



Universidad de las Ciencias Informáticas

Facultad 2

Plataforma Nacional de Consulta de Medicamentos

Trabajo de Diploma para optar por el título

de Ingeniero en Ciencias Informáticas

Autores:

Nelson Alejandro Cruz Borrego

Denis Duranza Díaz

Tutores:

Msc. Madelis Pérez Gil

Ing. Darianis Pérez Aguada

La Habana, noviembre de 2023

“Año 65 de la Revolución”

Agradecimientos

Nelson Alejandro Cruz Borrego

*Quisiera agradecer, primeramente, a mi **familia**, en especial a mi madre, mi padre, mi hermana y mi abuela que siempre me apoyan incondicionalmente cuando los necesito y me ofrecen sus valiosos consejos. A mis **tutoras** Darianis y Madelis que me han ayudado mucho durante todo este proceso de investigación y me han brindado sus conocimientos técnicos y experiencia para el mejor desarrollo de este trabajo. A mis **compañeros** de apartamento, con los que he compartido muchos momentos juntos que son muy preciados para mí. Además, quiero agradecer a todos los **amigos** que he conocido estos años estudiando en la universidad. Quiero agradecer también a todos y cada uno de los **profesores** de los cuales aprendí mucho no solo de las asignaturas que cada uno impartía sino también de la vida en general. A todo aquel que haya tenido que ver con la realización de este Trabajo de Diploma que culmina cinco años de arduo estudio y dedicación.*

Denis Duranza Diaz

*En primer lugar, a mi **compañero de tesis**, la mitad de este logro es de él, a mis **dos tutoras** Darianis y Madelis que desde el minuto uno se preocuparon y ocuparon para que las cosas salieran con el nivel que requería. A mis **amistades**, tanto las que hoy siguen aquí conmigo como los que no se encuentran tan cerca. A mi **hermana** y mi **cuñado** que fueron un importante apoyo en todo este tiempo de estudios. Por último y más importante a mi **madre** y a mi **padre** que siempre han estado ahí para mí sin importar si mis decisiones eran las correctas o no. Muchas Gracias.*

“El futuro de nuestra Patria tiene que ser necesariamente un futuro de hombres de ciencia, tiene que ser un futuro de hombres de pensamiento, porque precisamente es lo que estamos sembrando.”

Fidel Castro

Declaración de autoría

Declaración de autoría

Declaro por este medio que yo, con carné de identidad soy el autor principal del trabajo titulado “Plataforma Nacional de Consulta de Medicamentos” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los 13 días del mes de noviembre de 2023.

Nelson Alejandro Cruz Borrego

Autor

MSc. Madelis Pérez Gil

Tutor

Denis Duranza Díaz

Autor

Ing. Darianis Pérez Aguada

Tutor

Resumen

En la sociedad actual, se enfrentan desafíos significativos en la gestión de la información, especialmente en lo que respecta a la disponibilidad de medicamentos en las farmacias. Este problema es particularmente prevalente en Cuba, donde la falta de información accesible puede dificultar la adquisición oportuna de medicamentos esenciales. En respuesta a esta problemática, se propone el desarrollo de un sistema informático denominado Plataforma Nacional de Consulta de Medicamentos (PNCM). Esta plataforma tiene como objetivo principal digitalizar y centralizar la información relacionada con la disponibilidad de medicamentos, proporcionando así una solución eficaz a este problema persistente. Para el desarrollo de este sistema, se optó por utilizar la metodología de desarrollo de software XP, reconocida por su eficacia en proyectos que requieren flexibilidad y adaptabilidad. En cuanto a la tecnología, se seleccionó Express como marco de trabajo debido a su robustez y versatilidad. Además, se utilizó MongoDB como sistema gestor de bases de datos, dada su capacidad para manejar grandes volúmenes de datos y su compatibilidad con JavaScript, el lenguaje de programación que se eligió para este proyecto. Finalmente, se incorporaron tecnologías adicionales y modernas como React.js y Node.js para optimizar la funcionalidad y la experiencia del usuario del sistema.

Palabras Clave: farmacia, medicamentos, plataforma, servicio farmacéutico

Abstract

In today's society, significant challenges are faced in information management, especially with regard to the availability of medications in pharmacies. This problem is particularly prevalent in Cuba, where the lack of accessible information can difficult the timely acquisition of essential medicines. In response to this problem, the development of a computer system called National Medication Consultation Platform (NMCP) has been proposed. The main objective of this system is to digitize and centralize information related to the availability of medicines, providing of this way an effective solution to this persistent problem. For the development of this system, we decided to use the XP software development methodology, recognized for its effectiveness in projects that require flexibility and adaptability. In terms of technology, we selected Express as the framework due to its robustness and versatility. In addition, MongoDB was used as the database management system, given its ability to handle large volumes of data and its compatibility with JavaScript, the programming language chosen for this project. Finally, additional and modern technologies such as React.js and Node.js were used to optimize the functionality and user experience of the system.

Keywords: *medicines, pharmacy, pharmaceutical service, platform*

Índice general

Introducción	1
Capítulo 1: Fundamentación teórica.....	6
1.1. Introducción.....	6
1.2. Conceptos fundamentales	6
1.3. Estudio del Estado del Arte.....	7
1.3.1. Sistemas informáticos similares en el mundo	7
1.3.2. Sistemas informáticos similares en Cuba	9
1.3.3. Conclusiones del Estudio del Estado del Arte	9
1.4. Metodología de desarrollo de software	10
1.4.1. Programación Extrema.....	13
1.5. Tecnologías para el desarrollo de la propuesta de solución	14
1.5.1. Tecnologías utilizadas en el Frontend	14
1.5.2. Tecnologías utilizadas en el Backend	15
1.5.3. Lenguaje de programación.....	16
1.5.4. Herramienta CASE para el modelado del sistema Visual Paradigm for UML 16.2....	17
1.6. Conclusiones parciales.....	17
Capítulo 2: Análisis y diseño	19
2.1. Introducción	19
2.2. Proceso de consulta de medicamentos en las farmacias cubanas	19
2.3. Exploración.....	20
2.3.1. Historias de usuario	20
2.4. Planificación	29
2.4.1 Estimación del esfuerzo	29
2.4.2. Plan de iteraciones.....	30
2.4.3. Plan de entregas	32
2.5. Diseño del sistema	32
2.5.1. Patrón arquitectónico	33
2.5.2. Patrones de diseño	33
2.5.3. Tarjetas CRC.....	37
2.5.4. Modelo de datos.....	39
2.6. Conclusiones parciales.....	40
Capítulo 3: Implementación y pruebas	41
3.1. Introducción	41
3.2. Implementación	41
3.2.1. Diagrama de despliegue	41

Índice general

3.2.2. Estándares de codificación	42
3.3. Desarrollo por iteraciones	44
3.3.1. Iteración 1	45
3.3.2. Iteración 2	48
3.3.3. Iteración 3	51
3.3.4. Iteración 4	54
3.4. Conclusiones parciales	64
Conclusiones generales	65
Recomendaciones	66
Referencias bibliográficas	67
Anexos	70
Acta de Aceptación	70

Índice de figuras

Figura 2.1. Diagrama del proceso de consulta de medicamentos en las farmacias cubanas. ...	19
Figura 2.2. Prototipo de Historia de Usuario 1.....	22
Figura 2.3. Prototipo de Historia de Usuario 2.....	23
Figura 2.4. Prototipo de Historia de Usuario 3.....	24
Figura 2.5. Prototipo de Historia de Usuario 4.....	25
Figura 2.6. Prototipo de Historia de Usuario 5.....	26
Figura 2.7. Prototipo de Historia de Usuario 6.....	27
Figura 2.8. Prototipo de Historia de Usuario 7.....	28
Figura 2.9. Prototipo de Historia de Usuario 8.....	29
Figura 2.10. Evidencia del patrón Experto.....	34
Figura 2.11. Evidencia del patrón Creador.	35
Figura 2.12. Evidencia del patrón Bajo Acoplamiento.....	36
Figura 2.13. Evidencia del patrón Alta Cohesión.	36
Figura 2.14. Evidencia del patrón Observador.....	37
Figura 2.15. Modelo de datos.....	39
Figura 3.1. Diagrama de despliegue.	42
Figura 3.2. Uso de importaciones.....	43
Figura 3.3. Uso de la notación camelCase.....	43
Figura 3.4. Estándar para comentarios.	44
Figura 3.5. Test unitario 1 de la primera iteración.....	47
Figura 3.6. Test unitario 2 de la primera iteración.	48
Figura 3.7. Test unitario 1 de la segunda iteración.	51
Figura 3.8. Test unitario 2 de la segunda iteración.	51
Figura 3.9. Test unitario 1 de la tercera iteración.....	53
Figura 3.10. Test unitario 2 de la tercera iteración.....	54
Figura 3.11. Test unitario de la cuarta iteración.	57
Figura 4.1. No Conformidades por Iteración.....	63

Índice de tablas

Tabla 1.1. Comparación de Metodologías.	10
Tabla 1.2. Comparación de algunas metodologías ágiles.	12
Tabla 2.1. Historia de Usuario 1.	21
Tabla 2.2. Historia de usuario 2.	22
Tabla 2.3. Historia de usuario 3.	23
Tabla 2.4. Historia de usuario 4.	24
Tabla 2.5. Historia de usuario 5.	25
Tabla 2.6. Historia de usuario 6.	26
Tabla 2.7. Historia de usuario 7.	27
Tabla 2.8. Historia de usuario 8.	28
Tabla 2.9. Estimación de esfuerzo por historia de usuario.	30
Tabla 2.10. Plan de duración de las iteraciones.	31
Tabla 2.11. Plan de entregas.	32
Tabla 2.12. Tarjeta CRC 1.	38
Tabla 2.13. Tarjeta CRC 2.	38
Tabla 2.14. Tarjeta CRC 3.	39
Tabla 3.1. TI1: Buscar medicamentos.	45
Tabla 3.2. TI2: Buscar farmacias.	46
Tabla 3.3. TI3: Geolocalizar las farmacias del país.	48
Tabla 3.4. TI4: Mostrar información sobre las farmacias en el mapa.	49
Tabla 3.5. TI5: Geolocalizar al usuario.	50
Tabla 3.6. TI6: Resaltar las farmacias más cercanas a la posición del usuario.	52
Tabla 3.7. TI7: Calcular la ruta a la farmacia seleccionada.	52
Tabla 3.8. TI8: Suscripción del usuario.	54
Tabla 3.9. TI9: Modificar suscripción.	55
Tabla 3.10. TI10: Eliminar suscripción.	56
Tabla 4.1. CP01: Búsqueda.	58
Tabla 4.2. CP02: Geolocalizar Farmacias.	58
Tabla 4.3. CP03: Mostrar Información de las Farmacias en el Mapa.	59
Tabla 4.4. CP04: Geolocalizar al Usuario.	59
Tabla 4.5. CP05: Resaltar Farmacias Cercanas.	60
Tabla 4.6. CP06: Calcular Ruta a Farmacia.	60
Tabla 4.7. CP07: Suscripción a la PNCM.	61
Tabla 4.8. CP08: Modificar Suscripción.	61
Tabla 4.9. CP09: Eliminar Suscripción.	62

Introducción

La informatización de la salud es un proceso estratégico que busca mejorar la calidad y la eficiencia de los servicios sanitarios mediante el uso de las tecnologías de la información y las comunicaciones (TIC). El Ministerio de Salud Pública (MINSAP) retoma como una de sus prioridades, el desarrollo de este proceso en todas sus instituciones y áreas de salud con el objetivo de lograr un impacto en la atención a los pacientes, en la optimización de la gestión administrativa y en el uso eficiente de los recursos. Uno de los sectores que se beneficia de esto es el sistema de farmacias en Cuba, que está compuesto por 2180 farmacias en todo el país, 17 empresas de farmacia y ópticas, 532 dispensarios, 77 centros de producción local y 600 farmacias en unidades asistenciales. (Sánchez Padrón, 2021)

En el sector farmacéutico las tecnologías de la información y las comunicaciones permiten mejorar el control y la distribución de los medicamentos, así como facilitar el acceso y la información a los usuarios. Algunos resultados del uso de estas tecnologías son: el carnet de vacunación digital, que permite a los ciudadanos acceder a su historial de vacunas y recibir recordatorios sobre las próximas dosis; la auto-pesquisa, que permite a los usuarios reportar su estado de salud y recibir orientaciones sobre posibles síntomas; y el portal Infomed, que constituye la intranet de la Salud Pública Cubana y ofrece servicios como correo electrónico, biblioteca virtual, cursos online y redes sociales. La digitalización de la salud en Cuba es un reto ante el modelo económico y social actual, pero también una oportunidad para seguir avanzando en la garantía del derecho a la salud de toda la población. (Ramos, 2007)

La informatización también contribuye a optimizar los recursos, reducir los costos y aumentar la eficiencia y la seguridad de los procesos sanitarios. Es un factor clave para el desarrollo y la innovación del sistema de salud cubano el cual propone ir poco a poco mejorando sus servicios, los cuales actualmente se ven afectados por distintos factores que comprometen su calidad como pueden ser:

- Incrementos de las faltas y bajas coberturas (no más de 120 días).
- Déficit y fluctuación de personal calificado profesional o técnico
- Procesos manuales con alto volumen de documentos que deben reflejar la trazabilidad y fiabilidad de la información.
- Insuficiente disponibilidad de transporte administrativo y de carga.
- Dificultades con la infraestructura, equipamiento, mobiliarios e insumos.

Introducción

Los servicios farmacéuticos brindados están constituidos por una red de unidades de farmacias de la comunidad, hospitales e instituciones asistenciales pertenecientes al Sistema Nacional de Salud (SNS). Las unidades de farmacias comunitarias, sin importar su lugar de ubicación y nivel de distribución deberán tener aseguradas las existencias de los medicamentos previstos en su Cuadro Básico de Medicamentos (CBM) incluyendo los fitofármacos y apifármacos, así como los productos químicos dispensariales en función de la necesidad sentida en el área donde se encuentra la farmacia.

Las farmacias comunitarias se clasifican sobre la base de tres principios fundamentales:

1. Necesidad de establecer niveles de distribución de medicamentos.
2. Vinculación de las farmacias a los servicios y unidades de salud.
3. Capacidad y niveles de prestación de servicios en farmacia.

En sus unidades se dispensarán solo medicamentos y productos relacionados con la salud autorizados por el MINSAP:

1. Medicamentos incluidos en el Cuadro Básico de Medicamentos (CBM) y productos naturales según los niveles de distribución establecidos.
2. Medicamentos del Formulario Nacional de Fitofármacos y Apifármacos.
3. Medicamentos químicos-dispensariales establecidos en el formulario relativo vigente.
4. Medicamentos elaborados por formulaciones magistrales.
5. Otros productos relacionados con la salud como suplementos nutricionales, antioxidantes, material gastable, insumos médicos, equipos de diagnóstico, etc.

La metodología de dispensación relaciona tres elementos básicos:

1. Verificación de la prescripción médica, se deberá verificar que el modelo utilizado sea el oficialmente aprobado según el tipo de medicamento que se precise (receta médica, certificado médico, receta doble foliada y otros), que todos los escaques estén debidamente cumplimentados de forma clara y legible (firma y cuño del prescriptor, datos del paciente, cuño y nombre de la institución de salud, nombre, dosificación, forma farmacéutica, vía de administración del medicamento, fecha de emisión). En caso de existir dudas o no estar precisada alguna información determinante, no se dispensará el medicamento sin la previa aclaración y/o rectificación del prescriptor.

Introducción

2. Verificación del medicamento a dispensar, se comprobará fecha de vencimiento, lote, características organolépticas, integridad del envase, que las especificaciones del nombre, dosificación y vía de administración se ajusten a las indicadas en la prescripción y precio. Igualmente se debe comprobar si el medicamento está sometido a alguna regulación adicional en la prescripción.
3. Asesoramiento e información al paciente sobre la farmacoterapia. En cuanto a la información a brindar al paciente debe enfatizarse en la necesidad de ajustarse a la posología y duración del tratamiento prescrito, posibilidad de ocurrencia de reacciones adversas e Interacciones con otros medicamentos o alimentos, así como los requisitos de conservación y almacenamiento. (M. B. Ortega et al., 2022)

El acceso a la información sobre la disponibilidad de medicamentos en la red de farmacias es un problema crítico que afecta directamente la salud y el bienestar de la población. La falta de información clara y precisa sobre qué medicamentos están disponibles y dónde pueden obtenerse puede generar dificultades para que las personas sigan sus tratamientos médicos y manejen sus condiciones de salud de manera efectiva. Esto puede tener consecuencias graves para la salud, ya que puede llevar a problemas relacionados con la automedicación, puede generar incertidumbre y ansiedad en las personas que más necesitan de estos medicamentos. Además, puede aumentar el riesgo de interrupciones en los tratamientos y reducir la eficacia de los mismos. Es un problema que requiere atención y soluciones efectivas para garantizar que todas las personas tengan acceso a la información que necesitan para mantenerse saludables.

