Usuario.	
Descripción: Prueba para la funcionalidad autenticar usuario.	
Condiciones de Ejecución: El usuario debe acceder a la aplicación. No insertar en el campo usuario números ni mayúsculas. No insertar en el campo contraseña menos de 8 caracteres.	
Entrada/Pasos de Ejecución: El usuario debe presionar el botón MiCuenta y en el submenú indicar la opción login donde deberá completar los campos correspondientes.	
Resultado Esperado: El usuario insertó los datos correctos (usuario: string y contraseña: string).	
Evaluación de Prueba: Satisfactoria	

Tabla 27 Prueba de Aceptación: Gestionar usuario

Caso de prueba de Aceptación	
Código: HU2_CP1	**Historia de Usuario: 2**
Nombre: Gestionar Usuario.	
Descripción: Prueba para la funcionalidad gestionar usuario.	
Condiciones de Ejecución: El administrador debe acceder a la aplicación y encontrarse autenticado. El cliente debe acceder a la aplicación y encontrarse autenticado. Cumplir con los tipos de datos correspondientes.	
Entrada/Pasos de Ejecución: El administrador debe presionar el botón Inicio e indicar la opción que desee realizar eliminación. En caso del cliente debe acceder al botón mi cuenta y luego podrá gestionar su cuenta. (id: number, nombre: string, apellidos: string, usuario: string, correo: string, país: string, dirección: string, sexo: string, contraseña: string, token: string)	

Resultado Esperado: El usuario insertó los datos correctos. El administrador insertó los datos correctos.
Evaluación de Prueba: Satisfactoria

Tabla 28 Prueba de Aceptación: Gestionar alojamiento

Caso de prueba de Aceptación	
Código: HU3_CP1	**Historia de Usuario: 3**
Nombre: Gestionar Alojamiento.	
Descripción: Prueba para la funcionalidad gestionar alojamiento.	
Condiciones de Ejecución: El administrador debe acceder a la aplicación y encontrarse autenticado. El cliente debe acceder a la aplicación y encontrarse autenticado.	
Entrada/Pasos de Ejecución: El administrador debe acceder a la ruta misalojamientos e indicar la opción que desee realizar. En caso del cliente debe acceder a la ruta misalojamientos y luego podrá gestionar sus alojamientos. (id: number, nombre: string, tipopropietario: string, tipoalojamiento: string, idioma: string, diasdecancelaciongratis: number, precio: number, moneda: string, descripción: string, wifi: string, baño: string, canthabitaciones: number, dirección: string, parqueo: string, imagen1: string, imagen2: string, imagen3: string, idUsuario: number)	
Resultado Esperado: El usuario insertó los datos correctos. El administrador insertó los datos correctos.	
Evaluación de Prueba: Satisfactoria	

Tabla 29 Prueba de Aceptación: Buscar alojamiento

Caso de prueba de Aceptación	
Código: HU4_CP1	**Historia de Usuario: 4**
Nombre: Buscar Alojamiento.	

Descripción: Prueba para la funcionalidad buscar alojamiento.
Condiciones de Ejecución: El usuario online debe acceder a la aplicación
Entrada/Pasos de Ejecución: Insertar el nombre del alojamiento que desea buscar.
Resultado Esperado: El usuario online insertó los datos correctos.
Evaluación de Prueba: Satisfactoria

Tabla 30 Prueba de Aceptación: Gestionar imagen

Caso de prueba de Aceptación	
Código: HU5_CP1	**Historia de Usuario: 5**
Nombre: Gestionar imagen.	
Descripción: Prueba para la funcionalidad gestionar imagen.	
Condiciones de Ejecución: El usuario debe acceder a la aplicación y encontrarse autenticado.	
Entrada/Pasos de Ejecución: El administrador debe acceder a la ruta misalojamientos e indicar la opción que desee realizar. En caso del cliente debe acceder a la ruta misalojamientos y luego podrá gestionar sus imágenes.	
Resultado Esperado: El usuario online insertó los datos correctos.	
Evaluación de Prueba: Satisfactoria	

Tabla 31 Prueba de Aceptación: Mostrar ayuda

Caso de prueba de Aceptación	
Código: HU6_CP1	**Historia de Usuario: 6**
Nombre: Mostrar Ayuda	

Descripción: Prueba para la funcionalidad mostrar ayuda.
Condiciones de Ejecución: El usuario debe acceder a la aplicación.
Entrada/Pasos de Ejecución: El administrador debe acceder al botón Ayuda.
Resultado Esperado: El usuario online obtuvo los datos correctos.
Evaluación de Prueba: Satisfactoria

Luego de aplicar las distintas pruebas, se pudo comprobar que el flujo de trabajo de la función es correcto ya que los resultados obtenidos son equivalentes a los esperados. Durante las pruebas unitarias y de aceptación se obtuvieron algunas no conformidades la cuales fueron resultas en cada una de las iteraciones. A continuación, se muestran las distintas no conformidades por tipo de prueba e iteraciones.

