



**UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN**

**FACULTAD DE CIENCIA, ARTE Y TECNOLOGÍA**

**LICENCIATURA EN ANALISIS DE SISTEMAS INFORMATICOS**

**TRABAJO FINAL DE GRADO**

**INTERCONECTIVIDAD REST PARA EL SERVICIO DE TRANSFERENCIA DE  
INFORMACIÓN DE LA SEPTIMA REGION SANITARIA DE ITAPÚA**

**LINEA DE INVESTIGACIÓN:  
INGENIERÍA DE SOFTWARE, TRANSFERENCIA DE INFORMACIÓN**

**ESTUDIANTE:**

**MARIO DAVID SEGOVIA TROCHE**

**Encarnación – Paraguay**

**Agosto – 2021**

# **DICTAMEN DE EVALUACIÓN DEL TRABAJO DE FIN DE GRADO**

Visto lo estipulado en el Manual de Investigación de la Universidad Autónoma de Encarnación (UNAE) referente a los procesos de evaluación, tras la evaluación de la 1<sup>a</sup> instancia (Orientador), la validación de la 2<sup>a</sup> instancia (decanato y CIDUNAE), la evaluación de la 3<sup>a</sup> instancia (Evaluador) y la presentación oral, se resuelve que el Trabajo de Fin de Grado presentado el \_\_\_\_/08/2021 en Encarnación (ciudad) y titulado “INTERCONECTIVIDAD REST PARA EL SERVICIO DE TRANSFERENCIA DE INFORMACIÓN DE LA SEPTIMA REGION SANITARIA DE ITAPÚA” realizado por Mario David Segovia Troche resulta aprobado con calificación final \_\_\_\_\_ (número y letra).

## Descargo fundamentado:

### Firmas, aclaraciones y CI miembros de la mesa

1.  2.

Sello Unidad Académica

# INTERCONECTIVIDAD REST PARA EL SERVICIO DE TRANSFERENCIA DE INFORMACIÓN DE LA SEPTIMA REGION SANITARIA DE ITAPÚA

Autor: Mario David Segovia Troche

Línea de Investigación: Ingeniería del Software y Transferencia de Información.

## Resumen

El Dengue es actualmente una de las enfermedades que más impacto genera en la salud de la población y en los sistemas sanitarios de vastas regiones del planeta, abarcando una gran cantidad de países entre las cuales se encuentra el Paraguay. Con la finalidad de erradicar y controlar la enfermedad, se aplicaron varias estrategias a través de proyectos, planes, investigaciones, entre otras iniciativas involucrando distintas áreas del conocimiento y las ciencias. De las diferentes iniciativas realizadas, existen varios ejemplos en las que se utilizaron recursos de las Tecnologías de la Información y Comunicación (TIC), como también las ciencias de la informática y la computación, que dieron resultados exitosos y beneficiosos. A partir de estos hechos antes mencionados, surge la propuesta de una plataforma web para el registro de personas infectadas con el virus del Dengue, con la finalidad de ser una herramienta tecnológica de utilidad para la lucha contra la enfermedad, destinada a los responsables de la Séptima Región Sanitaria del Departamento de Itapúa, Paraguay. La plataforma web propuesta se modela en una estructura modular, en la cual cada módulo desarrollado tiene una serie de funcionalidades específicas que trabajan en conjunto como un sistema unitario. En este material se describe el desarrollo del módulo de interconectividad API REST, el cual cumple la función de ser el núcleo de toda la plataforma web para el intercambio y almacenamiento de datos generados. Para el logro de los objetivos propuestos, se utilizó una metodología de investigación aplicada, realizando una recolección de información y obtener conocimiento con el fin de obtener un resultado que beneficie a la sociedad con una solución práctica. Para el desarrollo del módulo de interconectividad API REST, se optó por una metodología ágil SCRUM, por sus características que ofrecen una gran flexibilidad y las herramientas necesarias para una planificación y ejecución exitosa de las actividades para llegar al producto final. A partir de la selección de los recursos tecnológicos como el lenguaje de programación, entorno de desarrollo, sistemas de gestión de bases de datos, entre otros, teniendo en cuenta su calidad de software libre, se programaron y configuraron todas las funcionalidades requeridas. Para la verificación del correcto funcionamiento y el comportamiento esperado de las funcionalidades del módulo, se aplicaron pruebas de forma constante durante todo el proceso de desarrollo, de acuerdo a los lineamientos establecidos por la metodología SCRUM. De esta forma se logró completar el módulo de interconectividad API REST, con la integración exitosa a la plataforma web, con la cual se pretende ofrecer una solución para la aplicación de estrategias en el combate y control del Dengue que afecta a la población y a las instituciones sanitarias.