Dada la situación anterior se plantea el siguiente **problema a resolver**: El proceso de acceso a la información sobre la disponibilidad de medicamentos en las farmacias cubanas se torna engorroso, debido a que a la misma no se puede acceder de forma remota, fiable y de manera permanente, provocando molestias y desinformación en la población.

Siendo el **objeto de estudio** de esta investigación, el proceso de los servicios farmacéuticos.

Para resolver el problema anterior se establece como **objetivo general**, desarrollar la Plataforma Nacional de Consulta de Medicamentos (PNCM), en función de conocer de forma remota, fiable y de manera permanente la disponibilidad de medicamentos en las farmacias cubanas.

Se define como **campo de acción**, el proceso de los servicios de las consultas de disponibilidad de medicamentos en las farmacias cubanas.

Introducción

Para alcanzar el objetivo trazado y dar solución al problema planteado se elaboraron las siguientes **tareas de investigación**:

1. Revisión de la bibliografía para elaborar el marco teórico conceptual en lo referente a aplicaciones informáticas relacionadas con productos farmacéuticos estableciendo similitudes con la investigación en curso.
2. Análisis de los sistemas homólogos existentes, en función de conocer aspectos regulares en el diseño de un sistema informático con fines farmacéuticos.
3. Identificación de las herramientas y lenguajes informáticos, además de la metodología de desarrollo de software para lograr una aplicación funcional como solución al problema planteado en la investigación.
4. Identificación de las principales funcionalidades del sistema a desarrollar para la posterior implementación del mismo, teniendo en cuenta los requisitos definidos por el cliente.
5. Estudio de los diferentes tipos de pruebas para verificar el correcto funcionamiento del sistema desarrollado.

Para el cumplimiento del objetivo propuesto y de las tareas de investigación se emplean métodos teóricos y empíricos de la investigación científica.

Métodos teóricos:

- Analítico - Sintético: Permite procesar la información y llegar a conclusiones a partir de la revisión de conceptos, teorías, técnicas y herramientas. De aquí se extraen las ideas fundamentales y al mismo tiempo se detalla la información necesaria para que de esta manera identificar las tecnologías y herramientas que se utilizarán en el desarrollo de la aplicación.

Métodos empíricos:

- Entrevista: Se utilizará la entrevista como una conversación planificada con los clientes para obtener información acerca del problema que se investiga. Su uso constituye un medio para el conocimiento cualitativo de las características particulares de un proceso y puede influir en el posterior análisis y diseño del producto de software.
- Observación: Se utilizó para obtener información e identificar algunas características mediante la percepción de objetos y recopilación de conocimientos para analizar la

Introducción

situación actual de sistemas similares a la solución propuesta. Se usa además para diagnosticar el problema e identificar posibles funcionalidades de la propuesta de solución. (Falcón & Serpa, 2021)

Estructura del trabajo investigativo:

El documento se encuentra estructurado en: Resumen, introducción, 3 capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos.

Capítulo 1,” Fundamentación Teórica: En este capítulo se realiza un estudio a fondo de los conceptos fundamentales relacionados con el objeto de estudio, se investigan los sistemas homólogos a nivel internacional y nacional, se describe la metodología mejor acorde al desarrollo del software y las herramientas y lenguaje utilizados.

Capítulo 2,” Análisis y Diseño: En este capítulo se presenta la propuesta de solución denominada Plataforma Nacional de Consulta de Medicamentos (PNCM). Se ofrece una descripción detallada del proceso de consulta de medicamentos en las farmacias cubanas, junto con su modelado correspondiente. También se describen las principales funcionalidades y los patrones de diseño utilizados para esta solución.

Capítulo 3,” Implementación y Pruebas: Se implementa la solución propuesta, dándole cumplimiento a cada uno de los requerimientos pactados con el cliente y se muestran los resultados obtenidos de las pruebas realizadas.

Capítulo 1: Fundamentación teórica

1.1. Introducción

En este capítulo se describen los conceptos básicos asociados a la investigación para proporcionar un mejor entendimiento del tema. Se realiza un estudio del arte para caracterizar aplicaciones similares al sistema que se quiere desarrollar y buscar aspectos que coincidan. Se fundamenta el proceso de desarrollo de software que se lleva a cabo, indicando la metodología de desarrollo de software y modelo de desarrollo de software. Se describen las tecnologías y herramientas que se van a usar en el proyecto, indicando por qué son necesarias en cada caso.

1.2. Conceptos fundamentales

En el presente epígrafe se expone las principales definiciones o conceptos para una mejor comprensión de la presente investigación.

Medicamentos: son compuestos químicos utilizados en el tratamiento, cura, alivio sintomatológico, prevención o diagnóstico de enfermedades. Estos compuestos no pueden utilizarse o comercializarse sin ser analizados y autorizados por la Administración de Medicamentos y Alimentos (FDA), lo que contribuye a garantizar su idoneidad para el uso en humanos. (Marin Guzman, 2022)

Farmacias: son establecimientos farmacéuticos autorizados para la dispensación y expendio de medicamentos de uso y consumo humano, especialidades farmacéuticas, productos naturales procesados de uso medicinal, productos biológicos, insumos y dispositivos médicos, cosméticos, productos dentales, así como para la preparación y venta de fórmulas oficinales y magistrales. (Cevallos, Zambrano, 2020)

Servicio farmacéutico: es el servicio médico responsable que implica la interacción directa del farmacéutico con el paciente con acciones de prevención de enfermedades con el uso de fármacos que respondan a la condición del paciente y la mejora de la salud. (Marin Guzman, 2022)

Distra: Sistema de gestión empresarial de la empresa XETID que permite integrar y optimizar los procesos empresariales.

Distra-Farmacia: Es parte del sistema de gestión de Distra, enfocado en el área farmacéutica.

Capítulo 1. Fundamentación teórica

Geolocalización: es el proceso de determinar la ubicación geográfica de objetos o personas en un territorio o espacio determinado, utilizando puntos de coordenadas como la latitud, longitud y altura. Esta información se puede representar visualmente en un mapa, lo que permite una mejor comprensión y análisis de la posición virtual de un objeto o persona en tiempo real. (Herdoíza Carrillo, 2022)

Navegador web: es un software, que permite el acceso a la Web, interpretando la información de distintos tipos de archivos y sitios web para que estos puedan ser visualizados por el usuario. (Murillo, 2017)

API (Interfaz de Programación de Aplicaciones): es un sistema que forma una interfaz, que en muchos casos trabaja con un estándar para el manejo de la información. Por lo general se desarrolla en el lado del servidor. Mediante la programación en JavaScript se solicitan datos a un servidor que tiene diversos servicios, esta comunicación entre el Frontend y el Backend se conoce como API. (Aguirre, 2022)

Framework (Marco de trabajo): es una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que se puede añadir las últimas piezas para construir una aplicación concreta. (Villena et al., s. f.)

Librería: En programación, es un archivo o conjunto de archivos de código que se utilizan para facilitar la programación. (Villena et al., s. f.)

1.3. Estudio del Estado del Arte

Se investigaron aplicaciones similares a la propuesta de solución tanto en el ámbito internacional como en el nacional. A continuación, se analizan las mismas, haciendo énfasis en determinados aspectos que puedan ser de utilidad para esta investigación.

1.3.1. Sistemas informáticos similares en el mundo

Luda Farma

Es una herramienta digital creada en España en el 2020 por la compañía Luda Partners en medio de la situación epidemiológica debido al Covid-19, permite a los médicos ayudar a sus pacientes a la hora de localizar medicamentos con problemas de suministro o desabastecimiento. Es utilizada en la zona europea y solo se encuentra disponible para las farmacias que pertenecen a la red Luda Farma.

Capítulo 1. Fundamentación teórica

En la actualidad más de 3300 farmacias ya forman parte de la red digital de Luda Partners y tienen a su disposición la herramienta digital Luda Farma para ayudar a los pacientes a la hora de localizar medicamentos, lo que permite continuar con tratamientos pautados. («LUDA Farma, herramienta para localizar medicamentos desde las farmacias», s. f.)

Esta herramienta es muy similar a la propuesta de solución, pero carece de la funcionalidad de las notificaciones a los correos electrónicos cuando un medicamento esté disponible, además para utilizar este sistema es obligatorio pertenecer a la red de Luda Farma.

Farmaguía

Es el localizador de farmacias de la provincia de Barcelona en tiempo real. Encuentra de forma rápida y eficaz las farmacias y sus servicios, y también hospitales y centros de atención primaria. (*FARMAGUIA - Localizador de farmacias y sus servicios, farmacias de guardia, farmacias de Barcelona en tiempo real*, s. f.)

Este sitio web carece de información sobre los medicamentos de las farmacias. Esta herramienta digital aportó ideas para la creación y diseño del mapa del sistema propuesto.

Vademecum Mobile 2.0

Es el sitio web de farmacia en español más consultado. Vademécum Mobile 2.0, su aplicación, creada en el 2013 por la compañía Vademecum, permite a la comunidad farmacéutica consultar con facilidad la información sobre todos los medicamentos a la venta en España. Proporciona acceso a funcionalidades de pago o gratuitas; se pueden consultar medicamentos genéricos. (Farmamundi, 2022)

Este es sistema español que no tiene ninguna funcionalidad de mapas, es enfocado en la realización de consultas.

Medicamento Accesible Plus

Esta aplicación fue promovida por el Consejo General del Colegio Oficial de Farmacéuticos, entre otras entidades. Su primera versión fue presentada en Madrid el 11 de marzo de 2014. Es capaz de realizar dos tipos de búsquedas.

Búsqueda por Nombre del Medicamento: permite realizar búsquedas de medicamentos por su nombre.

Capítulo 1. Fundamentación teórica

Búsqueda por Código Nacional del Medicamento: permite realizar búsquedas de medicamentos por su código nacional. (Farmamundi, 2022)

En esta aplicación como en el caso anterior no tiene implementado ningún servicio de mapas y su principal uso es la búsqueda de medicamentos.

1.3.2. Sistemas informáticos similares en Cuba

Formulario Nacional de Medicamentos (FNM)

Es una aplicación solo disponible en dispositivos Android. Creada por un colectivo de profesionales del Ministerio de Salud Pública y la primera versión fue presentada oficialmente en agosto de 2014. Permite consultar información confiable y actualizada sobre más de 750 fármacos. Su objetivo principal es satisfacer las necesidades de médicos, farmacéuticos, enfermeros y todos los profesionales relacionados con la información sobre medicamentos y su uso. Esta aplicación permitió identificar qué información sobre los medicamentos es la más vital para los usuarios. (APKlis, s. f.)

Guía Farma

Es un complemento sin fines de lucro para permitir la consulta en iPhones y iPads, con funcionalidades muy parecidas a las de FNM, es un trabajo realizado por los especialistas del Comité Editorial del Formulario Nacional de Medicamentos de Cuba. La primera versión estuvo disponible el 2 de mayo de 2020. Los contenidos que se encuentran en esta aplicación están dirigidos fundamentalmente a profesionales de la salud. La información que se suministra no debe ser utilizada, bajo ninguna circunstancia, como base para la prescripción de tratamientos o medicamentos, sin previa orientación médica. (APKlis, s. f.)

Mapa de Cuba offline

Es una aplicación para dispositivos Android que funciona solo con GPS. Proporciona mapas altamente detallados que están adaptados para trabajar con dispositivos móviles. Entre sus funcionalidades se destaca la ubicación sin conexión a internet en el mapa de Cuba (APKlis, s. f.)

1.3.3. Conclusiones del Estudio del Estado del Arte

El análisis de los distintos casos realizado anteriormente permitió arribar a varias conclusiones sustanciales para la investigación. Como se puede observar, la mayoría de estos softwares son utilizados en aplicaciones móviles y para los que utilizan sistema Windows son privados, por tal razón no es posible la modificación de su código para adecuarlo a las necesidades del cliente además que no cumplen con los requisitos en su totalidad. También la adquisición de los mismos

Capítulo 1. Fundamentación teórica

es costosa y se ve afectada por la influencia del bloqueo, por lo que ninguno de estos sistemas resuelve el problema. Sin embargo, cuentan con funcionalidades que se pueden tener presentes en el desarrollo de una nueva solución, entre las que se destacan:

- Búsqueda de medicamentos.
- Obtener información confiable y actualizada sobre los medicamentos.
- Ubicar las farmacias en un mapa.
- Geolocalización en el mapa.

1.4. Metodología de desarrollo de software

Las metodologías tradicionales se centran en llevar una extensa documentación para garantizar el cumplimiento de un plan de proyecto definido previamente en una fase inicial. Las metodologías ágiles nacieron en la industria del desarrollo de software, cuando las compañías de este sector comprendieron que la forma tradicional de trabajo retrasaba mucho la entrega del producto final. Unos procesos basados normalmente en un contrato cerrado, con escasa comunicación de los trabajadores, que conducían a entregables de mala calidad. (Dallos Bustos et al., 2020)

El modelado ágil (MA) es una metodología basada en la práctica para modelar y documentar con eficacia los sistemas basados en software. En pocas palabras, es un conjunto de valores, principios y prácticas para hacer modelos de software aplicables de manera eficaz y ligera a un proyecto de desarrollo de software. Los modelos ágiles son más eficaces que los tradicionales porque son sólo buenos, sin pretender ser perfectos. (Pressman, 2010)

Ventajas de las metodologías ágiles

- Favorecen la productividad.
- Adecuadas para proyectos con requisitos imprecisos y cambiantes, donde hay un alto riesgo técnico.
- Promueven las relaciones interpersonales y el aprendizaje de los desarrolladores.

Tabla 1.1. Comparación de Metodologías. Fuente (MONTERO,CEVALLOS, 2018).

Metodologías Tradicionales	Metodologías Ágiles
Predictivos.	Adaptativos.

Capítulo 1. Fundamentación teórica

Orientado a procesos.	Orientado a personas.
Proceso rígido. Se concibe como un proyecto.	Proceso flexible. Un proyecto es subdividido en varios subproyectos más pequeños.
Poca comunicación con el cliente.	Comunicación constante con el cliente.
Entrega del software al finalizar el desarrollo.	Entregas constantes de software.
Documentación extensa.	Poca documentación.

Según la comparativa realizada, se decide seleccionar la Metodología Ágil.

Comparación de Metodología Ágiles

AUP: en su sigla en inglés proceso unificado ágil (PUA) adopta una filosofía “en serie para lo grande” e “iterativa para lo pequeño” a fin de construir sistemas basados en computadora. Al adoptar las actividades en fase clásicas del PU -concepción, elaboración, contrición y transición-, el AUP brinda un revestimiento en serie que permite que el equipo visualice el flujo general del proceso de un proyecto de software, las actividades son: modelado, implementación, pruebas, despliegue, configuración y administración del proyecto, administración del ambiente. (Pressman, 2010)

Scrum: es un método de desarrollo ágil de software concebido por Jeff Sutherland y su equipo de desarrollo. Las actividades estructurales que incorporan son: 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 (el número de estos que requiere cada actividad estructural varía en función de la complejidad y tamaño del producto) se adapta al problema en cuestión. (Pressman, 2010)

XP: la programación extrema es una de las metodologías de desarrollo de software ágil mejor conocidas y más ampliamente utilizadas. El nombre lo acuñó Kent Beck debido a que el enfoque se desarrolló llevando a niveles “extremos” las prácticas reconocidas, como el desarrollo iterativo. XP se basa en la simplicidad, la comunicación y la realimentación del código desarrollado. (Sommerville, 2011)

Capítulo 1. Fundamentación teórica

A continuación, se presenta la tabla, donde se seleccionaron los elementos comparativos, según lo que es necesario para el sistema a ser desarrollado.

Tabla 1.2. Comparación de algunas metodologías ágiles. Fuente: (Pressman, 2010)

Características	SCRUM	AUP	XP
Documentación	Mucha	Regular o mucha	Poca
Relación con el cliente	Poca	Regular	Constante
Uso en proyecto	Pequeño y mediano	Pequeño y mediano	Pequeño y mediano
Enfoque	En las personas	En las actividades	En las personas
Refactorización del código	No	No	Sí

Luego de analizar estas metodologías se escoge XP por las siguientes razones:

- Al tener en cuenta las características del equipo de trabajo, el cual está conformado por 2 desarrolladores y la permanente interacción con el cliente, resulta apropiado la implementación de prácticas correspondientes a la metodología XP.
- El cliente necesita el sistema lo más rápido posible, porque le hacen falta funcionalidades importantes en su quehacer diario para cumplir con sus tareas. XP es apropiado porque permite en cada iteración implementar, hacer las pruebas y entregar una o varias historias de usuario.
- El cliente está disponible para formar parte del equipo de desarrollo para culminar el proyecto lo más rápido posible
- Es abierta a los cambios y genera poca documentación lo que hace la entrega del software menos complicada y más satisfactoria tanto para los clientes como para el equipo de desarrollo
- XP tiene un enfoque en la refactorización del código.

