  
 **UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN**

**FACULTAD DE CIENCIA, ARTE Y TECNOLOGÍA**

**LICENCIATURA EN ANALISIS DE SISTEMAS INFORMATICOS**

**TRABAJO FINAL DE GRADO**

**PLATAFORMA VIRTUAL PARA EL REGISTRO, CONTROL Y GESTIÓN DE DATOS DE PACIENTES INFECTADOS CON DENGUE DESTINADO A LA SEPTIMA REGION SANITARIA DEL DEPARTAMENTO DE ITAPUA, REPUBLICA DEL PARAGUAY**

**LINEA DE INVESTIGACIÓN:  
  
ESTUDIANTE:**

**MARIO DAVID SEGOVIA TROCHE**

**TUTOR:**

**OSVALDO MICNIUK  
  
Encarnación – Paraguay**

**Mes – 2020**

INDICE

[1. INTRODUCCIÓN 3](#_Toc67737751)

[2. MARCO TEÓRICO 5](#_Toc67737752)

[2.1. Consecuencias sociales y económicas de la enfermedad del Dengue en el Paraguay 5](#_Toc67737753)

[2.2. Utilización de las Tecnologías de la Información y Comunicación en el control y la erradicación del Dengue. 6](#_Toc67737754)

[2.3. Desarrollo de una API REST utilizando una metodología de desarrollo ágil para una plataforma de registro de personas infectadas con el virus del Dengue. 7](#_Toc67737755)

[3. PREGUNTAS, HIPOTESIS Y OBJETIVOS 9](#_Toc67737756)

[3.1. Preguntas: 9](#_Toc67737757)

[3.2. Hipótesis: 9](#_Toc67737758)

[3.3 Objetivos 10](#_Toc67737759)

[4. METODOLOGÍA 11](#_Toc67737760)

[4.1. Resumen de las etapas de trabajo. Delineación de la estructura de la plataforma y del módulo API REST principal 11](#_Toc67737761)

[4.2. Descripción de las actividades a realizarse durante cada etapa 12](#_Toc67737762)

[5. RESULTADOS Y DISCUSIÓN 15](#_Toc67737763)

[6. CONCLUSIÓN 16](#_Toc67737764)

[7. RECOMENDACIONES 17](#_Toc67737765)

[8. REFERENCIAS BIBLIOGRÁFICAS. 18](#_Toc67737766)

[9. ANEXO 20](#_Toc67737767)

# 1. INTRODUCCIÓN

Este trabajo tiene por finalidad realizar una plataforma virtual para el registro, control y gestión de datos de pacientes infectados con Dengue, que sea de utilidad para los responsables de la región sanitaria del departamento de Itapúa, República del Paraguay, en la toma de decisiones y definir estrategias en la lucha contra esta enfermedad.

El Dengue en el Paraguay crea una gran carga económica y social en el sistema de salud, con epidemias que desde hace años generan una cantidad de fallecidos y enfermos cuya atención médica representan un alto costo (Arbo, 2019).

En este sentido, se buscan soluciones y estrategias para erradicar o disminuir el impacto de este flagelo utilizando diversas herramientas, y una de ellas es la tecnología. Existen iniciativas como la utilización de plataformas digitales y aplicaciones para móviles con el fin de involucrar a la comunidad a participar activamente en la detección y eliminación de criaderos del mosquito Aedes Aegypti, transmisor del virus del Dengue (Parra, 2017).

Otras aplicaciones utilizadas son modelos matemáticos computacionales para predecir el comportamiento y expansión de posibles epidemias mediante datos y estadísticas históricas (Mello Román et al, 2019). Así también tenemos portales de gestión de datos e información sobre la enfermedad para su mejor manejo y reducir el impacto de éste sobre nuestro país (Pane et al, 2018).

Actualmente hay estudios aplicando las últimas tecnologías, como la aplicación de redes neuronales para mostrar escenarios posibles que puedan presentarse de esta dolencia (Ughelli et. al, 2017), y otras como investigación en sistemas de información geográficos para localizar las zonas de más incidencia y registrar en mapas los casos reportados (Condor Camara et. al, 2018).

Así pues, el objetivo es desarrollar una plataforma de registro y gestión de personas infectadas con el virus del Dengue destinado para la Séptima Región Sanitaria del departamento de Itapúa.

Esto será realizado mediante la planificación de actividades y el flujo de trabajo estructurado mediante una metodología de desarrollo ágil (Navarro Cadavid et. al, 2013). El software se desarrolla en un entorno web a partir de módulos independientes que trabajan en forma conjunta ([Molina Ríos et. al,](http://repositori.uji.es/xmlui/browse?type=author&authority=00e9b65e-d5f5-4133-8488-98af581641f5) 2018).

El modulo API (Application Programming Interfaces) se establece como el centro de los procesos y almacenamiento de datos de todo el sistema. A partir del protocolo de comunicación REST (Representational State Transfer), la API ofrece una interfaz que permite el intercambio de datos con los otros módulos (Revuelta Arribas, 2020). Para el almacenamiento de los datos se utiliza un Sistema de Gestión de Bases de Datos (SGBD) (Marín, 2019).

El desarrollo de la API REST se establece en cuatro etapas o fases:

Primera etapa: análisis y definición de requisitos y requerimientos, como el alcance del proyecto, software de programación y desarrollo necesarios.

Segunda etapa: diseño de la API, creación de los casos de uso, interfaces y diagramas, establecimiento de estándares de programación.

Tercera etapa: Programación del software a partir del diseño realizado, codificación en el lenguaje elegido, respaldar los repositorios con el software de versionado.

Cuarta etapa: Testeo del software, corrección de errores y documentación.

# 2. MARCO TEÓRICO

## 2.1. Consecuencias sociales y económicas de la enfermedad del Dengue en el Paraguay

El dengue es una enfermedad causado por un virus, del género Flavivirus, de la familia Flaviviridae, del cual se conocen cuatro serotipos identificados como: DENV-1, DENV-2, DENV-3 y DENV-4. Se transmite a los humanos a través de la picadura del mosquito Aedes Aegypti, una especie que esta mayormente activa en horas de la mañana y antes de oscurecer, y se reproduce colocando su huevos en espacios y lugares donde se almacene o acumule agua, como floreros, tanques, piscinas y todo tipo de desechos de la actividad humana como neumáticos viejos, elementos de plásticos como bolsas, vasos, recipientes de toda clase. Los síntomas típicos de esta enfermedad suelen presentarse como fiebre alta, dolores de cabeza, musculares y articulares., Los cuadros más graves pueden generar sangrados internos y daños profundos en órganos, hasta llevar a la muerte del paciente. Se extiende en regiones de climas tropicales y subtropicales, como África, norte de Australia, México, Centroamérica y Sudamérica (Lage et. al, 2015).

En el mundo se calcula que más de la mitad de la población mundial, alrededor de 3.600 millones de habitantes residen en zonas donde son susceptibles de contraer la enfermedad del Dengue. Según la Organización Mundial de la Salud (OMS), alrededor de 34 millones de casos de Dengue clínico ocurren cada año, así como 2 millones de casos de Dengue hemorrágico y más de 20 mil muertes (Arbo, 2019).

El Paraguay ha sufrido varias epidemias de Dengue desde la llegada de esta enfermedad al país. Se registran epidemias en los años 1989, 1990, 2007 y 2013, y un crecimiento sostenido de casos reportados desde los años 2009 hasta 2015, registrándose más de 300.000 casos de los cuales más de 100 fueron pacientes fallecidos. Específicamente en el año 2013 se reportaron 149.429 casos, el registro de mayor cantidad de casos confirmados (Arbo, 2019). Según datos del boletín correspondiente al periodo comprendido entre las fechas del 29 de diciembre del año 2019 al 25 del mes de abril del año 2020 de la Dirección General de Vigilancia de la Salud, se reportan 40.250 casos de Dengue, y un total de 59 fallecidos.

Según estimaciones realizadas, cada paciente representa un gasto de servicios e insumos sanitarios en guaraníes de aproximadamente 5.700.000 o en dólares americanos de 1.054, tomando como promedio 4 días de internación, sin considerar otros gastos indirectos, como el costo de transporte, los acompañantes, alimentos, así como la pérdida de ingresos por la imposibilidad de asistir a su puesto de trabajo, elevando significativamente el costo económico a los involucrados (Arbo, 2019; Rodriguez-Castro et. al, 2019).

Cifras obtenidas en un estudio realizado sobre el impacto económico del Dengue en el Instituto de Previsión Social del Paraguay, en la epidemia ocurrida entre el mes de diciembre del año 2006 hasta el mes de julio del año 2007, de un total de 5960 casos, los costos por internación de pacientes, insumos, contratación de médicos, estudios laboratoriales y subsidios por reposo totalizaron un monto de más 12 mil millones de guaraníes o más de 2,5 millones de dólares americanos (Flores et. al, 2015).

Según resultados obtenidos en un trabajo de investigación en relación a los costos del Dengue en el Paraguay entre los años 2010 al 2013, se reportaron al Ministerio de Salud Pública y Bienestar Social (MSPBS) 272.997 casos probables y confirmados de Dengue. Así también un número estimativo de 2.162.538 casos de pacientes ambulatorios o que no requerían de hospitalización y 45.801 casos de enfermos hospitalizados. En cuanto a los costos, calculados en dólares americanos, en relación a los pacientes ambulatorios, en gastos médicos directos se estima una cantidad de 113.533.245 US$, en gastos médicos no directos la cantidad de 23.787.918 US$ y en gastos indirectos la cantidad de 134.077.356 US$, con un total global de 276.804.864 US$. En relación a pacientes hospitalizados, los gastos médicos directos ascienden a 13.740.300 US$, los gastos médicos no directos a 5.816.727 US$ y gastos indirectos a 7.648.767 US$, totalizando 23.450.112 US$. Contrastando estos datos con el presupuesto anual total destinado a la salud publica en el Paraguay, que asciende a aproximadamente 800 millones de dólares americanos, en el año 2010 los costos por la enfermedad del Dengue fue del 2,9% del presupuesto anual de salud, en el año 2011 subió a un 6,6%, al año 2012 un 2,14%, y en el año 2013 un 20,9%. Estos montos son altamente significantes, y repercuten negativamente en el presupuesto destinado al sistema sanitario del país (Cuellar et al, 2020).

Estos números suponen una elevada carga económica y social en el sistema de atención médica y en la población en general.

## 

## 2.2. Utilización de las Tecnologías de la Información y Comunicación en el control y la erradicación del Dengue.

De las distintas estrategias que se utilizan para eliminar o disminuir los casos de Dengue en la población, una de ellas son las Tecnologías de la Información y la Comunicación (TIC), así como las relacionadas con la informática y la computación.

El proyecto TopaDengue, llevado a cabo por estudiantes de la Universidad Católica de la Ciudad de Asunción, Paraguay, con financiamiento del Consejo Nacional de Ciencia y Tecnología (CONACYT), tenía por finalidad reducir el alcance de las epidemias de Dengue mediante la eliminación y control de criaderos del vector transmisor, el mosquito Aedes Aegypti, a través de las TIC, utilizando una plataforma web, combinado con una aplicación móvil, llamada DengueChat, por medio de la cual, la comunidad tenía una activa participación en los reportes sobre criaderos. Se desarrolló entre los años 2018 y 2019, en la ciudad de Asunción, Paraguay, en la zona llamada Bañado Sur, comprendiendo los barrios San Ignacio, San Cayetano y Caacupemí. Los resultados obtenidos evidenciaron reducciones en los índices de infestación del mosquito en las zonas intervenidas con respecto a las zonas de control. En relación a la plataforma web DengueChat, es una iniciativa desarrollada por el Laboratorio de Aplicaciones Sociales de la Universidad de California en Berkeley. Esta plataforma, que es de libre acceso, combina juegos digitales, comunicación e interacción entre los participantes a través de chats o mensajería instantánea y redes sociales, a las que se pueden acceder por medio de teléfonos celulares, computadoras y otros aparatos digitales. Los usuarios de esta plataforma son motivados a identificar y eliminar criaderos de mosquitos, por los cuales son recompensados con distinciones y otros tipos de reconocimientos, y principalmente socializando con la mayor cantidad de personas involucradas, con el fin de concienciar sobre el problema de la enfermedad y que mediante la participación de la mayor cantidad posible de individuos de una comunidad, se puede erradicar este problema que genera el Dengue. Además de propiciar la participación ciudadana, permite recolectar datos sobre los índices de infestación del mosquito vector transmisor del Dengue, con la cual tener información de la situación en tiempo real, que puede ser utilizada por los responsables o autoridades para la toma de decisiones. Entre las ciudades donde se está utilizando esta plataforma con iniciativas locales podemos mencionar a la Ciudad de Managua, del país de Nicaragua, a la ciudad de Armenia, de Colombia y lugares donde se realizaron ensayos piloto podemos mencionar a la ciudad de Rio de Janeiro, Brasil y las ciudades de Cuernavaca y Tepalcingo, Mexico (Parra, 2019).

Otra solución desarrollada es la plataforma de Datos Abiertos y Alertas sobre Dengue, un software libre para la recolección, almacenamiento, análisis y visualización de datos, y a partir del procesamiento de grandes cantidades de datos, poder obtener información útil, como filtrar datos específicos y buscar patrones de interés para investigadores o generar alertas tempranas sobre posibles epidemias. Este proyecto también contó con la financiación del CONACYT y el desarrollo de este software dio como resultado un sistema integrado por un módulo de recolección y publicación de datos, un módulo de análisis y visualización de datos y un módulo de alertas tempranas. Todas las herramientas utilizadas en el desarrollo del software son open source o de uso libre, con el objetivo que pueda ser utilizado por los potenciales interesados sin costo alguno. Otras características a destacar son la flexibilidad para poder recolectar datos de distintas fuentes y tipos, y almacenarlas en una base de datos integrados y ofrecer un acceso de forma fácil para los posibles usuarios, y poder aplicar sobre ellos distintas metodologías de análisis de datos. El entorno de desarrollo o framework utilizado también ofrece la posibilidad de ser extensible y poder agregar más funcionalidades y mejoras (Pane et al, 2018).

También se estudian modelos matemáticos computacionales, estadística, redes neuronales artificiales, minería de datos para aplicar en datos recolectados, y generar proyecciones de posibles escenarios que se presentarían de la enfermedad. Pruebas realizadas presentaron resultados que serían de utilidad para detectar tempranamente casos de Dengue (Ughelli et. al, 2017; Mello Román et al, 2019).

Los Sistemas de Información Geográfica (SIG), se utilizan para obtener datos de referencias geográficas y ubicación en mapas, mediante estos se pueden delimitar las áreas o zonas donde se tienen brotes de personas infectadas, índices de infestación del mosquito transmisor o lugares propicios para criaderos de larvas del vector, y aplicar medidas de contención o protocolos establecidos de acción, como realizar pruebas para detectar casos posibles que no se reportaron, fumigación para eliminar los mosquitos, o realizar limpiezas de criaderos de larvas. Este proyecto se llevó a cabo entre el mes de setiembre del año 2017 hasta el mes de diciembre del año 2019 en las ciudades de Lima y de Iquitos – Loreto, del Perú. Por medio de encuestas y digitalización de fichas de notificación de cuadros de Dengue se obtenían los datos de los pacientes, con sus datos de georeferencia. A partir de estos datos se cargaban en la plataforma SIG, con el cual era posible visualizar en los mapas digitales la áreas de influencia de la enfermedad (Condor Camara et. al, 2018).

Otras aplicaciones y trabajos de investigación que utilizaron las TIC para su realización que podemos mencionar:

- Medisys, creado en Italia, un sistema de alertas mediante el análisis de noticias y tweets relacionados al Dengue.

- DengueTrends, de la empresa Google, es una aplicación que genera estadísticas a partir de búsquedas de palabras claves relacionadas con la enfermedad del Dengue.

- Denfree, proyecto que tiene la finalidad de detectar las causas más importantes que ocasionan la transmisión, infección y epidemia de dengue, conjuntamente con el desarrollo de nuevas herramientas para la detección y diagnóstico de infecciones asintomáticas.

- IDAMS, iniciativa que busca el desarrollo de herramientas innovadoras y nuevas para ser utilizadas al control del Dengue en un entorno global

- DengueTools, herramienta para establecer una distribución espacio-temporal de enfermedades de transmisión por vectores para un control efectivo (Pane et al, 2015).

De esta forma, podemos encontrar una gran cantidad de proyectos, trabajos e investigaciones sobre la enfermedad del Dengue que implementan y utilizan herramientas TIC y otras tecnologías de la computación e informática para su realización y concreción de los objetivos.

## 

## 2.3. Desarrollo de una API REST utilizando una metodología de desarrollo ágil para una plataforma de registro de personas infectadas con el virus del Dengue.

En la revisión de antecedentes de trabajos de investigación y otros proyectos relacionados a la enfermedad del Dengue, se evidencia que en varios casos se basaron en la implementación de herramientas TIC y otras tecnologías relacionadas a la computación y la informática para lograr sus objetivos propuestos. De esta forma, se considera factible llevar a cabo la propuesta de una plataforma web de registro de personas infectadas con el virus del Dengue, utilizando una variedad de herramientas informáticas para su desarrollo, que deben establecerse de manera clara y precisa, y una planificación cuidadosa para obtener un resultado exitoso.

Muchas de las intervenciones tecnológicas implican de cierta manera desarrollo de software, con una estructura modular que funcionan en conjunto como un sistema unitario y en el cual el concepto de sinergia toma un mayor significado. Por lo tanto, como todo proceso metodológico, es necesario establecer la forma de trabajo, las actividades que se llevaran a cabo y las herramientas tecnológicas a ser utilizadas.

Primeramente, antes de iniciar todo desarrollo de software, se debe especificar qué tipo de metodología se utilizará. Se entiende por metodologías de desarrollo de software a un conjunto de métodos, procesos y técnicas que se utilizan de guía para ejecutar un proyecto con el objetivo de crear un nuevo software que sea funcional y de utilidad para los usuarios finales ([Molina Ríos et. al,](http://repositori.uji.es/xmlui/browse?type=author&authority=00e9b65e-d5f5-4133-8488-98af581641f5) 2018).

Las metodologías de desarrollo de software se clasifican en metodologías tradicionales y metodologías ágiles. Las metodologías tradicionales se caracterizan por un enfoque rígido en la planificación y el proceso, las etapas de trabajo se deben establecer de forma precisa y los periodos de tiempo deben cumplirse sin retrasos, se respalda en una documentación minuciosa y exhaustiva, la comunicación entre los involucrados es mínima, ya que los encargados de cada parte del proyecto se dedican exclusivamente a su trabajo establecido y los flujos de actividades no posibilitan opciones de cambios o ajustes, los cuales si son necesarios deben realizarse a través de procesos burocráticos que afectan a todo el proyecto.(Navarro Cadavid et. al, 2013).

Las metodologías de desarrollo ágil permite un flujo de trabajo dinámico, con un enfoque hacia el producto y evitando una documentación excesiva. También está centrado en el trabajo colaborativo, con un intercambio constante de información entre los involucrados en el proyecto. El cliente o usuario final también tiene una participación activa en el proceso de desarrollo del producto, interactuando con los involucrados dando retroalimentación para introducir mejoras o corregir errores. La flexibilidad es otro punto que caracteriza a este tipo de metodología, permitiendo realizar cambios o introducir correcciones durante el proceso de trabajo sin afectar al proyecto en general. Este tipo de metodología permite crear productos en periodos cortos de tiempo. Como ejemplo de metodologías ágiles más utilizadas podemos citar Scrum y Extreme Programming (XP) (Navarro Cadavid et. al, 2013).

La metodología ágil Scrum ofrece un marco de trabajo con una variedad de métodos, artefactos y herramientas para desarrollar un software de manera efectiva y eficaz. Se caracteriza por equipos de trabajo auto-gestionados y multifuncionales, los integrantes se organizan de forma autónoma y colaboran activamente entre sí para llegar a los objetivos. Los avances del proyecto se realizan a partir de iteraciones y en cada iteración se van incrementando las funcionalidades del producto o se van agregando o corrigiendo características según los requerimientos del cliente. Scrum establece tres roles para el equipo de trabajo, el scrum master, el product owner y el equipo de desarrollo. El scrum master que tiene la función de líder del equipo de desarrollo y velar por el correcto desempeño del grupo, el product owner o dueño del producto, se encarga de controlar el correcto desarrollo del producto y gestionar las funcionalidades para obtener un resultado exitoso y por ultimo tenemos el equipo de desarrollo, que se encarga de realizar todas las labores requeridas para obtener el producto final (Navarro Cadavid et. al, 2013).

En cuanto a los elementos principales que componen la metodología Scrum, se destacan los eventos como el Sprint, definido como un periodo especifico de tiempo donde se realiza el trabajo de desarrollo del producto y debe dar como resultado un incremento a ser evaluado por los responsables. El Sprint a su vez se compone de otros elementos como el Sprint planning, en el cual se planifican todas las actividades a ser desarrolladas en el Sprint, el Daily Scrum, que son reuniones diarias de corta duración donde se discuten las tareas realizadas y a realizarse, así como las dificultades encontradas, la revisión del Sprint, que se realiza al final de un Sprint donde el product owner tiene participación activa, se realiza una revisión general de las tareas realizadas y no realizadas, los problemas que se presentaron y la verificación del producto resultante. Otro evento a destacar es la retrospectiva de Sprint, un análisis del trabajo del equipo desarrollador, que se hizo bien y que mal, y crear un plan de mejoras para el siguiente Sprint. También debemos mencionar los artefactos de Scrum, piezas claves en este marco de trabajo. El Product Backlog es una lista de funcionalidades o características que el producto final debe tener de acuerdo a las necesidades solicitadas por el cliente, y puede ir cambiando, incrementándose o modificándose de acuerdo a las necesidades del dueño, por lo tanto no es una lista fija, sino es dinámica y evoluciona a lo largo del desarrollo del proyecto. El Sprint backlog es una lista derivada del product backlog, y se establece de acuerdo a las prioridades que cada ítem representa para el dueño, y se desarrollan en el Sprint planificado finalizando en el incremento del producto. El monitoreo de progreso es un control de los avances que se realizan en el Sprint, consistente en la revisión de las tareas que ya se cumplieron y las tareas que están pendientes. Por ultimo tenemos los Incrementos, que son la sumatoria de todos los ítems terminados de la lista del Product Backlog y que van determinando el estado de avance del proyecto (Navarro Cadavid et. al, 2013).

El sistema se modela en un formato de funcionamiento dentro de un entorno web, alojado en un servidor que nos ofrece accesibilidad desde internet. Esto nos permite utilizar la plataforma y disponer de los datos sin restricciones de horarios, y desde cualquier dispositivo que tenga acceso a internet. De esta forma, se trabaja con el esquema cliente/servidor, donde los usuarios que necesiten utilizar la plataforma podrán ingresar al sistema por medio de cualquier navegador web, ya sea desde una computadora, una tablet, un teléfono inteligente u otro dispositivo electrónico con conectividad a internet, a través de una interfaz de páginas web con la cual podrán interactuar con el servidor que contenga el sistema. Asi también no se necesita de ninguna instalación de software específico ni de equipos electrónicos especiales para utilizar la plataforma por parte de los usuarios y el sistema alojado en un servidor permite un mayor control y mejor mantenimiento, y aplicar medidas de seguridad y protección de datos más confiables, y copias de respaldo en caso de problemas o inconvenientes con el servicio, que garantiza una rápida respuesta en caso de alguna contingencia y restablecer el funcionamiento normal. Otra ventaja considerable es la posibilidad de crear una arquitectura de software basada en bloques o módulos independientes, que pueden ser desarrollados de acuerdo a las necesidades o requerimientos de funcionalidades, y pueden agregarse, modificarse o eliminarse sin que el sistema completo se vea afectado en su funcionamiento ([Molina Ríos et. al,](http://repositori.uji.es/xmlui/browse?type=author&authority=00e9b65e-d5f5-4133-8488-98af581641f5) 2018).

El software a desarrollar se pretende estructurar con una API principal que servirá de núcleo y va a concentrar el procesamiento y almacenamiento seguro de los datos. Una API, cuyas siglas provienen de los términos del inglés Application Programming Interfaces, que en español se traduce como Interfaces de Programación de Aplicaciones, se entiende como un conjunto de protocolos y definiciones que se establecen para poder establecer una comunicación entre dos o más aplicaciones, permitiendo su interacción y el intercambio de datos. Esta característica es la que posibilita el desarrollo de distintas aplicaciones con determinadas funcionalidades de forma independiente, y una vez terminados, todos los módulos puedan ser acoplados y funcionar conjuntamente. Las APIs se clasifican en locales y remotas. Las APIs locales se caracterizan por utilizarse dentro de un solo equipo computacional, en la comunicación entre el hardware que compone la computadora o la comunicación entre el software. Como ejemplo podrían ser: sistemas operativos, librerías de software, el hardware del equipo. Las APIs remotas son las comunicaciones entre distintos tipos de hardware o software utilizando un protocolo común establecido, normalmente separados físicamente. Se consideran APIs remotas: sistemas de bases de datos, binarias, web. Las APIs web son actualmente las más conocidas y con una amplia aplicación en desarrollos de software, basando su estándar de comunicación común utilizado en internet, el Protocolo de Transferencia de Hipertexto, más conocido por las siglas HTTP, abreviatura del inglés Hipertext Transfer Protocol (Revuelta Arribas, 2020).

Existen dos especificaciones de APIs web más utilizadas actualmente, denominadas SOAP y REST. Las APIs basadas en SOAP, del inglés “Simple Object Acces Protocol,” que en español podríamos traducir como “Protocolo de Acceso a Objetos Simples”, se caracteriza por utilizar XML, del inglés “Extensible Markup language”, en español “Lenguaje de Marcado Extensible” como formato de trasferencia de datos, y utiliza herramientas de desarrollo muy específicas para su implementación. La especificación REST, del inglés “Representational State Tranfer”, que traducido al español sería “Transferencia de Estado Representacional”, se resalta que puede implementar como formato de trasferencia de datos JSON, del inglés “JavaScrip Object Notation”, y en español “Notación de objeto JavaScript”, un formato sencillo y liviano, que actualmente tiene una gran aceptación y es cada vez más utilizado como alternativa al XML. La especificación REST se basa en los métodos HTTP para la comunicación e intercambio de datos entre las aplicaciones, por lo tanto utiliza los mismos métodos como GET para obtener datos, POST para el envío de datos para su almacenamiento, PUT para la modificación de datos existentes y DELETE para eliminar datos, por lo tanto esta especificación es una de las más utilizadas en la actualidad por su facilidad en la implementación, flexibilidad en su uso, eficiencia en el intercambio de datos y la gran cantidad de herramientas de desarrollo disponibles para su utilización. (Revuelta Arribas, 2020).