Palabras claves: Enfermedad tropical, Transferencia de información, Tecnología de la información

INTERCONECTIVIDAD REST MOMARANDU ÑEMBOHASA TEKUAIVORE GUARÃ  
SEPTIMA REGIÓN SANITARIA ITAPÚAGUI

Apohära: Mario David Segovia Troche

Tembikuaareka rape: Ingeniería del Software y Transferencia de Información.

ÑEMOMBYKY

Dengue ko'ágá ha'e peteì mba'asy ojapova heta apañuái tesáipe oikova ko yvy ári ha tesái motenondeháva, oparupi retà ko mba'asy oikatuva ojesururu, ko retà atyra oì avei Paraguai. Oipe'a guarà ko mba'asy ha anive ojapo vaí yvypórape, heta tembiapo ojejapo oipurúva opiachagua arandu ha kua'apy. Ko tembiapo apytépe heta oì oipuruva Tecnologías de la Información y Comunicación (TIC), informática ha computación kua'apy, osè poràva. Upe haguere, ojejapo ko plataforma web mboguapy haguà umi oguerekóva Dengue mba'asy, ha peicha pe mba'ekuaàrà oñembyatyva oikatu ojepuru ha oikuaveta mba'e tembiapo ojejapota oipe'a hagu'a ko mba'asy umi tesái tenondeha oimeva Septima Región Sanitariape, oimeva Itapua Paraguaipe. Ko plataforma web ojemboja'o hetà pehenguepe ojejapohaguà, ha upéi oñembyaty oikohaguà peteicha ojoykére. Ko jehaipyrepe ojehài mba'eicha ojejapo ko módulo de interconectividad API REST, kòa ombo'apota plataforma web mbytéicha, mbohasàhaguà ha ñongatu mba'ekuaa. Ojehupytyhaguà ojejaposeva, ojeipuru metodología de investigación aplicada, oñembyaty momarandu ha ojeheka kuaa ojejapohaguà tembiapokue ojapo poràta avanoòkuèrape. Tembiapo ojejapótava oikòhaguà ko módulo de interconectividad API REST, ojeipuru metodología ágil SCRUM, oguerekóva flexibilidad ha herramientas oikatuva mohenda ojejapovaerà pya'e ha ndoipuruiva heta ára eguahè haguà tembiapokue ojeipotáva. Péicha upéi ojeporavó recursos tecnológicos, ha'eicha lenguaje de programación, entorno de desarrollo, sistemas de gestión de bases de datos, ha opaichagua oikatùva ojeporu, ha'éva software libre, ha ojejapo programación y configuración ojeipotáva, oguahèhaguà pe funcionalidades oñemohendava. Ojehechajey haguà oiko poràpa ojejapovaekue, ojeipuru techaukahà opa tembiapo ojejapova, pèicha ojejapojev opapeve ojehupytyseva, he'iaicha metodología SCRUM. Péicha ojejapopaito ko módulo de interconectividad API REST, oñembyaty porà plataforma web ndive, ha ombo'apo porà ojoykére, ojeporuhaguà Dengue mba'asy pe'a yvypòragui ha pytyvòhaguà tesái motenondehakuérape.

ÑE'Ê TEKOTEVÈVA: Mba'asy tropical, Momarandu ñembohasa, Momarandu mba'aporekokuaa

# REST INTERCONNECTIVITY FOR TRANSFER SERVICE OF INFORMATION FOR THE SEVENTH SANITARY REGION OF ITAPÚA

Author: Mario David Segovia Troche.

Research Line: Software Engineering and Information System.