1.4.1. Programación Extrema

Valores de XP

Comunicación: este valor se centra en asegurarse de que todos los miembros del equipo sepan que se espera de ellos, y que otras personas están trabajando en ello. El objetivo es lograr una comunicación eficaz entre los clientes y desarrolladores. (Pressman, 2010)

Simplicidad: para alcanzarla XP restringe a los desarrolladores para que diseñen sólo las necesidades inmediatas, en lugar de considerar las del futuro. El objetivo es crear un diseño sencillo que se implemente con facilidad en forma de código. Si hay que mejorar el diseño, se rediseñará en un momento posterior. (Pressman, 2010)

Retroalimentación: se obtiene del software implementado, del cliente y de otros miembros del equipo de software. El equipo debe obtener impresiones de la idoneidad de su trabajo lo antes posible. Fallar pronto puede ser útil, especialmente si al hacerlo se obtiene nueva información, mientras aún hay tiempo de mejorar el producto. (Pressman, 2010)

Fases de XP

Planeación: comienza escuchando la actividad para recabar requerimientos que permite que los miembros técnicos del equipo XP entiendan el contexto del negocio para el software y adquieran la sensibilidad de la salida y características principales y funcionalidad que se requieren. El proyecto comienza recopilando las historias de usuarios, las que constituyen a los tradicionales casos de uso. Una vez obtenidas estas historias de usuarios, los programadores evalúan rápidamente el tiempo de desarrollo de cada una. (Pressman, 2010)

Diseño: sigue rigurosamente el principio “manténlo sencillo”. Un diseño sencillo siempre se prefiere sobre una representación más compleja. Los conceptos más importantes de diseño en XP son los siguientes: simplicidad, recodificación, metáforas y soluciones. (Pressman, 2010)

Codificación: Después de que las historias han sido desarrolladas y de que se ha hecho el trabajo de diseño preliminar, el equipo no inicia la codificación, sino que desarrolla una serie de pruebas unitarias a cada una de las historias que se van a incluir en la entrega en curso (incremento de software). Una vez creada la prueba unitaria, el desarrollador está mejor capacitado para centrarse en lo que debe implementarse para pasar la prueba. Una vez que el código está terminado, se le aplica de inmediato una prueba unitaria, con lo que se obtiene

retroalimentación instantánea para los desarrolladores. Un concepto clave en esta fase es la programación por parejas. (Pressman, 2010)

Pruebas: Se deben realizar pruebas unitarias, de integración, de validación y de aceptación. 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). Las pruebas de la integración y validación del sistema pueden efectuarse a diario. Y las pruebas de aceptación 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 como parte de la liberación del software. (Pressman, 2010)

1.5. Tecnologías para el desarrollo de la propuesta de solución

Stack del proyecto

Se utilizó en el proyecto el stack MERN (MongoDB - Express - Reactjs - Nodejs) que consiste en un conjunto de frameworks (marcos de trabajo) y herramientas que ayudan al desarrollo de las aplicaciones. Se nombrarán todas las tecnologías y librerías usadas pertenecientes a cada ámbito.

Entorno de desarrollo

Visual Studio Code v1.82 es un editor de programación multiplataforma desarrollo por Microsoft. Es un proyecto de software libre que se distribuye bajo la licencia MIT, aunque los ejecutables se distribuyen bajo una licencia gratuita. De forma predeterminada, Visual Studio Code es compatible con gran variedad de lenguajes de programación. Además, trae un conjunto de características que facilitan el proceso de codificación como puede ser el uso de Snippets, Resaltado de sintaxis, Autocompletado, Debugger, etc. Todas estas características pueden ser extendidas a través de complementos, disponibles en un repositorio central. (*Visual Studio Code - Code Editing. Redefined*, 2023)

1.5.1. Tecnologías utilizadas en el Frontend

Reactjs: Es una librería JavaScript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre. En el proyecto hay más de mil desarrolladores libres. (*React*, 2023)

Leaflet: Para la creación del mapa se seleccionó Leaflet porque es una librería de código abierto para crear mapas interactivos de diversos tipos, está específicamente diseñada para ser

Capítulo 1. Fundamentación teórica

visualizada en dispositivos móviles. Su diseño es sencillo, por lo que mejora los resultados de los desarrolladores no expertos. Tiene una API fácil de usar y está bien documentada, además de contar con un código fuente simple y legible, y una comunidad desarrolladora muy atractiva. (García Alonso, 2021)

Las ventajas de utilizar Leaflet son:

- Funcionamiento óptimo
- Recomendado para accesorios móviles
- Sencillez de manejo y aprendizaje
- Complementado por multitud de plugins

Axios: Es un cliente HTTP basado en Promesas para Javascript. Se usará para las llamadas a la API del backend. (Axios, 2023)

MaterialUI: Es un framework de estilos que ofrece componentes de React para un desarrollo web más rápido y sencillo. Tiene gran cantidad de componentes adaptados a los estándares de diseño Material, con un diseño muy moderno. (Material UI, 2023)

1.5.2. Tecnologías utilizadas en el Backend

MongoDb: es una base de datos de código abierto y líder en bases de datos NoSQL. Es una base de datos distribuida, basada en documentos en formato BSON, que es una representación binaria de los objetos JSON. Además, MongoDB permite una gran escalabilidad y flexibilidad en la gestión de la base de datos, beneficiando al sistema en propuesto. (MongoDB, 2023)

Express: Es un marco de trabajo de Node.js de código abierto muy ligero, extremadamente mínimo y flexible diseñado para crear aplicaciones web, utilizando API. Express se ejecuta como un módulo o un elemento dentro del entorno Node.js. A diferencia de otros marcos, es mucho más minimalista y ofrece características avanzadas. Al trabajar como una capa sobre Node.js, puede ayudar a diseñar aplicaciones de una sola página, múltiples páginas e híbridas. Proporciona seguridad, líneas de códigos reducidas, por lo tanto, un desarrollo mucho más rápido y fácil accesibilidad a complementos de terceros lo convierte en el marco Node.js más popular entre los desarrolladores. (Express - Infraestructura de aplicaciones web Node.js, 2023)

Nodejs: Es el entorno de servidor web de código abierto más popular de la actualidad desarrollado por Ryan Dahl y Joyent Inc (actualmente propiedad de Samsung Electronics). Al ser un entorno de tiempo de ejecución, Node.js ayuda a ejecutar códigos JavaScript fuera de un navegador. Utiliza el motor de Google Chrome para ejecutar JavaScript. En términos simples,

Capítulo 1. Fundamentación teórica

ayuda a traducir los códigos de JavaScript al lenguaje de nivel de máquina, por lo tanto, una máquina puede entender los códigos de JavaScript sin la necesidad de un navegador. (*About Node,Js*, 2023)

Nodemailer: Es un módulo para aplicaciones Node.js que permite el envío de correos electrónicos de forma muy sencilla. El proyecto se inició en 2010, cuando no existía una opción sensata para enviar mensajes de correo electrónico; hoy en día es la solución a la que recurren la mayoría de los usuarios de Node.js de forma predeterminada. (*Nodemailer*, 2023)

1.5.3. Lenguaje de programación

JavaScript: es un lenguaje de programación que se usa para procesar información y manipular documentos. Al igual que cualquier otro lenguaje de programación, JavaScript provee instrucciones que se ejecutan de forma secuencial para indicarle al sistema lo que se quiere hacer (realizar una operación aritmética, asignar un nuevo valor a un elemento, etc.). Cuando el navegador encuentra este tipo de código en el documento, ejecuta las instrucciones al momento y cualquier cambio realizado en el documento se muestra en pantalla. (Gauchat, 2017)

Se hará uso de JavaScript para el desarrollo de la aplicación por ser la base del entorno planteado por los autores.

Por las conceptos y características planteados se decide utilizar React JS como librería de interfaz de usuario, se asume la tecnología MERN Stack, por agregar React JS en su entorno de trabajo y ser conocida por los programadores (XP señala que la tecnología a utilizar debe ser dominada por el equipo de desarrollo).

De acuerdo a lo planteado, los autores declaran el listado de las tecnologías a utilizar con sus respectivas versiones para el desarrollo de la solución propuesta:

- React JS versión 18.2.0
- Material UI versión 5.12.3
- Leaflet versión 1.9.3
- Axios versión 1.4.0
- Node JS versión 18.15.0
- Express versión 4.18.2
- MongoDB versión 4.0
- Nodemailer versión 6.9.4

1.5.4. Herramienta CASE para el modelado del sistema Visual Paradigm for UML 16.2

Una Herramienta de Ingeniería de Software Asistida por Computadora (CASE, del inglés Computer Aided Software Engineering) es una herramienta individual para ayudar al desarrollador de software o administrador de proyecto durante una o más fases del desarrollo de software o mantenimiento (González, 2016). En el modelado de la propuesta de solución se utilizará Visual Paradigm para UML para generar los artefactos tales como: diagrama de procesos del negocio, diagrama de despliegue, etcétera.

El Visual Paradigm 16.2 es una herramienta CASE diseñada para una amplia gama de usuarios, incluyendo ingenieros de software, analistas de sistemas, analistas de negocios y arquitectos de sistemas, o para cualquier persona que esté interesada en la construcción de forma fiable de los sistemas de software a gran escala con un enfoque orientado a objetos. Ofrece soporte para varias versiones de UML (González, 2016)

Resultados del análisis

Se utilizó como herramienta Visual Paradigm en su versión 16.2 debido no solo a ser la herramienta con la que más afinidad posee el equipo de trabajo, también se puede destacar que posee capacidades de ingeniería directa e inversa, licencia gratuita y comercial, exportación como HTML, diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad, también el uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.

1.6. Conclusiones parciales

Luego de llevar a cabo una investigación exhaustiva y una revisión bibliográfica, se obtuvieron las siguientes conclusiones:

- Después de analizar las aplicaciones homólogas se obtuvieron ideas para las funcionalidades y tecnologías que se utilizarán.
- La propuesta de solución planteada utilizará la novedad de realizar consultas a medicamentos junto con geolocalización para las farmacias del país.
- Fue seleccionada la metodología XP, debido a que cumplía con los requisitos para la solución propuesta, teniendo en cuenta el ambiente de desarrollo creado para la misma. Además, hace posible en cada iteración una entrega de software funcional según las necesidades del cliente.

Capítulo 1. Fundamentación teórica

- La selección del marco de trabajo MERN (MongoDB, Express, React.js y Node.js) permitirá la creación de un sistema informático web de una sola página, logrando que el sistema informático web no haga que el navegador se recargue y que sean más rápidas las peticiones al servidor.

Capítulo 2: Análisis y diseño

2.1. Introducción

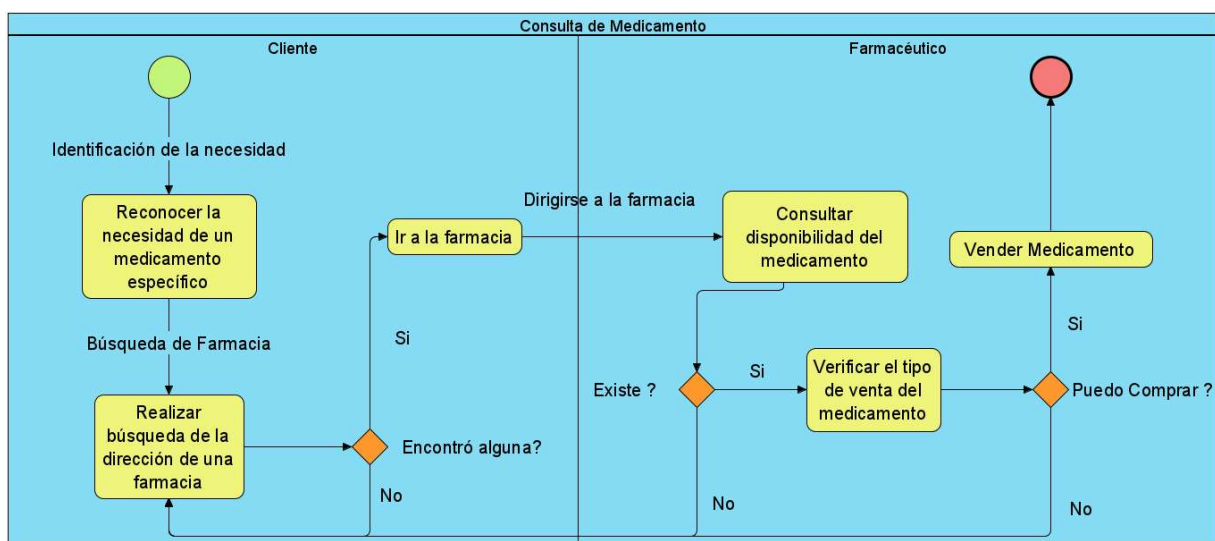
En este capítulo se presenta un diagrama que ilustra el proceso del negocio, y se introduce la propuesta de la PNCM, una herramienta innovadora que permite realizar consultas para obtener información detallada sobre los medicamentos disponibles en cada farmacia del país. Se describen las principales funcionalidades de esta solución. En los siguientes epígrafes se detallarán los artefactos generados en las fases de Exploración, Planificación y Diseño, utilizando la metodología de desarrollo de software XP.

2.2. Proceso de consulta de medicamentos en las farmacias cubanas

Actualmente el proceso de consulta de medicamentos en las farmacias del país comienza cuando una persona padece de alguna enfermedad o por otro motivo necesita saber en qué farmacia está disponible el medicamento que necesita. Así que el cliente al requerir información confiable se dirige a la farmacia y le pregunta al farmacéutico si el medicamento está a la venta en la farmacia y en caso afirmativo el farmacéutico verifica el tipo de venta y si el cliente cumple con los requisitos le vende el medicamento.

A continuación, se muestra el diagrama del proceso de negocio descrito anteriormente.

Figura 2.1. Diagrama del proceso de consulta de medicamentos en las farmacias cubanas. Fuente: (Elaboración propia)



2.3. Exploración

Durante esta etapa inicial, los clientes delimitan de manera general las Historias de Usuario (HU) que serán pertinentes para la primera entrega del producto. Paralelamente, el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se implementarán en el proyecto. Se lleva a cabo una evaluación de la tecnología y se exploran las posibilidades arquitectónicas del sistema mediante la construcción de un prototipo. El período de exploración puede variar desde unas pocas semanas hasta varios meses, dependiendo del grado de familiaridad que los programadores tengan con la tecnología en cuestión. (Beck, 2000)

2.3.1. Historias de usuario

Historias de usuario (HU): En la programación extrema, son requerimientos que se expresan como escenarios, 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. Entre las liberaciones del sistema existe un breve lapso. (Sommerville, 2011)

Con el objetivo de determinar que HU son más esenciales resolver, según las necesidades del cliente y poder realizar una correcta planificación de la implementación, cada HU es clasificada por el propietario del producto según su prioridad para el negocio en:

Alta: HU que resultan fundamentales en el desarrollo del sistema.

Media: HU que resultan funcionalidades necesarias, pero no imprescindibles.

Baja: HU que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura y no tienen nada que ver con el sistema en desarrollo.

También se tiene en cuenta la dificultad y posible existencia de errores durante la implementación de cada HU, por ello el equipo de desarrollo clasifica a cada una según el riesgo en su desarrollo:

Alto: cuando en la implementación de la HU se considera la posible existencia de errores que lleven a la inoperatividad del código.

Medio: cuando pueden aparecer errores en la implementación de la HU que puedan retrasar la entrega de la versión.

Bajo: cuando pueden ocurrir errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del proyecto.

Las HU son representadas mediante tablas divididas por las siguientes secciones:

Capítulo 2. Análisis y diseño

1. **Número:** esta sección representa el número, incremental en el tiempo, de la historia de usuario que se describe.
2. **Nombre de Historia de Usuario:** identifica la HU que se describe entre los desarrolladores y el cliente.
3. **Usuario:** rol del usuario que realiza la funcionalidad.
4. **Prioridad en negocio:** se le otorga una prioridad (Alta, Media, Baja) a las HU de acuerdo a la necesidad de desarrollo.
5. **Riesgo en Desarrollo:** se le otorga una medida de (Alto, Medio, Bajo), a la ocurrencia de errores en el proceso de desarrollo de la HU.
6. **Iteración asignada:** número de la iteración donde va a desarrollarse la HU.
7. **Puntos Estimados:** es el tiempo estimado en semanas ideales (40 horas) que se demorará el desarrollo de la HU.
8. **Descripción:** breve descripción de la HU.
9. **Observaciones:** señalamiento o advertencia del sistema.

Debido al análisis realizado al proceso de consulta de medicamentos en las farmacias del país, todos los aspectos relacionados con el mismo quedaron totalmente claros. Lo que permitió la elaboración de las historias de usuario, en la etapa de exploración. No siendo necesario realizar modificaciones ni adicionar nuevas HU a lo largo de las iteraciones.

A continuación, se exponen las HU definidas por el cliente en conjunto con el equipo de desarrollo:

Tabla 2.1. Historia de Usuario 1. Fuente: (Elaboración propia).