Considerando que la plataforma propuesta es un software planteado para funcionar en un entorno web, es primordial establecer el lenguaje de programación adecuado para este proyecto. Entendemos por lenguaje de programación como un conjunto de instrucciones que se utilizan para realizar una serie de operaciones que una computadora pueda ejecutar y con ello realizar distintos tipos de tareas (Sala, 2003). Los lenguajes de programación fueron evolucionando a la par que las computadoras, adaptándose y diversificándose en distintos tipos y naturalezas, de acuerdo a sus campos de acción. De esta forma tenemos lenguajes de programación enfocados a software de diversas clases, como software para uso de empresas, aplicaciones de uso personal, software centrado en teléfonos inteligentes y resaltamos los lenguajes destinados al desarrollo web.

De los lenguajes de programación enfocados a la web, podemos mencionar como los más importantes ASP, de la empresa Microsoft, y PHP, una herramienta de código libre. El lenguaje PHP es una de las más utilizadas en la actualidad, desde su creación en el año 1994, fue creciendo en características y funcionalidades y gracias a un sólido respaldo de una gran comunidad de desarrolladores se ha consolidado como referencia en la elección de los programadores (Díaz & Banchoff Tzancoff, 2000).

Así como el surgimiento y crecimiento de distintos tipos de lenguajes de programación, esto también derivo en la creación de complementos y utilidades para los mismos con el fin de facilitar su utilización e implementación. De esta forma aparecieron los denominados “frameworks” o “entornos de desarrollo”, también se pueden mencionar como “entornos de trabajo”. Los frameworks son un conjunto de herramientas y conjuntos de archivos que ofrecen determinadas funcionalidades y ofrecen al desarrollador una estructura o un esqueleto que le facilite la construcción de su proyecto. De esta forma un programador utiliza estos frameworks para desarrollar su software agregando los complementos, librerías, paquetes y otras opciones que necesita de una forma rápida y sencilla, reduciendo de forma significativa el tiempo de trabajo requerido para completar un proyecto. En relación al lenguaje PHP, existen varias opciones de frameworks basados en este lenguaje como base, y entre las más utilizadas podemos mencionar a Symfony, CodeIgniter, CakePHP y Laravel (Sierra, Acosta, Ariza & Salas, 2013).

Laravel es uno de los frameworks basado en PHP más utilizados en la actualidad, dispone de una gran cantidad de librerías, paquetes de archivos, utilidades y herramientas que facilitan el desarrollo del software y el trabajo de los programadores. Es un framework de código abierto, fue creado en el año 2011 y actualmente continúa su crecimiento incorporando nuevas características, mejoras de rendimiento y optimizando los procesos de creación de aplicaciones para alcanzar una dinámica de trabajo ágil y veloz. Un punto a destacar en Laravel es la implementación del patrón de diseño Modelo Vista Controlador o MVC, una arquitectura de software que separa la interfaz del usuario de la lógica del negocio y del manejo de datos, consiguiendo un código de programación más ordenado , limpio y comprensible, facilitando su reutilización, mantenimiento y modificación. El diseño MVC se basa en tres componentes que interactúan entre si para realizar los procesos del software, primeramente tenemos el Modelo, que representa la estructura de la información que maneja la aplicación y por lo tanto gestiona el acceso a esta información que se realizan por medio de consultas o modificaciones, por otra parte tenemos el Controlador, que se encarga de responder a los eventos que ocurren en el software, normalmente que provienen de un usuario, procesando los pedidos que se requieren y se comunica con los modelos para acceder a la información que se precisa, básicamente actuando como un intermediario entre las vistas y los modelos, por ultimo tenemos la Vista, que se encarga de mostrar al usuario de una forma ordenada y de fácil comprensión toda la información que son procesados por el controlador y que son proveídos por los modelos. En una forma sencilla de descripción podemos decir que un usuario visualiza la información e interactúa con el sistema por medio de las Vistas, a su vez éstas trasladan los pedidos del usuario a los Controladores que procesan estos pedidos y se conectan con los Modelos para acceder a la información y realizar las manipulaciones solicitadas, ya sean crear nueva información, consultar, modificar o eliminar datos. Por estas características el framework Laravel es una de las opciones más sólidas y viables para realizar a cabo un proyecto de desarrollo web (Ovando Ortega, 2019).

En todo sistema informático que se pretenda desarrollar un aspecto clave y primordial es la forma en que se va a tratar y manipular los datos que el sistema va a generar y almacenar. En respuesta a esta necesidad se crearon los Sistemas de Gestión de Bases de Datos (SGBD), que es un conjunto de aplicaciones que nos posibilitan la creación, gestión y administración de bases de datos, además de ofrecer las estructuras necesarias para el resguardo y la búsqueda de información del modo más eficiente posible. Los SGDB se clasifican en no relacionales (NoSQL) y relacionales (SQL). Los SGBD no relacionales (NoSQL) se identifican por no utilizar estructuras fijas de datos, como tablas y funcionan en entornos distribuidos, una desventaja de estos tipos de SGBD seria no garantizar una integridad completa de los datos. Como ejemplo podemos mencionar a Mongo DB, Redis, Cassandra. Los SGBD relacionales (SQL) son los más utilizados en la actualidad, ya que sus comienzos datan del año 1970, teniendo una larga trayectoria de evolución hasta nuestros días. Su funcionamiento se basa en estructurar los datos en tablas y estableciendo vínculos entre ellos y establece normas rígidas para asegurar la integridad y seguridad de los datos. Entre los SGBD relacionales más conocidos podemos mencionar SQL Server, de la empresa Microsoft y ORACLE, que son de pago y Postgresql, Mysql, MariaDB son software libre que pueden implementarse en un proyecto sin la necesidad de pago por licencias. De las opciones de SGBD relacionales de licencia libre, PostgreSQL es de los más utilizados y más completos, ofreciendo una solución robusta y potente para distintos tipos de desarrollo de software (Marín, 2019).

De esta forma, es factible desarrollar una plataforma para el registro de pacientes con dengue, utilizando una metodología de desarrollo ágil, en un entorno web que permita su acceso desde internet, en una estructura modular que permita su desarrollo en bloques independientes que una vez culminados puedan trabajar en conjunto, en la cual uno de los módulos sirva como núcleo de todo el sistema para el procesamiento y almacenamiento de los datos, basando su funcionamiento en una API REST que se encargaría de toda la lógica para la manipulación de los datos y su almacenamiento en un SGBD y ofrecer una interfaz de comunicación e intercambio de datos con los demás módulos de acuerdo a sus funcionalidades y requerimientos.

# 3. PREGUNTAS, HIPOTESIS Y OBJETIVOS

## 3.1. Preguntas:

- ¿Por qué desarrollar una plataforma para el registro de personas infectadas con Dengue destinado a la Séptima Región Sanitaria del departamento de Itapúa, república del Paraguay?

- ¿Qué metodología de desarrollo de software presenta las características necesarias para el sistema propuesto?

- ¿Qué sistema de software es posible desarrollar para cumplir con los requerimientos de una API REST como núcleo principal de la plataforma?

- ¿Qué software de programación presenta las funcionalidades requeridas para la generación de la lógica de funcionamiento, el procesamiento y almacenamiento de datos de la API REST?

## 3.2. Hipótesis:

La plataforma de registro de personas infectadas con Dengue permitirá a la Séptima Región Sanitaria del departamento de Itapúa tener acceso a una base de datos constantemente actualizada de los casos de la enfermedad registrados de la zona geográfica a la que pertenece. Además, contendrá un registro de todos los establecimientos de salud dependientes, así como de laboratorios de análisis clínicos, que serán los que registren los casos confirmados. A partir de los datos acumulados en el sistema se pueden aplicar varios procesos de selección, filtrado y algoritmos que puedan proveer información útil. Esta información podrá ser utilizada para crear estrategias de lucha contra la enfermedad, establecer un mejor manejo de recursos y prevenir las consecuencias que la dolencia genera en la población.

La utilización de una metodología de desarrollo de software ágil nos permite establecer una dinámica de trabajo más rápida, y se adapta mejor a los requerimientos contemplados en el proyecto. Esta metodología nos permite un mejor manejo de las etapas necesarias para las actividades previstas y la administración del tiempo en periodos más reducidos. Se reduce de manera significativa la documentación al menor nivel posible, evitando con esto la burocracia y la pérdida de tiempo. Se hace énfasis en el trabajo en equipo y colaborativo entre los involucrados, con una comunicación constante entre los miembros. El flujo de actividades es flexible e iterativo, se realizan evaluaciones constantes de los avances, los cuales ayudan en la corrección de errores y la toma de decisiones para definir nuevos objetivos o metas.

Para el desarrollo de la plataforma de registro de personas contagiadas de dengue, el sistema se orienta en un entorno web, instalado en un servidor conectado a internet. La ventaja principal de esa disposición es la capacidad de tener acceso al sistema desde cualquier dispositivo electrónico conectado a la web, a toda hora y desde cualquier lugar. El. El.

Para desarrollar la API REST principal se utilizará software de código abierto, cuya principal ventaja es la gratuidad de uso y posee una comunidad de desarrolladores que constantemente van actualizando, implementando mejoras de rendimiento y agregando nuevas funcionalidades. Se tiene una gran cantidad de software de desarrollo disponible y para esta plataforma la opción más valida sería el entorno de desarrollo LARAVEL con el lenguaje de programación PHP, para modelar la API REST con los procedimientos para el tratamiento y procesamiento de los datos y establecer la interfaz de comunicación para interactuar con los otros módulos. En relación al almacenamiento de la información se utiliza un sistema de gestión de bases de datos PostgreSQL, un software consolidado y con herramientas sólidas para el manejo óptimo y de forma segura de los datos.

## 3.3 Objetivos

- Desarrollar una plataforma de registro de personas infectadas con el virus del Dengue destinado a la Séptima Región Sanitaria del Departamento de Itapúa, República del Paraguay

- Seleccionar una metodología de desarrollo de software de acuerdo a las necesidades del sistema propuesto.

- Desarrollar una API REST para el sistema informático de la plataforma de registros de casos de personas infectadas con Dengue.

- Probar el funcionamiento de la API REST. Realizar

# 4. METODOLOGÍA

## 4.1. Tipo y diseño de investigación

El presente trabajo aborda el tipo de investigación aplicada debido a que se realiza una investigación con la finalidad de obtener un producto que pueda ser utilizado para el beneficio de la sociedad. Se pretende obtener conocimiento mediante la indagación y la recolección de información para que pueda resultar en la solución de una problemática de forma práctica (Muñoz Razo, 2011).

Mediante esta investigación aplicada se desea realizar una herramienta tecnológica definido como un software representado en una API destinada a una plataforma web de registro de pacientes con dengue, para solucionar la problemática de la gestión eficiente de la información sobre la enfermedad antes mencionada que afecta a una institución sanitaria identificada como la Séptima Región Sanitaria, del departamento de Itapúa, del país Paraguay.

El diseño corresponde a una investigación de caso práctico ya que aborda una problemática específica, que es el déficit de un sistema informático de registro de pacientes con Dengue en la organización sanitaria del departamento de Itapúa, y a partir del cual se desarrolla una propuesta de solución consistente en una plataforma web de registro de pacientes con Dengue. Para lograr el objetivo propuesto se utilizan los recursos y conocimientos que ofrecen las disciplinas relacionadas a la informática y la computación, con los cuales se desea llegar al resultado de una solución que derive en un beneficio a la organización y la comunidad al cual es destinado.

4.2. Definición del objeto de estudio

Se define como objeto de estudio las funcionalidades del módulo API principal de la Plataforma Web de registro de pacientes infectados con el virus del Dengue, que tiene por finalidad gestionar y almacenar los datos que se generen en la interacción con los demás módulos que componen el sistema. Esto es porque la plataforma web está estructurada como un conjunto de módulos de los cuales la API principal tiene como función establecer una interfaz de comunicación con los demás módulos para establecer un intercambio eficiente y seguro de los datos que se generen por el uso de la plataforma, y establecer una base de datos centralizada para concentrar los datos y tener una control más estricto con la medidas de seguridad necesarias que garanticen la confiabilidad e integridad de la información.

4.3. Descripción de la población y muestra

Establecemos la población como la totalidad de las funcionalidades del módulo que compone la API principal. Debido a que se considera la totalidad no se extrae una muestra.

4.4. Procedimiento de recogida de información:

Se realizó una revisión bibliográfica con la finalidad de establecer las herramientas de desarrollo más adecuadas al software proyectado y a partir de la utilización de una metodología de desarrollo de software ágil denominado SCRUM se aplicaron las herramientas para definir las funcionalidades y requerimientos necesarios para el modulo API principal, consistentes en las historias de usuario, el product backlog y otros sugeridos por la metodología.

4.5. Procesamiento de los datos

A partir de la revisión bibliográfica se establecieron las metodologías de desarrollo de software y las herramientas informáticas que mejor se adapten al proyecto propuesto, como la metodología SCRUM, el lenguaje de programación PHP, el framework laravel, el sistema de gestión de bases de datos PostgreSQL.

Aplicando las herramientas de la metodología SCRUM, como las historias de usuario, se definieron las características, requerimientos y funcionalidades necesarias para el módulo API, continuando con los demás procesos de la metodología ágil seleccionada para realizar el producto deseado. Se planificó el product backlog, siguiendo con el Sprint Planning, y la ejecución de los Sprints, con los cuales se fueron cumpliendo las iteraciones y entregando los incrementos del software hasta llegar al producto final (Ilustración 1).

Ilustración 1. Esquema de la metodología de desarrollo SCRUM.

## 

## Fuente: (Navarro Cadavid et. al, 2013).

## 4.6 Alcance y Limitaciones del Software

Alcances:

- El software está diseñado para funcionar en un servidor web.

- El sistema ofrece una interfaz web para la gestión de Usuarios, Entidades, Roles y Permisos.

- La interfaz de comunicación de la API es por medio del protocolo REST.

- El formato para el intercambio de datos es JSON.

- El almacenamiento de datos se realiza en una Base de Datos PostgreSQL.

- La autenticación de los usuarios de realiza a través de Tokens de seguridad.

Limitaciones:

- El modulo debe tener conexión a internet para su funcionamiento.

- La comunicación con la API se realiza solamente con el protocolo REST, no contempla otro tipo de protocolo.

- El formato de datos de entrada y de salida de la API se limita al formato JSON, no admite otros formatos.

- La autenticación de usuarios se realiza únicamente por medio de Tokens, no se contemplan otros métodos de autenticación

# 5. RESULTADOS Y DISCUSIÓN

5.1. Planificación general del Módulo API

5.1.1. Selección de la metodología de desarrollo

Luego de una revisión bibliográfica de las diferentes metodologías de desarrollo de software, se optó por utilizar una metodología de desarrollo ágil denominada SCRUM, considerándose la más apropiada para este proyecto.

La metodología SCRUM se caracteriza por ofrecer un entorno de trabajo con una variedad de herramientas, métodos y recursos que se pueden implementar para lograr un desarrollo de software exitoso. Ofrece flexibilidad para ajustarse a las necesidades del proyecto y un flujo de trabajo dinámico que permite lograr los objetivos propuestos en periodos cortos de tiempo. La administración de las actividades y la gestión de avances se realizan en forma sencilla y clara sin caer en complicaciones ni en procesos burocráticos que entorpezcan los avances de las actividades.

Esta metodología se compone de varios instrumentos que se pueden utilizar de acuerdo a las necesidades del proyecto. En este proyecto se utilizaran el product backlog, consistente en una lista de funcionalidades que se definen mediante el análisis de requerimientos del software, una vez elaborado esta lista se continúa con el Sprint planning, que es la planificación de las etapas de desarrollo del software, en las cuales se establece un periodo de tiempo de duración y un conjunto de funcionalidades y tareas a ser realizadas, denominado Sprint backlog, que se realiza a partir del product backlog. Una vez cumplido un Sprint, se aplica el Sprint retrospective, definido como un análisis de los resultados del Sprint, cuáles fueron los avances y las dificultades o inconvenientes encontrados, con la finalidad de detectar los errores cometidos y corregirlos para el próximo Sprint, además de contemplar los aspectos a mejorar. Una vez terminado el Sprint, se obtiene un incremento del producto, y se continúa con el siguiente Sprint, hasta llegar al producto final.

5.1.2. Establecimiento de requerimientos y funcionalidades del modulo

La API REST principal tiene como finalidad servir de núcleo para la plataforma de gestión de pacientes con Dengue. Básicamente todos los módulos que componen la plataforma realizarán el intercambio de datos con la API principal, el cual se encargará de administrar y tener acceso exclusivo a la base de datos, por lo tanto todas las consultas, la creación o la edición de la información que se genere en la plataforma será gestionado por la API.

Por medio del protocolo REST los demás módulos podrán comunicarse con la API utilizando los siguientes métodos:

Método GET: para la consulta de datos, la API recibe la petición, realiza la consulta a la base de datos y retorna una respuesta en formato JSON.

Método POST: para almacenar datos, la API realiza los procesos requeridos para guardar los datos en la base de datos.

Método PUT: para editar o modificar datos, la API recibe los datos a ser modificados, consulta a la base de datos y realiza los cambios requeridos.

Método DELETE: para eliminar o dar de baja datos, la API recibe la petición, busca el dato y procede a su eliminación.

Todos los métodos REST serán accesibles por medio de endpoints o puntos de entrada, representados como URI, siglas en inglés de Universal Resource Identifier, o Identificador Universal de Recursos. El formato de entrada y salida de datos será en JSON, un formato ligero y sencillo que facilita su manejo y manipulación.

El modulo también ofrecerá una interfaz web para la gestión de los usuarios de la plataforma, así también como las entidades a las que pertenecen los usuarios, los roles y permisos.

5.1.3. Software y herramientas de desarrollo seleccionados para el módulo

De acuerdo a las necesidades del proyecto se establecen las siguientes herramientas informáticas:

Trello: aplicación web online para la gestión de proyectos a partir de tableros para administrar las tareas y una variedad de herramientas para el control y organización de los trabajos y actividades.

PHP: lenguaje de programación de código abierto con enfoque al entorno web

Laravel: es un entorno de desarrollo para crear aplicaciones y servicios web, utilizando el lenguaje PHP

Composer: Es un administrador de librerías y dependencias para PHP, utilizado en la instalación de proyectos realizados en el entorno de desarrollo Laravel.

PostgreSQL: Sistema de gestión de base de datos para la estructuración y administración de almacenamiento de los datos

Git: software para el control de versiones para el registro de las modificaciones y el avance de un proyecto.

Github: plataforma web que trabaja conjuntamente con Git para tener un control de versiones con repositorios en línea.

Postman: Aplicación para la prueba y testeo de software API REST.

Atom: editor de código para realizar los cambios en los archivos del proyecto.

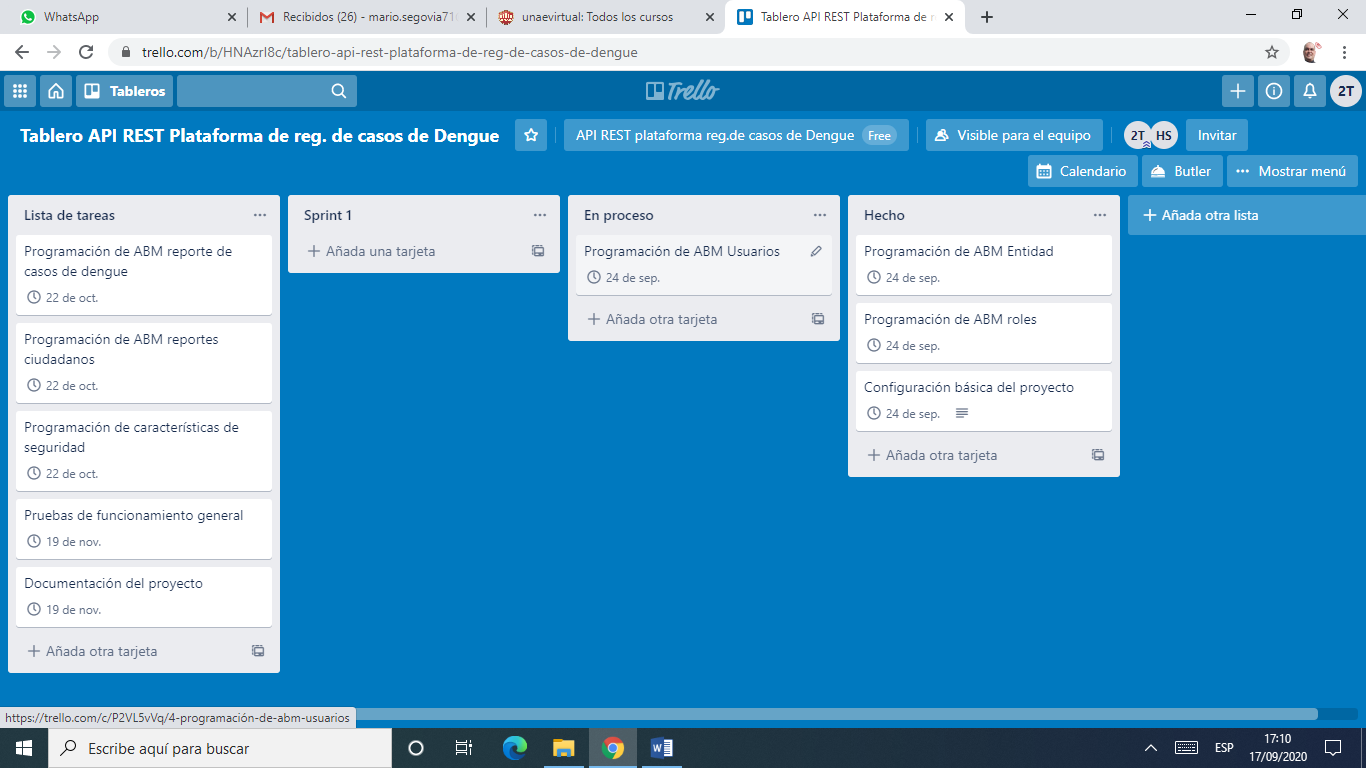
5.1.4. Planificación de actividades aplicando la metodología de desarrollo ágil SCRUM .

A partir del instrumento denominado product backlog de la metodología SCRUM, se realiza un análisis de los requerimientos para el modulo API, con las cuales se realiza una lista de funcionalidades y se les asigna una prioridad que se deben atender primero de acuerdo a su necesidad. A partir del product backlog se procede a realizar el sprint planning, la planificación de las etapas en las que se asigna una determinada cantidad de funcionalidades a ser completadas, establecer un periodo de tiempo para cumplir con las tareas y lograr un incremento del producto. Cada vez que se completa un Sprint, se realiza un Sprint retrospective, consistente en verificar los avances e inconvenientes encontrados durante la realización del Sprint, introducir mejoras y corregir los errores detectados, para continuar con el Sprint siguiente.

Se utilizó un tablero de la página web Trello para organizar las funcionalidades, representar el product backlog, los sprints con las funcionalidades correspondientes, el periodo de tiempo establecido, y los avances en las tareas. La interfaz de Trello se compone de columnas y tarjetas que se pueden ir moviendo de una columna a otra, de una forma sencilla e intuitiva que facilita la organización de las actividades de nuestro proyecto

Para el tablero Trello correspondiente al modulo API se estableciron cuatro columnas, una columna destinada para el product backlog, con las tareas pendientes a realizarse, otra columna representaba las tareas establecidas para le Sprint, una columna para las tareas en proceso de realización y por ultimo una columna con las tareas finalizadas (Ilustración 2).

Ilustración 2. Tablero Trello con las tareas del product backlog del módulo API.



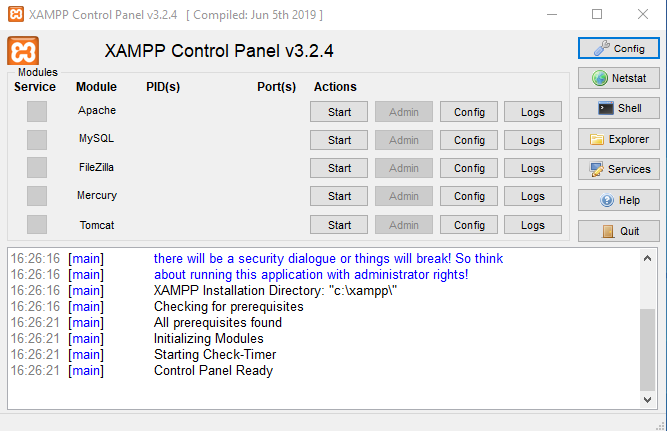
Fuente: Elaboración propia.

5.2. Desarrollo

5.2.1. Instalación de librerías del lenguaje PHP por medio del paquete de instalación XAMPP

Para la disponibilidad de las dependencias necesarias para el lenguaje PHP se utilizó el paquete de instalación XAMPP, que ofrece además un conjunto de aplicaciones como el servidor Apache, el sistema de gestión de base de datos MariaDB/Mysql y un panel de control para un fácil acceso a las configuraciones.

Ilustración 3. Panel de control de la aplicación XAMPP.