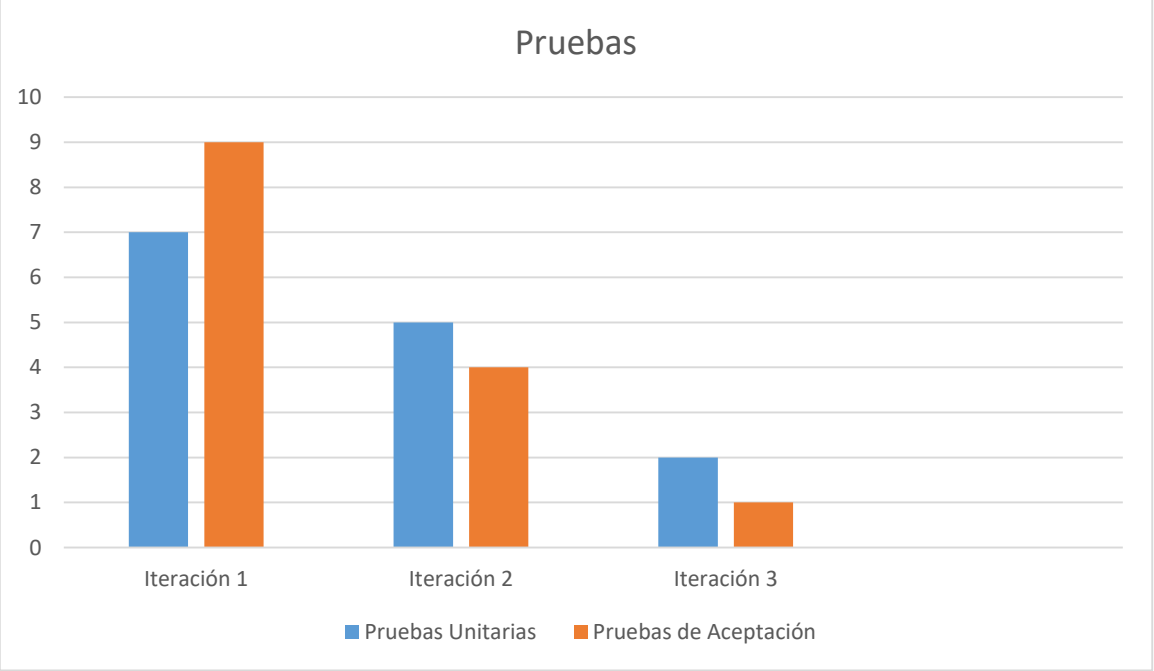


Figura 11 No conformidades encontradas por iteración.

3.3. Conclusiones Parciales

En el presente capítulo se realizaron las pruebas a la aplicación AlojamientosXetid tomando en cuenta las fases implementación y prueba de la metodología XP. Con el desarrollo de las funcionalidades del sistema fueron realizadas las pruebas correspondientes. Las no conformidades encontradas fueron solucionadas en las mismas iteraciones, por ello todas las tareas de ingenierías fueron reflejadas para la mejor comprensión del cliente. De esta manera se obtuvo una aplicación web con la calidad y las funcionalidades solicitadas por el cliente.

Conclusiones Generales

Una vez desarrollada la presente investigación, cumplidas las tareas de investigación y el objetivo general, se obtienen las siguientes conclusiones:

- El desarrollo del marco teórico permitió revisar los antecedentes relacionados al tema de la investigación relacionado con la gestión de la contratación y los principales conceptos asociados.
- Con el análisis de las aplicaciones web Homestay, Airbnb.apk, Hospedaje Cubano y Booking que gestionan la contratación de alojamientos se obtuvienron las características para tenerse en cuenta en el sistema desarrollado.
- Se obtuvo una caracterización del proceso de desarrollo basado en microservicios donde se profundizó en su modo de implementación, características y posibles resultados.
- El empleo de la metodología XP, tecnologías, lenguajes y herramientas HTML, CSS, JavaScript, Postgres, Visual Paradigm permitió la realización de la implementación.
- Con el empleo de los patrones arquitectónicos microservicios y modelo-vista de Angular, patrones de diseño Grasp y Gof, se realizó el correcto diseño de la aplicación.
- Se validó el sistema haciendo uso de las pruebas unitarias y de aceptación usando KarmaJs y Jasmine, comprobándose la funcionalidad del código. Además, se constató que el resultado es el esperados por el cliente.