Historia de Usuario	
Número: 1	Nombre: Buscar medicamentos
Usuario: Todos los usuarios	
Prioridad en negocio: Alta	Riesgo en desarrollo: Medio
Puntos estimados: 0.6	Iteración asignada: 1
Programador responsable: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: Los usuarios deben poder buscar medicamentos por nombre, provincia, municipio, tipo de venta, grupo y forma farmacéutica. La búsqueda debe mostrar una lista de medicamentos que coincidan con los criterios introducidos por el usuario. El usuario puede seleccionar un medicamento de la lista para ver más información sobre él.	

Capítulo 2. Análisis y diseño

Observaciones: El sistema debe consultar los servicios de Distra-Farmacia para obtener los medicamentos disponibles en el país. El sistema debe mostrar un mensaje de error si no se encuentra ningún medicamento que coincida con los criterios de búsqueda.

Figura 2.2. Prototipo de Historia de Usuario 1. Fuente: (Elaboración propia).

Browser

Paracetamol

Precio

Grupo

Venta

Dosificación

Descripción

Disponible en

Farmacia 4

Provincia

Municipio

Teléfono

Dirección

Tipo

Ver

Tabla 2.2. Historia de usuario 2. Fuente: (Elaboración propia).

Historia de Usuario	
Número: 2	Nombre: Buscar farmacias
Usuario: Todos los usuarios	
Prioridad en negocio: Alta	Riesgo en desarrollo: Medio
Puntos estimados: 0.6	Iteración asignada: 1
Programador responsable: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: Los usuarios deben poder buscar farmacias por nombre, dirección, provincia, municipio y tipo. La búsqueda debe mostrar una lista de farmacias que coincidan con los	

Capítulo 2. Análisis y diseño

criterios introducidos por el usuario. El usuario puede seleccionar una farmacia de la lista para ver más información sobre ella.

Observaciones: El sistema debe consultar los servicios de Distr-Farmacia para obtener información sobre las farmacias del país. El sistema debe mostrar un mensaje de error si no se encuentra ninguna farmacia que coincida con los criterios de búsqueda.

Figura 2.3. Prototipo de Historia de Usuario 2. Fuente: (Elaboración propia).

Browser

Buscar farmacias ... 🔍

Filtrar provincia o municipio

Farmacia 4

Provincia

Pinar del Río

Dirección

Calle 74 e/3 y 12

Municipio

San Luis

Tipo

Hospitalaria

Teléfono

500695114

Ver

Tabla 2.3. Historia de usuario 3. Fuente: (Elaboración propia).

Historia de Usuario	
Número: 3	Nombre: Geolocalizar las farmacias del país en el mapa
Usuario: Todos los usuarios	
Prioridad en negocio: Alta	Riesgo en desarrollo: Alta
Puntos estimados: 0.6	Iteración asignada: 2
Programador responsable: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	

Capítulo 2. Análisis y diseño

Descripción: El mapa debe mostrar todas las farmacias del país con un ícono o marcador que indique su ubicación exacta. El usuario puede hacer zoom o desplazarse por el mapa para ver las farmacias en diferentes zonas o regiones. El usuario puede seleccionar una farmacia del mapa para ver su información (nombre, dirección, teléfono, provincia, municipio, tipo de farmacia). El usuario puede cambiar entre las distintas capas del mapa (OSM, Satelital e Híbrido).

Observaciones: El sistema debe consultar los servicios de Distra-Farmacia para obtener información sobre las farmacias del país y sus coordenadas geográficas. El sistema debe usar una API externa para mostrar el mapa y sus funcionalidades.

Figura 2.4. Prototipo de Historia de Usuario 3. Fuente: (Elaboración propia).



Tabla 2.4. Historia de usuario 4. Fuente: (Elaboración propia).

Historia de Usuario	
Número: 4	Nombre: Mostrar información sobre las farmacias en el mapa
Usuario: Todos los usuarios	
Prioridad en negocio: Media	Riesgo en desarrollo: Alta
Puntos estimados: 0.6	Iteración asignada: 2

Capítulo 2. Análisis y diseño

Programador responsable: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz
Descripción: La información debe mostrarse en una ventana emergente cuando el usuario seleccione una farmacia del mapa. La información debe incluir nombre, dirección, teléfono, distancia aproximada a la ubicación del usuario y un botón para calcular la ruta a la farmacia desde la posición actual del usuario.
Observaciones: El sistema debe consultar los servicios de Distra-Farmacia para obtener información sobre las farmacias del país.

Figura 2.5. Prototipo de Historia de Usuario 4. Fuente: (Elaboración propia).

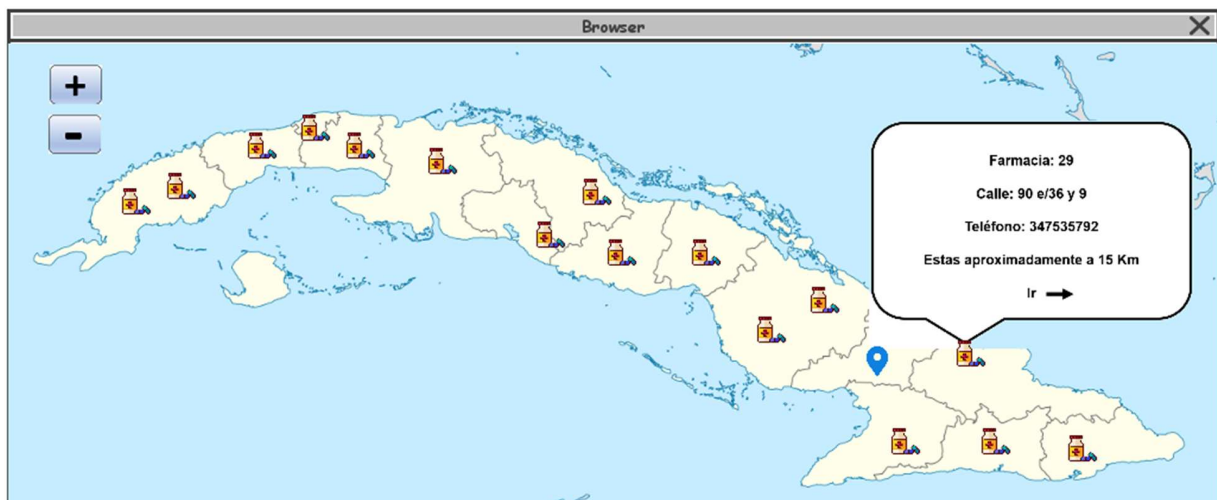


Tabla 2.5. Historia de usuario 5. Fuente: (Elaboración propia).

Historia de Usuario	
Número: 5	Nombre: Geolocalizar usuario en el mapa
Usuario: Todos los usuarios	
Prioridad en negocio: Media	Riesgo en desarrollo: Baja
Puntos estimados: 0.8	Iteración asignada: 2
Programador responsable: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: El sistema debe solicitar al usuario su permiso para acceder a su ubicación actual cuando acceda al sistema por primera vez o cuando cambie su configuración de privacidad. El sistema debe mostrar un campo donde el usuario pueda introducir una dirección	

Capítulo 2. Análisis y diseño

(municipio). El sistema debe mostrar el punto introducido por el usuario en el mapa con un icono o marcador que indique su ubicación exacta.

Observaciones: El sistema debe validar que el usuario tenga permiso para acceder a la funcionalidad de geolocalizar. El sistema debe validar que la dirección (municipio) introducido por el usuario sea válido.

Figura 2.6. Prototipo de Historia de Usuario 5. Fuente: (Elaboración propia).



Tabla 2.6. Historia de usuario 6. Fuente: (Elaboración propia).

Historia de Usuario	
Número: 6	Nombre: Resaltar las farmacias más cercanas a la posición del usuario
Usuario: Todos los usuarios	
Prioridad en negocio: Media	Riesgo en desarrollo: Baja
Puntos estimados: 1	Iteración asignada: 3
Programador responsable: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: El sistema debe resaltar las farmacias que se encuentren dentro de un radio de 5 Km con un color o ícono diferente al resto.	

Capítulo 2. Análisis y diseño

Observaciones: El sistema debe consultar los servicios de Distra-Farmacia para obtener las coordenadas de las farmacias del país. El sistema debe solicitar al usuario su permiso para acceder a su ubicación actual o permitirle introducir una dirección manualmente.

Figura 2.7. Prototipo de Historia de Usuario 6. Fuente: (Elaboración propia).



Tabla 2.7. Historia de usuario 7. Fuente: (Elaboración propia).

Historia de Usuario	
Número: 7	Nombre: Calcular la ruta a la farmacia seleccionada
Usuario: Todos los usuarios	
Prioridad en negocio: Media	Riesgo en desarrollo: Baja
Puntos estimados: 3	Iteración asignada: 3
Programador responsable: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: El sistema debe mostrar un mapa con la ruta trazada entre el origen y el destino con un color que indique el sentido y la distancia.	
Observaciones: El sistema debe validar que las direcciones de origen y destino sean válidas y se encuentren dentro del país. El sistema debe usar una API externa que permita realizar el cálculo de la ruta.	

Capítulo 2. Análisis y diseño

Figura 2.8. Prototipo de Historia de Usuario 7. Fuente: (Elaboración propia).



Tabla 2.8. Historia de usuario 8. Fuente: (Elaboración propia).

Historia de Usuario	
Número: 8	Nombre: Gestionar suscripción
Usuario: Todos los usuarios	
Prioridad en negocio: Media	Riesgo en desarrollo: Alta
Puntos estimados: 4	Iteración asignada: 4
Programador responsable: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: El sistema debe mostrar un formulario donde el usuario pueda introducir su correo electrónico, una lista de farmacias y medicamentos que podrá seleccionar, de los cuales recibirá notificaciones a partir de la disponibilidad de los mismos. El usuario podrá gestionar su suscripción, la gestión incluye las acciones de crear suscripción, eliminar suscripción, modificar suscripción y ver la información de la suscripción.	
Observaciones: El sistema debe validar que el usuario solo seleccione como máximo 5 farmacias y 5 medicamentos a notificar. El sistema debe validar que el usuario no tenga un correo ya existente en el sistema. El sistema debe validar que los datos introducidos por el usuario sean válidos y completos. El sistema debe almacenar sus datos en una base de datos segura. El sistema debe usar una API externa para enviar correos electrónicos. El sistema debe mostrar un mensaje de error si ocurre algún problema al registrar al usuario o al enviar el código de verificación.	

Capítulo 2. Análisis y diseño

Figura 2.9. Prototipo de Historia de Usuario 8. Fuente: (Elaboración propia).



2.4. Planificación

La actividad de planeación comienza al escuchar, lo que permite recabar requerimientos para que los miembros técnicos del equipo XP entiendan el contexto del negocio para el software y adquieran la sensibilidad de la salida y características principales y funcionalidad que se requieren. Escuchar lleva a la creación de algunas historias de 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 (es decir, una prioridad) 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. (Pressman, 2010) La planeación se puede realizar basándose en el tiempo o en el alcance, en este caso fue basándose en el alcance del sistema.

2.4.1 Estimación del esfuerzo

Para determinar el tiempo que requerirá el equipo de desarrollo para completar todas las historias de usuario y cuántos puntos deben desarrollarse, se lleva a cabo una estimación del esfuerzo por cada Historia de Usuario (HU). Dado que la velocidad del proyecto no se define claramente

Capítulo 2. Análisis y diseño

hasta después de varias iteraciones, se calculó de tal manera que la primera iteración fue de 7 puntos, la segunda de 3.2, la tercera de 6.6 y la cuarta iteración de 6 puntos, agrupando las 8 HU en cuatro iteraciones. La determinación del número de iteraciones se basó en dos indicadores fundamentales: la experiencia del equipo de desarrollo (analogía) y el ajuste permitido por la planificación orientada al alcance del sistema (cantidad total de puntos / velocidad del proyecto). En la Tabla 2.9 queda reflejada la estimación del esfuerzo realizada.

Tabla 2.9. Estimación de esfuerzo por historia de usuario. Fuente: (Elaboración propia).

Iteración	Historias de usuario		Puntos estimados (semanas)
1	1	Buscar medicamentos	0.6
	2	Buscar farmacias	0.6
2	3	Geolocalización de las farmacias del país en el mapa	0.6
	4	Mostrar información sobre las farmacias en el mapa	0.6
	5	Geolocalizar usuario	0.8
3	6	Resaltar las farmacias más cercanas a la posición del usuario	1
	7	Calcular la ruta a la farmacia seleccionada	3
4	8	Gestionar suscripción	4
Total			11.2

2.4.2. Plan de iteraciones

Tras la identificación y descripción de las Historias de Usuario (HU) y la estimación del esfuerzo dedicado a cada una de ellas, en XP se procede a la creación del artefacto denominado plan de iteraciones. Este artefacto indica cuáles HU serán implementadas en cada iteración para optimizar el rendimiento del equipo de desarrollo. Al concluir la última iteración, el sistema estará preparado para entrar en producción. A continuación, se planifica la fase de implementación, durante la cual las HU serán desarrolladas y probadas en un ciclo iterativo. Para un rendimiento organizado, el equipo establece un total de cuatro iteraciones. El plan de duración de las iteraciones se refleja en la Tabla 2.10.

Iteración 1.

Capítulo 2. Análisis y diseño

En esta iteración se implementarán HU de nivel de prioridad alto que estén relacionadas con la primera etapa (la búsqueda), creando con ella la línea base de la arquitectura del sistema. Además, se obtendrá una primera versión del producto en la que el cliente podrá probar dos de las funcionalidades solicitadas (la búsqueda de medicamentos y la búsqueda de farmacias) descritas en las HU 1 y 2.

Iteración 2.

En esta iteración se implementarán el resto de las HU con prioridad alta en el negocio y se corregirán errores o no conformidades de los usuarios con los componentes implementados en la iteración anterior. Como resultado se obtendrá una versión más completa sobre las funcionalidades del sistema. Al culminar esta iteración los usuarios tendrán acceso entonces a funcionalidades que cubren hasta la geolocalización en el mapa.

Iteración 3.

Esta iteración es la dedicada a la implementación HU de nivel de prioridad media. En esta iteración se implementarán las HU 6 y 7. Se corregirán errores o no conformidades de los usuarios con las funcionalidades desarrolladas en la iteración anterior. Como resultado se obtendrá una versión del sistema con todas las funcionalidades de consulta y de visualización en el mapa.

Iteración 4.

En esta iteración se implementará la última HU (gestionar suscripción) con prioridad en el negocio media, completando el desarrollo de la aplicación. Se obtendrá la versión 1.0 de la PNCM donde los usuarios podrán ejecutar todas las funcionalidades y explotar el sistema a gusto.

Tabla 2.10. Plan de duración de las iteraciones. Fuente: (Elaboración propia).

Iteración	Historias de usuario		Duración (semanas)
1	1	Buscar medicamentos	1.2
	2	Buscar farmacias	
2	3	Geolocalización de las farmacias del país en el mapa	2
	4	Mostrar información sobre las farmacias en el mapa	

Capítulo 2. Análisis y diseño

	5	Geolocalizar usuario	
3	6	Resaltar las farmacias más cercanas a la posición del usuario	4
	7	Calcular la ruta a la farmacia seleccionada	
4	8	Gestionar suscripción	4
Total			11.2

2.4.3. Plan de entregas

Con el propósito de mantener al cliente centrado en la mejora del producto, y permitir que sea él quien realice sus propias pruebas de campo, se efectúan múltiples entregas del sistema. A continuación, se presenta una tabla que ilustra lo mencionado, especificando la fecha en la que se lleva a cabo la reunión entre el equipo de desarrollo y los propietarios del software para la liberación de las distintas versiones del producto.

Tabla 2.11. Plan de entregas. Fuente: (Elaboración propia).

Iteración	Fecha de entrega
Iteración 1	22 de mayo del 2023
Iteración 2	5 de junio del 2023
Iteración 3	3 de julio del 2023
Iteración 4	31 de julio del 2023

2.5. Diseño del sistema

El diseño XP sigue rigurosamente el principio MS (mantenlo sencillo). Un diseño sencillo siempre se prefiere sobre una representación más compleja. XP estimula el uso de las tarjetas CRC como un mecanismo eficaz para pensar en el software en un contexto orientado a objetos. 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. (Pressman, 2010)

2.5.1. Patrón arquitectónico

Patrón de arquitectura: Son reglas que determinan el contexto bajo el cual se llevará a cabo el desarrollo, estas reglas tienen como finalidad la obtención de las características esperadas del software en cuestión. Los patrones arquitectónicos de software definen un enfoque específico para el manejo de alguna característica de comportamiento del sistema. (Pressman, 2010)

Modelo Vista Controlador

El Modelo Vista Controlador (Model-View-Controller MVC) es un patrón de diseño arquitectónico de software, que sirve para clasificar la información, la lógica del sistema y la interfaz que se le presenta al usuario. En este tipo de arquitectura existe un sistema central o controlador que gestiona las entradas y la salida del sistema, uno o varios modelos que se encargan de buscar los datos e información necesaria y una interfaz que muestra los resultados al usuario final. (Sunardi y Suharjito, 2019)

El patrón Modelo Vista Controlador posee los siguientes componentes (Giridhar, 2016):

- Modelo: este componente se encarga de manipular, gestionar y actualizar los datos. Si se utiliza una base de datos aquí es donde se realizan las consultas, búsquedas, filtros y actualizaciones.
- Vista: este componente se encarga de mostrarle al usuario final las pantallas, ventanas, páginas y formularios; el resultado de una solicitud. Desde la perspectiva del programador este componente es el que se encarga del frontend; la programación de la interfaz de usuario si se trata de una aplicación de escritorio, o bien, la visualización de las páginas web (CSS, HTML, HTML5 y Javascript).
- Controlador: este componente se encarga de gestionar las instrucciones que se reciben, atenderlas y procesarlas. Por medio de él se comunican el modelo y la vista: solicitando los datos necesarios; manipulándolos para obtener los resultados; y entregándolos a la vista para que pueda mostrarlos.