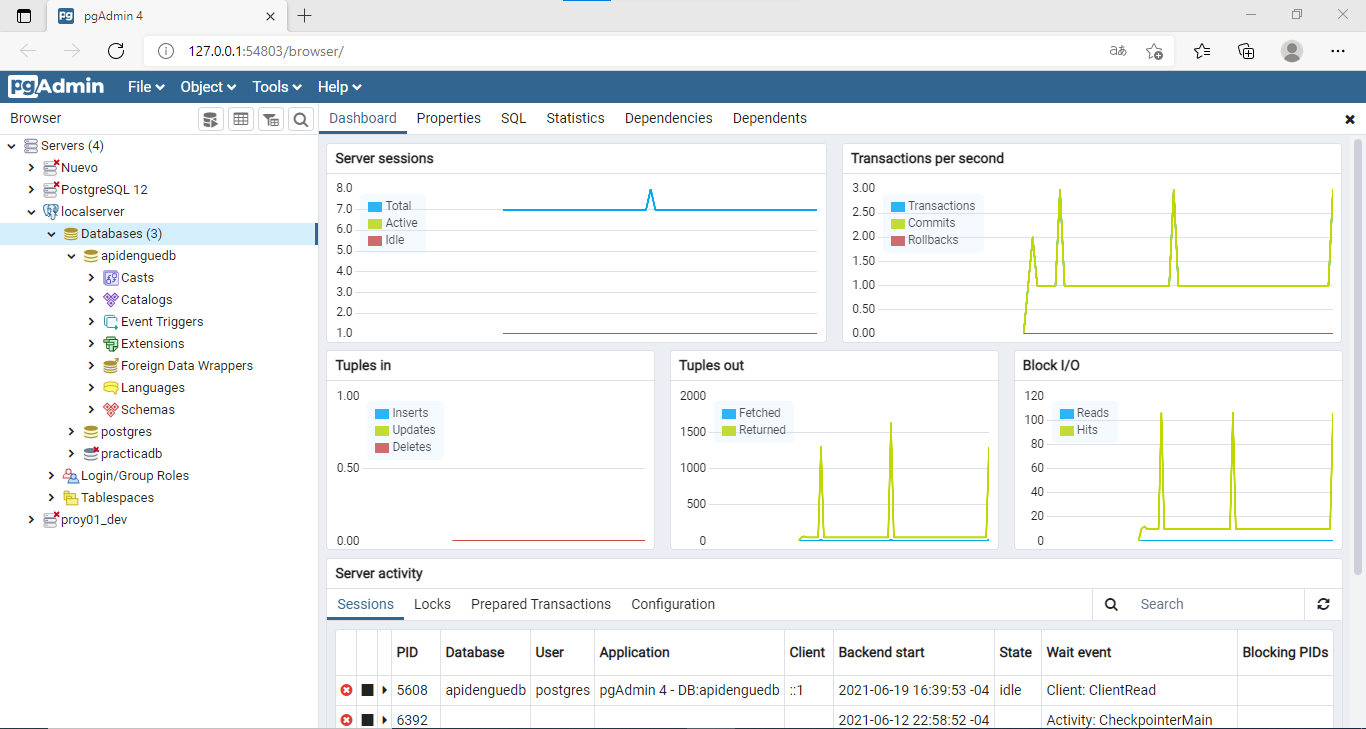


Fuente: elaboración propia

5.2.2. Instalación y configuración del gestor de base de datos PostgreSQL.

Como sistema de gestión de base de datos se optó por PostgreSQL. La instalación se realizó por medio de un archivo ejecutable que se puede descargar de la página web oficial. La instalación incluye la aplicación PgAdmin, una interfaz gráfica de entorno web para la gestión de las bases de datos existentes, de una manera sencilla y fácil (Ilustración 4).

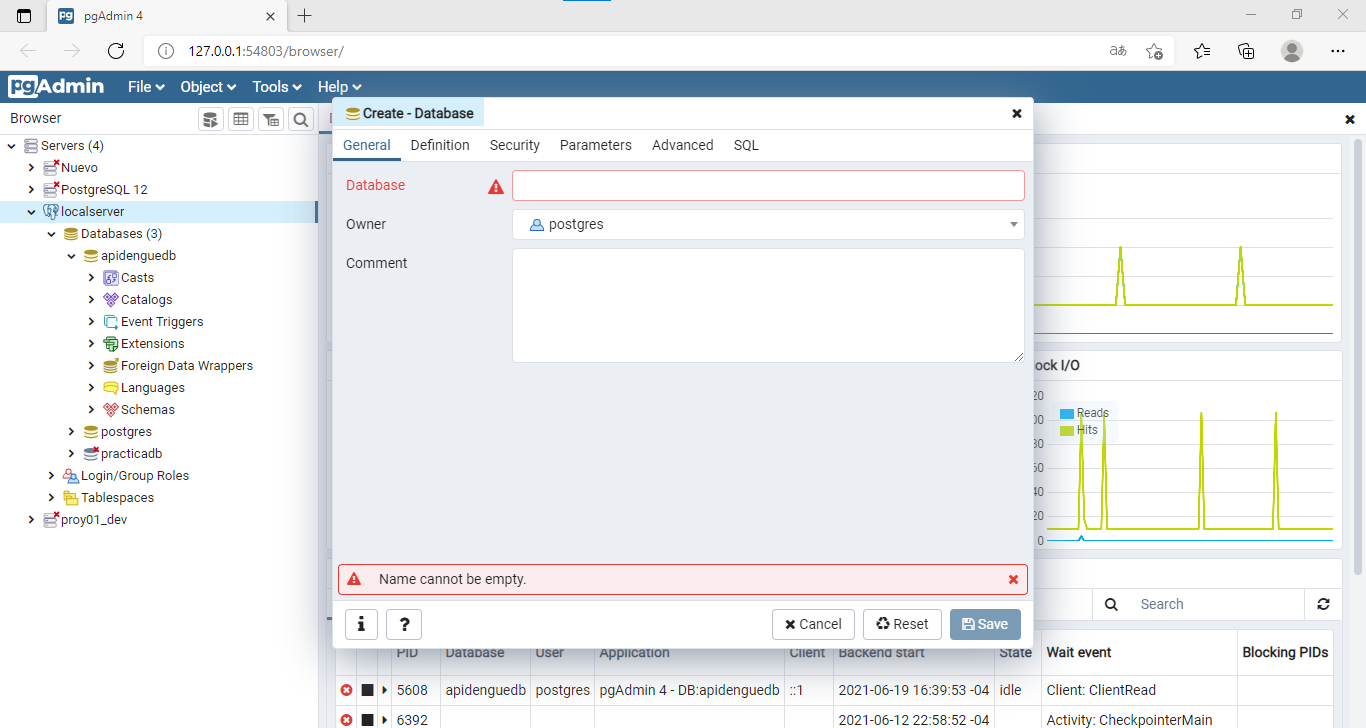
Ilustración 4. Panel principal de PgAdmin.



Fuente: Elaboración propia.

Una vez instalada el PostgreSQL se procedió a crear la base de datos a utilizarse para el modulo, a través de la interfaz de PgAdmin con la opción create database

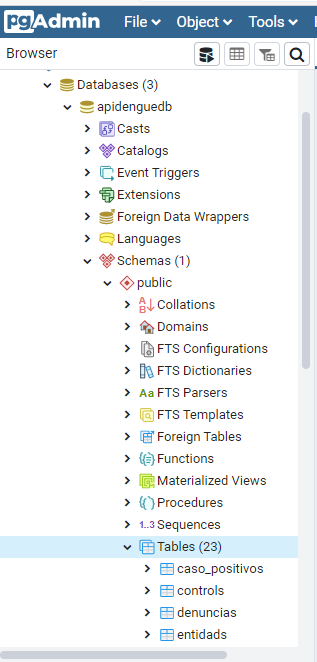
Ilustración 5. Menu de PgAdmin para la creación de una base de datos.



Fuente: Elaboración propia

Una vez creada la base de datos ya podemos visualizar el contenido, como las tablas de datos, y disponemos de una gran variedad de herramientas para gestionar el contenido de nuestra base de datos

Ilustración 6. Visualización de la base de datos creada para el proyecto

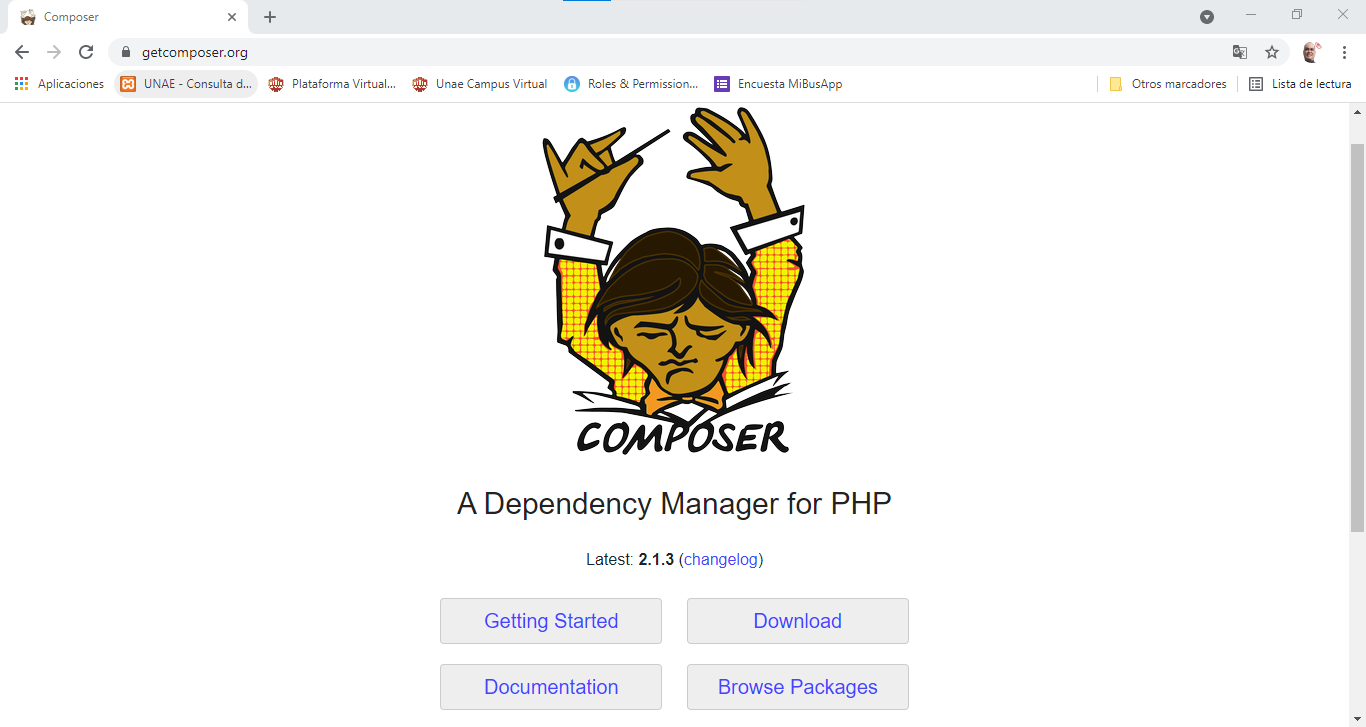


Fuente: Elaboración propia

5.2.3. Instalación de Composer

Como prerrequisito para la realización de un proyecto basado en el entorno de trabajo Laravel, se necesita de una aplicación para la gestión de paquetes y dependencias para el lenguaje PHP denominado Composer. Para la instalación se debe descargar de la página web oficial el archivo ejecutable (Ilustración 5).

Ilustración 7. Página web del gestor de dependencias para PHP Composer.



Fuente: Elaboración propia.

5.2.4. Creación del proyecto Laravel

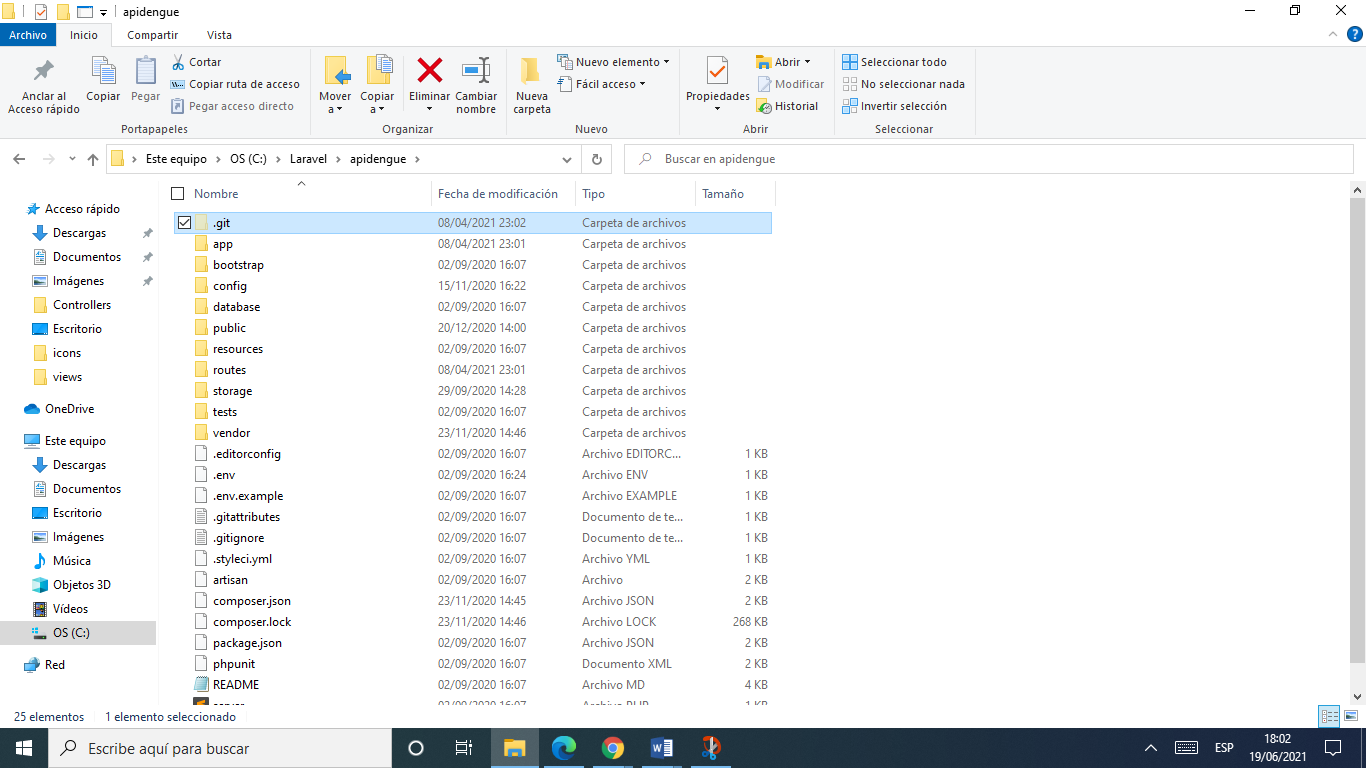
Para la creación del proyecto Laravel se cumplieron con los prerrequisitos necesarios, como la instalación del gestor de dependencias Composer, y de las librerías del lenguaje PHP, así como la creación de una base de datos destinado para el módulo.

A partir de una interfaz de línea de comandos se ejecutó la siguiente línea para crear la base del proyecto:

composer create-project laravel/laravel apidengue

Mediante este comando se crea un directorio o carpeta donde se establecen todas las dependencias y archivos necesarios para el funcionamiento del proyecto (Ilustracion 8)

Ilustración 8. Directorio del proyecto Laravel

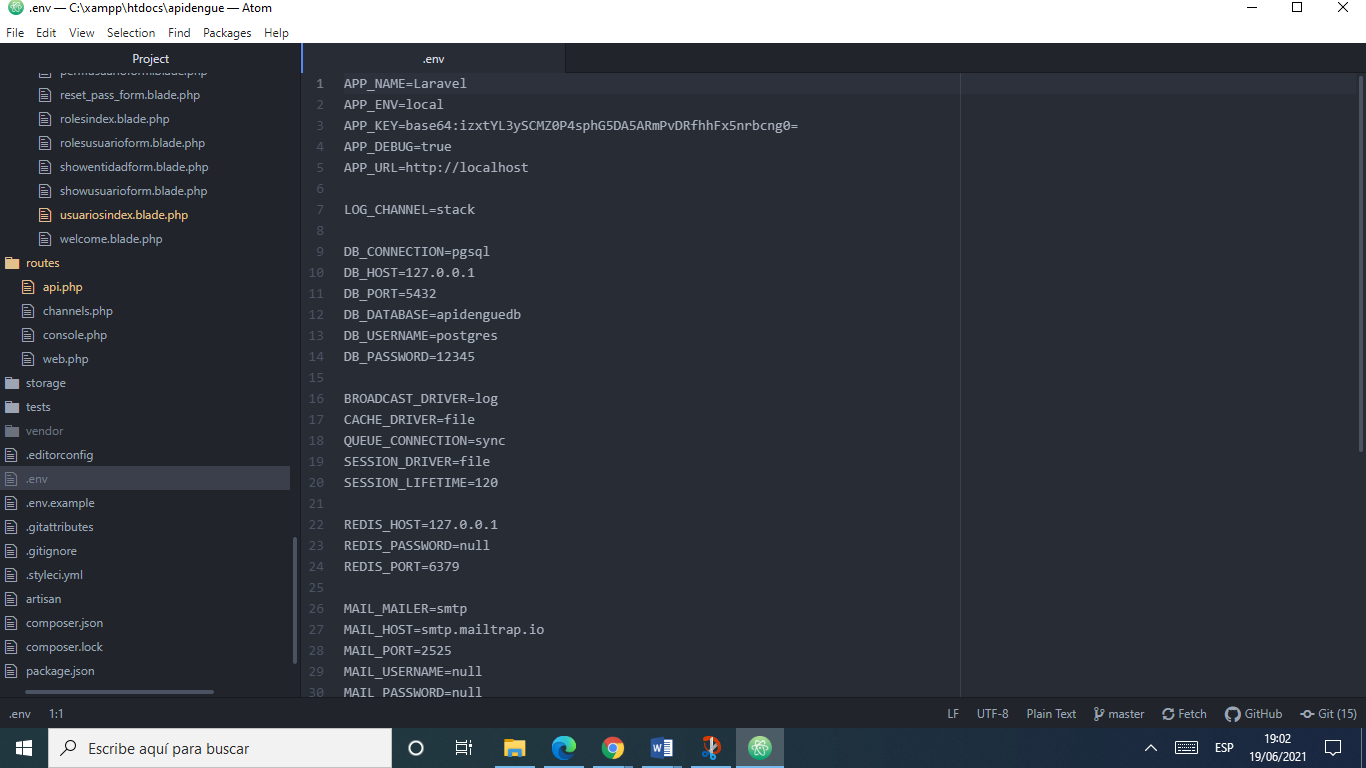


Fuente: Elaboración propia.

5.2.5. Configuraciones iniciales del proyecto Laravel

Una vez creado el proyecto, se procede a configurar el archivo “.env” para realizar la conexión con la base de datos. La edición del archivo se realiza con el editor de código Atom (Ilustración 9).

Ilustración 9. Configuración del archivo “.env” para establecer la conexión con la base de datos



Fuente: elaboración propia

La configuración tiene los siguientes parámetros:

- Pgsql: establece que la base de datos es postgreSQL

- 127.0.0.1: representa la dirección ip de la base de datos

- 5432: puerto de comunicación con la base de datos

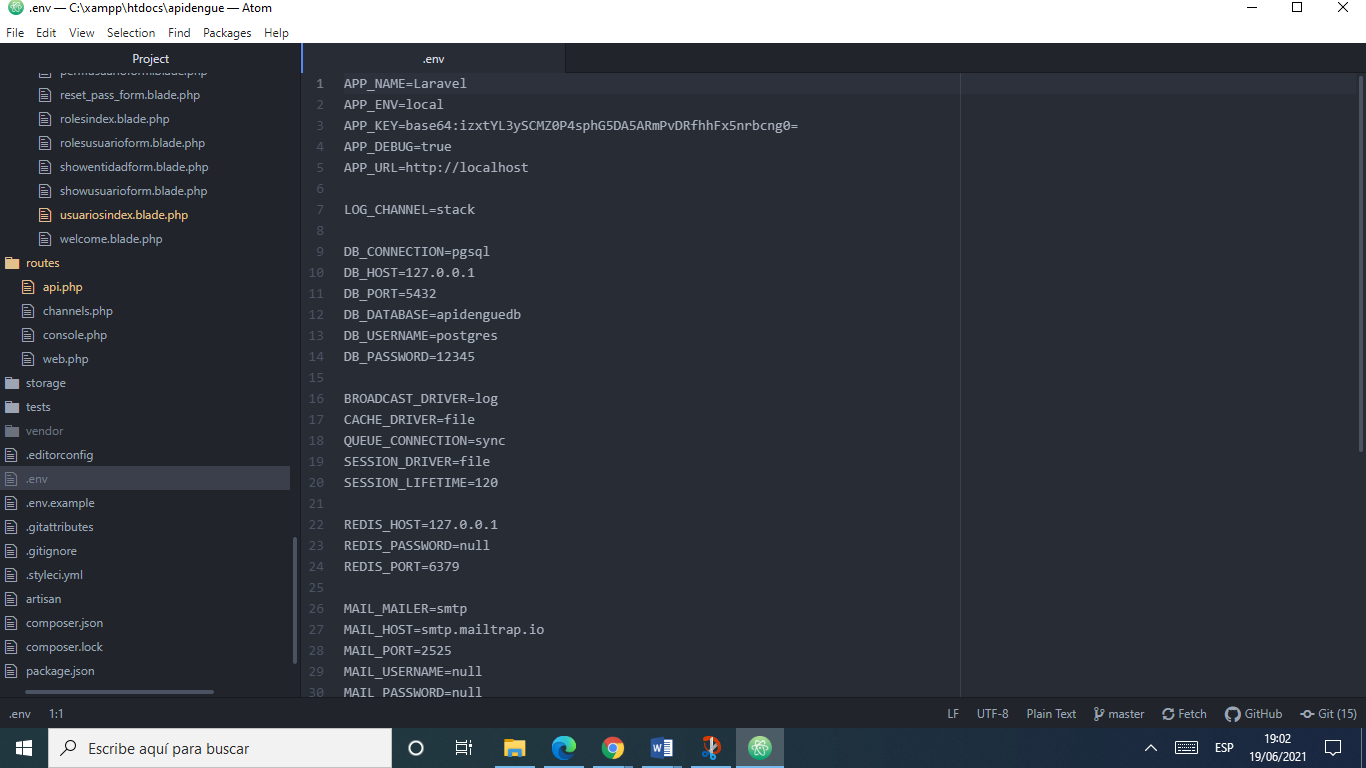
- Apidengue: nombre de la base de datos

- Postgres: el nombre de usuario de la base de datos

- 12345: contraseña de usuario de la base de datos.

Si los parámetros son correctos la aplicación ya puede conectarse e interactuar con la base de datos (Ilustración 10).

Ilustración 10. Configuración del archivo “.env” para la conexión con la base de datos.

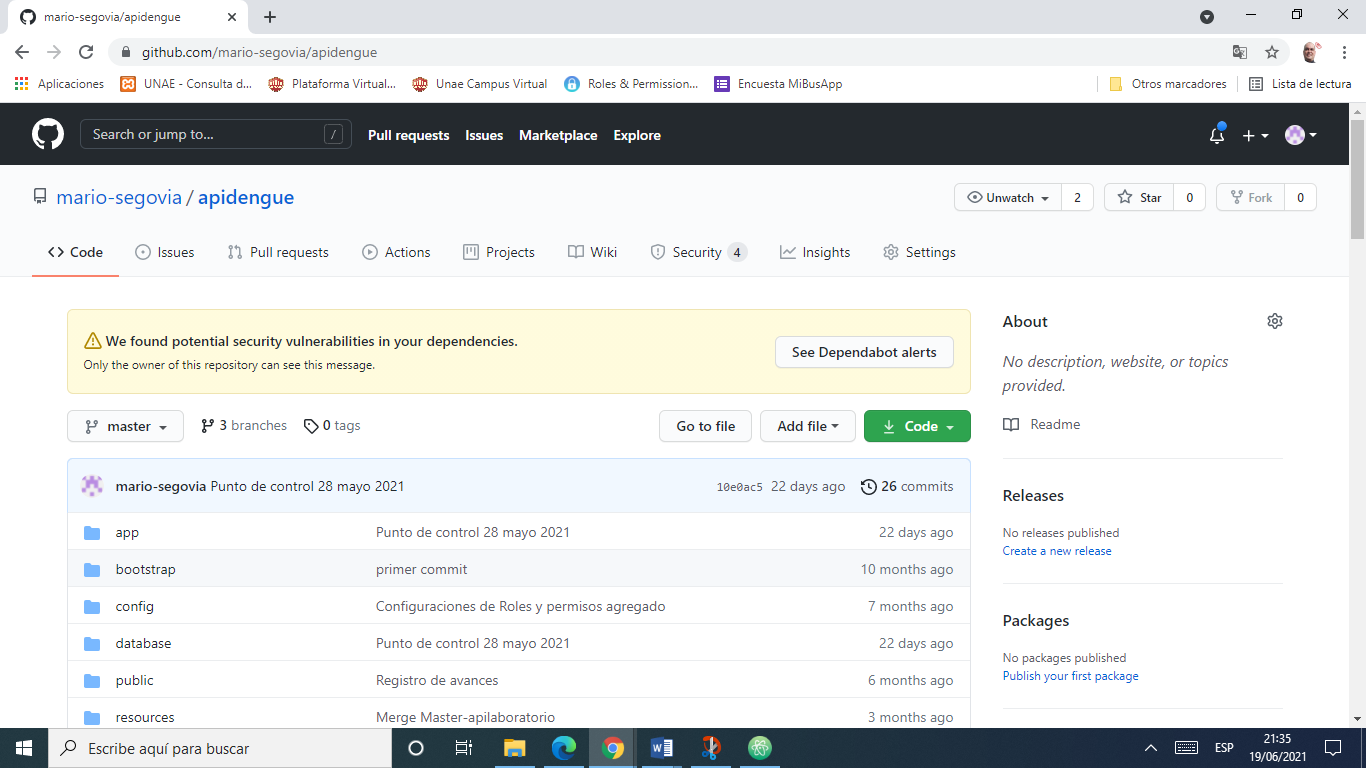


Fuente: Elaboración propia.

5.2.6. Creación de repositorio en Github para el control de versiones del proyecto.

Para el control y registro de los avances y cambios en el código del proyecto se utilizó un repositorio en Github (Ilustración 11).