## Summary

Dengue actually is one of the diseases that have a huge impact on population health and sanitary systems on vast regions of the planet, embracing a large number of countries, including Paraguay. With the purpose to eradicate and control the disease, several strategies were used through projects, plans, investigations and others initiatives involving different knowledge areas and sciences. From the different initiatives that are made, several examples of those were made using resources from information and communication technologies, as well as computing sciences and related, with successful and beneficial results. From these aforementioned facts, emerge a proposal of a web platform for the register of people infected with Dengue virus, in order to be a technological utility tool to struggle against Dengue disease, to be used by the people in charge of the Séptima Región Sanitaria de Itapúa, Paraguay. The web platform to be done is modeled in a modular structure, when each module has several specific functionalities that works together as a unitary system. In this document is described the API REST interconnectivity module development were the function to be the core of the entire platform for interchange and storage of the generated data. To achieve the proposed objectives, was used an applied investigation methodology, conducting an information gathering and get knowledge in order to obtain a result that benefits society with a practical solution. For the development of the API REST interconnectivity module, was chosen the SCRUM agile methodology, due to its characteristics that offer great flexibility and the necessary tools for a successful planning and execution of activities to reach the final product. Based on the selection of technological resources such as the programming language, development environment, database management systems, among others, taking into account their quality of free software, all the required functionalities were programmed and configured. To verify the correct functioning and expected behavior of the module's functionalities, tests were applied constantly throughout the development process, according to the guidelines established by the SCRUM methodology. In this way, the API REST interconnectivity module was completed, with the successful integration to the web platform, with which it is intended to offer a solution for the application of strategies in the fight and control of Dengue that affects the population and sanitary institutions.

KEY WORDS: Tropical diseases, Information transfer, Information technology

## INDICE

	Página
1. INTRODUCCIÓN .....	8
2. MARCO TEÓRICO.....	12
2.1. Consecuencias sociales y económicas de la enfermedad del Dengue en el Paraguay ..	12
2.2. Utilización de las Tecnologías de la Información y Comunicación en el control y la erradicación del Dengue.....	24
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.....	35
3. PREGUNTAS, HIPOTESIS Y OBJETIVOS.....	46
3.1. Preguntas: .....	46
3.2. Hipótesis:.....	46
3.3 Objetivos .....	47
4. METODOLOGÍA .....	48
4.1. Tipo y diseño de investigación.....	48
4.2. Definición del objeto de estudio.....	48
4.3. Descripción de la población y muestra .....	49
4.4. Procedimiento de recogida de información.....	49
4.5. Procesamiento de los datos .....	49
4.6 Alcance y Limitaciones del Software.....	50
4.7. Validación de los datos .....	50
5. RESULTADOS Y DISCUSIÓN .....	52
5.1. Selección de la metodología de desarrollo .....	52
5.2. Establecimiento del modelo de negocio del módulo.....	52
5.3. Definición del entorno tecnológico. ....	53
5.4. Planificación de actividades aplicando la metodología de desarrollo ágil SCRUM.....	54
5.5. Instalación de librerías del lenguaje PHP por medio del paquete de instalación XAMPP .....	55
5.6. Instalación y configuración del gestor de base de datos PostgreSQL.....	56
5.7. Instalación de Composer .....	58
5.8. Creación del proyecto Laravel .....	58
5.9. Configuraciones iniciales del proyecto Laravel .....	59
5.10. Creación del repositorio en Github para el control de versiones del proyecto. .....	60
5.11. Programación de las funcionalidades del módulo API. ....	63

5.11.1. Programación de Modelos en el proyecto Laravel .....	64
5.11.2. Programación de Controladores en el proyecto Laravel .....	68
5.11.3. Programación de rutas para los puntos de acceso o endpoints en el módulo API. .....	70
5.12. Programación del módulo de gestión de usuarios de la API.....	72
5.13. Programación de funcionalidades de los módulos de la plataforma.....	82
5.14. Programación de características de seguridad del módulo API.....	85
5.15. Programación de características de control de acceso del módulo API.....	85
5.16. Pruebas de funcionamiento .....	86
6. CONCLUSIÓN.....	90
7. REFERENCIAS BIBLIOGRÁFICAS.....	92

## 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 este 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.

Se aplica una metodología de investigación aplicada, con el objetivo de obtener conocimiento e información mediante la investigación para ofrecer una solución que responda a una problemática dada, de una forma práctica, resultando en un beneficio a la sociedad. Esto será realizado mediante la planificación de actividades y el flujo de trabajo estructurado mediante una metodología de desarrollo ágil. Mediante la utilización de las técnicas, actividades y artefactos de la metodología ágil SCRUM, se realizan los procedimientos de recolección de

información, procesamiento y validación de datos (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, 2018).