2.5.2. Patrones de diseño

Patrones de diseño: es una solución estándar a un problema, los patrones de diseño definen y organizan un vocabulario de elementos básicos en un conjunto de componentes y conectores de

tal manera que se garanticen una adecuada composición de los elementos que lo conforman según esquema de organización asociado. (Blas, 2019)

Patrones GRASP (General Responsibility Assignment Software Patterns - Patrones Generales de Software para Asignar Responsabilidades)

Ayudan al entendimiento y la aplicación de los conceptos sobre la Programación Orientada a Objetos y a diseñar software con mayor calidad aplicando buenas prácticas que provienen de la experiencia de años de desarrollo. (G. A. V. Ortega, 2021)

Experto: Este patrón sugiere asignar responsabilidades a las clases que tienen la información necesaria para cumplirlas. Esto puede requerir información distribuida en otros objetos, es decir, otros expertos. (G. A. V. Ortega, 2021)

Figura 2.10. Evidencia del patrón Experto. Fuente: (Elaboración propia).

```
<MedCard
  setActiveCard={setActiveCard}
  id={m.id}
  setPharmacyFound={setPharmacyFound}
  key={m.id}
  nombre={m.nombre}
  descripcion={m.descripcion}
  precioDeVenta={m.precioDeVenta}
  tipoDeVenta={m.tipoDeVenta}
  grupoFarmacologico={m.grupoFarmacologico}
  formaDeDosificacion={m.formaDeDosificacion}
/>
```

Creador: Este patrón se refiere a qué clases deben ser responsables de crear instancias u objetos de otra clase. (G. A. V. Ortega, 2021)

Capítulo 2. Análisis y diseño

Figura 2.11. Evidencia del patrón Creador. Fuente: (Elaboración propia).

```
<LayersControl position="topright">
  <LayersControl.Overlay checked name="Farmacias">
    <LayerGroup>{arrayIcons}</LayerGroup>
  </LayersControl.Overlay>
  <LayersControl.BaseLayer checked name="OSM">
    <LayerGroup>
      <TileLayer
        | url={ "http://tile.openstreetmap.org/{z}/{x}/{y}.png" }
      />
      ;
    </LayerGroup>
  </LayersControl.BaseLayer>
  <LayersControl.BaseLayer name="Híbrido">
    <LayerGroup>
      <TileLayer
        | url={ "http://mtl.google.com/vt/lyrs=y&x={x}&y={y}&z={z}" }
      />
      ;
    </LayerGroup>
  </LayersControl.BaseLayer>
  <LayersControl.BaseLayer name="Satelital">
    <LayerGroup>
      <TileLayer
        | url={
          | "http://www.google.cn/maps/vt?lyrs=s@189&gl=cn&x={x}&y={y}&z={z}"
        }
      />
      ;
    </LayerGroup>
  </LayersControl.BaseLayer>
</LayersControl>
```

Bajo Acoplamiento: Este patrón busca evitar la gran dependencia generada entre las clases que forman parte del sistema. El objetivo es tener poca dependencia entre las clases, ya que una alta dependencia entre clases creará una gran cantidad de software. (G. A. V. Ortega, 2021)

Capítulo 2. Análisis y diseño

Figura 2.12. Evidencia del patrón Bajo Acoplamiento. Fuente: (Elaboración propia).



```
167 |         </LayersControl.BaseLayer>
168 |     </LayersControl>
169 |
170 |     <DraggableMarker />
171 |
172 |     <LocationButton />
173 |
174 |     <RestartButton />
175 |
176 |     {newLocation.length === 0 && <ManualLocation />}
177 |
178 |     {wp.length > 1 && newLocation.length === 0 && (
179 |         <RoutingMachine waypoints={wp} />
180 |     )}
181 |     {newLocation.length > 0 && (
182 |         <RoutingMachine waypoints={[location, newLocation]} />
183 |     )}
184 | </MapContainer>
185 | </>
186 | );
187 | };
188 | export default Map;
```

Alta Cohesión: Este patrón busca crear clases que contengan la suficiente información, sin llegar al exceso de contenido, para que puedan funcionar de manera óptima y así evitar la alta dependencia con otras clases que le suministren datos necesarios para su existencia. (G. A. V. Ortega, 2021)

Figura 2.13. Evidencia del patrón Alta Cohesión. Fuente: (Elaboración propia).



```
304 | <Button
305 |   variant="contained"
306 |   onClick={
307 |     updateSubscription
308 |     ? handleSubmit(postUpdate)
309 |     : handleSubmit(postForm)
310 |   }
311 | >
312 |   {updateSubscription ? "Modificar Suscripción" : "Suscribirse"}
313 | </Button>
```

Patrones GOF (Gang of Four - El grupo de los Cuatro)

Es un patrón para solucionar problemas de diseño recurrentes. Clasifica los patrones en 3 grandes categorías basadas en su propósito: creacionales, estructurales y de comportamiento. (Soto et al., 2021)

Comportamiento: Proporcionan una solución para estructurar datos y los objetos para así organizar el procesamiento entre objetos. (Soto et al., 2021)

Observador: Es un patrón de comportamiento que notifica automáticamente a otros objetos cuando su estado cambia.

Figura 2.14. Evidencia del patrón Observador. Fuente: (Elaboración propia).

```
setInterval(async () => {  
  const oldPharmacy = await dataBasePharmacy.find({});  
  if (oldPharmacy.length === 0) {  
    await savePharmacy();  
  } else {  
    await axios  
      .get("http://localhost:5001/getPharmacy")  
      .then(async (r) => {  
        const newDataBasePharmacy = r.data.farmacias.map((f) => {  
          return {  
            id: f.id,  
            idMedicamentos: f.idMedicamentos,  
          });  
        });  
  
        const newEntrance = diferenciaDataBaseMedicines(  
          oldPharmacy[0].pharmacy,  
          newDataBasePharmacy  
        );  
  
        if (newEntrance.length > 0) {  
          await dataBasePharmacy  
            .updateOne({  
              pharmacy: newDataBasePharmacy,  
            })  
            .then()  
            .catch((e) => console.log("Algo salió mal"));  
          await notifyUsers(newEntrance, r.data.farmacias);  
        }  
      })  
      .catch((e) => console.log("Algo salió mal", e));  
}, 1000 * 60 * 60);
```

2.5.3. Tarjetas CRC

Para el diseño de las aplicaciones, la metodología XP no requiere la representación del sistema mediante diagramas de clases utilizando notación Lenguaje Unificado de Modelado (UML) por

Capítulo 2. Análisis y diseño

sus siglas del inglés Unified Modeling Language. En su lugar se usan otras técnicas como las tarjetas CRC como una extensión informal de UML.

Tarjetas CRC (Clase-Responsabilidad-Colaborador): Estas tarjetas ayudan a identificar y organizar fácilmente las clases más relevantes para los requerimientos de un sistema; además, se dividen en tres secciones: en la parte superior se encuentra el nombre de la clase, en la parte izquierda se enlistan todas las acciones que la clase pueda realizar (responsabilidades) y en la parte derecha, otras clases que interactúen con la actual (colaboradores), ayudándola a llevar a cabo sus responsabilidades. Las tarjetas CRC son el único producto del trabajo de diseño que se genera como parte del proceso XP. (Pressman, 2010)

Las tarjetas CRC elaboradas son las siguientes:

Tabla 2.12. Tarjeta CRC 1. Fuente: (Elaboración propia).

Tarjeta CRC	
Clase: Suscripción	
Responsabilidad	Colaboración
Almacenar la información referente a la suscripción del usuario.	mongoose Código_de_Validación

Tabla 2.13. Tarjeta CRC 2. Fuente: (Elaboración propia).

Tarjeta CRC	
Clase: Código_de_Validación	
Responsabilidad	Colaboración
Almacenar la información referente a los códigos de validación de usuarios.	mongoose Suscripción

Capítulo 2. Análisis y diseño

Tabla 2.14. Tarjeta CRC 3. Fuente: (Elaboración propia).

Tarjeta CRC	
Clase: Datos_de_Farmacias	
Responsabilidad	Colaboración
Almacenar la información referente a todas las farmacias.	mongoose

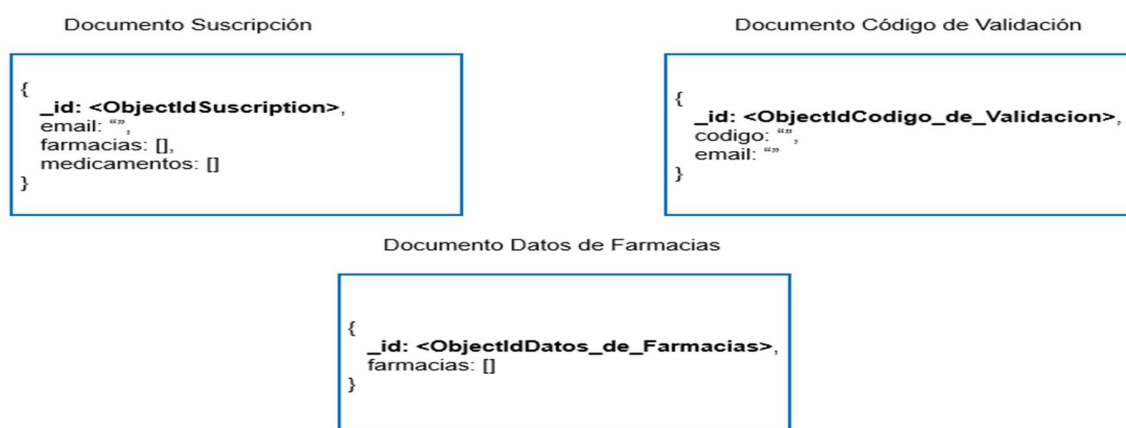
2.5.4. Modelo de datos

Un modelo es un conjunto de herramientas conceptuales para describir datos, sus relaciones, su significado y sus restricciones de consistencia. (*Data Modeling Introduction — MongoDB Manual*, s. f.)

MongoDB es un sistema de base datos potente, flexible, de alto rendimiento, de expansión fácil de almacenamiento de datos. Las entidades son representadas en forma de documentos orientados y es no relacional. (*Data Model Design — MongoDB Manual*, s. f.)

La solución propuesta cuenta con 3 documentos y se puede observar a continuación.

Figura 2.15. Modelo de datos. Fuente: (Elaboración propia).



2.6. Conclusiones parciales

Sobre este capítulo se puede concluir que:

- La identificación de 8 historias de usuario en la fase de planeación de la metodología permitió determinar cuáles son las partes del software a ser implementado con prioridad.
- La estimación del tiempo identificó que el proyecto era entregable en el tiempo adecuado según XP y según el fondo de tiempo para un trabajo de este tipo.
- La elaboración de 3 tarjetas CRC facilitó la detección temprana de errores en el diseño del sistema.
- El uso de los patrones MVC, GRASP y GOF proporcionó un criterio objetivo para asignar responsabilidades a las clases y definir las relaciones que se establecen entre ellas, permitiendo un diseño flexible y fácil de mantener.

Capítulo 3: Implementación y pruebas

3.1. Introducción

Una vez finalizada la fase de diseño, que establece los fundamentos estructurales del sistema, se da inicio a la etapa de implementación y pruebas. Durante este período, las Historias de Usuario (HU) se desglosan en Tareas de Ingeniería (TI) para simplificar su comprensión e implementación. Se desarrollan pruebas unitarias, se llevan a cabo las TI y se realizan pruebas de aceptación. La fase de implementación y prueba representa el intervalo con los resultados más tangibles y significativos, ya que culmina con un sistema terminado y validado por el cliente. Estas pruebas son vitales para garantizar que el sistema funcione como se espera y para identificar cualquier problema o error antes de que el sistema se ponga en producción.

Este capítulo aborda elementos como el diagrama de despliegue, los estándares de codificación que regulan el código producido, las iteraciones llevadas a cabo para la implementación de las HU, las pruebas realizadas y sus resultados.

3.2. Implementación

Las funcionalidades son desarrolladas en esta fase, generando al final de cada una un entregable funcional que implementa las HU asignadas a la iteración. Como las HU no tienen suficiente detalle como para permitir su análisis y desarrollo, al principio de cada iteración se realizan las TI necesarias de análisis, recabando con el cliente todos los datos que sean necesarios.

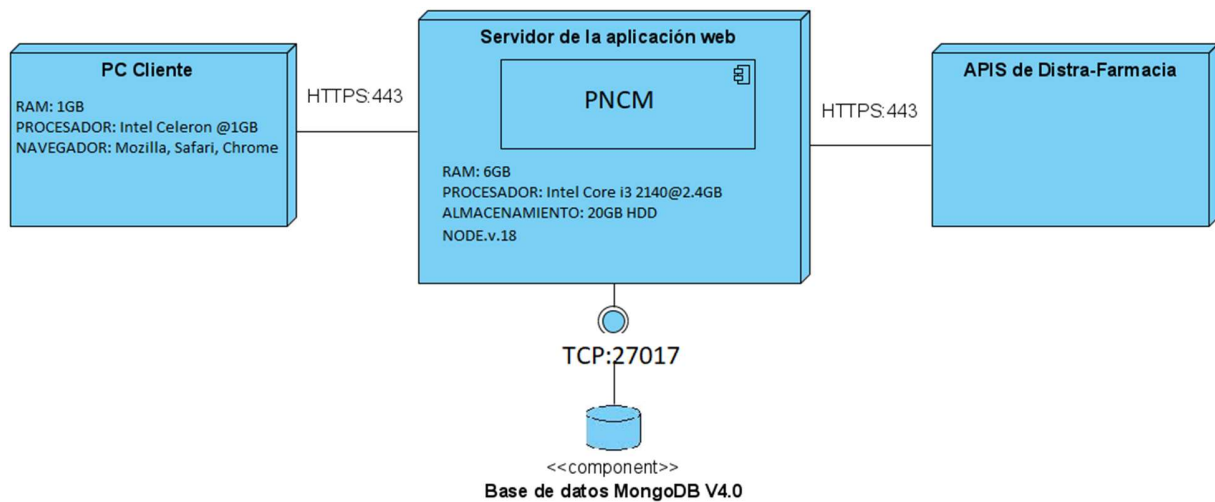
El cliente, por lo tanto, también debe participar activamente durante esta fase del ciclo. Las iteraciones son también utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida clara de avance.

3.2.1. Diagrama de despliegue

El modelo de despliegue consiste en un modelo de objetos que tiene como objetivo describir la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre las estaciones de trabajo (denominados nodos). La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Los nodos representan objetos físicos existentes en tiempo de ejecución, sirven para modelar recursos que tiene memoria y capacidad de proceso y puede ser tanto ordenadores como dispositivos, memoria o personas. (Sommerville, 2011) A continuación, se muestra el diagrama de despliegue para la solución propuesta.

Capítulo 3. Implementación y pruebas

Figura 3.1. Diagrama de despliegue. Fuente: (Elaboración propia).



A continuación, se muestra una descripción del diagrama de despliegue propuesto para la PNCM.

PC cliente: Esta computadora es la que utilizan los usuarios para interactuar con el sitio web.

Servidor: Ordenador de alto rendimiento donde estará alojada y configurada la plataforma.

APIs de Distrá-Farmacia: Servicios que ofrecen datos sobre medicamentos y farmacias para ser consumidos en la PNCM.

Base de datos MongoDB: Es la base de datos que utiliza la PNCM para guardar información referente a la suscripción del usuario, los códigos de validación y las farmacias.

HTTPS: Protocolo de transferencia de hipertexto seguro.

TCP: Protocolo de la capa de transporte

3.2.2. Estándares de codificación

Los estándares de codificación o programación son términos que describen convenciones para escribir código fuente en los lenguajes de programación. En este caso se utilizó el estándar de codificación de JavaScript. (*GitHub - Ogranada/Js-Coding-Standards*, 2023)

- Codificación UTF-8 en todos los archivos.
- Evitar usar espacios en blanco innecesarios.
- Los espacios entre el nombre y el valor de una variable son muy importantes, puesto que mejoran la legibilidad del código.

Capítulo 3. Implementación y pruebas

- Las llaves de las funciones deben empezar en la misma línea que se declara la función y debe haber un espacio entre el paréntesis de cierre de argumentos y la llave de inicio de cuerpo de función.
- Las importaciones deben estar en líneas separadas.