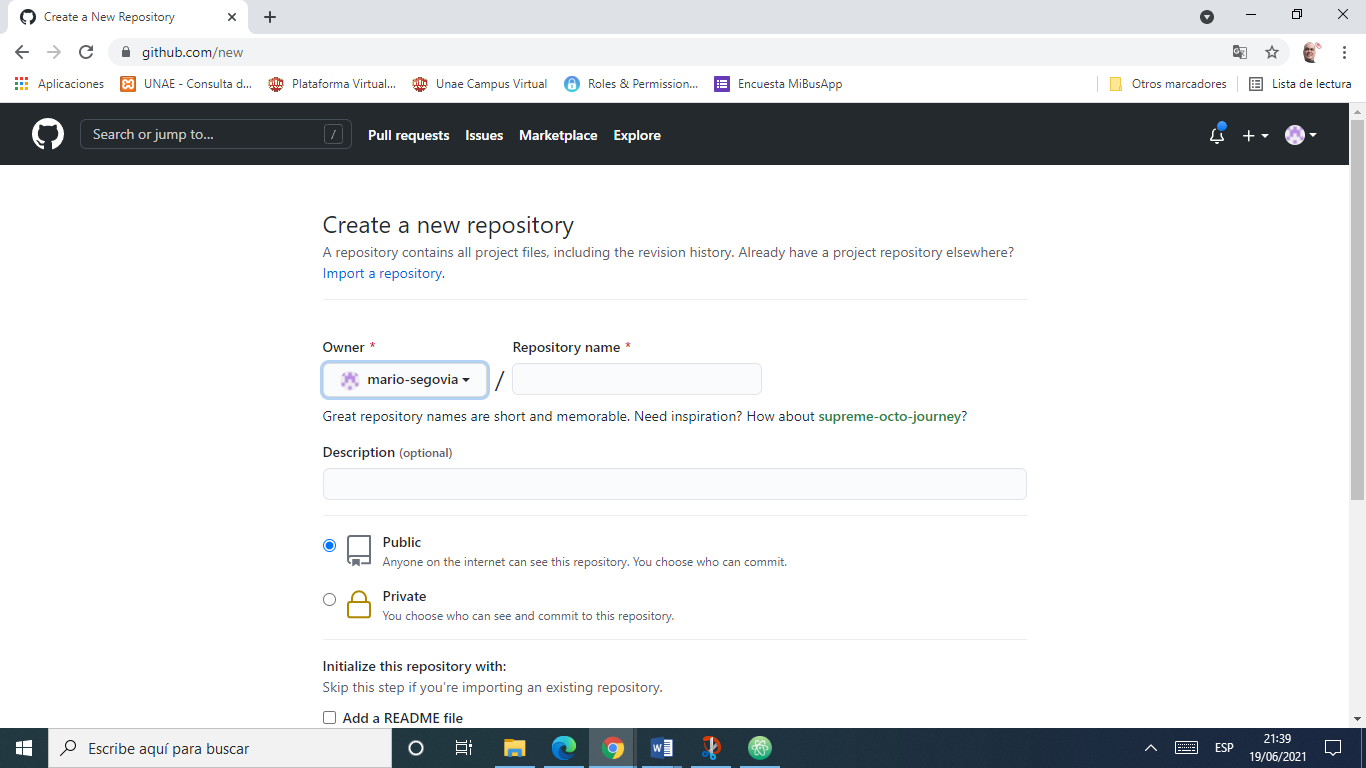
Ilustración 11. Repositorio del módulo API en Github.



Fuente: Elaboración propia

Los pasos realizados fueron primeramente la creación de un nuevo repositorio en la página web de Github (Ilustracion 12).

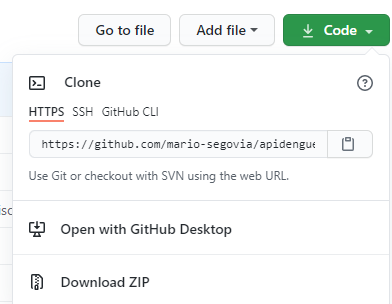
Ilustración 12. Formulario para la creación de un repositorio nuevo en Github.



Fuente: elaboración propia.

Una vez creado el repositorio se obtiene el enlace al repositorio para vincular la carpeta donde se encuentra nuestro proyecto, en la *Ilustración 12* podemos visualizar el enlace HTTPS que utilizaremos para configurar la conexión.

Ilustración 13. Enlace al repositorio Github.



Fuente: Elaboración propia.

Para inicializar el repositorio git local y comenzar los registros de cambios se deben ejecutar los siguientes comandos:

- git init

Este comando crea e inicializa los archivos para registrar los cambios.

- git add .

Se preparan todos los archivos existentes en el directorio para ser rastreados.

- git commit -m "first commit"

Se guardan el estado de los archivos en el repositorio local

- git remote add origin https://github.com/NOMBRE\_USUARIO/NOMBRE\_PROYECTO.git

Se establece la conexión con el repositorio remoto de Github.

- git push -u origin master

Este último comando envía al repositorio remoto los cambios registrados en el repositorio local, sincronizando los archivos y mantener el historial de cambios.

Una vez hecho estos procedimientos, el proyecto está listo para registrar los avances, cambios y modificaiones de acuerdo a las actualizaciones periódicas que realicemos en los repositorios. En este caso, para cada vez que se desea realizar un nuevo punto de registro de avances y cambios, se utilizan los siguientes comandos:

- git add .

Para detectar los cambios y nuevos archivos en el repositorio y preparar su registro

- git commit -m "mensaje del commit"

Guardar los cambios detectados con el comando anterior, agregando un mensaje para identificar el commit.

- git push origin master

Guardar y sincronizar los cambios detectados en el repositorio local con el remoto.

5.2.7. Programación de las funcionalidades del módulo API.

Con la configuración del entorno de trabajo completo, se procede a la codificación de las funcionalidades establecidas en el product backlog, de acuerdo al orden de prioridades asignadas en la planificación.

Siguiendo la pauta del patrón de diseño MVC o Modelo Vista Controlador, de la cual los proyectos basados en Laravel basan principalmente su funcionamiento, se procede a la abstracción y representación de las funcionalidades basándose en esta estructura.

El patrón MVC se basa en tres componentes que interactúan entre sí para realizar los procesos necesarios que requiere el software para su funcionamiento, que se describen a continuación:

- Modelo: es la representación de la información y de los datos que maneja el modulo, mediante este componente se realizan los procesos de creación, modificación y acceso a la información, y está relacionado con la conexión a la base de datos.

- Controlador: Se encarga de atender los eventos que ocurren en el software, normalmente las interacciones que realizan los usuarios por medio de las vistas, y se comunican con el Modelo para procesar los datos que se requieran, básicamente actúa como un intermediario en la comunicación e intercambio de datos entre las vistas con los modelos.

- Vista: es la interfaz con la cual los usuarios pueden interactuar con el software, se encarga de presentar los datos que requiere el usuario de una forma sencilla y fácil de entender y envía las solicitudes al controlador para realizar los procesos correspondientes.

Cabe destacar que el modulo API presenta una interfaz de comunicación a través de Endpoints o puntos de acceso, representados por medio de URIs o Identificador Universal de Recursos. Por medio de estos puntos de acceso los demás módulos que componen la plataforma podrán interactuar con la API para el intercambio de datos.

5.2.8. Programación de Modelos en el proyecto Laravel.

Para la creación de los modelos en Laravel se utiliza el siguiente comando:

- php artisan make:model Modelo

Este comando crea un archivo que representa al Modelo de una funcionalidad específica, en la cual se deben agregar los atributos o campos de datos que lo componen, asi como los procedimientos y las relaciones que establecerá con los otros modelos.

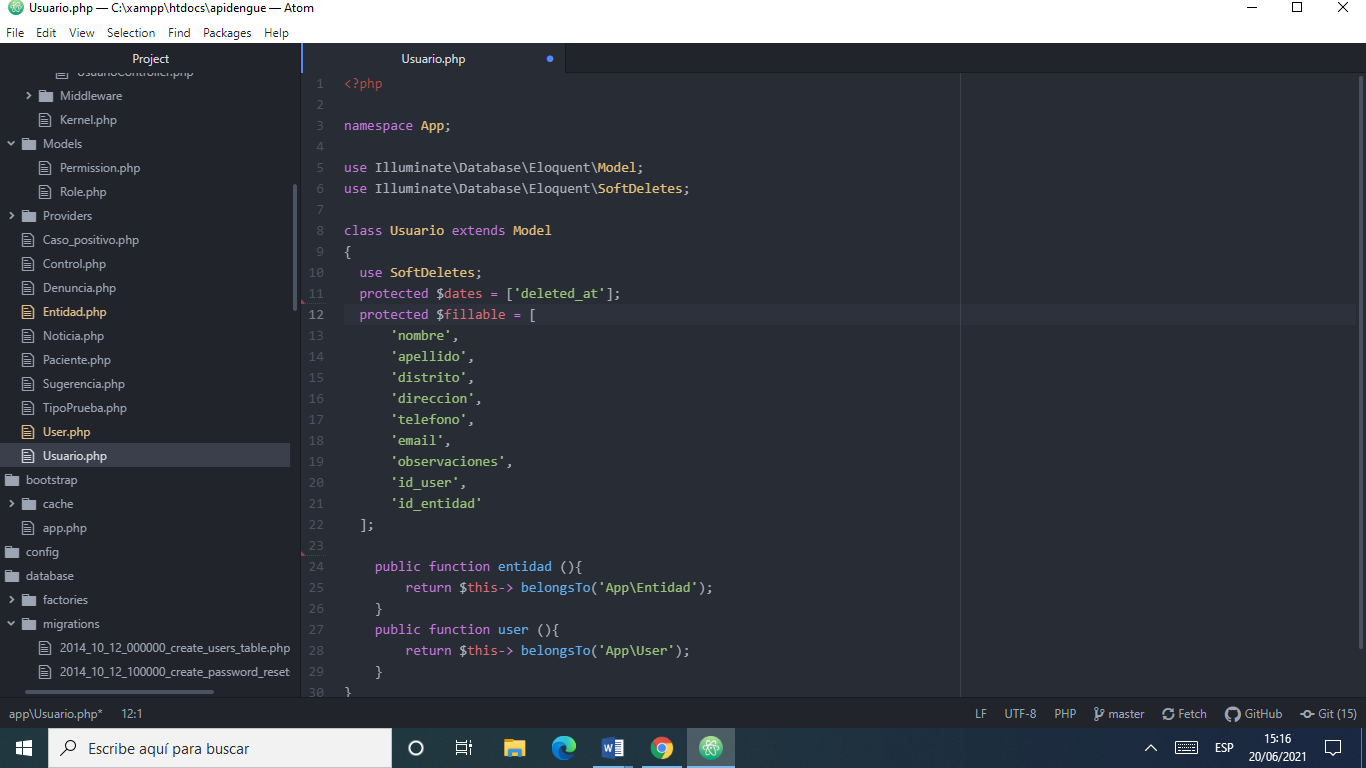
En este proyecto para la creación de los Modelos se utilizó el siguiente comando:

- php artisan make:model Modelo -a

Este comando agrega el parámetro “-a” el cual además de crear el Modelo, crea los demás componentes necesarios como el Controlador y la migración a la base de datos, cuyas funciones detallaremos más adelante.

En la *Ilustración 14* podemos ver un ejemplo de un Modelo en Laravel.

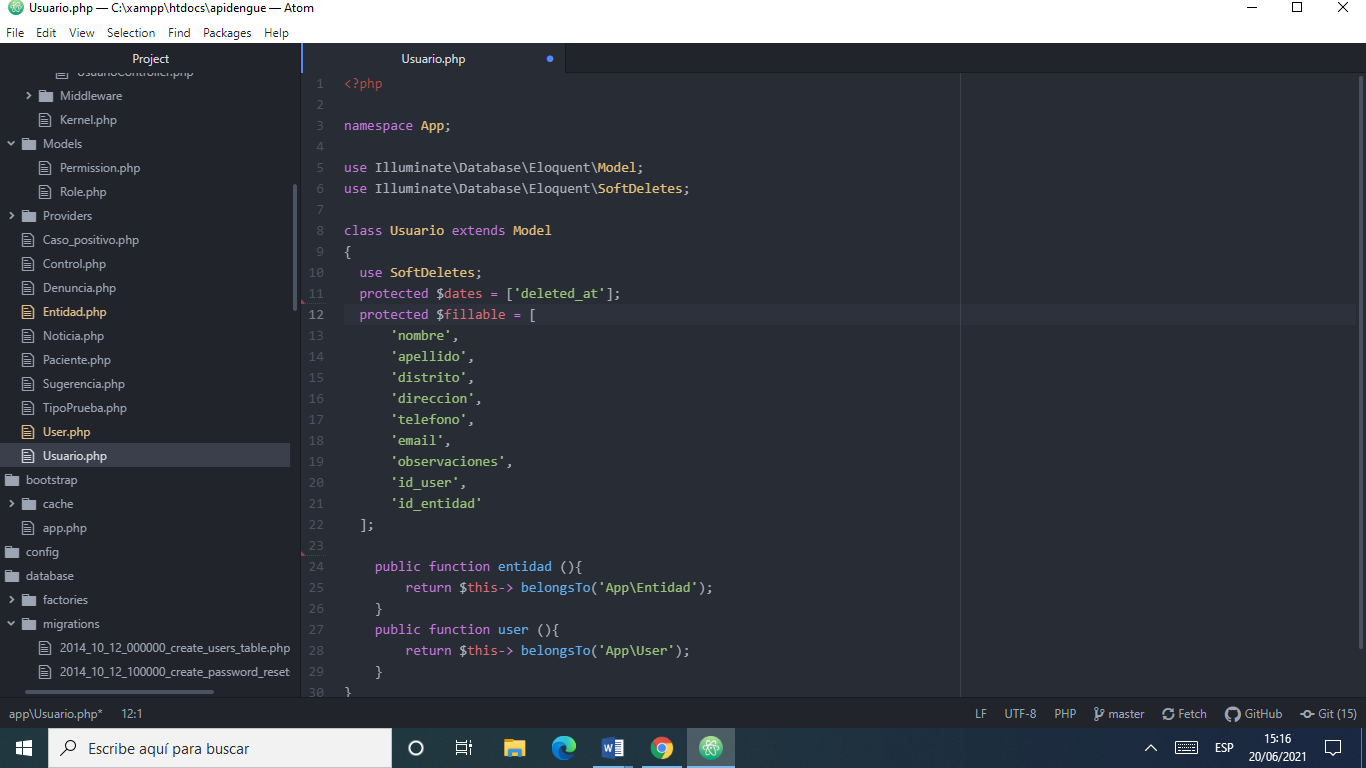
Ilustración 14. Ejemplo de Modelo de Laravel.



Fuente: Elaboración propia.

La *Ilustración 15* nos muestra la configuración de los campos de datos que componen el Modelo.

Ilustración 15. Atributos o campos de datos que componen el Modelo.

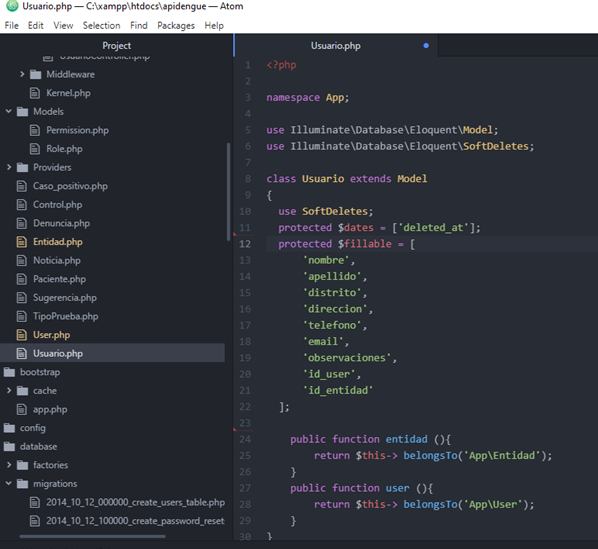


Fuente: Elaboración propia.

Fuente: Elaboración propia.

En la I*lustración 16* podemos visualizar las relaciones del Modelo con los demás Modelos, con los cuales se crean los vínculos para interactuar entre ellos.

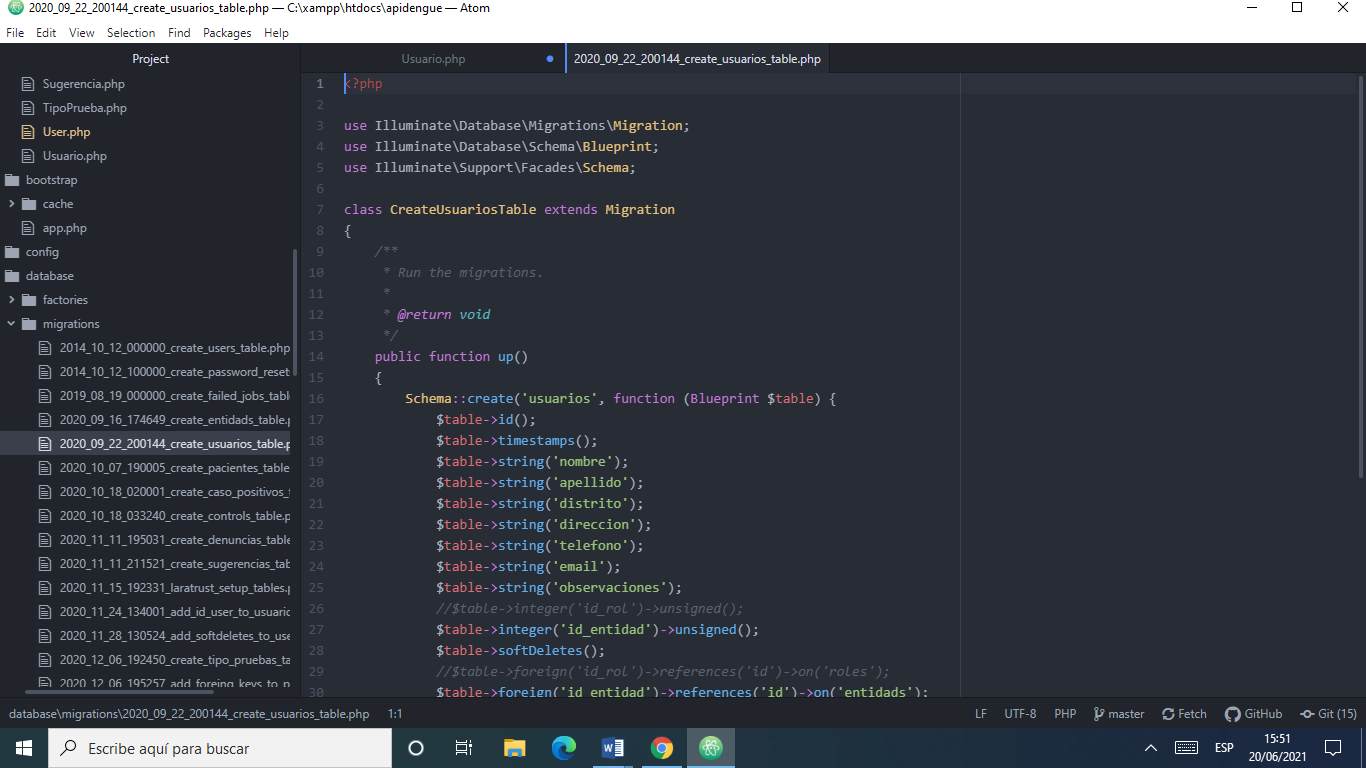
Ilustración 16. Relaciones de dependencia con otros modelos



Fuente: Elaboración propia.

Al crear un Modelo, normalmente ya creamos de forma simultánea una migración, que es un archivo que nos permite realizar el proceso de crear la tabla en la base de datos que represente a la estructura de datos del modelo (Ilustración 16).

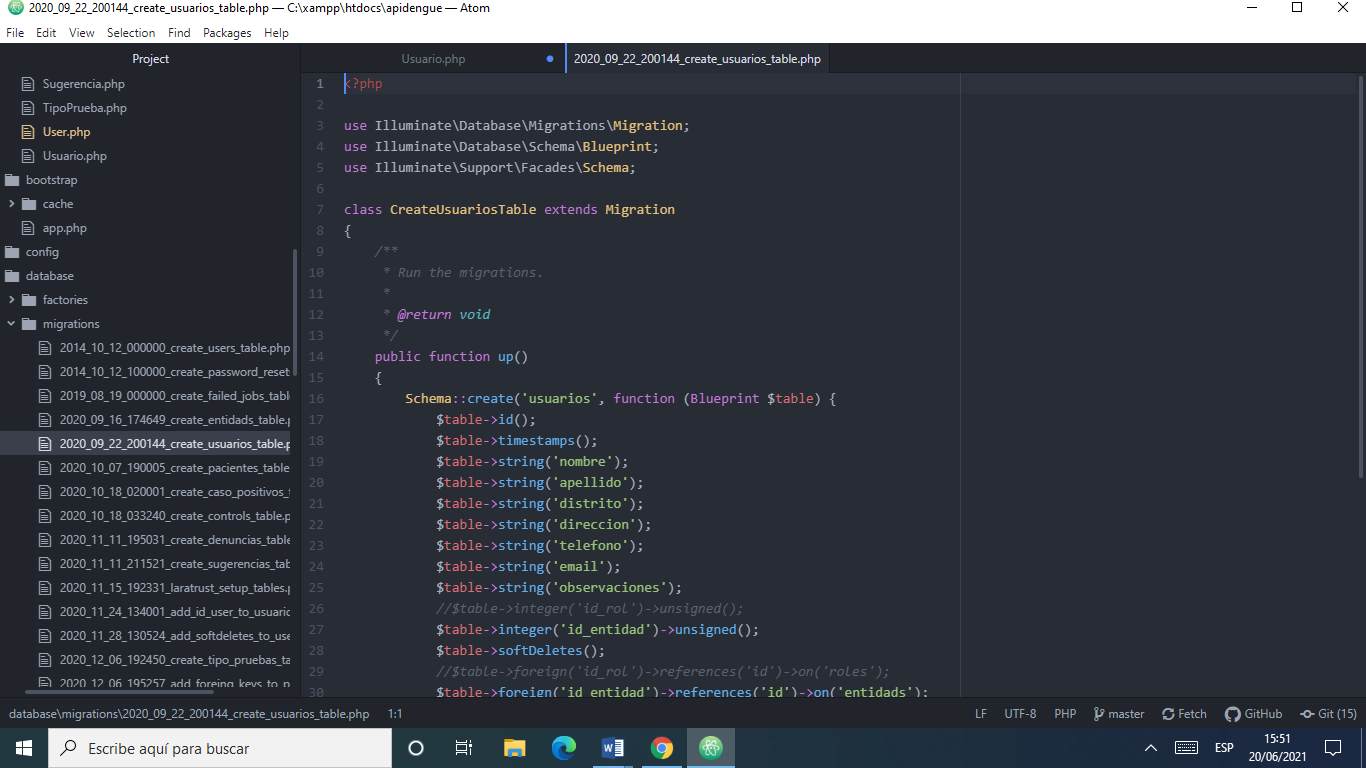
Ilustración 17. Archivo de migración.



Fuente: Elaboración propia.

En la migración se deben especificar todos los campos de datos que contendrá la tabla en la base de datos, asi como el tipo de datos, como texto, numero, booleano, relaciones con otras tablas, entre otros (Ilustración 18).

Ilustración 18. Campos de datos de la migración.



Fuente: Elaboración propia.

Laravel ofrece además la posibilidad de crear migraciones para realizar otro tipo de operaciones además de crear tablas en la base de datos, como ser agregar nuevos campos a tablas ya existentes, cambiar el tipo de datos, eliminar tablas o campos de datos, establecer relaciones, entre otras opciones. El comando para crear una nueva migración es el siguiente:

- php artisan make:migration create\_ejemplo\_table

Una vez configurada correctamente un archivo de migración, se procede a la ejecución de la migración para llevar a cabo los procesos para crear las tablas correspondientes en la base de datos. El comando para ejecutar las migraciones se presenta a continuación:

- php artisan migrate

De esta forma ya se tienen en la base de datos las tablas que representan a los Modelos del proyecto y ya se puede interactuar para el intercambio de datos.

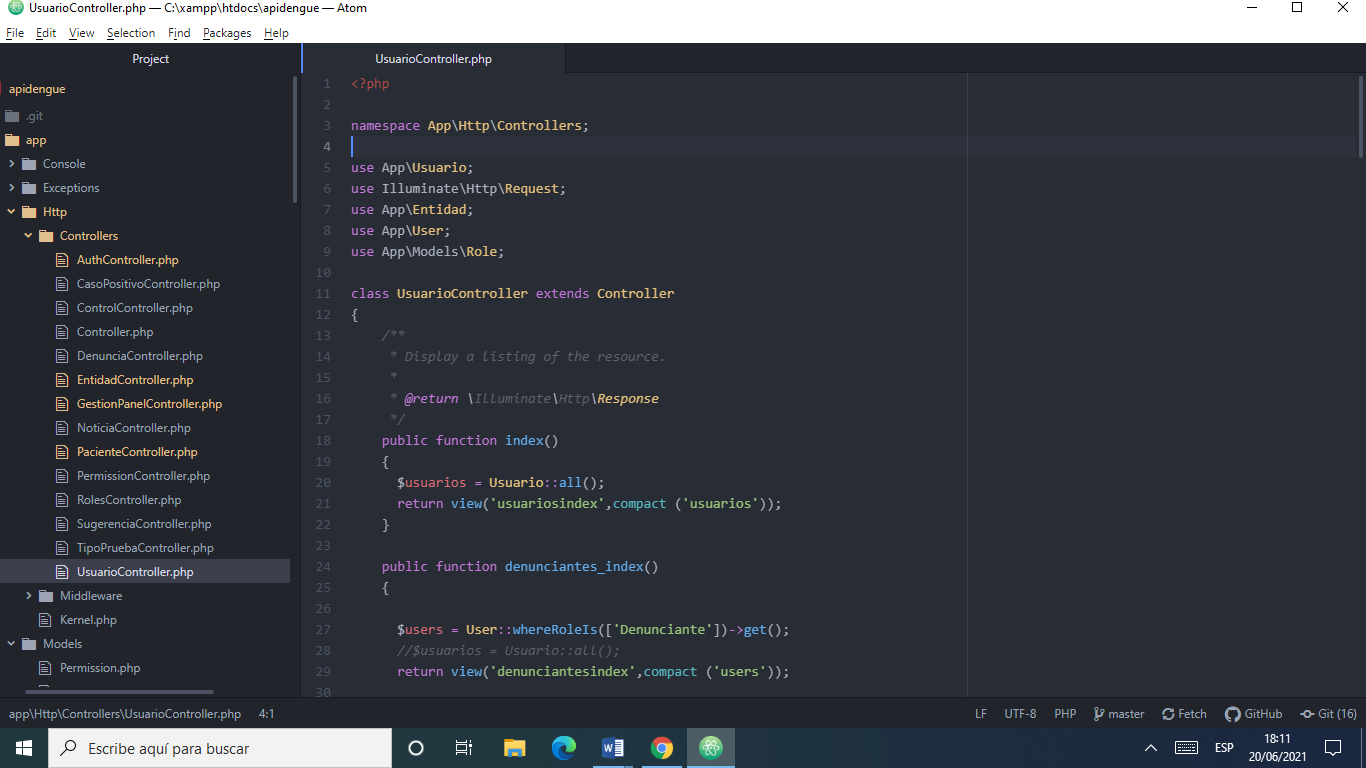
5.2.9. Programación de Controladores en el proyecto Laravel.