Recomendaciones

Se recomienda para posibles investigaciones:

Desarrollar un módulo para conocer la disponibilidad de los alojamientos contratados.

Referencias Bibliográficas

FERNÁNDEZ SANZ, L. y BERNAD SILVA, P., 2014. Gestión de riesgos en proyectos de desarrollo de software en España: estudio de la situación. *Revista Facultad de Ingeniería Universidad de Antioquia*, no. 70.

El bloqueo impide mayor y mejor acceso de Cuba a Internet. *Cubadebate* [en línea], 2015. [Consulta: 25 mayo 2019]. Disponible en: http://www.cubadebate.cu/noticias/2015/10/09/el-bloqueo-impide-mayor-y-mejor-acceso-de-cuba-a-internet/.

ISO 9000:2000 Sistemas de Gestión de la Calidad. Conceptos y Vocabulario., 2000. pp. 42.

PRESSMAN, R.S. y CONTRERAS, T.B., 2010. Desarrollo ágil. *Pablo Roig Vásquez. Ingeniería del software. Ciudad de México: McGrawHill*, pp. 69.

Luis Calabria, Pablo Piriz. 2003. Metodología XP. Uruguay: s.n., 2003.

VIDAL LEDO, M.J. y ARAÑA PÉREZ, A.B., 2012. Gestión de la información y el conocimiento. *Educación Médica Superior*, vol. 26, no. 3, pp. 474-484. ISSN 0864-2141.

Diccionario Ilustrado, Editorial Sopena Argentina, Argentina, 2010

¿Qué es Homestay.com? | Homestay.com Help. [en línea], [sin fecha]. [Consulta: 9 mayo 2019]. Disponible en: https://www.homestay.com/es/help/article/205363681-que-es-homestay-com.

RODRÍGUEZ, Y.R., KILE, S.B.F. y ROMERO, A.E.R., 2013. Aplicación de la estrella de Boehm y Turner al proyecto laboratorios virtuales. *UCV-HACER. Revista de Investigación y Cultura*, vol. 2, no. 1, pp. 20–29.

EGUILUZ, J., [sin fecha]. Introducción a CSS., pp. 241.

VERDECIA-ROSALES, A., 2018. TENDENCIAS DEL CONSUMIDOR DIGITAL PARA EL PRODUCTO TURÍSTICO. *Redel. Revista granmense de Desarrollo Local*, vol. 2, no. 3.

MÁRQUEZ LOBILLO, P. (2011). “El consumidor en la contratación electrónica servicios urísticos”RDM,209-242.

SOMMERVILLE, I., 2011. Ingeniería del software. S.l.: Pearson Educación.

Javier Eguiluz Pérez, Introducción a CSS, Versión Impresa 2008 - Buscar con Google. [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

Javier Eguiluz Pérez, Introducción a JavaScript, Versión Impresa 2009 - Buscar con Google. [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

Javier Eguiluz Pérez, Introducción a XHTML - Buscar con Google. [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

Larman, Craig. UML y Patrones - Una introducción al análisis y diseño orientado a objetos y el proceso unificado. s.l.:

Pablo F. Burgueño, 11 de junio de 2010 https://www.pablofb.com/la-contratacion-electronica-en-el-ordenamiento-juridico-español - Buscar con Google.

Convelia (Consultores legales tecnológicos) https://www.convelia.com/contrato-electronico-definicion-y-validez - Buscar con Google. [en línea],

Penadés, Patricio Letelier. 2010. Metodologías ágiles para el desarrollo de software: Extrema Programming. Valencia: s.n., 2010.

PAVÓN MESTRAS, Juan. Estructura de las Aplicaciones Orientadas a Objetos. El patrón ModeloVista-Controlador (MVC). Madrid: Departamento de Informática. Universidad Complutense de Madrid., 2011.