El módulo de Interfaz (API, por sus siglas en inglés de 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 de Estados Representacionales (REST, por sus siglas del inglés 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 describe en los siguientes capítulos:

- Selección de la metodología de desarrollo: a partir de una investigación bibliográfica y una evaluación y comparación de las metodologías de desarrollo, se decidió por la selección de la metodología de desarrollo ágil SCRUM, por ser la opción más adecuada para este proyecto.
- Establecimiento del modelo de negocio del módulo: se define la función que tendrá el modulo API REST dentro la plataforma propuesta. Se describe las características y el funcionamiento esperado del módulo.
- Definición del entorno tecnológico: luego del análisis de los requerimientos se establecen las herramientas tecnológicas para el desarrollo del software, como el lenguaje de programación, el entorno de desarrollo, y otras aplicaciones adicionales.
- Planificación de actividades aplicando la metodología SCRUM: a partir de los artefactos establecidos por SCRUM, se aplican al proceso de planificar todas las actividades necesarias para el desarrollo del módulo.
- Instalación de librerías del Lenguaje PHP: como requisito inicial para el inicio del desarrollo se instalan los archivos necesarios para la configuración correcta y el uso del lenguaje PHP.
- Instalación del Sistema de Gestión de Bases de Datos PostgreSQL: se instalan las dependencias necesarias para la base de datos a ser utilizado para el almacenamiento de la información.
- Instalación del gestor de paquetes Composer: aplicación necesaria para la gestión de librerías para el entorno de desarrollo Laravel.

- Creación del proyecto Laravel: en este paso se crea la base de todo el módulo, sobre el cual se construirá todo el software utilizando las utilidades proveídas por el entorno de desarrollo Laravel.
- Configuraciones iniciales del Proyecto Laravel: se realizan los primeros pasos para establecer la conexión y pruebas de comunicación con la base de datos y el ajuste de otros parámetros.
- Creación del repositorio Github: para el control de los cambios y avances del proyecto se utilizó la herramienta de control de versiones Git, respaldando la información en un repositorio remoto en la plataforma Github.
- Programación de las funcionalidades de la API REST: se describe la codificación de las funcionalidades del módulo mediante la arquitectura Modelo Vista Controlador (MVC), diseño en que se basa Laravel para la construcción del software.
- Programación del módulo de gestión de usuarios de la API: descripción de la construcción de las interfaces para la gestión de todos los elementos necesarios para el registro de los datos de las personas que utilizarán la plataforma.
- Programación de las funcionalidades de los módulos de la plataforma: se describen los módulos que integran la plataforma y que tienen conexión con el modulo API, especificándose las funcionalidades programadas para cada uno de ellos.
- Programación de características de seguridad del módulo API: descripción de la programación y configuración de parámetros para la autenticación segura de los usuarios del sistema
- Programación de características de control de acceso del módulo API: instalación de las librerías para la configuración para establecer los permisos que permitan a los usuarios acceder a determinados datos o limitar su campo de acción dentro de la plataforma.
- Pruebas de funcionamiento: aplicación a todas las funcionalidades de testeos para la comprobación del correcto desempeño y comportamiento esperado de las características del módulo.