Los controladores se pueden generar automáticamente al crear un modelo, pero también se pueden generar por medio de este comando:

- php artisan make:controller EjemploController

Los controladores se componen de métodos que al ser invocados realizan un proceso determinado y normalmente retornan una respuesta (Ilustración 19).

Ilustración 19. Controlador de Laravel.

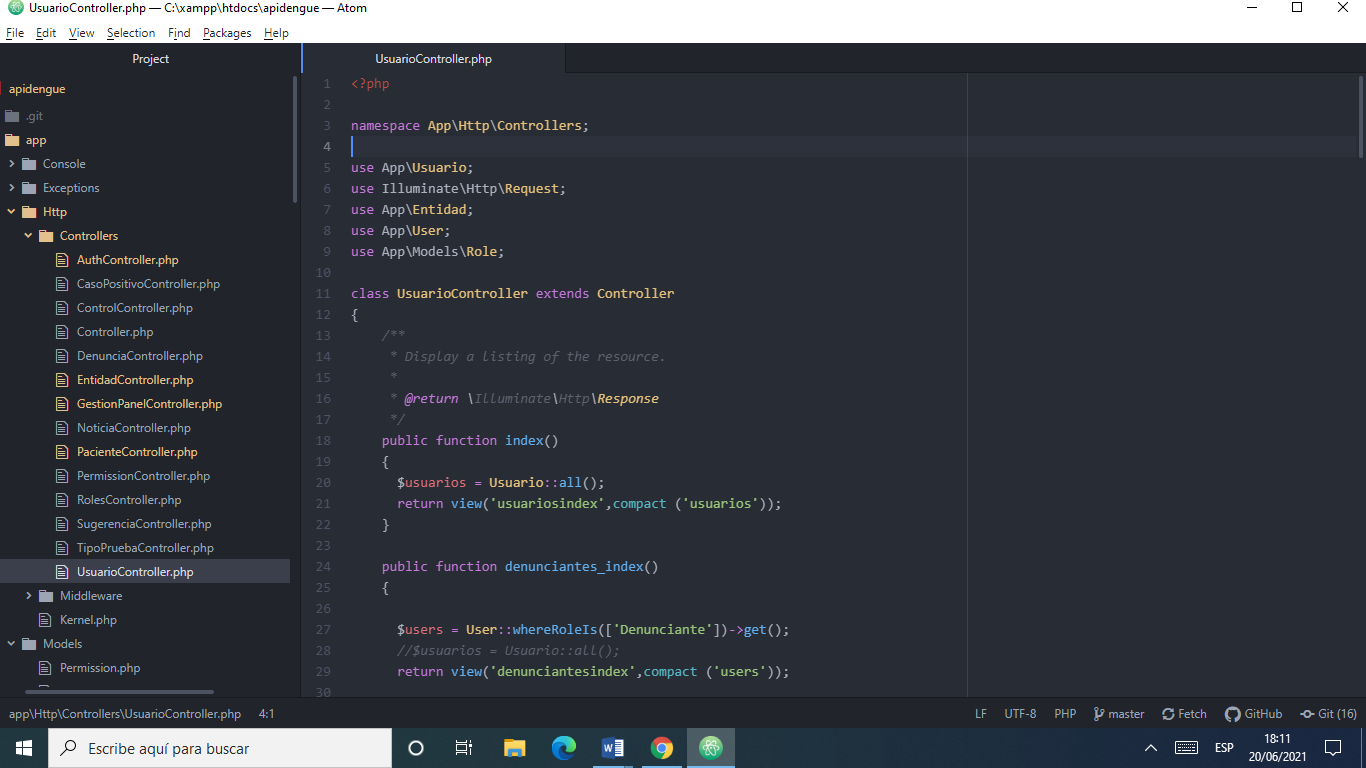


Fuente: Elaboración propia.

En este proyecto los controladores asociados a los modelos presentan unos métodos comunes asociados a los procesos de uso habitual en la gestión de los datos, que son crear, modificar, mostrar y eliminar datos que se encuentren almacenados en la base de datos, estos son los métodos a considerar:

- index: este método se utiliza para devolver la totalidad de registros existentes en la base de datos relacionados a un modelo específico (Ilustración 20).

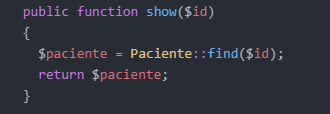
Ilustración 20. Método index de un controlador.



Fuente: Elaboración propia.

- show: método para mostrar un registro especifico de un modelo dado (Ilustración 21).

Ilustración 21. Método show de un controlador



- store: método utilizado para guardar un nuevo registro de un modelo específico (Ilustración 22).

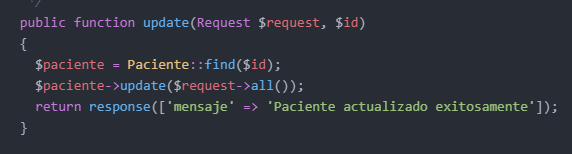
Ilustración 22. Método store de un controlador



Fuente: Elaboración propia.

- update: método para guardar los cambios de un registro del modelo (Ilustración 23).

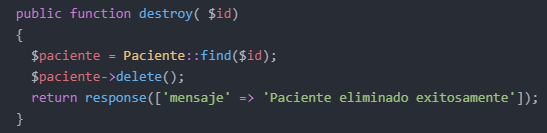
Ilustración 23. Método update de un controlador



Fuente: Elaboración propia.

- destroy: método para eliminar un registro del modelo (Ilustración 24).

Ilustración 24. Método destroy de un controlador.



Fuente: Elaboración propia.

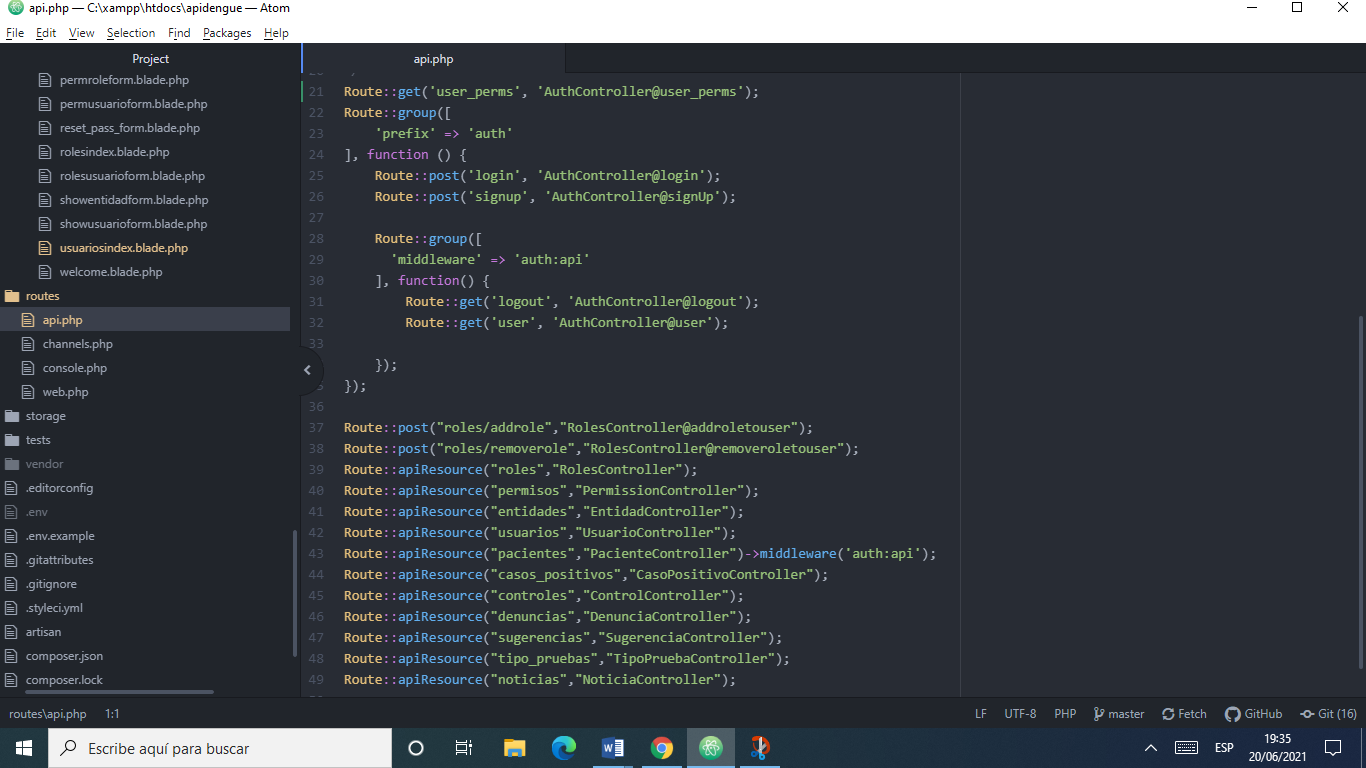
Así también estos métodos pueden utilizarse para realizar diversos procesos de acuerdo a las necesidades del proyecto de una forma flexible y sencilla.

Fuente: Elaboración propia.

5.2.10. Programación de rutas para los puntos de acceso o endpoints en el módulo API.

La configuración de las rutas para crear las URIs o endpoints, que utilizaran los demás módulos de la plataforma para interactuar con la API se realiza en el archivo “api” situado en el directorio “routes” (Ilustración 25).

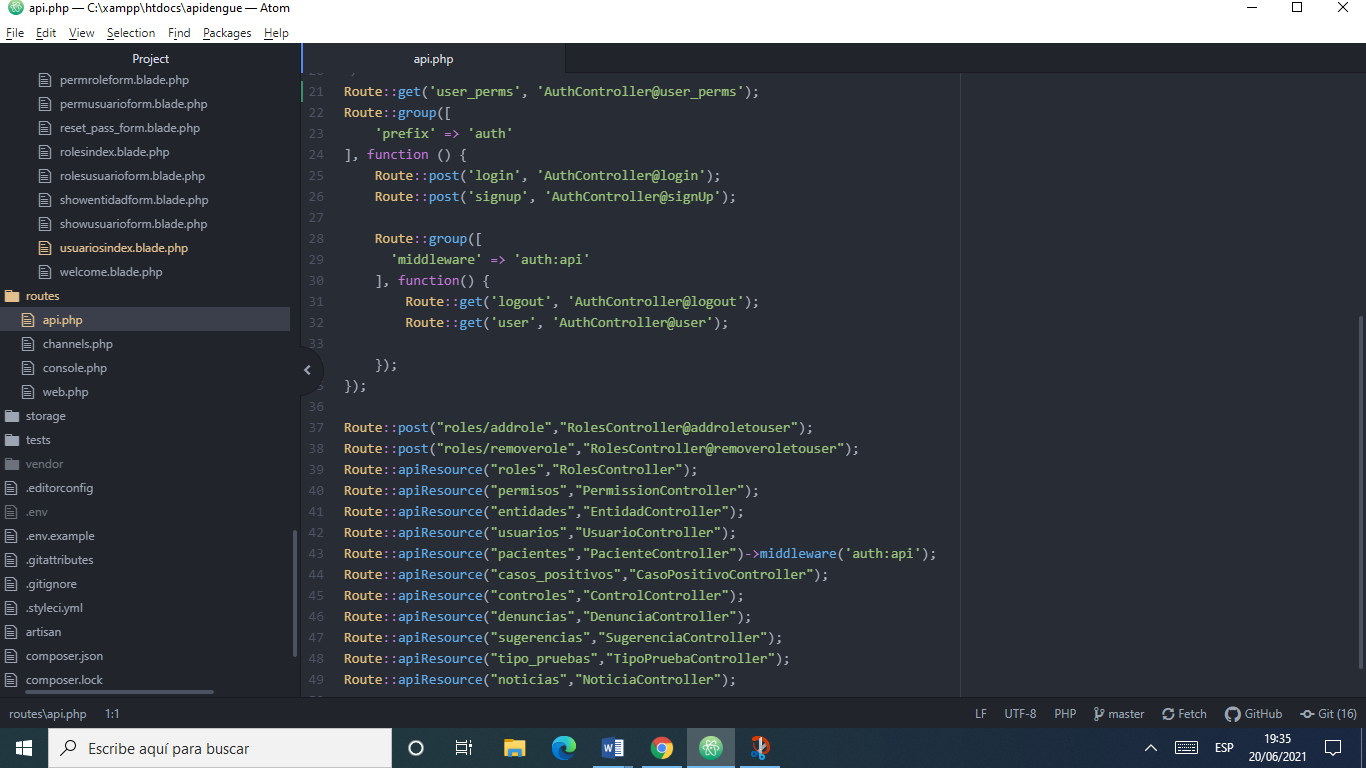
Ilustración 25. Archivo “api”, para la configuración de rutas del modulo.



Fuente: Elaboración propia.

Las rutas se construyen con la instancia Route, seguido del método, que pueden ser get, post put, delete, el prefijo de la ruta con el que se identifica el recurso, y el controlador al que apuntan estas rutas (Ilustración 26).

Ilustración 26. Rutas de acceso a los recursos del módulo.



Fuente: Elaboración propia.

Laravel ofrece varias configuraciones posibles para facilitar la construcción de las rutas de acceso. Una de estas configuraciones que más se utilizó en este proyecto es la opción “apiResource”, que mediante esta característica asignada a un controlador automáticamente se generan las siguientes rutas:

- https://servidor/proyecto/api/recurso

Por el método GET se obtienen como respuesta todos los registros existentes del recurso, por medio del método index del controlador.

- https://servidor/proyecto/api/recurso/{parámetro}

Por el método GET se obtiene como respuesta el registro del recurso identificado por el parámetro agregado a la ruta, por medio del método show del controlador.

- https://servidor/proyecto/api/recurso

Por el método POST se puede crear un nuevo registro del recurso, por medio del método store del controlador

- https://servidor/proyecto/api/recurso/{parámetro}

Por el método PUT se realiza la modificación del registro del recurso identificado por el parámetro agregado a la ruta, por medio del método update del controlador.

- https://servidor/proyecto/api/recurso/{parametro}

Por el método DELETE se elimina el registro del recurso identificado por el parámetro agregado a la ruta, por medio del método destroy del controlador.

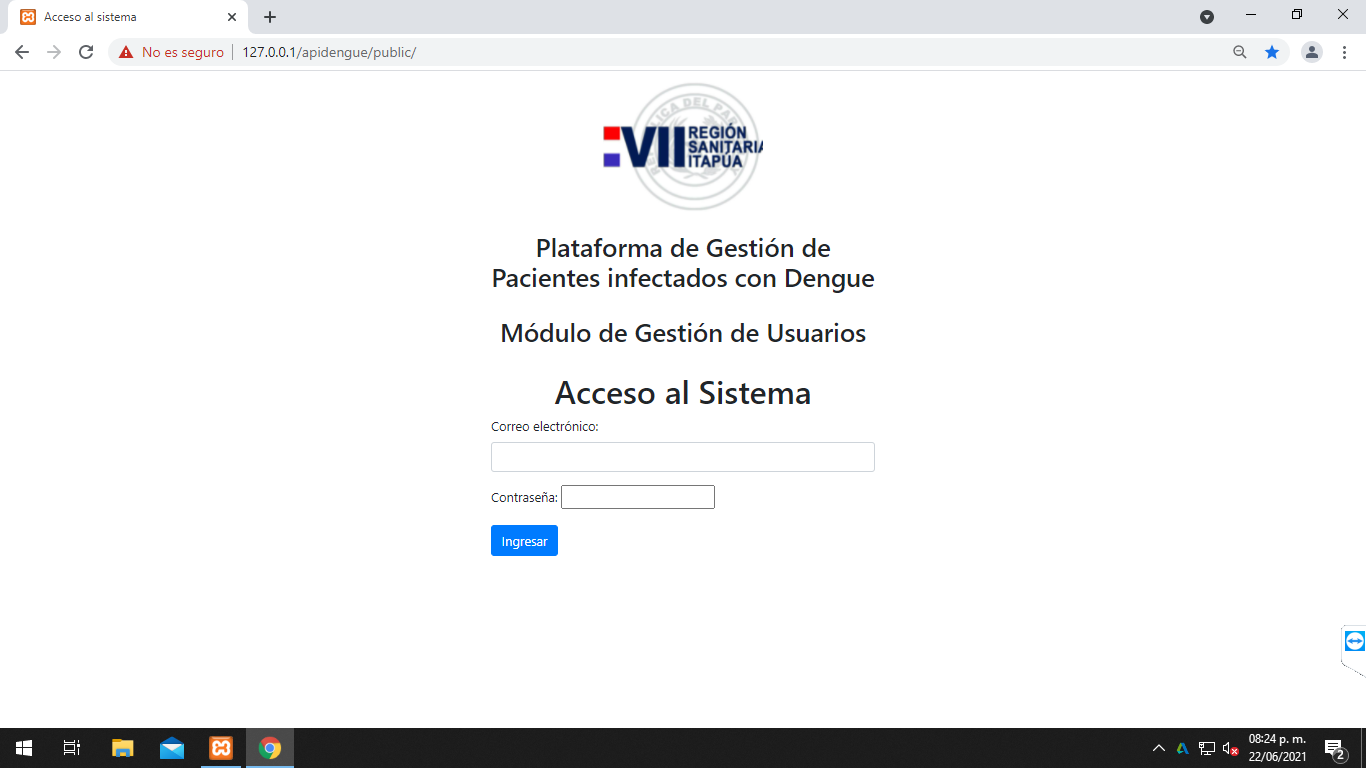
De esta forma se pueden utilizar varias configuraciones para crear rutas y vincular a los métodos de los controladores para tener acceso a los recursos de la API.

5.2.11. Programación del módulo de gestión de usuarios de la API.

El modulo API consta de las funcionalidades para la gestión de usuarios para la plataforma de gestión de pacientes con dengue, así como la gestión de entidades, roles, permisos y denunciantes, por medio de una interfaz de páginas web.

La gestión de usuarios se compone de una página de ingreso por medio de un correo electrónico y una contraseña (Ilustración 27).

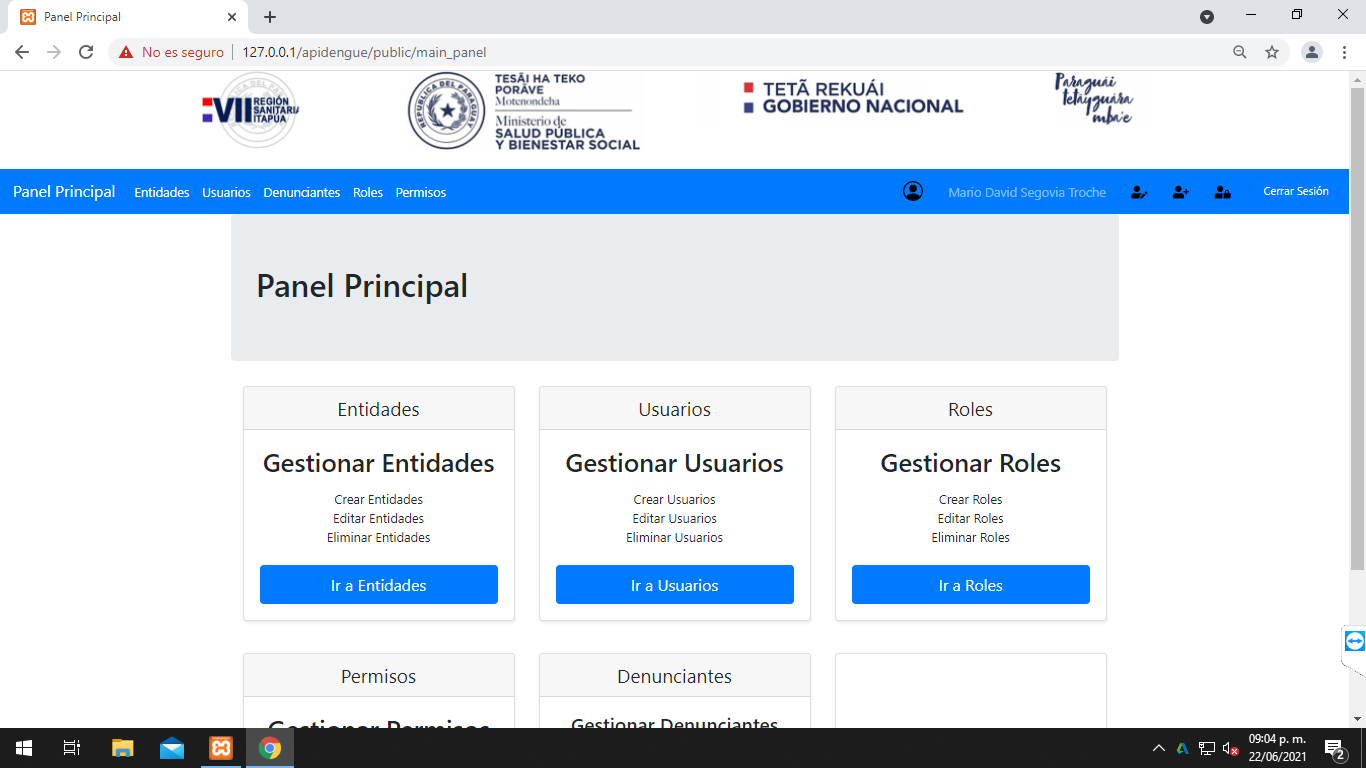
Ilustración 27. Página de ingreso al módulo de gestión de usuarios.



Fuente: Elaboración propia.

Al ingresar al sistema se presenta la página del panel principal, en la cual se visualizan las secciones que integran el modulo, que sirven como accesos directos a cada sección (Ilustración 28).

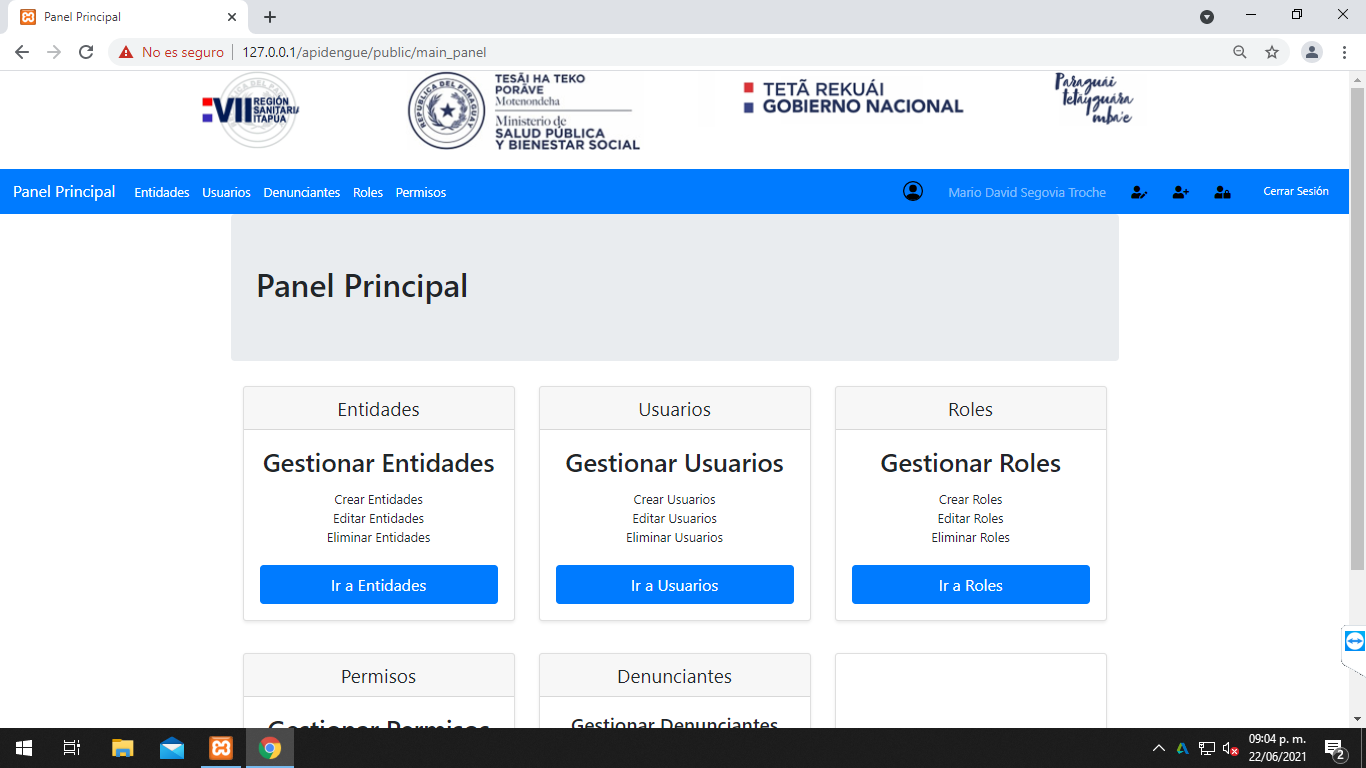
Ilustración 28. Panel principal del módulo de gestión de usuarios.

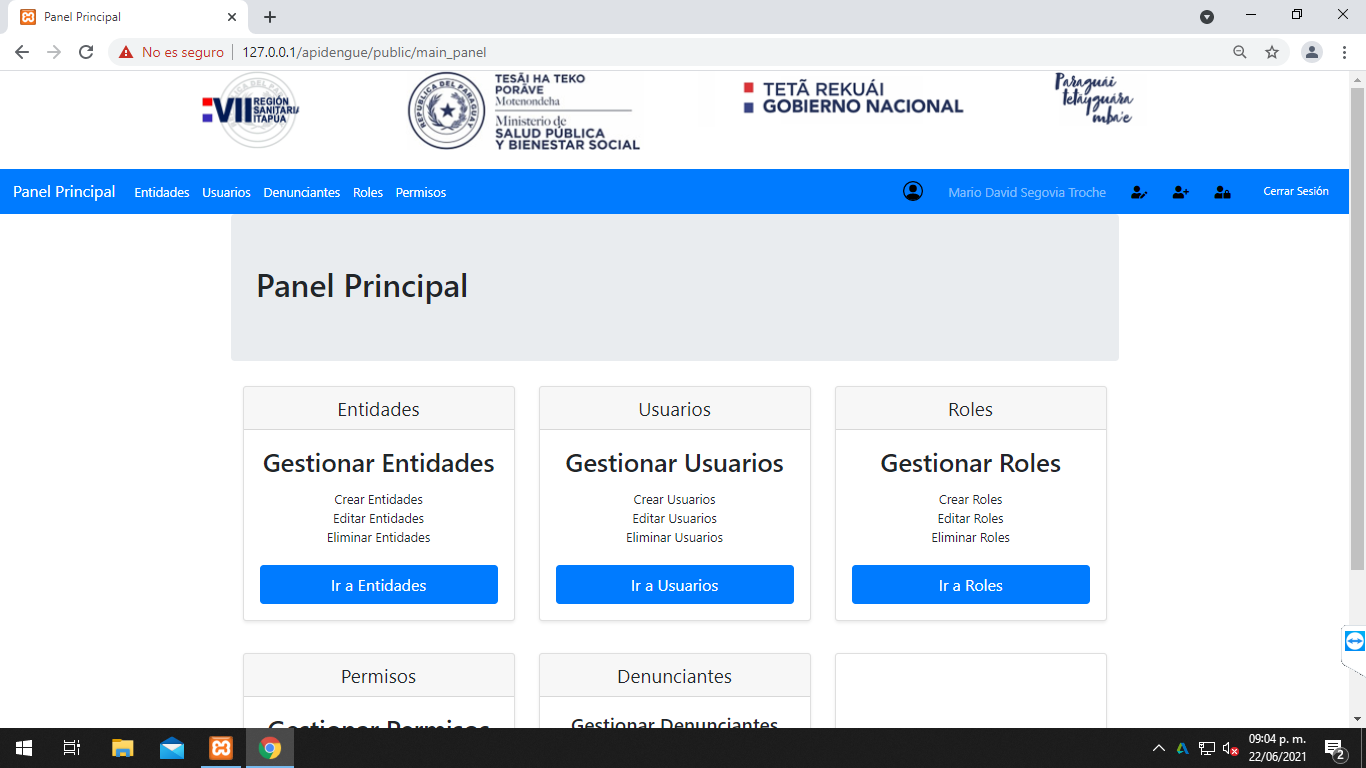


Fuente: Elaboración propia.