Figura 3.2. Uso de importaciones. Fuente: (Elaboración propia).

```
1  import React, { useContext, useEffect } from "react";
2  import Search from "../Search/Search";
3  import "../home.css";
4  import Context from "../../Components/Context/Context";
5
```

- Los nombres de variables y funciones deben ser escritos con notación camelCase (es una convención de nomenclatura en la programación en la que se escribe una frase sin espacios, pero con cada palabra después de la primera comenzando con una letra mayúscula).

Figura 3.3. Uso de la notación camelCase. Fuente: (Elaboración propia).

```
3  export function myIcon(url) {
4    return new Icon({
5      iconUrl: url,
6      iconSize: [38, 38],
7      iconAnchor: [22, 34],
8      popupAnchor: [-10, -70],
9    });
10 }
```

- Uso de comentarios en algunas partes del código que pueden resultar complejas.

Figura 3.4. Estándar para comentarios. Fuente: (Elaboración propia).

```
12  //Algoritmo de Haversine
13  export function getKilometros(lat1, lon1, lat2, lon2) {
14      const rad = function (x) {
15          return (x * Math.PI) / 180;
16      };
17      let R = 6378.137; //Radio de la tierra en km
18      let dLat = rad(lat2 - lat1);
19      let dLong = rad(lon2 - lon1);
20      let a =
21          Math.sin(dLat / 2) * Math.sin(dLat / 2) +
22          Math.cos(rad(lat1)) *
23          Math.cos(rad(lat2)) *
24          Math.sin(dLong / 2) *
25          Math.sin(dLong / 2);
26      let c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
27      let d = R * c;
28      return d.toFixed(3); //Retorna tres decimales
29  }
30
```

3.3. Desarrollo por iteraciones

En la fase de implementación y pruebas se realizan las iteraciones previstas. Durante el transcurso de las iteraciones se realiza la implementación de las historias de usuario seleccionadas para cada una de estas. Al inicio de las mismas, se lleva a cabo una revisión del plan de iteraciones y se modifica de ser necesario. Como parte de este plan, se descomponen las HU en TI. Estas tareas son para el uso de los programadores, pueden escribirse utilizando un lenguaje técnico y no necesariamente deben ser comprensibles para el cliente. Antes de implementar las tareas, se elaboran los test unitarios para cada una de ellas, los cuales prueban la funcionalidad y detallan los fallos de la misma. Una vez culminada la iteración se realiza la entrega de una versión del producto al propietario del software, y se ejecutan las pruebas de aceptación, validando la satisfacción del cliente con el sistema. (Beck, 2000)

Siguiendo el plan establecido, se realizaron cuatro iteraciones de desarrollo en el sistema, resultando en un producto listo para su despliegue. A continuación, se detallan cada una de las iteraciones.

Capítulo 3. Implementación y pruebas

3.3.1. Iteración 1

Durante esta iteración se abordaron las HU de mayor prioridad y se construyó la base de la arquitectura del sistema con el fin de obtener un producto con las funcionalidades críticas para ser mostrado al cliente y obtener una rápida retroalimentación de este. Se implementaron las HU: Buscar Medicamentos y Buscar Farmacias. Para su implementación las mismas fueron divididas en TI.

Tabla 3.1. TI1: Buscar medicamentos. Fuente: (Elaboración propia).

Tarea de Ingeniería Buscar Medicamentos	
Número de tarea: 1	Número de HU: 1
Tipo de Tarea: Desarrollo	Puntos estimados: 0.6
Fecha de inicio: 11 de mayo del 2023	Fecha de fin: 15 de mayo del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: 1. Entender la API: El programador debe familiarizarse con la documentación de la API proporcionada por DISTRA. Esto incluye entender los endpoints disponibles, los parámetros de solicitud que estos endpoints aceptan y las respuestas que devuelven. 2. Diseño de la Interfaz de Usuario (UI): Diseñar una interfaz de usuario intuitiva para la búsqueda de medicamentos. Esto podría incluir una barra de búsqueda para introducir el nombre del medicamento y botones para iniciar la búsqueda y limpiar los resultados. 3. Manejo de Errores de Entrada del Usuario: Implementar una validación de la entrada del usuario para asegurar que se ha introducido un término de búsqueda válido. Esto incluye mostrar un mensaje de error si el usuario intenta buscar sin introducir nada, si la búsqueda no devuelve resultados o si el usuario intenta buscar con menos de tres caracteres. 4. Procesamiento de la Respuesta de la API: Una vez que la API devuelve una respuesta, esta debe ser procesada. Esto podría implicar extraer la información relevante de la respuesta y formatearla de una manera que pueda ser fácilmente mostrada en la UI.	

Capítulo 3. Implementación y pruebas

5. Mostrar los Resultados de la Búsqueda: Los resultados de la búsqueda deben ser mostrados al usuario de una manera fácil de entender. Esto podría implicar mostrar la información del medicamento en una tabla o lista.

Tabla 3.2. TI2: Buscar farmacias. Fuente: (Elaboración propia).

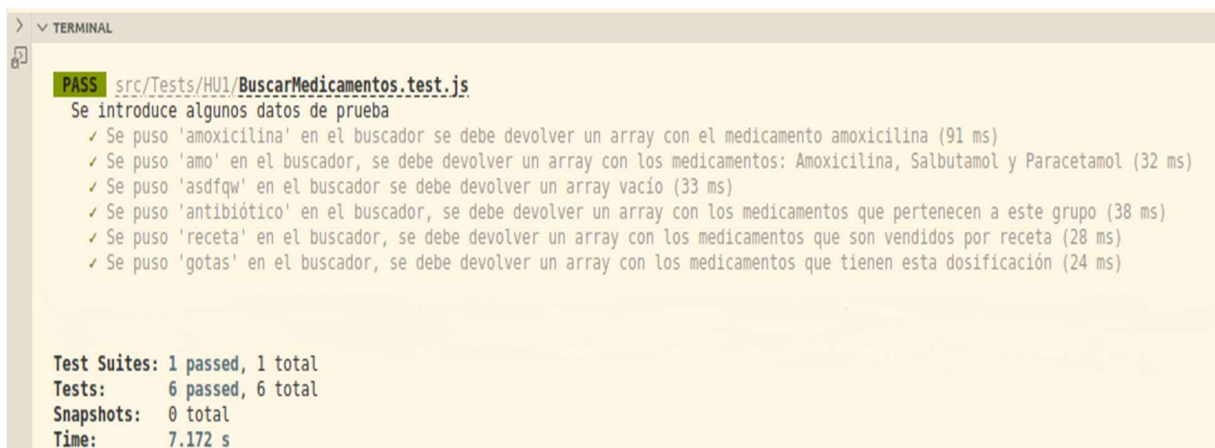
Tarea de Ingeniería Buscar Farmacias	
Número de tarea: 2	Número de HU: 2
Tipo de Tarea: Desarrollo	Puntos estimados: 0.6
Fecha de inicio: 16 de mayo del 2023	Fecha de fin: 19 de mayo del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: 1. Entender la API: El programador debe familiarizarse con la documentación de la API proporcionada por DISTRA. Esto incluye entender los endpoints disponibles, los parámetros de solicitud que estos endpoints aceptan y las respuestas que devuelven. 2. Diseño de la Interfaz de Usuario (UI): Reutilizar la interfaz de usuario anteriormente implementada para realizar nuevas búsquedas de las farmacias. 3. Manejo de Errores de Entrada del Usuario: Implementar una validación de la entrada del usuario para asegurar que se ha introducido un término de búsqueda válido. Esto incluye mostrar un mensaje de error si el usuario intenta buscar sin introducir nada, si la búsqueda no devuelve resultados o si el usuario intenta buscar con menos de tres caracteres. 4. Procesamiento de la Respuesta de la API: Una vez que la API devuelve una respuesta, esta debe ser procesada. Esto podría implicar extraer la información relevante de la respuesta y formatearla de una manera que pueda ser fácilmente mostrada en la UI. 5. Mostrar los Resultados de la Búsqueda: Los resultados de la búsqueda deben ser mostrados al usuario de una manera fácil de entender. Esto podría implicar mostrar la información de las farmacias en una tabla o lista.	

Pruebas unitarias

En XP, el desarrollo dirigido por tests o pruebas es una de las prácticas que guían la etapa de implementación y pruebas. Las pruebas unitarias permiten saber si cada función en un archivo de programa simple funciona correctamente, ayudando a independizar un módulo y facilitando la posibilidad de realizar una prueba independientemente del resto del sistema. El objetivo principal de estos tests es aislar, cada parte del programa y mostrar que las partes individuales son correctas, brindando coherencia entre código escrito y el resultado que debe proporcionar. (Rodríguez & Bonilla, 2020) El requisito mínimo que se establece para culminar una iteración es que todas las funcionalidades implementadas superen cada uno de sus test unitarios.

En las Figuras 3.5 y 3.6 se muestran los resultados de la ejecución de los test unitarios de la primera iteración.

Figura 3.5. Test unitario 1 de la primera iteración. Fuente: (Elaboración propia).



```
> ▾ TERMINAL
8
PASS src/Tests/HUI/BuscarMedicamentos.test.js
Se introduce algunos datos de prueba
  ✓ Se puso 'amoxicilina' en el buscador se debe devolver un array con el medicamento amoxicilina (91 ms)
  ✓ Se puso 'amo' en el buscador, se debe devolver un array con los medicamentos: Amoxicilina, Salbutamol y Paracetamol (32 ms)
  ✓ Se puso 'asdfqw' en el buscador se debe devolver un array vacío (33 ms)
  ✓ Se puso 'antibiótico' en el buscador, se debe devolver un array con los medicamentos que pertenecen a este grupo (38 ms)
  ✓ Se puso 'receta' en el buscador, se debe devolver un array con los medicamentos que son vendidos por receta (28 ms)
  ✓ Se puso 'gotas' en el buscador, se debe devolver un array con los medicamentos que tienen esta dosificación (24 ms)

Test Suites: 1 passed, 1 total
Tests:      6 passed, 6 total
Snapshots:  0 total
Time:       7.172 s
```

Capítulo 3. Implementación y pruebas

Figura 3.6. Test unitario 2 de la primera iteración. Fuente: (Elaboración propia).

```
> v TERMINAL
PASS src/Tests/HU2/BuscarFarmacias.test.js
Se introduce algunos datos de prueba
✓ Se puso 'Farmacia 10' en el buscador se debe devolver un array con la Farmacia 10 (96 ms)
✓ Se puso 'La Habana' en el buscador, se debe devolver un array con las Farmacias que pertenecen a esta provincia (32 ms)
✓ Se puso 'Hospitalaria' en el buscador se debe devolver un array con todas las farmacias que tienen esta clasificación (32 ms)
✓ Se puso 'Playa' en el buscador se debe devolver un array con todas las farmacias que pertenecen a este municipio (23 ms)

Test Suites: 1 passed, 1 total
Tests:      4 passed, 4 total
Snapshots:  0 total
Time:       8.014 s
```

3.3.2. Iteración 2

Durante esta iteración se implementaron las HU: Geolocalización de las farmacias del país en el mapa, Mostrar información sobre las farmacias en el mapa, Geolocalizar Usuario. Para ello se definieron 3 TI.

Tabla 3.3. TI3: Geolocalizar las farmacias del país. Fuente: (Elaboración propia).

Tarea de Ingeniería Geolocalizar las Farmacias	
Número de tarea: 3	Número de HU: 3
Tipo de Tarea: Desarrollo	Puntos estimados: 0.6
Fecha de inicio: 22 de mayo del 2023	Fecha de fin: 24 de mayo del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: 1. Entender la API: El programador debe familiarizarse con la documentación de la API proporcionada por DISTRA. Esto incluye entender los endpoints disponibles, los parámetros de solicitud que estos endpoints aceptan y las respuestas que devuelven. 2. Diseño de la Interfaz de Usuario (UI): Diseñar una interfaz de usuario intuitiva para la geolocalización. Esto podría incluir un mapa interactivo y	

Capítulo 3. Implementación y pruebas

un campo de entrada para que el usuario pueda introducir la ubicación que desea geolocalizar.
3. Procesamiento de la Respuesta de la API: Una vez que la API devuelve una respuesta, esta debe ser procesada. Esto podría implicar extraer las coordenadas de la ubicación de la respuesta.
4. Marcar la Ubicación en el Mapa: Usar las coordenadas obtenidas de la API para marcar la ubicación en el mapa. Esto podría implicar añadir un ícono en las coordenadas correspondientes en el mapa.

Tabla 3.4. TI4: Mostrar información sobre las farmacias en el mapa. Fuente: (Elaboración propia).

Tarea de Ingeniería Mostrar Información de Farmacias en el Mapa	
Número de tarea: 4	Número de HU: 4
Tipo de Tarea: Desarrollo	Puntos estimados: 0.6
Fecha de inicio: 25 de mayo del 2023	Fecha de fin: 29 de mayo del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: 1. Entender la API: El programador debe familiarizarse con la documentación de la API proporcionada por DISTRA. Esto incluye entender los endpoints disponibles, los parámetros de solicitud que estos endpoints aceptan y las respuestas que devuelven. 2. Diseño de la Interfaz de Usuario (UI): Diseñar una interfaz de usuario intuitiva para mostrar un marcador en el mapa que represente la posición de la farmacia ubicada, que muestre un Popup al hacer clic sobre ella. 3. Procesamiento de la Respuesta de la API: Una vez que la API devuelve una respuesta, esta debe ser procesada. Esto podría implicar extraer la información que se va a mostrar en el Popup. 4. Mostrar el Popup de Información: Cuando el usuario hace clic en un marcador, se debe mostrar el Popup de información con los datos obtenidos de la API.	

Capítulo 3. Implementación y pruebas

Tabla 3.5. T15: Geolocalizar al usuario. Fuente: (Elaboración propia).

Tarea de Ingeniería Geolocalizar al usuario	
Número de tarea: 5	Número de HU: 5
Tipo de Tarea: Desarrollo	Puntos estimados: 0.8
Fecha de inicio: 30 de mayo del 2023	Fecha de fin: 2 de junio del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: 1. Acceso a la Geolocalización del Usuario: El programador debe implementar una función que solicite el permiso del usuario para acceder a su ubicación geográfica. Esta función debe manejar adecuadamente cualquier error que pueda ocurrir durante la solicitud, como, por ejemplo, si el usuario deniega el permiso. 2. Obtener la Geolocalización del Usuario: Si el usuario otorga el permiso, la aplicación debe obtener las coordenadas de geolocalización del usuario. Esto puede hacerse utilizando la API de Geolocalización del navegador o una API proporcionada por un tercero. 3. Diseño de la Interfaz de Usuario (UI): Diseñar una interfaz de usuario intuitiva para el mapa. Esto podría incluir un mapa interactivo que muestre la ubicación actual del usuario. 4. Mostrar el Marcador en el Mapa: Usar las coordenadas obtenidas para mostrar un marcador en la ubicación del usuario en el mapa. Esto podría implicar añadir un ícono en las coordenadas correspondientes en el mapa.	

Test Unitarios

En las Figuras 3.7 y 3.8 se muestran los resultados de la ejecución de los test unitarios de la segunda iteración.

Capítulo 3. Implementación y pruebas

Figura 3.7. Test unitario 1 de la segunda iteración. Fuente: (Elaboración propia).

```
> ▼ TERMINAL
PASS src/Tests/HU3/GeolocalizarFarmacia.test.js
Se realizaron pruebas al mapa
  ✓ Se trata de renderizar el mapa sin pasarle ningunas coordenadas de farmacias (175 ms)
  ✓ Se trata de renderizar el mapa con coordenadas de farmacias (41 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        5.042 s
```

Figura 3.8. Test unitario 2 de la segunda iteración. Fuente: (Elaboración propia).

```
> ▼ TERMINAL
PASS src/Tests/HU4/MostrarInfoFarmacias.test.js
Se renderizó el mapa con información de las farmacias
  ✓ Se espera que la info del teléfono de la farmacia sea 14229499 (215 ms)
  ✓ Se espera que la info del nombre de la farmacia sea Farmacia 1 (43 ms)
  ✓ Se espera que la info de la dirección de la farmacia sea Calle 35 e/80 y 7 (40 ms)
  ✓ Se espera que la info de las coordenadas de la farmacia sea 21.425009456585325, -83.7244758468594 (31 ms)

Test Suites: 1 passed, 1 total
Tests:       4 passed, 4 total
Snapshots:   0 total
Time:        7.865 s
```

3.3.3. Iteración 3

Durante el desarrollo de esta iteración se implementaron las HU: Resaltar las farmacias más cercanas a la posición del usuario y Calcular la ruta a la farmacia seleccionada. Para ello se definieron 2 TI.

Capítulo 3. Implementación y pruebas

Tabla 3.6. T16: Resaltar las farmacias más cercanas a la posición del usuario. Fuente: (Elaboración propia).