Bibliografía

Metodologías Ágiles. ¿Cómo desarrollo utilizando XP? Ramírez, Danay Pérez, y otros. La Habana: s.n., 2008, CUJAE.

Ingeniería de Software, 9na Edición, Ian Sommerville, Pearson Educación, México, 2011

Sánchez, N., El marco lógico. Metodología para la planificación, seguimiento y evaluación de proyectos, Visión gerencial: 6 (2), 328-343 (2007)

FERNÁNDEZ SANZ, L. y BERNAD SILVA, P., 2014. Gestión de riesgos en proyectos de desarrollo de software en España: estudio de la situación. *Revista Facultad de Ingeniería Universidad de Antioquia*, no. 70.

PRESSMAN, R.S. y CONTRERAS, T.B., 2010. Desarrollo ágil. *Pablo Roig Vásquez. Ingeniería del software. Ciudad de México: McGraw-Hill*, pp. 69.

PRESSMAN, R., 2005. Agile Development. *Software engineering, a practitioner's approach,* pp. 85–87.

RODRÍGUEZ, Y.R., KILE, S.B.F. y ROMERO, A.E.R., 2013. Aplicación de la estrella de Boehm y Turner al proyecto laboratorios virtuales. *UCV-HACER. Revista de Investigación y Cultura*, vol. 2, no. 1, pp. 20–29.

Aplicaciones móviles cubanas: ¿Por dónde van los pasos? (+ Fotos, Video e Infografía) | Cubadebate. [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

Comercio electrónico. *Cubadebate* [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

Comercio electrónico: Un servicio on-line en Cuba. *Cubadebate* [en línea], 2017. [Consulta: 10 mayo 2019].

Conoce Cuba | Directorio. [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

Conoce Cuba | Directorio de lugares y servicios. [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

Definición de modelo de datos - Qué es, Significado y Concepto. [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

El Confidencial Digital - elconfidencialdigital. [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

Guía Excelencias Cuba 2019. [en línea], [sin fecha]. [Consulta: 10 mayo 2019]. Disponible en: http://guiaexcelenciascuba.com/.

La evolución de la Web. [en línea], [sin fecha]. [Consulta: 10 mayo 2019]. Disponible en: http://www.evolutionoftheweb.com/?hl=es.

Metodologías Ágiles de desarrollo de software (XP) Fases. [en línea], [sin fecha]. [Consulta: 10 mayo 2019]. Disponible en: http://boards5.melodysoft.com/UBV_INGS/metodologias-agiles-de-desarrollo-43.html.

Patrones de GRASP - Adictos al trabajo. [en línea], [sin fecha]. [Consulta: 10 mayo 2019]. Disponible en: https://www.adictosaltrabajo.com/2003/12/22/grasp/.

Pruebas unitarias y de aceptación con metodología xp - Buscar con Google. [en línea], [sin fecha]. [Consulta: 10 mayo 2019].

¿Qué son los microservicios? [en línea], [sin fecha]. [Consulta: 10 mayo 2019]. Disponible en: https://www.redhat.com/es/topics/microservices.

QUINTERO, J.A.M. y REYES, D.F.N., [sin fecha]. Proceso de Pruebas Unitarias Bajo Entornos de Desarrollo Ágiles: Un Estudio Sistemático. , pp. 146.

Quintero y Reyes - Proceso de Pruebas Unitarias Bajo Entornos de Desa.pdf [en línea], [sin fecha]. S.l.: s.n. [Consulta: 10 mayo 2019].

VIZZUALITY, G., Hyperakt, [sin fecha]. Infografía: la evolución de los navegadores y la Web. [en línea]. [Consulta: 10 mayo 2019]. Disponible en: http://www.evolutionofweb.com.

HASSAN, S., BAHSOON, R. y KAZMAN, R., 2019. Microservice Transition and its Granularity Problem: A Systematic Mapping Study. En: arXiv: 1903.11665, *arXiv:1903.11665 [cs]* [en línea], [Consulta: 15 mayo 2019]. Disponible en: http://arxiv.org/abs/1903.11665.

WIZENTY, P., SORGALLA, J., RADEMACHER, F. y SACHWEH, S., 2017. MAGMA: build management-based generation of microservice infrastructures. *Proceedings of the 11th European Conference on Software Architecture: Companion Proceedings*. S.l.: ACM, pp. 61–65.

LIJPHART, A., 1975. *The politics of accommodation: Pluralism and democracy in the Netherlands*. S.l.: Univ of California Press.