En esta vista se agrega también una barra de navegación, que es común a todas las vistas y también ofrece acceso a todas las secciones del módulo, incluyendo información sobre el usuario activo, como el nombre y opciones para ver datos del usuario, editar datos, cambiar contraseña y cerrar sesión (Ilustración 28).

Ilustración 29. Barra de navegación del módulo de gestión de usuarios.

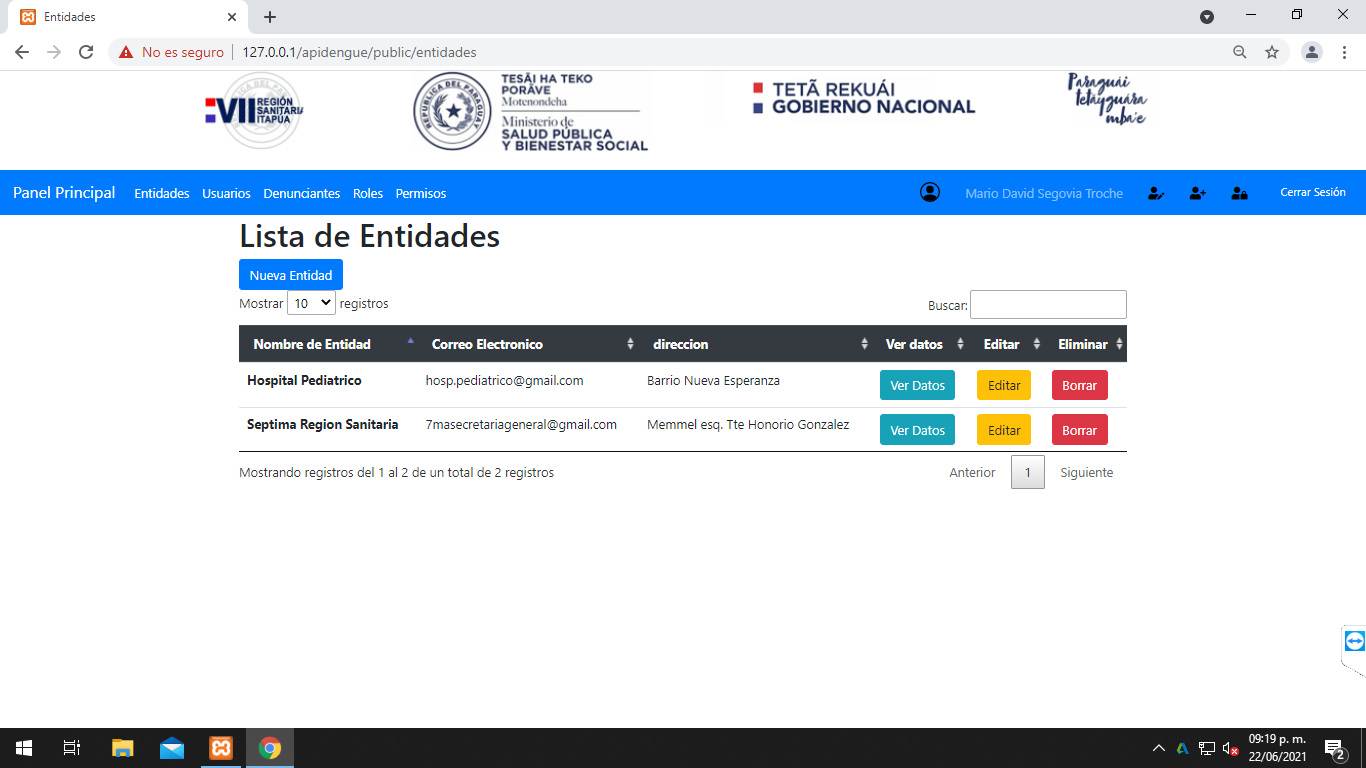


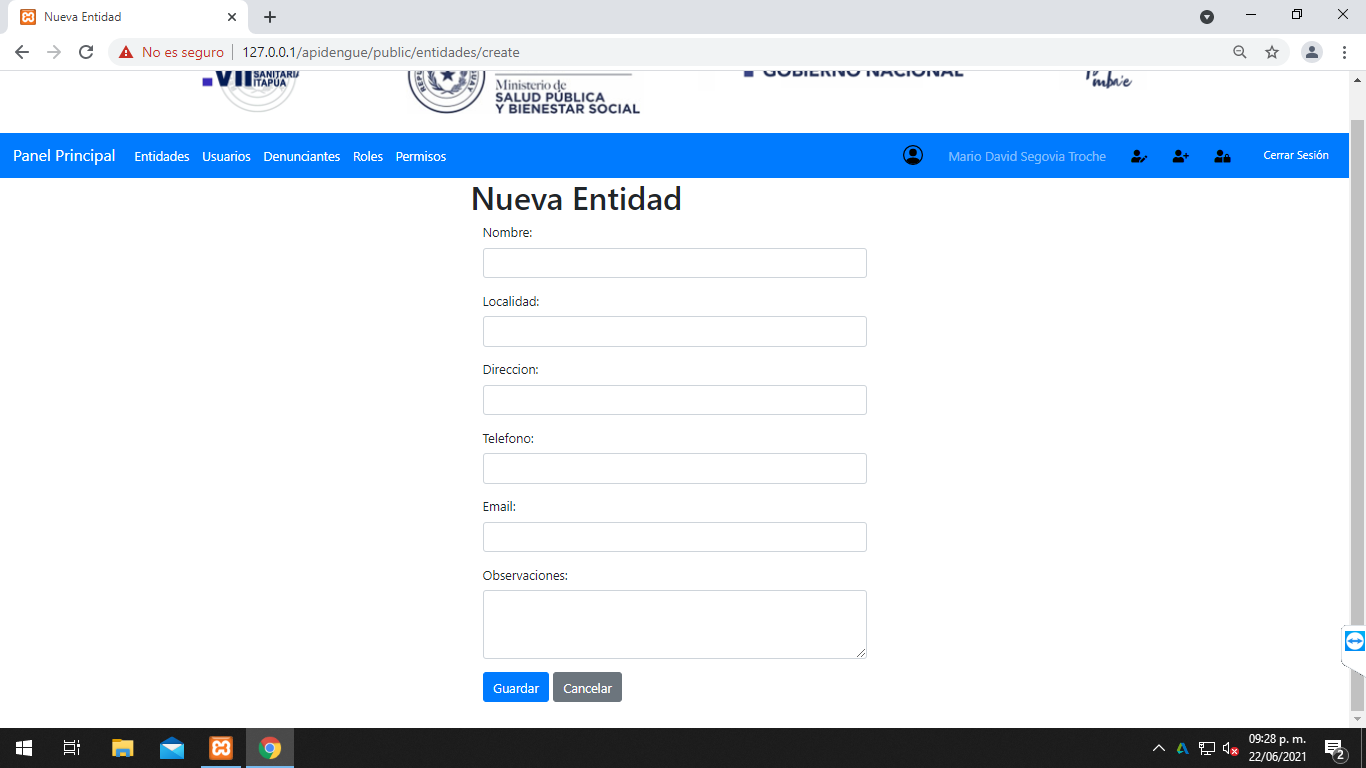


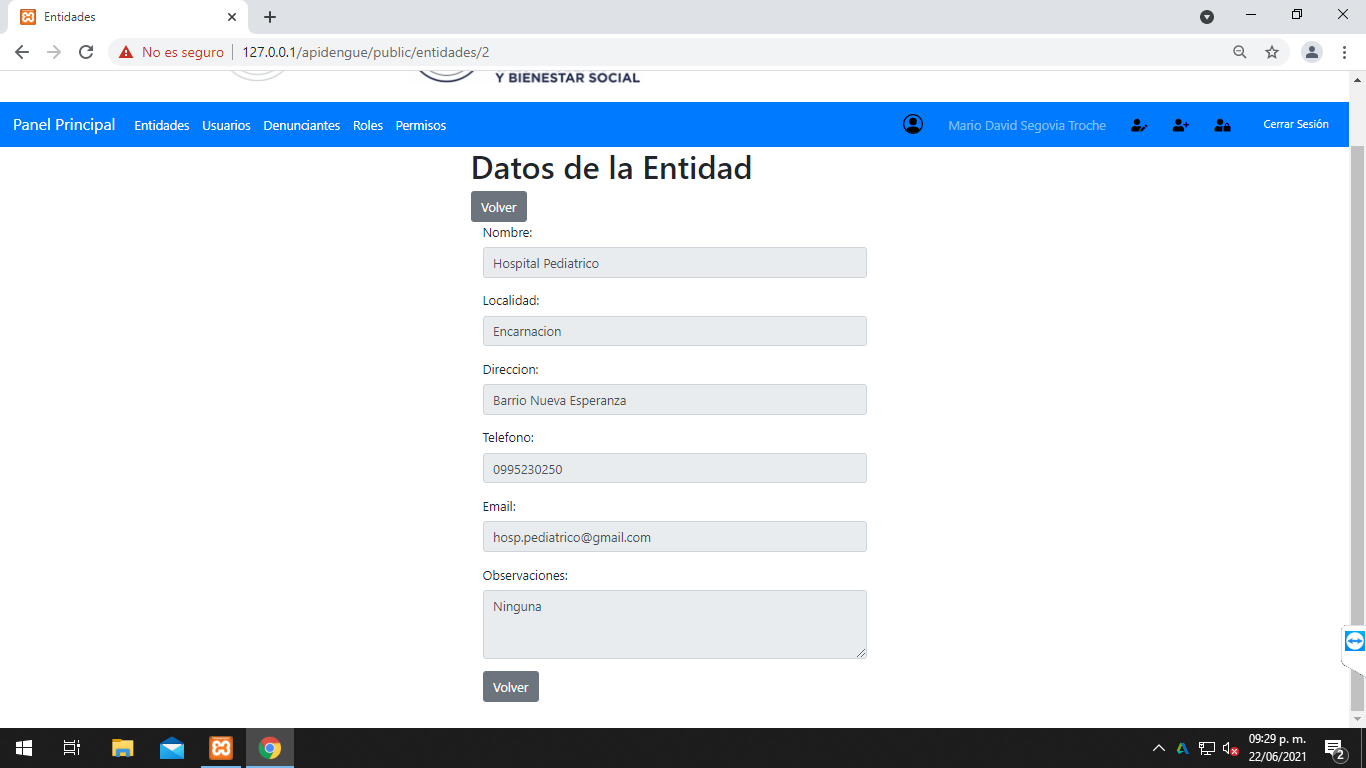
Fuente: Elaboración propia.

La sección de Entidades tiene la finalidad de registrarlos entes sanitarios que estarán relacionados con los usuarios que utilicen la plataforma, como pueden ser hospitales, laboratorios de análisis clínicos, centros de salud, unidades de salud familiar, sanatorios, entre otros. Esta sección se compone de las vistas para la lista de entidades donde se visualizan todos los registros de las entidades existentes, y ofrece las opciones para crear una nueva entidad, ver los datos completos, editar o borrar una entidad (Ilustración 30)..

Ilustración 30. Vistas de lista de entidades, nueva entidad y datos de entidad.



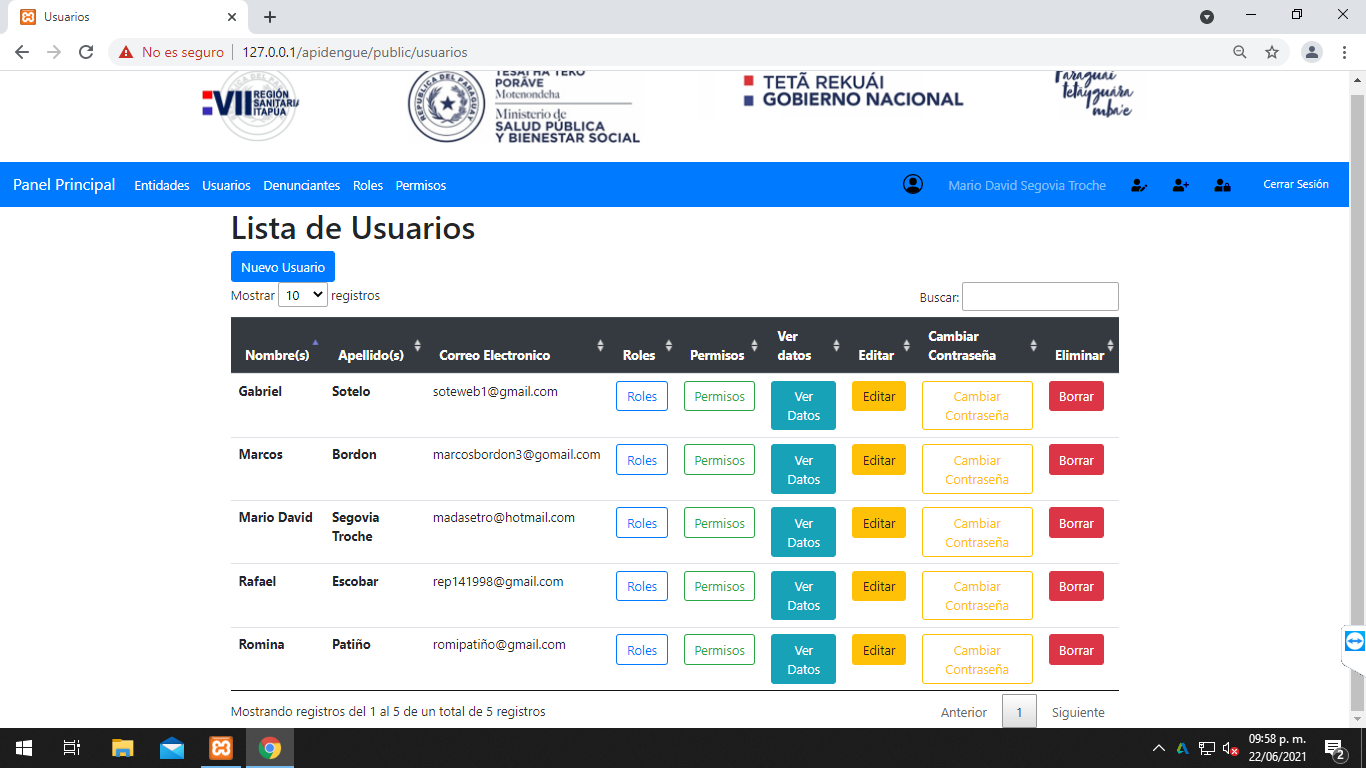


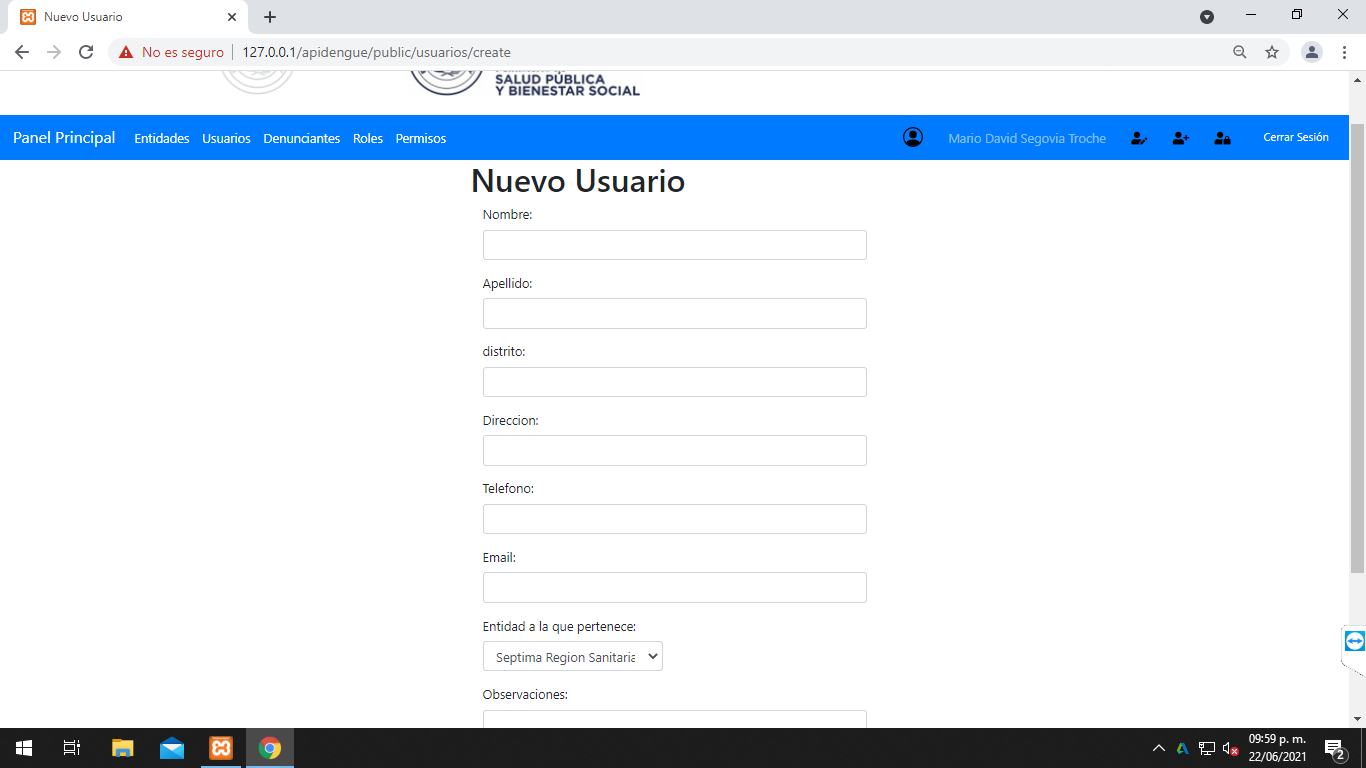


Fuente: Elaboración propia.

La sección de usuarios está destinado para gestionar a las personas que utilizarán los recursos de la plataforma, se puede visualizar la lista de usuarios, crear un nuevo usuario (Ilustración 31), ver o editar los datos de los usuarios (Ilustración 32), cambiar la contraseña de acceso y agregar o quitar los roles o permisos a los usuarios para el control de acceso a los recursos del sistema (Ilustración 33).

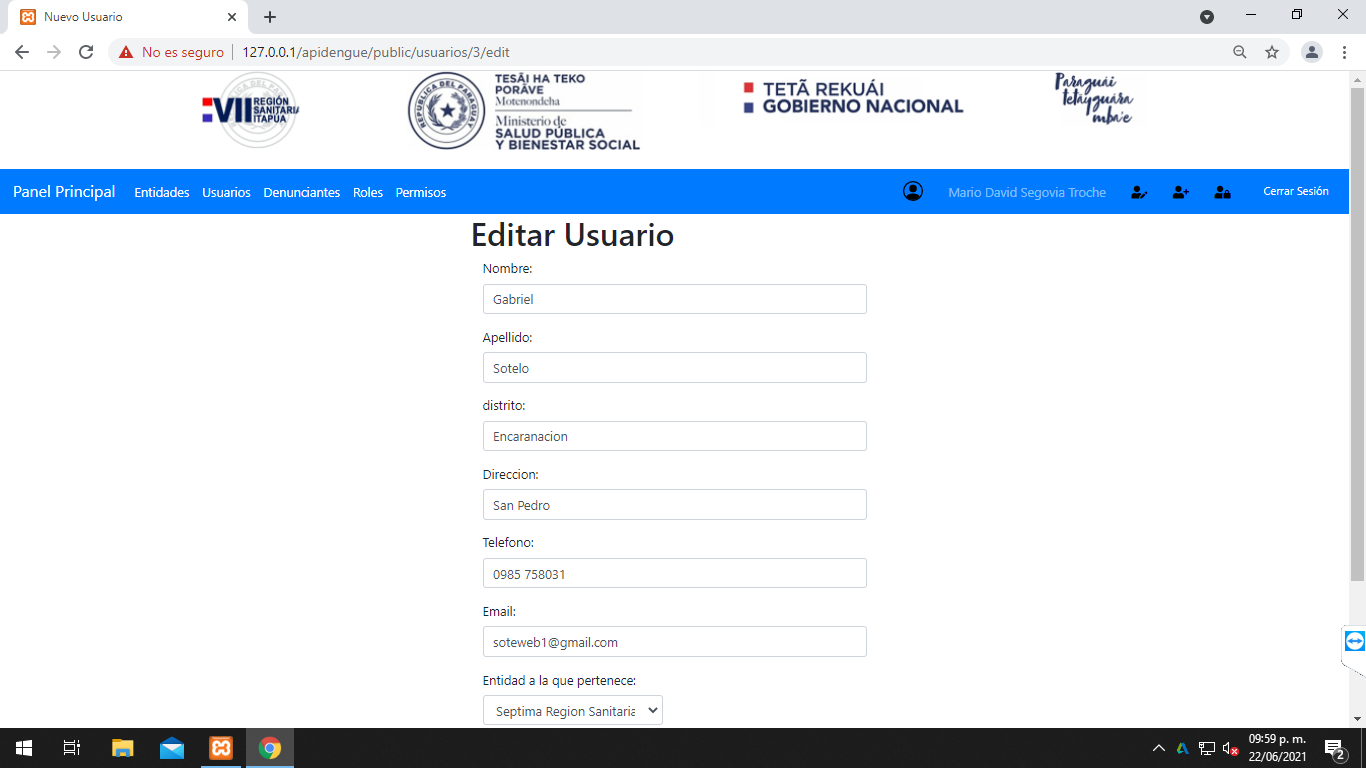
Ilustración 31 Vistas de lista de usuario y nuevo usuario..





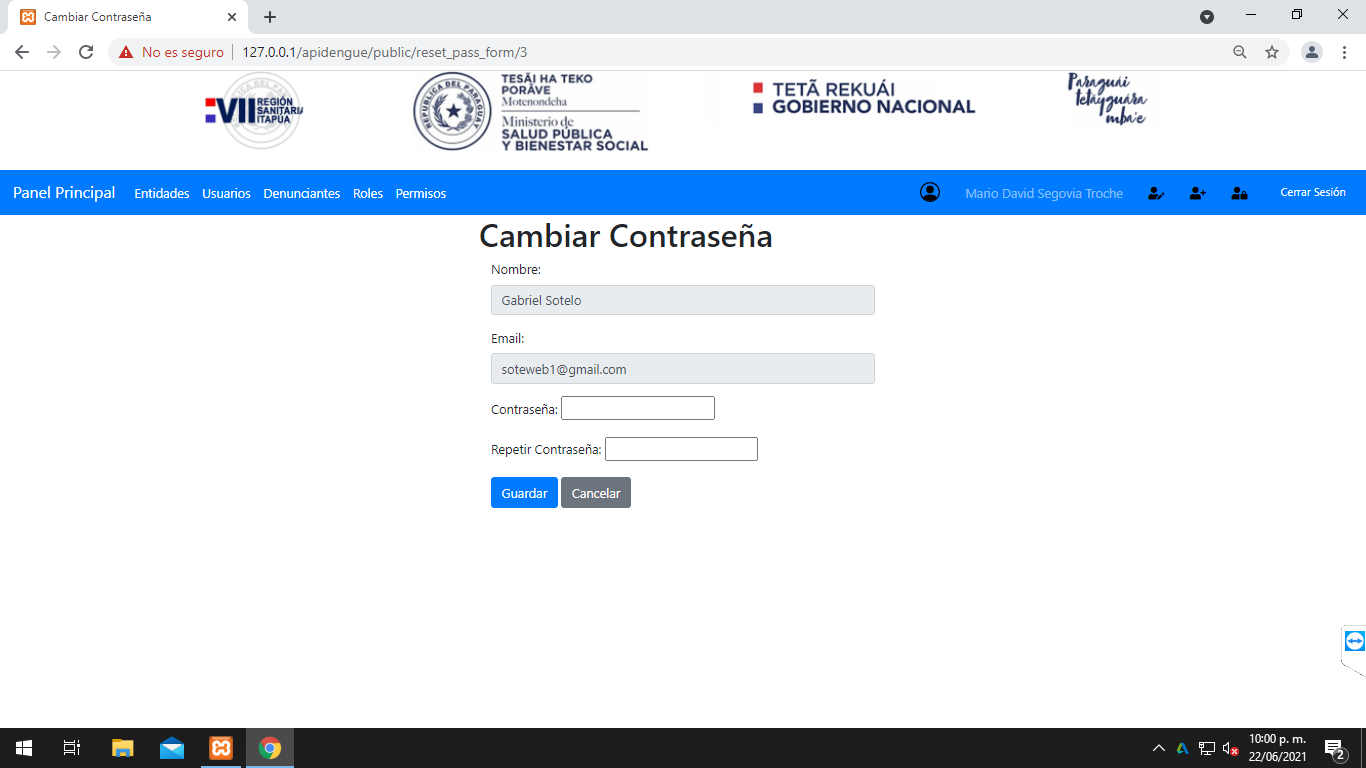
Fuente: Elaboración propia.