Tarea de Ingeniería Resaltar las Farmacias Cercanas	
Número de tarea: 6	Número de HU: 6
Tipo de Tarea: Desarrollo	Puntos estimados: 1
Fecha de inicio: 5 de junio del 2023	Fecha de fin: 9 de junio del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: 1. Entender el Algoritmo de Haversine: El programador debe familiarizarse con el algoritmo de Haversine, que se utiliza para calcular la distancia entre dos puntos en la superficie de una esfera a partir de sus longitudes y latitudes. 2. Obtener las Coordenadas de los Marcadores: El programador debe implementar una función que obtenga las coordenadas de todos los marcadores en el mapa. 3. Calcular la Distancia a la Posición Específica: Para cada marcador, el programador debe implementar una función que utilice el algoritmo de Haversine para calcular la distancia entre el marcador y la posición específica. 4. Resaltar los Marcadores: Si la distancia calculada es menor o igual a 5km, el marcador debe ser resaltado con otro color. Esto podría implicar cambiar el ícono del marcador o cambiar su color. 5. Diseño de la Interfaz de Usuario (UI): Diseñar una interfaz de usuario intuitiva que permita al usuario ver claramente los marcadores resaltados.	

Tabla 3.7. T17: Calcular la ruta a la farmacia seleccionada. Fuente: (Elaboración propia).

Tarea de Ingeniería Calcular Ruta a la Farmacia Seleccionada	
Número de tarea: 7	Número de HU: 7
Tipo de Tarea: Desarrollo	Puntos estimados: 3
Fecha de inicio: 12 de junio del 2023	Fecha de fin: 30 de junio del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción:	

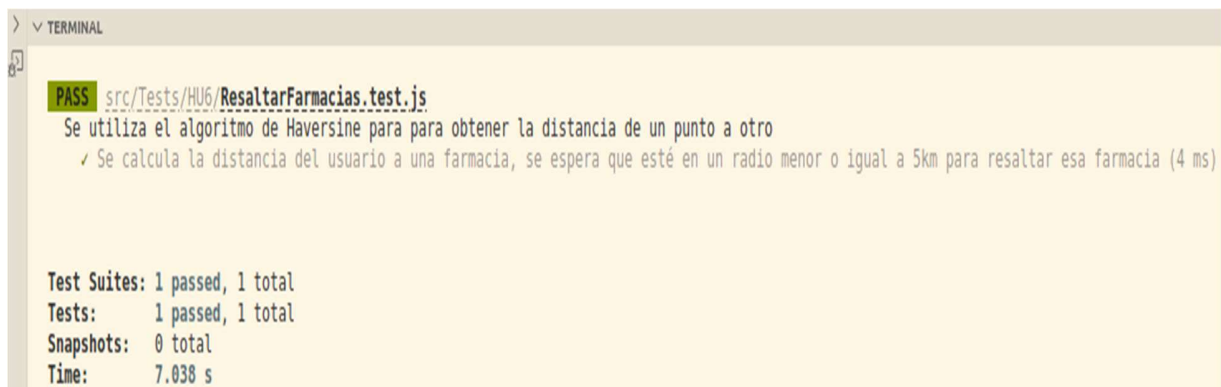
Capítulo 3. Implementación y pruebas

1. Entender la API y la Librería: El programador debe familiarizarse con la documentación de la API que proporciona la ruta entre dos puntos y con la librería `leaflet-routing-machine` que se utilizará para dibujar la ruta en el mapa.
2. Obtener las Coordenadas de los Puntos: El programador debe implementar una función que obtenga las coordenadas del punto de partida y del punto de llegada. Esto puede hacerse utilizando la API del mapa o una API proporcionada por un tercero.
3. Solicitud a la API de Rutas: Implementar una función que haga una solicitud a la API de rutas con las coordenadas del punto de partida y del punto de llegada. Esta función debe manejar adecuadamente cualquier error que pueda ocurrir durante la solicitud.
4. Procesamiento de la Respuesta de la API: Una vez que la API devuelve una respuesta, esta debe ser procesada. Esto podría implicar extraer la ruta de la respuesta.
5. Dibujar la Ruta en el Mapa: Usar la librería `leaflet-routing-machine` para dibujar la ruta en el mapa. La ruta debe ser de color negro y blanco.

Test Unitarios

En las Figuras 3.9 y 3.10 se muestran los resultados de la ejecución de los test unitarios de la tercera iteración.

Figura 3.9. Test unitario 1 de la tercera iteración. Fuente: (Elaboración propia).



```
> TERMINAL
PASS src/Tests/HU6/ResaltarFarmacias.test.js
Se utiliza el algoritmo de Haversine para obtener la distancia de un punto a otro
✓ Se calcula la distancia del usuario a una farmacia, se espera que esté en un radio menor o igual a 5km para resaltar esa farmacia (4 ms)

Test Suites: 1 passed, 1 total
Tests:      1 passed, 1 total
Snapshots:  0 total
Time:       7.038 s
```

Capítulo 3. Implementación y pruebas

Figura 3.10. Test unitario 2 de la tercera iteración. Fuente: (Elaboración propia).

```
> v TERMINAL
PASS src/Tests/HU7/CalcularRuta.test.js (5.313 s)
Se envía un array con un par de coordenadas para trazar la ruta entre estos dos puntos haciendo uso de la api de leaflet-routing-machine
✓ Se espera que la api devuelva la ruta y se renderice en el mapa (1488 ms)

Test Suites: 1 passed, 1 total
Tests:      1 passed, 1 total
Snapshots:  0 total
Time:       9.28 s
```

3.3.4. Iteración 4

Durante esta iteración se implementó la HU: Gestionar suscripción, para ello se definieron 3 TI.

Tabla 3.8. TI8: Suscripción del usuario. Fuente: (Elaboración propia).

Tarea de Ingeniería Suscripción del Usuario	
Número de tarea: 8	Número de HU: 8
Tipo de Tarea: Desarrollo	Puntos estimados: 1.4
Fecha de inicio: 3 de julio del 2023	Fecha de fin: 11 de julio del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: 1. Diseño del Formulario de Suscripción: Diseñar un formulario de suscripción que solicite al usuario su correo electrónico, las farmacias y los medicamentos de su interés. El formulario debe permitir seleccionar como máximo 5 farmacias y 5 medicamentos. 2. Verificación del Correo Electrónico: Implementar una función que envíe un código de verificación al correo electrónico proporcionado por el usuario. Esta función debe manejar adecuadamente cualquier error que pueda ocurrir durante el envío del correo electrónico.	

Capítulo 3. Implementación y pruebas

3. Validación del Código de Verificación: Implementar una función que valide el código de verificación introducido por el usuario. Si el código es correcto, se debe proceder con la suscripción. Si el código es incorrecto, se debe mostrar un mensaje de error al usuario.
4. Suscripción del Usuario: Si el código de verificación es correcto, el usuario debe ser suscrito a la PNCM y mostrar un mensaje al usuario. Esto podría implicar almacenar la información del usuario en una base de datos y configurar las notificaciones para las farmacias y medicamentos seleccionados.
5. Manejo de Errores del Formulario: Implementar una validación de los datos introducidos en el formulario para asegurar que son válidos. Esto incluye mostrar un mensaje de error si el usuario intenta suscribirse sin introducir un correo electrónico, si selecciona más de 5 farmacias o medicamentos, o si introduce un código de verificación incorrecto.

Tabla 3.9. TI9: Modificar suscripción. Fuente: (Elaboración propia).

Tarea de Ingeniería Modificar Suscripción	
Número de tarea: 9	Número de HU: 8
Tipo de Tarea: Desarrollo	Puntos estimados: 1.6
Fecha de inicio: 12 de julio del 2023	Fecha de fin: 21 de julio del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: 1. Diseño del Formulario de Suscripción: Diseñar un formulario que permita al usuario modificar su suscripción. Esto podría incluir campos para cambiar las farmacias y los medicamentos de su interés, con un límite de cinco seleccionados por cada uno. 2. Verificación del Correo Electrónico: Implementar una función que envíe un código de verificación al correo electrónico proporcionado por el usuario. Esta función debe manejar adecuadamente cualquier error que pueda ocurrir durante el envío del correo electrónico.	

Capítulo 3. Implementación y pruebas

3. Validación del Código de Verificación: Implementar una función que valide el código de verificación introducido por el usuario. Si el código es correcto, se debe proceder con la modificación de la suscripción. Si el código es incorrecto, se debe mostrar un mensaje de error al usuario.
4. Modificación de la suscripción del Usuario: Si el código de verificación es correcto, la suscripción del usuario debe ser modificada según las nuevas selecciones y mostrar un mensaje al usuario. Esto podría implicar actualizar la información del usuario en una base de datos y reconfigurar las notificaciones para las nuevas farmacias y medicamentos seleccionados.
5. Manejo de Errores del Formulario: Implementar una validación de los datos introducidos en el formulario para asegurar que son válidos. Esto incluye mostrar un mensaje de error si el usuario intenta suscribirse sin introducir un correo electrónico, si selecciona más de 5 farmacias o medicamentos, o si introduce un código de verificación incorrecto.

Tabla 3.10. TI10: Eliminar suscripción. Fuente: (Elaboración propia).

Tarea	
Número de tarea: 10	Número de HU: 8
Nombre de la tarea: Eliminar suscripción	
Tipo de Tarea: Desarrollo	Puntos estimados: 1
Fecha de inicio: 24 de julio del 2023	Fecha de fin: 28 de julio del 2023
Programadores responsables: Nelson Alejandro Cruz Borrego y Denis Duranza Díaz	
Descripción: 1. Diseño del Enlace de Eliminación de la Suscripción: Diseñar un enlace que permita al usuario eliminar su suscripción. Este enlace debe ser incluido en cada correo de notificación enviado al usuario. 2. Implementación del Enlace de Eliminación de la Suscripción: Implementar una función en el backend de la aplicación que maneje la solicitud de eliminación de la suscripción cuando el usuario hace clic en el enlace. Esta	

Capítulo 3. Implementación y pruebas

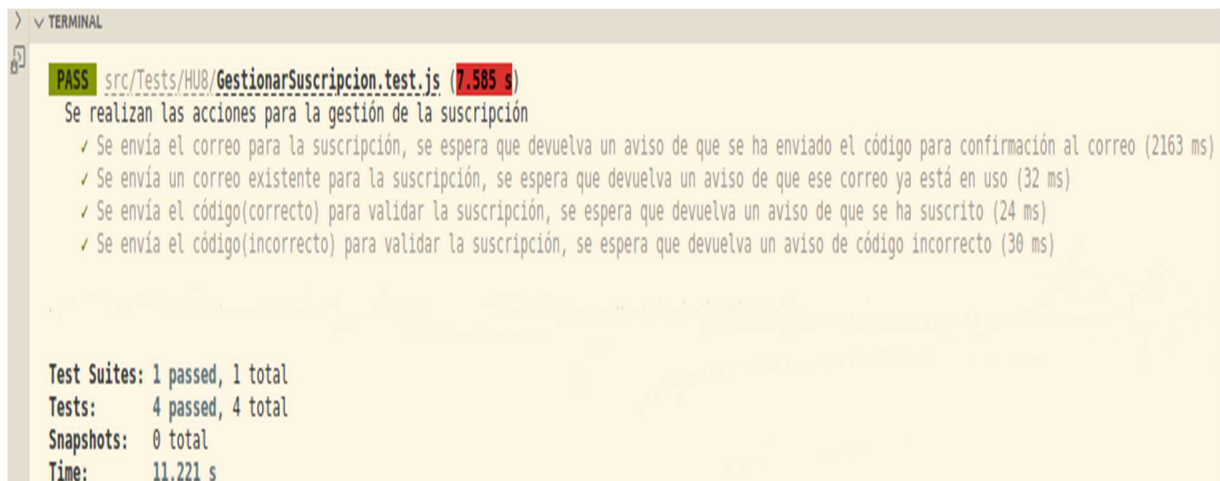
función debe manejar adecuadamente cualquier error que pueda ocurrir durante la eliminación de la suscripción.

3. Eliminación de la Suscripción del Usuario: Si el usuario hace clic en el enlace de eliminación de la suscripción, su suscripción debe ser eliminada. Esto podría implicar eliminar la información del usuario de una base de datos y detener las notificaciones para el usuario.
4. Confirmación de la Eliminación de la Suscripción: Después de que la suscripción del usuario ha sido eliminada, se debe mostrar un mensaje al usuario que su suscripción ha sido eliminada con éxito.

Test Unitarios

En la Figura 3.11 se muestran los resultados de la ejecución de los test unitarios de la cuarta iteración.

Figura 3.11. Test unitario de la cuarta iteración. Fuente: (Elaboración propia).



```
> v TERMINAL
PASS src/Tests/HUB/GestionarSuscripcion.test.js (7.585 s)
Se realizan las acciones para la gestión de la suscripción
✓ Se envía el correo para la suscripción, se espera que devuelva un aviso de que se ha enviado el código para confirmación al correo (2163 ms)
✓ Se envía un correo existente para la suscripción, se espera que devuelva un aviso de que ese correo ya está en uso (32 ms)
✓ Se envía el código(correcto) para validar la suscripción, se espera que devuelva un aviso de que se ha suscrito (24 ms)
✓ Se envía el código(incorrecto) para validar la suscripción, se espera que devuelva un aviso de código incorrecto (30 ms)

Test Suites: 1 passed, 1 total
Tests:      4 passed, 4 total
Snapshots:  0 total
Time:       11.221 s
```

Pruebas de aceptación

Estas se llevan a cabo después que el usuario ha definido sus necesidades respecto a lo que desea que realice la aplicación. Son básicamente pruebas funcionales, sobre el sistema completo

Capítulo 3. Implementación y pruebas

que se deben realizar en intervalos regulares, y busca validar los resultados obtenidos. Estos tests pueden ser automáticos o manuales, siendo los primeros los preferidos en todo momento, a pesar de que en muchas ocasiones es bastante difícil la construcción de estos. (Rodríguez & Bonilla, 2020)

Tabla 4.1. CP01: Búsqueda. Fuente: (Elaboración propia).

Caso de Prueba Búsqueda	
Código: CP01	Número de HU: 1
Descripción: Esta prueba verifica que la funcionalidad de búsqueda de medicamentos y farmacias funcione correctamente.	
Condiciones de Ejecución: La aplicación web está en funcionamiento.	
Entrada / Pasos de ejecución: 1. Navegar a la página de búsqueda. 2. Ingresar el nombre del medicamento o de la farmacia en el campo de búsqueda. 3. Hacer clic en el botón de búsqueda.	
Resultado Esperado: La aplicación muestra una lista de medicamentos que coinciden con el término de búsqueda ingresado.	

Tabla 4.2. CP02: Geolocalizar Farmacias. Fuente: (Elaboración propia).

Caso de Prueba Geolocalizar Farmacias	
Código: CP02	Número de HU: 3
Descripción: Esta prueba verifica que la funcionalidad de geolocalizar las coordenadas de las farmacias recibidas por la API funcione correctamente.	
Condiciones de Ejecución: La aplicación web está en funcionamiento y la API está enviando los datos de las coordenadas.	
Entrada / Pasos de ejecución: 1. Navegar a la pestaña del mapa. 2. Verificar que todas las ubicaciones recibidas por la API están marcadas en el mapa con un ícono.	

Capítulo 3. Implementación y pruebas

Resultado Esperado: Todas las ubicaciones recibidas por la API deben estar marcadas en el mapa cuando se abre la pestaña del mapa.

Tabla 4.3. CP03: Mostrar Información de las Farmacias en el Mapa. Fuente: (Elaboración propia).

Caso de Prueba Mostrar Información de las Farmacias en el Mapa	
Código: CP03	Número de HU: 4
Descripción: Esta prueba verifica que, al hacer clic en las farmacias marcadas en el mapa, se muestra información sobre ellas.	
Condiciones de Ejecución: La aplicación web está en funcionamiento, la API está enviando datos de las coordenadas y de la información de las respectivas farmacias.	
Entrada / Pasos de ejecución: 1. Navegar a la pestaña del mapa. 2. Hacer clic en una de las farmacias marcadas en el mapa.	
Resultado Esperado: Al hacer clic en una farmacia marcada en el mapa, se debe mostrar información como teléfono, dirección, nombre, tipo, distancia a la que se encuentra. Esta información es proporcionada por la API.	

Tabla 4.4. CP04: Geolocalizar al Usuario. Fuente: (Elaboración propia).

Caso de Prueba Geolocalizar al Usuario	
Código: CP04	Número de HU: 5
Descripción: Esta prueba verifica que al seleccionar una ubicación desde el menú que aparece en el mapa, el marcador se coloca en esa ubicación.	
Condiciones de Ejecución: La aplicación web está en funcionamiento.	
Entrada / Pasos de ejecución: 1. Navegar a la pestaña del mapa. 2. Tocar en el menú seleccionable con distintas ubicaciones. 3. Seleccionar una ubicación del menú.	
Resultado Esperado: Al seleccionar una ubicación del menú, el marcador del mapa se coloca en la ubicación seleccionada.	

Capítulo 3. Implementación y pruebas

Tabla 4.5. CP05: Resaltar Farmacias Cercanas. Fuente: (Elaboración propia).