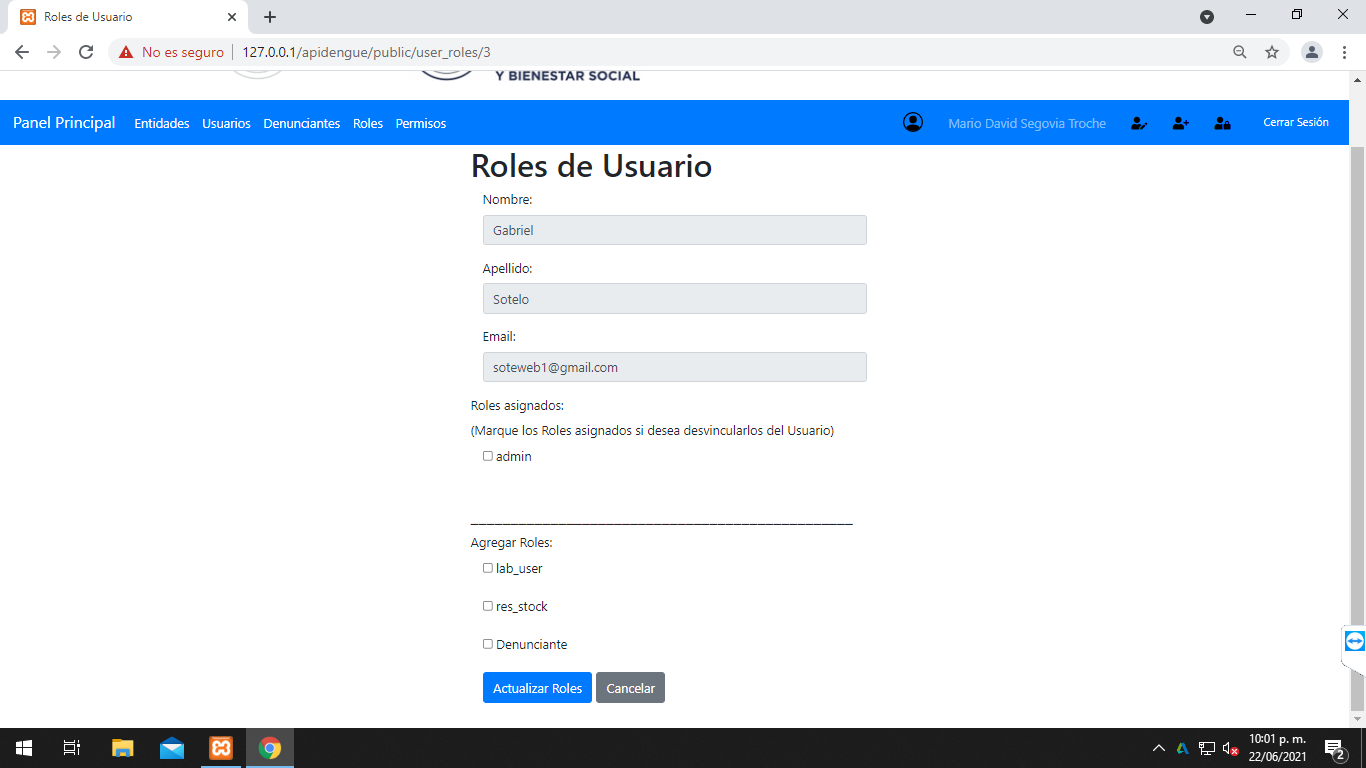
Ilustración 32. Vistas de datos de usuario y editar usuario.

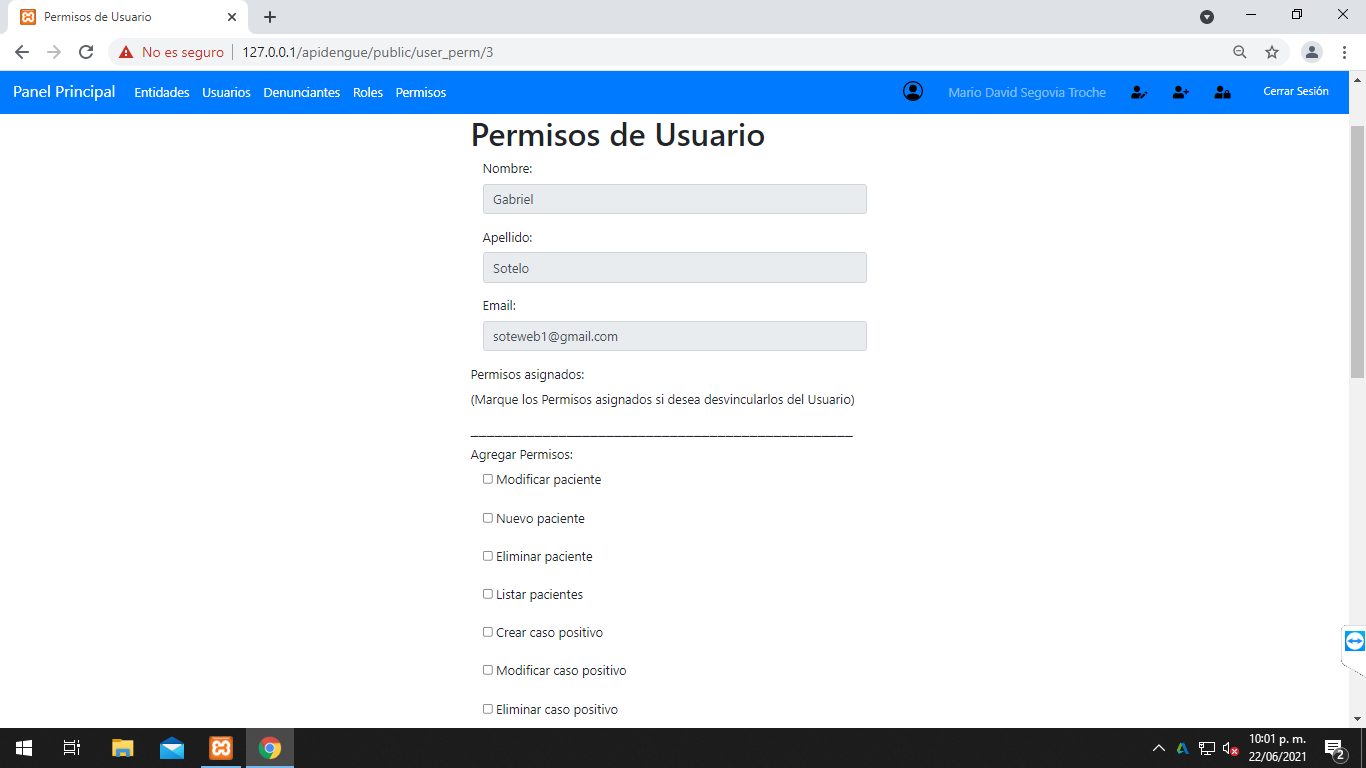


Fuente: Elaboración propia.

Ilustración 33. Vistas para cambiar contraseña, roles y permisos de usuario



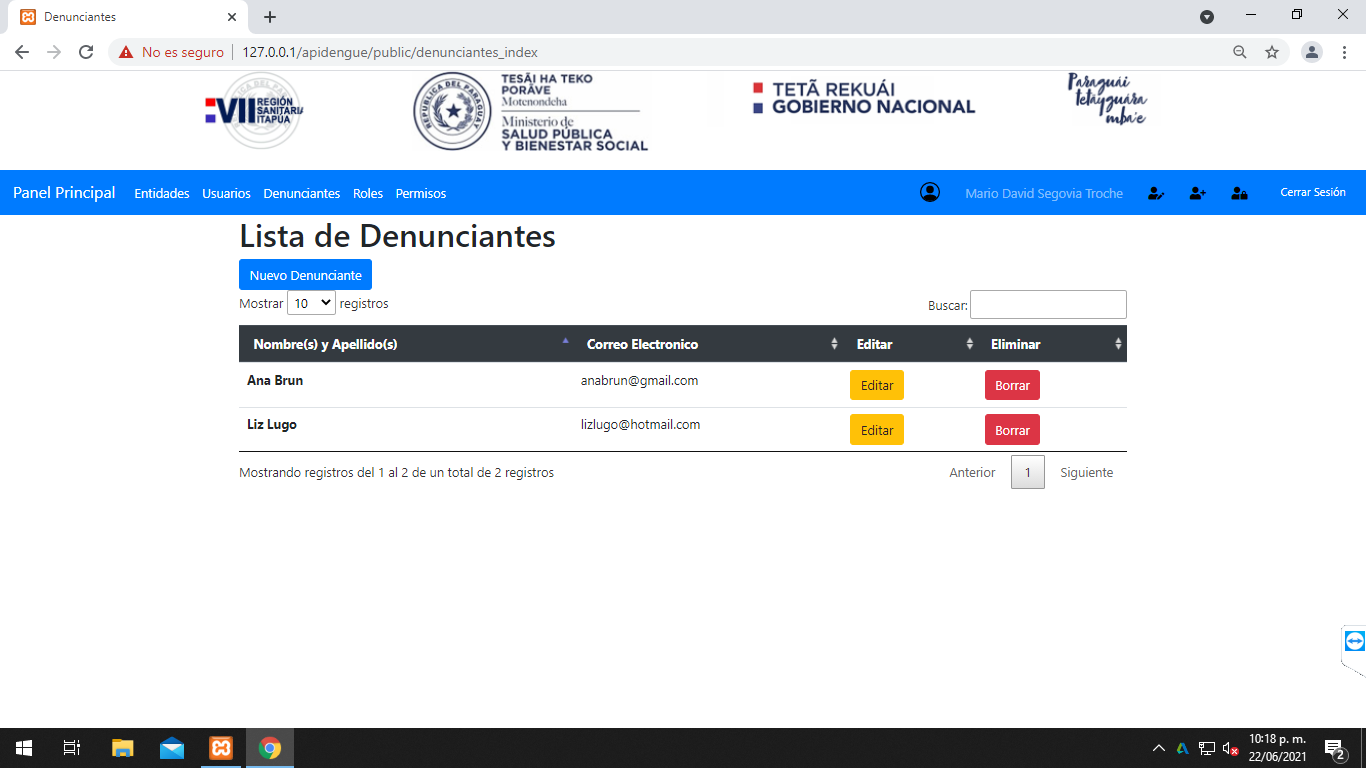


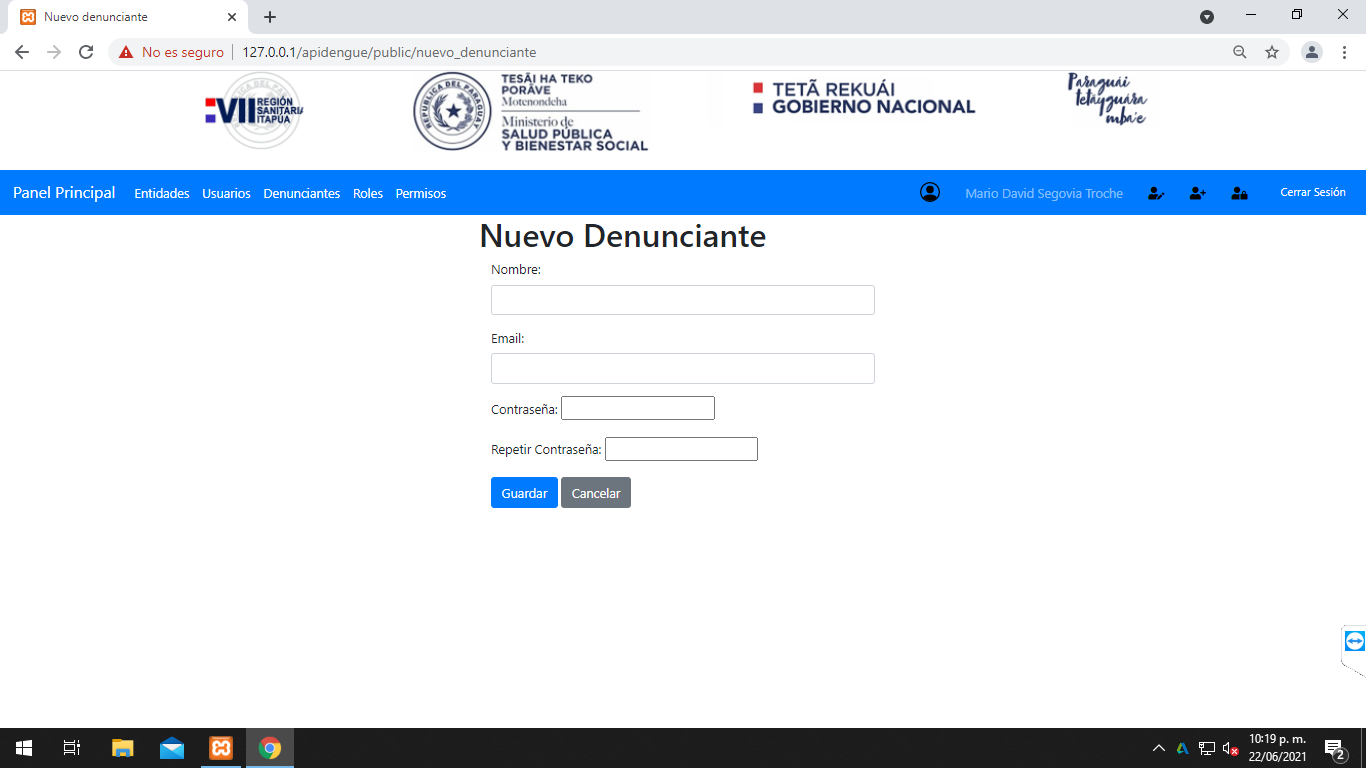


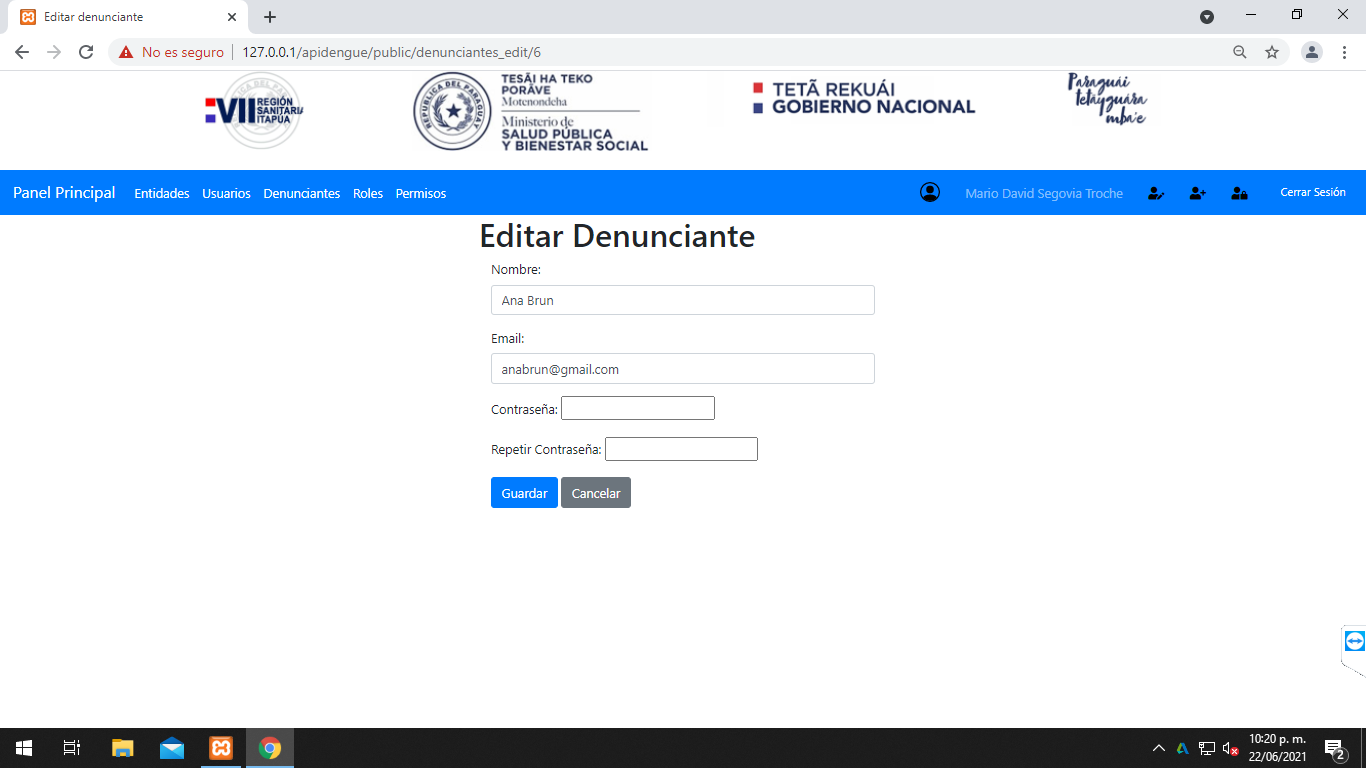
Fuente: Elaboración propia.

La sección de Denunciantes gestiona a los usuarios de las páginas web y aplicación para móviles de acceso público, en la cual la ciudadanía puede realizar sus denuncias sobre situaciones relacionadas a la enfermedad del Dengue. Esta sección consta de una lista de los denunciantes, crear un nuevo registro de un denunciante, editar o eliminar un registro (Ilustración 34).

Ilustración 34. Vistas de lista de denunciantes, crear y editar un registro.



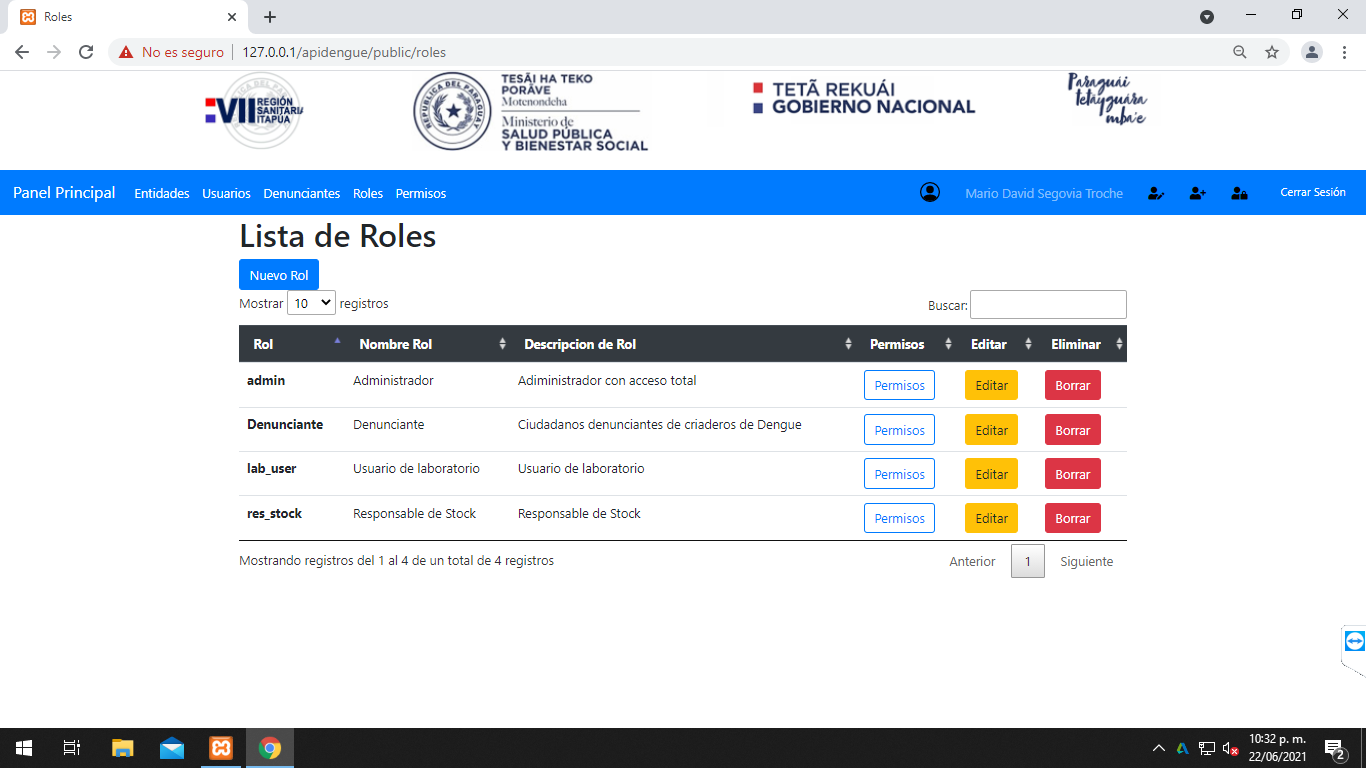


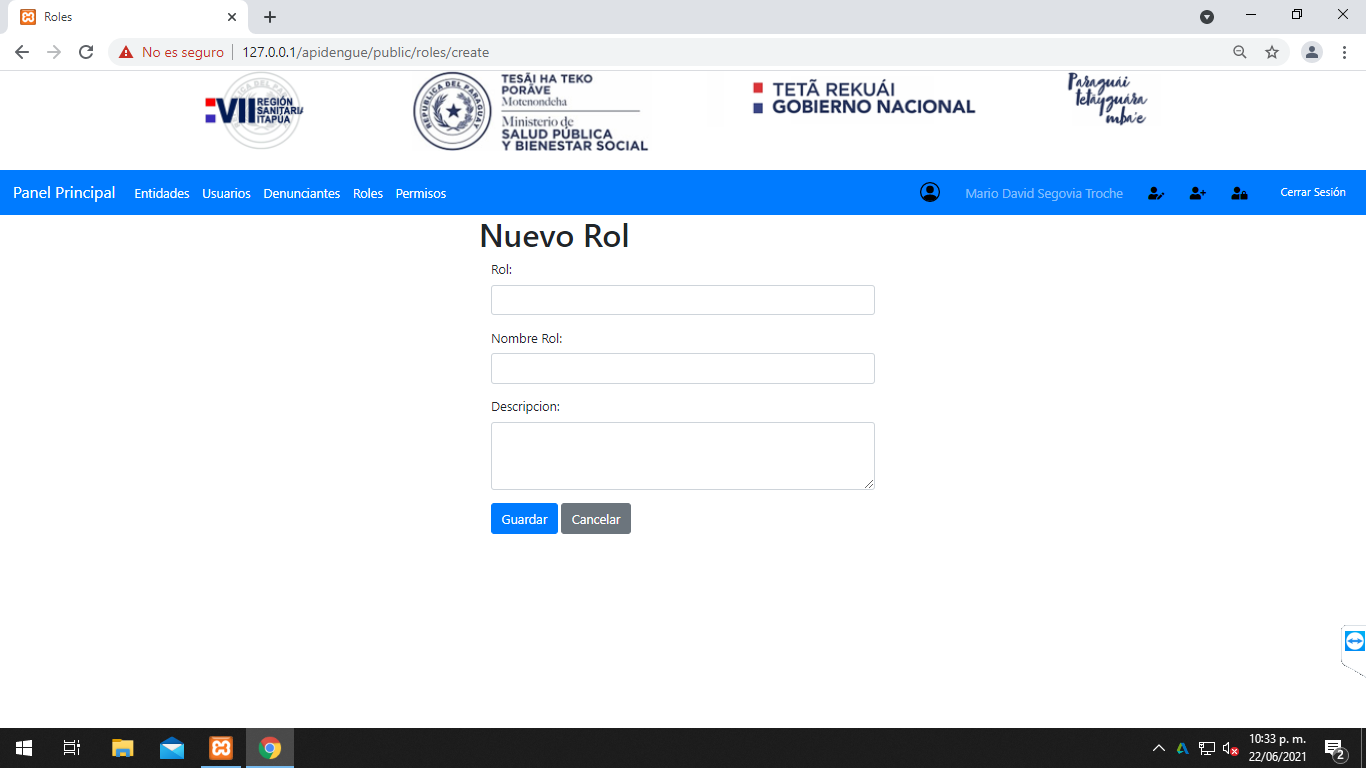


Fuente: Elaboración propia.

La sección de roles permite el control de acceso a los usuarios a los diferentes módulos que componen la plataforma. Dependiendo de los roles asignados a cada usuario podrán acceder a las funcionalidades de la plataforma o tendrán denegado el acceso. Se pueden ver la lista de roles, crear un nuevo rol (Ilustración 35), editar o borrar un rol y agregar o quitar permisos (Ilustración 36).

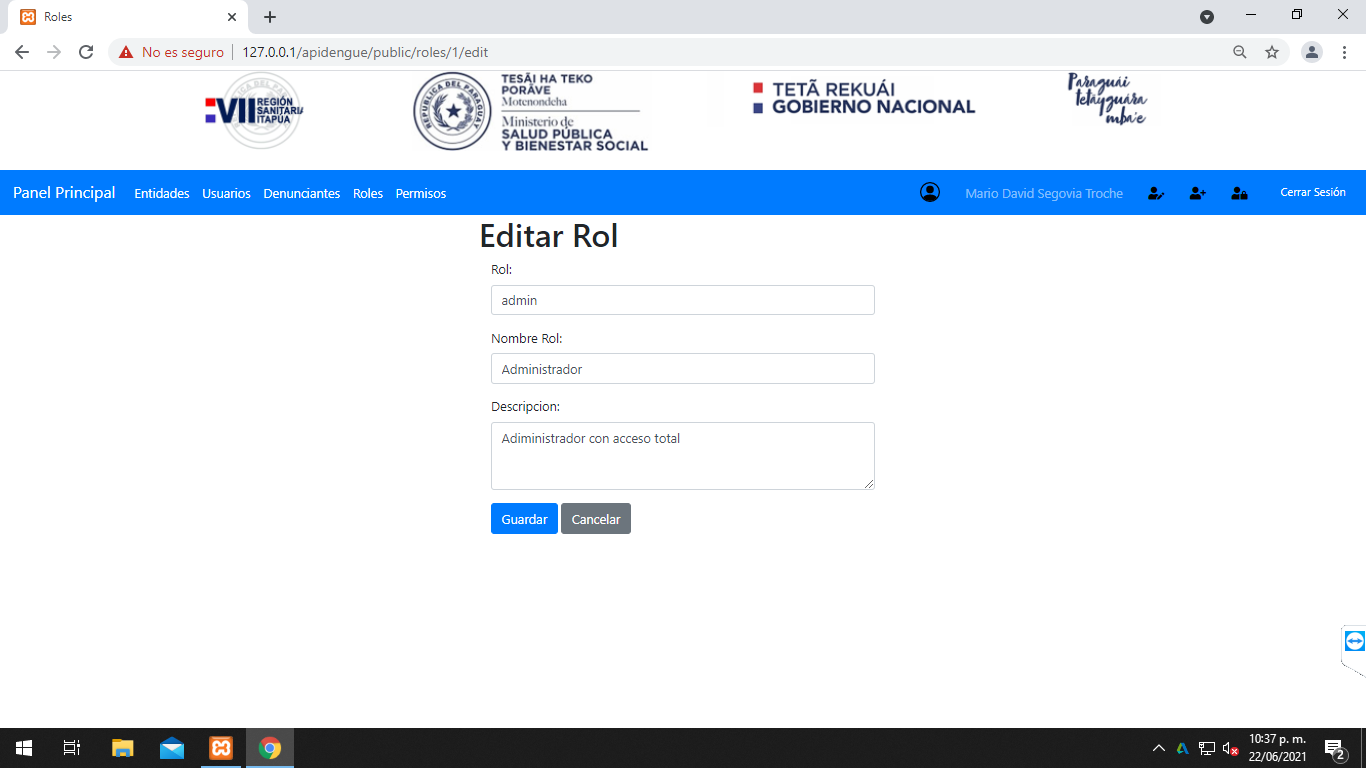
Ilustración 35. Vista de la lista de roles y crear un nuevo rol.

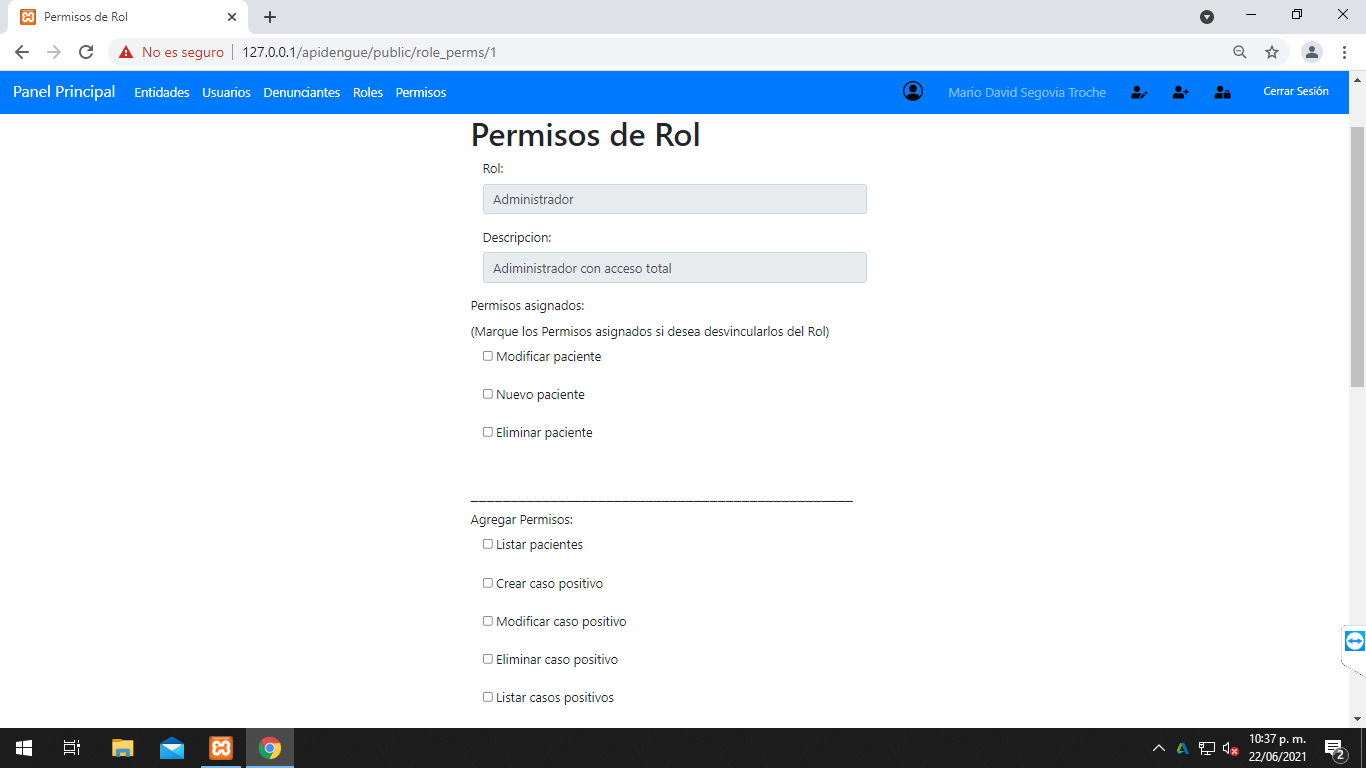




Fuente: Elaboración propia.

Ilustración 36. Vistas para editar rol y agregar o quitar permisos a los roles.

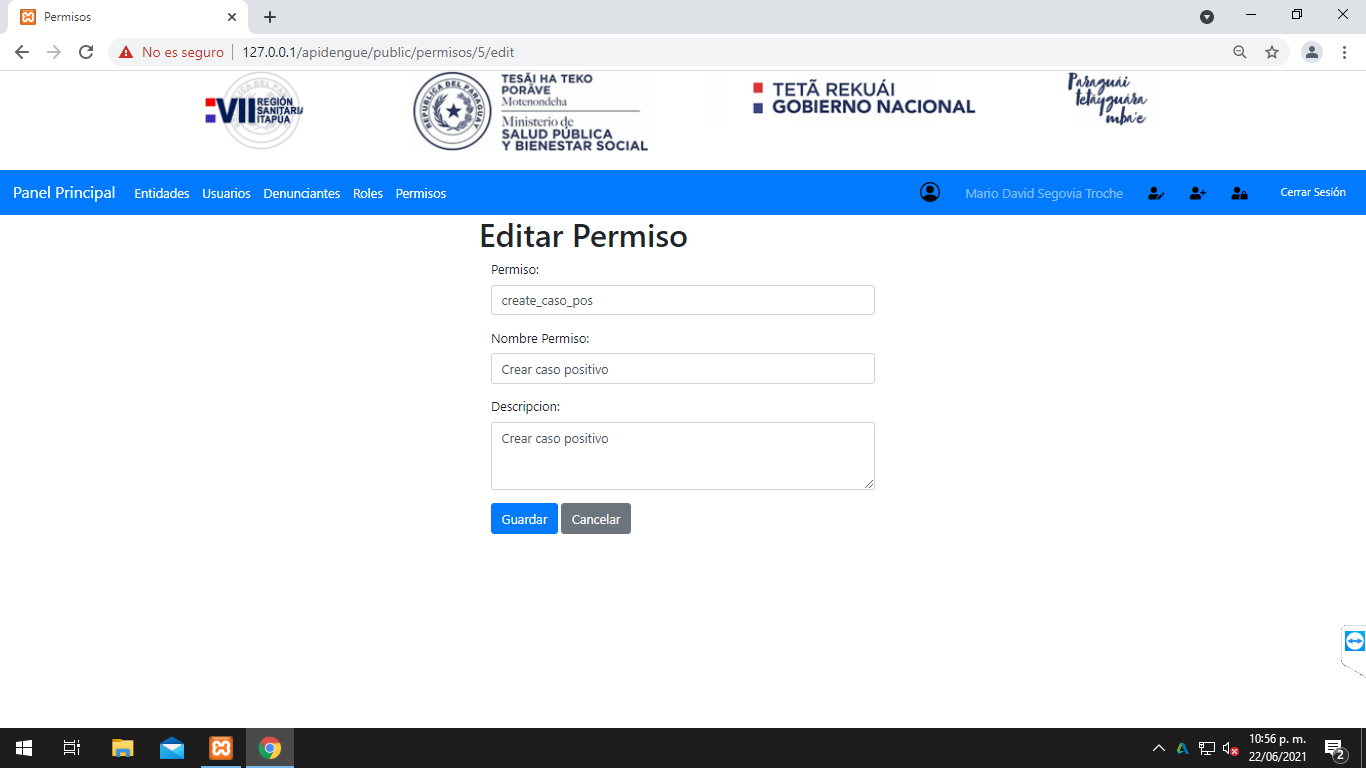
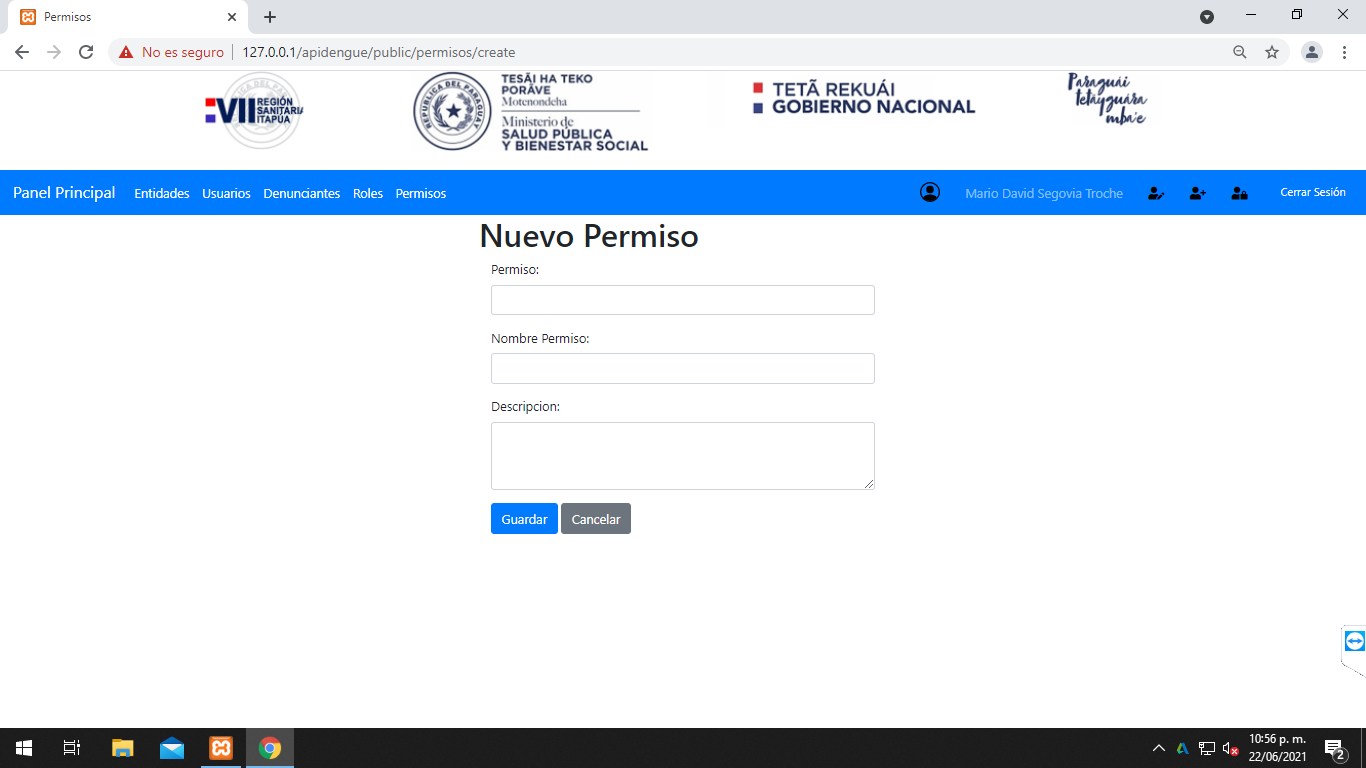
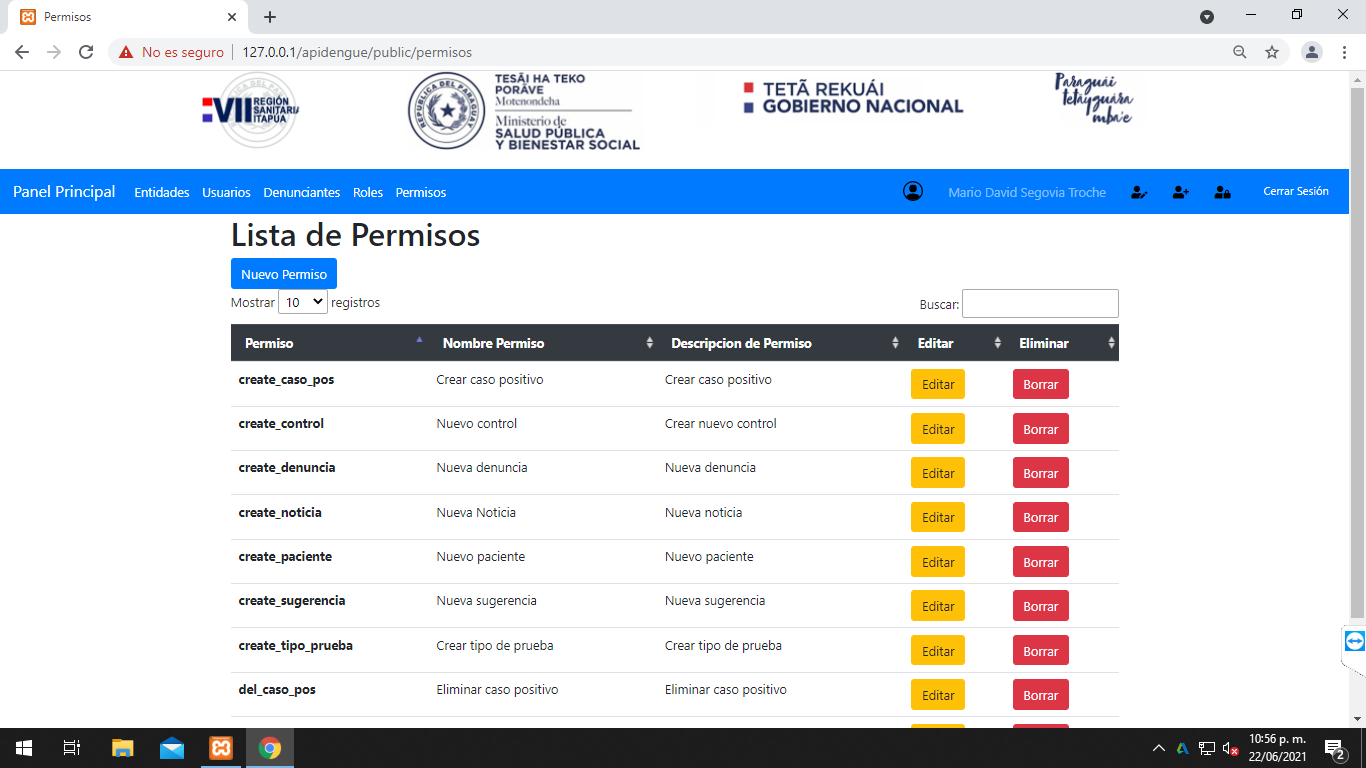




Fuente: Elaboración propia.

La sección de permisos permite agregar o denegar acciones más específicas dentro de la plataforma, tanto para los usuarios como para los roles, estableciendo un control de acceso más detallado y más individualizado. Están disponibles las vistas de lista de permisos, crear nuevo permiso, editar permiso y la opción de borrar permisos (Ilustración 37).

Ilustración 37. Vistas de lista de permisos, crear y editar permisos.



Fuente: Elaboración propia.

5.2.12. Programación de funcionalidades de los módulos de la plataforma.

La programación de las funcionalidades de los módulos que componen la plataforma se estableció según los requerimientos y necesidades en el product backlog del proyecto. Por cada módulo se realizó el análisis correspondiente de los componentes necesarios para lograr el funcionamiento deseado, como la creación y codificación de los modelos, controladores, rutas y otras configuraciones necesarias.

A continuación se realiza una breve descripción de la función de cada módulo de la plataforma, así como los modelos, teniendo en cuenta que por cada modelo se incluyen la configuración del controlador, la migración a la base de datos, rutas de acceso y otras características relacionadas:

- Modulo de registro de pacientes infectados con dengue:

Función:

Registro de pacientes infectados con el virus del Dengue, a través de una interfaz de páginas web destinado a los distintos establecimientos sanitarios como ser laboratorios de análisis clínicos, hospitales regionales, sanatorios, entre otros. También los responsables de la Séptima Región Sanitaria podrán visualizar los datos cargados y poder obtener reportes

Modelos creados: Paciente, Caso\_positivo, Control, TipoPrueba

Programación de características adicionales: controladores y rutas adicionales para el acceso a datos específicos de la base de datos.

- Modulo portal web de denuncias ciudadanas:

Función:

Página web destinado al registro de denuncias ciudadanas sobre posibles fuentes de propagación del vector de la enfermedad del Dengue.

Modelos creados: DenunciaWeb, Asignacion, Soporte, Asignacion.

- Modulo Aplicación móvil de denuncias ciudadanas:

Función:

Aplicación móvil destinada a teléfonos inteligentes con sistema operativo Android, para la realización de denuncias de criaderos del mosquito del Dengue y otros posibles focos de propagación del vector de la enfermedad, por parte de ciudadanos

Modelos creados: Denuncia, Sugerencia.

- Modulo Portal de Información sobre el Dengue:

Función:

Portal web de noticias referentes a la enfermedad del Dengue.

Modelos creados: Noticia

Programación de características adicionales: se crearon controladores y rutas para el acceso a datos específicos.

- Modulo de Control de Stock de insumos destinados al Dengue:

Función:

Control de stock e inventario de insumos destinados a la enfermedad del Dengue de las distintas dependencias de la Séptima Región Sanitaria de Itapúa

Modelos creados: Stock, Asset, Transaction.

- Módulo de Mapas georreferenciados y de calor de casos de Dengue:

Función:

Representación de casos de pacientes infectados con Dengue en mapas digitales en una interfaz web, ofreciendo distintos tipos de visualización a partir de opciones de filtrado de datos. Destinado a los responsables de la Séptima Región Sanitaria de Itapúa.

Programación de características específicas: Se programaron controladores y rutas para el acceso a datos específicos.

- Módulo de Reportes Gráficos:

Función:

Representar en distintos tipos de gráficos los datos almacenados en la plataforma

Programación de características específicas: controladores y rutas para el acceso a datos específicos.

- Modulo de Mapas digitales de casos de dengue destinado a la ciudadanía

Función:

Portal destinado a la población en general para informar sobre puntos de brotes de la enfermedad del Dengue en la región.

Programación de características específicas: controladores y rutas para el acceso a datos específicos.

5.2.13. Programación de características de seguridad del módulo API.

La seguridad de las rutas del módulo API están protegidas por medio de Tokens de seguridad. Los usuarios deben autenticarse con un usuario y contraseña, con la cual reciben un token que utilizarán para identificarse en cada petición que realicen a la API. Para este proyecto se utilizó la librería Passport, que provee de las características necesarias para crear y gestionar los tokens y proteger las rutas. El comando para la instalación de la librería se utiliza el siguiente comando:

- composer require laravel/passport

Una vez instalado las librerías se realiza la migración de las tablas necesarias en la base de datos para el registro de los tokens. Se utiliza el comando de migración de laravel:

- php artisan migrate

El siguiente paso es generar las claves de encriptación de los tokens con el siguiente comando:

- php artisan passport:install

Las siguientes configuraciones se realizan para establecer a esta librería como la encargada de proteger las rutas de la API, y finalmente se agrega el comando “middleware(‘auth:api’)” para designar las rutas a ser protegidas por esta funcionalidad (Ilustración 38).

Ilustración 38. Ruta protegida por token de seguridad.



Fuente: Elaboración propia.

5.2.14. Programación de características de control de acceso del módulo API.

El control de acceso a las funcionalidades de la API por parte de los usuarios de la plataforma se administra por medio de la librería Laratrust instalada en el módulo. Esta librería ofrece todas las opciones y características necesarias para permitir la gestión de roles y permisos de los usuarios.

La instalación de la librería se realiza con el siguiente comando:

- composer require santigarcor/laratrust

A continuación se debe establecer el archivo de configuración con el siguiente comando:

- php artisan vendor:publish --tag="laratrust"

Seguidamente se instala todas las configuraciones de la librería, se crean los modelos de Roles y Permisos, y las migraciones necesarias para crear las tablas en la base de datos. Se ejecuta el siguiente comando:

- php artisan laratrust:setup

Continuamos la configuración actualizando las clases de nuestro proyecto con este comando:

- composer dump-autoload

Finalmente terminamos con la migración a la base de datos de las tablas necesarias:

- php artisan migrate

De esta forma ya podemos utilizar las características de esta librería para el control de acceso de los usuarios. El nivel de acceso se puede configurar utilizando los roles o permisos, que se pueden utilizar de la siguiente forma:

- Roles: se puede utilizar esta característica para otorgar acceso de una forma más general, como acceso a todas las funcionalidades de uno o más módulos.

- Permisos: utilizado para otorgar acceso más específico a alguna funcionalidad de un módulo, posibilitando dar permisos más detallados.

# 6. cONCLUSIÓN

# 7. RECOMENDACIONES

# 8. REFERENCIAS BIBLIOGRÁFICAS.

Arbo, Antonio (2019). *Dengue: pesada carga para la salud pública del Paraguay*. http://dx.doi.org/10.18004/imt/20191411-2

Parra, Cristhian (2017). *TopaDengue.PY: tecnologías de la información y la comunicación para la promoción y soporte de iniciativas comunitarias contra arbovirus*. https://cicco.on.worldcat.org/oclc/1147979481

Mello Román, Jorge D., Mello-Román, Julio C., Gómez Guerrero, Santiago, & García Torres, Miguel (2019). *Predictive Models for the Medical Diagnosis of Dengue: A Case Study in Paraguay*. https://doi.org/10.1155/2019/7307803

Pane, Juan, Paciello, Julio, Lisnichuk, Yohanna, Martínez, Héctor, Valdez, Santiago & Durañona, Nelson (2018). *DATOS ABIERTOS Y ALERTAS SOBRE DENGUE*. https://publicaciones.fctunca.edu.py/jspui/handle/123456789/77

Ughelli, V., Lisnichuk, Y., Paciello, J., & Pane, J. (2017). *Prediction of Dengue Cases in Paraguay Using Artificial Neural Networks*. https://csce.ucmss.com/cr/books/2017/LFS/CSREA2017/HIM3277.pdf

Condor Camara, Daniel, Nolasco Cardenas, Oscar Patricio, Carrasco Escobar, Gabriel & Egoavil Ayala, Miguel (2018*). Sistema de Información basado en Tecnologías de Información y Comunicación para geolocalización de Zika, Dengue, Chikungunya y Malaria.*

http://coloquioenfermeria2018.sld.cu/index.php/coloquio/2018/paper/view/440/96

Navarro Cadavid, Andrés, Fernández Martínez, Juan Daniel, & Morales Vélez, Jonathan (2013). *Revisión de metodologías ágiles para el desarrollo de software.* https://www.redalyc.org/pdf/4962/496250736004.pdf

Molina Ríos, J.R., Zea Ordóñez, M.P., Contento Segarra, M.J. & García Zerda, F.G. (2018). *Comparación de metodologías en aplicaciones web*. 3C Tecnología: glosas de innovación aplicadas a la pyme, 7(1).1-19. DOI: <http://dx.doi.org/10.17993/3ctecno.2018.v7n1e25.1-19/>

Revuelta Arribas, Alberto (2020). *Creación de un complemento para Google Apps Script para la gestión de datos UPM.* http://oa.upm.es/58093/1/TFG\_ALBERTO\_REVUELTA\_ARRIBAS.pdf

Marín, Rafael (2019). *Los gestores de bases de datos más usados en la actualidad*. https://revistadigital.inesem.es/informatica-y-tics/los-gestores-de-bases-de-datos-mas-usados

Lage, Ricardo Juan, Herrera Graña, Tahimí, Simpson Johnson, Bertha & Zulueta Torres, Zulema (2015). *Aspectos actualizados sobre dengue*. https://dialnet.unirioja.es/servlet/articulo?codigo=6027633

Rodríguez-Castro, Aldo Ismael, Rolón, Jorge, & Ríos-González, Carlos Miguel. (2019). *Costos de internación del dengue en un hospital de tercer nivel de atención de Paraguay, 2017*. Revista del Instituto de Medicina Tropical, 14(1), 14-20. https://dx.doi.org/10.18004/imt/201914114-20

Flores, Laura, Giménez Caballero, Edgar, Díaz Duba, Sebastián & Torales, Judith. (2015). *Impacto Económico del dengue en el Instituto de Previsión Social: epidemia diciembre 2006 - julio 2007*. Memorias del Instituto de Investigaciones en Ciencias de la Salud, 13(2), 78-85. https://dx.doi.org/10.18004/Mem.iics/1812-9528/2015.013(02)78-085

Cuellar, C. M. D., Lovera, D., Merlo, O., & Arbo, A. (2020*). Impacto económico del dengue en Paraguay*. Revista chilena de infectología, 37(4), 356-361. doi: 10.4067/S0716-10182020000400356. PMID: 33399655.

Parra, C., Rojas, R., Espinoza, G. A., & Coloma, J (2019). *Participación en programas salud pública desde la comunidad: el caso TopaDengue*. https://www.researchgate.net/publication/334279550\_Participation\_in\_Public\_Health\_Programs\_from\_the\_Community\_the\_TopaDengue\_case

Pane Juan, Ojeda Verena & Valdez, Natalia (2015). *Dengue Open Data*. https://idatosabiertos.org/wp-content/uploads/2015/10/7.Dengue-Pane-Ojeda-Valdez.pdf

Díaz, F. J., & Banchoff Tzancoff, C. M. (2000). *PHP: una solución" open source" para el desarrollo de páginas Web dinámicas. In VI Congreso Argentino de Ciencias de la Computación.* <http://sedici.unlp.edu.ar/handle/10915/23700>

Sierra, F., Acosta, J., Ariza, J., & Salas, M. (2013). *Estudio y análisis de los framework en php basados en el modelo vista controlador para el desarrollo de software orientado a la web. Investigación y desarrollo en TIC, 4(2), 14-26*. <http://revistas.unisimon.edu.co/index.php/identic/article/view/2480>

Sala, J. J. R. (2003). *Introducción a la programación. Teoría y práctica (Vol. 3, p. 2)*. <http://virtual.usalesiana.edu.bo/web/conte/archivos/572.pdf>

Ovando Ortega, D. J. (2019). *Bootstrap y Laravel, herramientas para el desarrollo de aplicaciones web*. <http://repositorio.upsin.edu.mx/Fragmentos/tesinas/142016030030OvandoOrtegaDenzelJavier10843.pdf>

Muñoz Razo, C. (2011). *Cómo elaborar y asesorar una investigación de tesis*. Pearson Educación.

# 9. ANEXO