Caso de Prueba Resaltar Farmacias Cercanas	
Código: CP05	Número de HU: 6
Descripción: Esta prueba verifica que cuando el marcador de ubicación está en un radio menor o igual a 5 Km de los marcadores de las farmacias, estos se muestren con un ícono diferente.	
Condiciones de Ejecución: La aplicación web está en funcionamiento. La API está enviando los datos de las farmacias correctamente.	
Entrada / Pasos de ejecución: 1. Navegar a la pestaña del mapa. 2. Mover el marcador de ubicación cerca de otros marcadores en el mapa.	
Resultado Esperado: Cuando el marcador de ubicación está en un radio menor o igual a 5 Km de los marcadores de las farmacias, estos se muestran con un ícono verde.	

Tabla 4.6. CP06: Calcular Ruta a Farmacia. Fuente: (Elaboración propia).

Caso de Prueba Calcular Ruta a Farmacia	
Código: CP06	Número de HU: 7
Descripción: Esta prueba verifica que, al hacer clic en una farmacia del mapa, se muestra la información de esa farmacia y al hacer clic en el botón "Ir", se dibuja una ruta desde la ubicación actual hasta la farmacia.	
Condiciones de Ejecución: La aplicación web está en funcionamiento. La API está enviando los datos de las farmacias correctamente. La aplicación tiene acceso a internet.	
Entrada / Pasos de ejecución: 1. Navegar a la pestaña del mapa. 2. Hacer clic en una farmacia del mapa. 3. Verificar que se muestra la información de esa farmacia. 4. Hacer clic en el botón "Ir".	

Capítulo 3. Implementación y pruebas

Resultado Esperado: Se debe dibujar una ruta con colores blanco y negro desde la ubicación actual hasta esa farmacia.

Tabla 4.7. CP07: Suscripción a la PNCM. Fuente: (Elaboración propia).

Caso de Prueba Suscripción a la PNCM	
Código: CP07	Número de HU: 8
Descripción: Esta prueba verifica que la funcionalidad de suscripción a la PNCM funcione correctamente. El usuario debe poder ingresar un correo electrónico, seleccionar hasta 5 farmacias y 5 medicamentos, y recibir un código de verificación por correo electrónico.	
Condiciones de Ejecución: La aplicación web está en funcionamiento. La aplicación tiene acceso a internet.	
Entrada / Pasos de ejecución: 1. Navegar a la pestaña de suscripción. 2. Ingresar un correo electrónico válido en el campo correspondiente. 3. Seleccionar hasta 5 farmacias de la lista. 4. Seleccionar hasta 5 medicamentos de la lista. 5. Hacer clic en el botón de suscripción. 6. Verificar que se recibe un correo electrónico con un código de verificación. 7. Ingresar el código de verificación en el campo correspondiente en la pestaña de suscripción. 8. Hacer clic en el botón de verificación.	
Resultado Esperado: Al completar todos los pasos, el usuario debe recibir un mensaje de que la suscripción se realizó correctamente.	

Tabla 4.8. CP08: Modificar Suscripción. Fuente: (Elaboración propia).

Caso de Prueba Modificar Suscripción	
Código: CP08	Número de HU: 8

Capítulo 3. Implementación y pruebas

Descripción: Esta prueba verifica que la funcionalidad de modificación de suscripción funcione correctamente. El usuario debe poder ingresar un correo electrónico, recibir un código de verificación por correo electrónico y modificar su suscripción.
Condiciones de Ejecución: La aplicación web está en funcionamiento. La aplicación tiene acceso a internet.
Entrada / Pasos de ejecución: 1. Navegar a la pestaña de suscripción. 2. Ingresar un correo electrónico válido en el campo correspondiente. 3. Hacer clic en el botón de modificación. 4. Verificar que se recibe un correo electrónico con un código de verificación. 5. Ingresar el código de verificación en el campo correspondiente. 6. Seleccionar hasta 5 farmacias de la lista. 7. Seleccionar hasta 5 medicamentos de la lista. 8. Hacer clic en el botón de modificar suscripción.
Resultado Esperado: Al completar todos los pasos, el usuario debe recibir un mensaje de que la modificación de la suscripción se realizó correctamente.

Tabla 4.9. CP09: Eliminar Suscripción. Fuente: (Elaboración propia).

Caso de Prueba Eliminar Suscripción	
Código: CP09	Número de HU: 8
Descripción: Esta prueba verifica que la funcionalidad de eliminación de suscripción funcione correctamente. El usuario debe poder hacer clic en el enlace que se envía junto con las notificaciones vía correo electrónico para poder eliminar su suscripción.	
Condiciones de Ejecución: La aplicación web está en funcionamiento. La aplicación tiene acceso a internet.	
Entrada / Pasos de ejecución: 1. Verificar que se recibe un correo electrónico de notificación con un enlace para eliminar la suscripción. 2. Hacer clic en el enlace de eliminación de suscripción en el correo electrónico.	

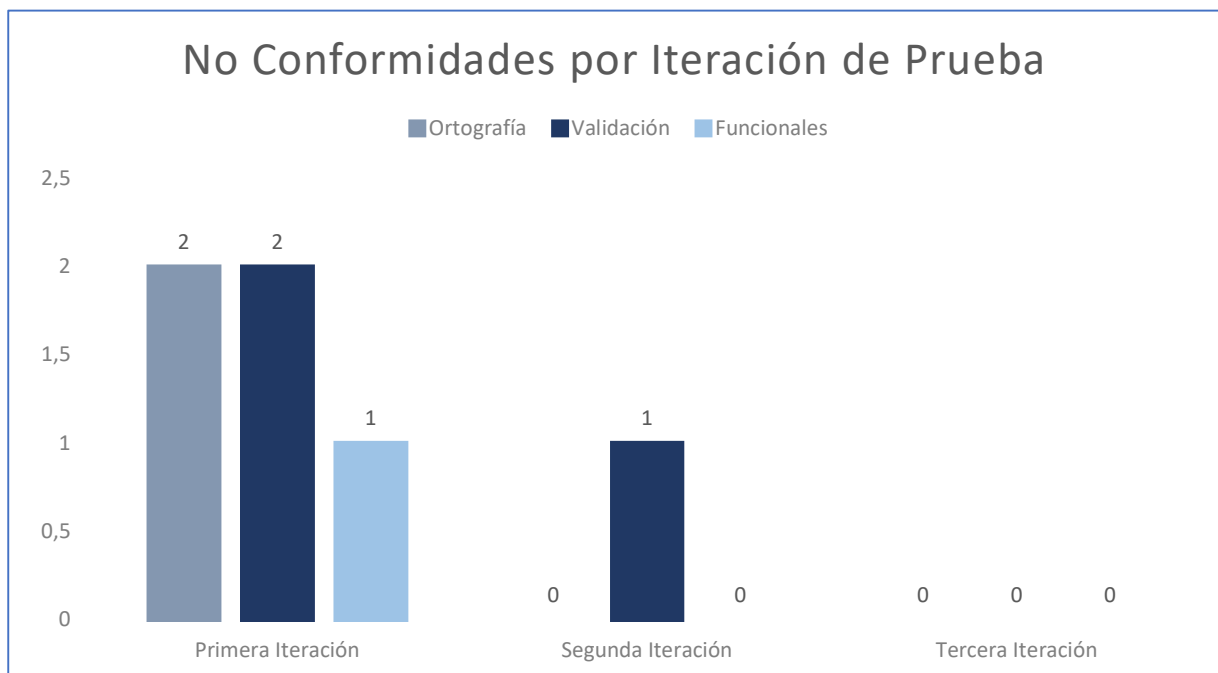
Capítulo 3. Implementación y pruebas

Resultado Esperado: Al hacer clic en el enlace de eliminación de suscripción en el correo electrónico, la suscripción del usuario debe ser eliminada y el usuario debe recibir un mensaje de que la eliminación de la suscripción se realizó correctamente.

Los resultados obtenidos fueron exitosos, validando el adecuado rendimiento de todas las características incorporadas en el sistema. Estas pruebas fueron realizadas por el Centro de Calidad de Software de la XETID, basándose en los diseños de casos de prueba que cubren las Historias de Usuarios de la PNCM. Las discrepancias identificadas fueron evaluadas y corregidas, garantizando un funcionamiento adecuado tanto con entradas válidas como no válidas.

A continuación, se muestran las no conformidades detectadas en cada iteración de prueba:

Figura 4.1. No Conformidades por Iteración. Fuente: (Elaboración propia).



En la primera iteración fueron detectadas un total de 5 No conformidades (NC): 2 de ortografía, 2 de validación y 1 funcional. En la segunda iteración fue detectada 1 NC de validación. En la tercera iteración no fueron detectados errores de ningún tipo, para un total de 0 NC, esto se ve reflejado en el Acta de Aceptación.

3.4. Conclusiones parciales

En este capítulo se reflejó la ejecución de la fase de implementación y pruebas, arribando a las siguientes conclusiones:

- Se realizaron las 4 iteraciones planificadas, implementándose un total de 8 HU a través de 10 TI.
- La realización del diagrama de despliegue permitió entender cómo se implementará el sistema.
- El empleo de los estándares de codificación definidos facilitó la lectura, comprensión y mantenimiento del código para los desarrolladores.
- La implementación de las TI hizo posible el cumplimiento del plan de entrega de las HU.
- La realización de pruebas unitarias y de aceptación permitió comprobar que las funcionalidades fueron implementadas correctamente y que el sistema satisface las necesidades del cliente.

Conclusiones generales

La implantación de la Plataforma Nacional de Consulta de Medicamentos desarrollada en esta investigación, permitirá a la población cubana consultar de forma remota, fiable y de manera permanente la disponibilidad de los medicamentos en cada una de las farmacias del país. Implementar servicios informáticos que mejoren la calidad de cada uno de los procesos que se viven a diario constituyen un reto de la rama de la Informática. De esta forma, la información que hoy solo se accede físicamente, será utilizada con el empleo de las nuevas tecnologías de la información y las comunicaciones (TIC), constituyendo un pilar clave del Programa Estratégico de Informatización del Ministerio de Salud Pública (MINSAP), dando de esta manera otro paso de avance a la informatización de la sociedad por la que aboga el país.

Recomendaciones

Durante la investigación y desarrollo de la Plataforma Nacional de Consulta de Medicamentos, se identificaron ideas que podrían ser consideradas para mejorar el resultado de este trabajo. Se recomienda tener en cuenta para futuras actualizaciones, la integración con aplicaciones como el Formulario Nacional de Medicamentos para nutrirse de sus datos y estadísticas. Y también desarrollar una versión para dispositivos móviles con el fin de promover la utilización de estas herramientas entre la población cubana.

Referencias bibliográficas

1. *About Node.js*. (2023). Node.Js. <https://nodejs.org/en/about>
2. Aguirre, S. (2022). *Crea tu API* (Vol. 202). RedUSERS.
3. *APKlis*. (s. f.). APKlis. Recuperado 29 de septiembre de 2023, de <https://www.apklis.cu>
4. *Axios*. (2023). <https://axios-http.com/>
5. Beck, K. (2000). *Extreme programming explained: Embrace change*. addison-wesley professional.
6. Blas, M. J. (2019). *Modelado y Verificación de Patrones de Diseño de Arquitectura de Software para Entornos de Computación en la Nube*. 17.
7. Cevallos, Zambrano, N. F. A., H. Y. L. (2020). *Ventaja competitiva de las farmacias como estrategia de posicionamiento en el mercado de Portoviejo*.
8. Dallos Bustos, L. P., Moncada Castaño, D. S., Ariza Hernández, D. C., & Franco Vargas, V. J. (2020). *Análisis comparativo entre metodologías ágiles y tradicionales para la gerencia de proyectos*. Universidad EAN.
9. *Data Model Design—MongoDB Manual*. (s. f.). Recuperado 11 de octubre de 2023, de <https://www.mongodb.com/docs/manual/core/data-model-design/>
10. *Data Modeling Introduction—MongoDB Manual*. (s. f.). Recuperado 11 de octubre de 2023, de <https://www.mongodb.com/docs/manual/core/data-modeling-introduction/>
11. *Express—Infraestructura de aplicaciones web Node.js*. (2023). <https://expressjs.com/es/>
12. Falcón, A. L., & Serpa, G. R. (2021). Acerca de los métodos teóricos y empíricos de investigación: Significación para la investigación educativa. *Revista Conrado*, 17(S3), 22-31.
13. *FARMAGUIA - Localizador de farmacias y sus servicios, farmacias de guardia, farmacias de Barcelona en tiempo real*. (s. f.). Recuperado 29 de septiembre de 2023, de <https://www.farmaguia.net/desktop/es/>
14. Farmamundi. (2022, enero 26). 10 aplicaciones imprescindibles para las farmacias de hoy. *Farmamundi*. <https://farmaceuticosmundi.org/10-aplicaciones-imprescindibles-para-las-farmacias-de-hoy/>
15. García Alonso, J. (2021). *Implementación de una herramienta Web Mapping para la gestión municipal*.
16. Gauchat, J. D. (2017). *Gran Libro de HTML5, CSS3 y JavaScript* (Tercera). MARCOMBO, S.A.
17. Giridhar, C. (2016). *Learning Python design patterns*.
18. *GitHub - 0granada/js-coding-standards: Estándares de codificación para el lenguaje javascript*. (2023). GitHub. <https://github.com/0granada/js-coding-standards>

Referencias bibliográficas

19. Gonzáles, F. & Y. (2016). *Módulo de HMAST para la administración y migración hacia SAMBA 4 del servicio directorio activo*. Universidad de las Ciencias Informáticas.
20. Herdoíza Carrillo, F. S. (2022). *Aplicación Móvil de Gestión de Actividades Turísticas y Recreativas Basadas en Geolocalización*. Pontificia Universidad Católica del Ecuador Sede Ambato.
21. LUDA Farma, herramienta para localizar medicamentos desde las farmacias. (s. f.). *LUDA Partners*. Recuperado 29 de septiembre de 2023, de <https://ludapartners.com/ludafarma/>
22. Marin Guzman, A. J. (2022). *Farmacovigilancia y seguridad del paciente en el servicio farmacéutico Droguería Vital Edge*. Universidad Nacional Abierta y a Distancia.
23. *Material UI: React components based on Material Design*. (2023). <https://mui.com/material-ui/>
24. *MongoDB: La Plataforma De Datos Para Aplicaciones*. (2023). MongoDB. <https://www.mongodb.com/es>
25. MONTERO,CEVALLOS, B. M., H. V. (2018). *Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software*.
26. Murillo, D. (2017). Navegadores web. *El Tecnológico*, 27(1), 13-14.
27. *Nodemailer*. (2023). <https://nodemailer.com/>
28. Ortega, G. A. V. (2021). Lineamientos para el diseño de aplicaciones web soportados en patrones GRASP. *Ciencia e Ingeniería*, 8(2), e5716304-e5716304.
29. Ortega, M. B., Nuñez, A. I. B., Coca, J. A., & Reyes, O. R. (2022). Conocimiento y desempeño de prescriptores de fitofármacos en un servicio de estomatología cubano. *Revista Cubana de Farmacia*, 55(2).
30. Pressman, R. S. (2010). *Ingeniería de Software. Un enfoque práctico* (Séptima).
31. Ramos, V. (2007). *Las TIC en el sector de la salud* (pp. 163, 41-45).
32. *React*. (2023). <https://react.dev/>
33. Rodriguez, C. A., & Bonilla, J. E. (2020). Pruebas en Programación Extrema. IV SIMPOSIO INTERNACIONAL DE SISTEMAS DE INFORMACIÓN E, 126.
34. Sánchez Padrón, Dra. G. (2021). *Anuario Estadístico de Salud*.
35. Sommerville, I. (2011). *Ingeniería de Software* (Novena). PEARSON EDUCACIÓN.
36. Soto, S. V. M., González, G. V. M., Betancur, R. A. M., & Castaño, L. E. G. (2021). Coffee Challenge: Un juego para la enseñanza de patrones de diseño de software. *Revista Politécnica*, 17(33), 36-46.
37. Sunardi y Suhajito, A. (2019). *MVC Architecture: A Comparative Study Between Laravel Framework and Slim Framework in Freelancer Project Monitoring System Web Based*.
38. Villena, A. H., Camacho, M. R. A. R., & Guerrero, V. H. V. (s. f.). *Tesina*.

Referencias bibliográficas

39. *Visual Studio Code—Code Editing. Redefined.* (2023). <https://code.visualstudio.com/>

Anexos

Acta de Aceptación



Habana, __13__ de __noviembre__ del 2023

Estimados Denis Duranza Diaz y Nelson Alejandro Cruz Borrego,

Nos complace informarle que su solicitud de aceptación de software ha sido aprobada. En nombre de XETID nos complace aceptar su software Plataforma Nacional de Consulta de Medicamentos para su uso en nuestra organización, expresamos nuestra satisfacción con respecto al software que se ha implementado recientemente en nuestra empresa.

Atentamente,

Nombre: __Kenier Cumba Bueno__
Cargo: __Director de UB Empresa Digital__