## 2. MARCO TEÓRICO

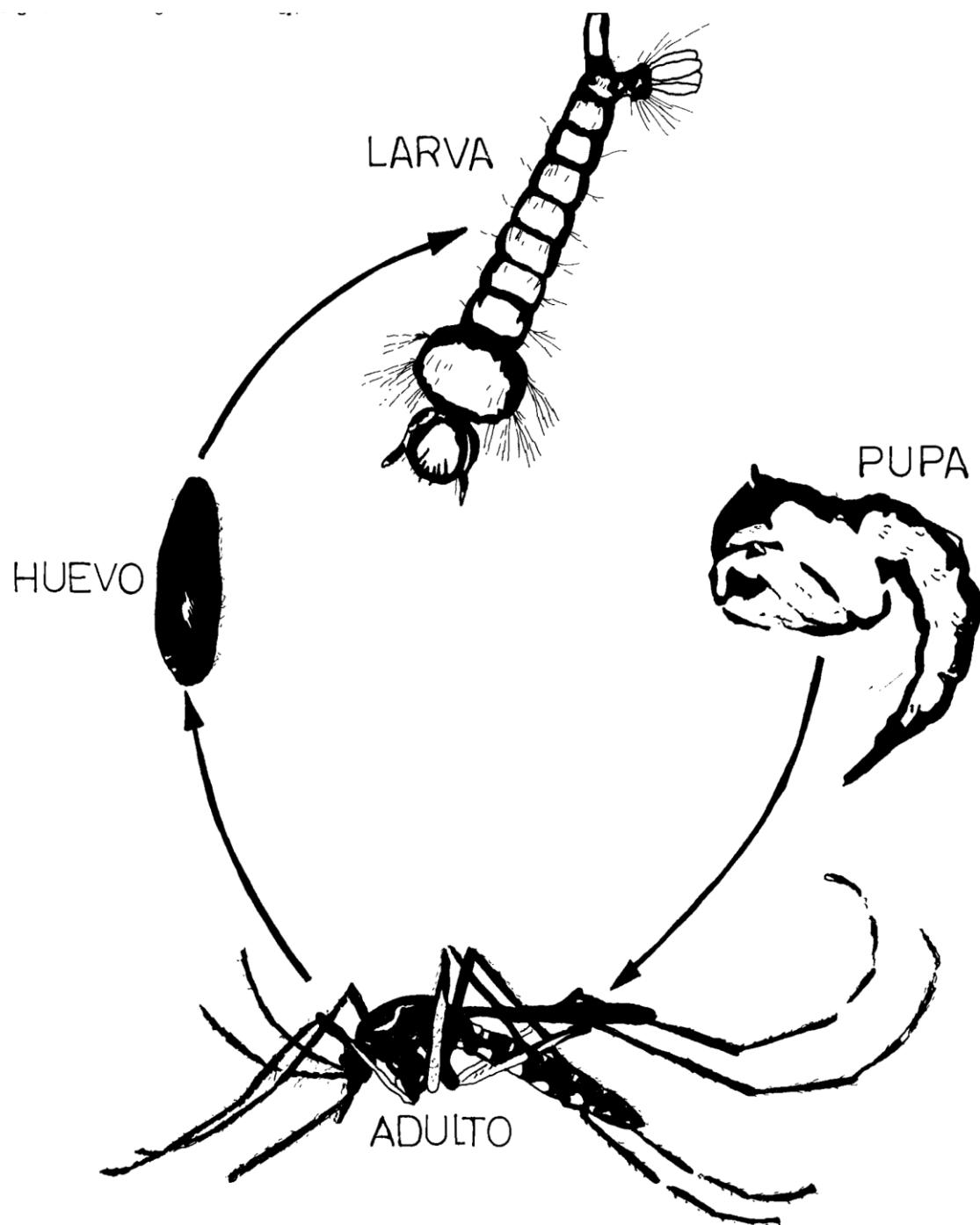
### 2.1. Consecuencias sociales y económicas de la enfermedad del Dengue en el Paraguay

El dengue es una enfermedad causada 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 está mayormente activa en horas de la mañana y antes de oscurecer, y se reproduce colocando sus 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. Tiene una capacidad de vuelo de 100 metros aproximadamente, pero pueden alcanzar recorridos de hasta 3 kilómetros para encontrar lugares adecuados para depositar sus huevos. El ciclo de transmisión se produce cuando un mosquito ingiere la sangre de una persona infectada, con lo cual adquiere el virus, que luego es transmitido a otras personas con la siguiente picadura del mosquito. Los pacientes infectados suelen tener un periodo de incubación de 5 a 8 días donde se presentan los primeros síntomas de la enfermedad. Los síntomas típicos de esta enfermedad suelen presentarse como fiebre alta, dolores de cabeza, musculares y articulares. Otras manifestaciones de la enfermedad serían la aparición de puntos rojos en las piernas y el tórax, que luego se diseminan por todo el cuerpo, inflamación de ganglios linfáticos, y en menor medida náuseas, vómitos, dolores abdominales con gastritis, estreñimiento, complicaciones hepáticas y renales, inflamación del bazo, sangrado de nariz y encías, distorsión en la percepción del sabor de las comidas o alimentos ingeridos. Los cuadros más graves pueden generar sangrados internos y daños profundos en órganos, hasta llevar a la muerte del paciente. Los signos de alarma que indicarían un cuadro grave de dengue serían dificultad para respirar, convulsiones, somnolencia o irritabilidad, taquicardia, hipotensión arterial o taquipnea, sangramiento por las mucosas, como nariz o encías, sudoración profusa y piel pegajosa en el resto del cuerpo, frialdad en manos y pies, palidez extrema, distensión y dolor abdominal. Se extiende en regiones de climas tropicales y subtropicales, como África, norte de Australia, México, Centroamérica y Sudamérica (Lage et. al, 2015).

El vector transmisor del Dengue es el mosquito *Aedes Aegypti*, como así también otras especies de la familia Aedes, como el albopictus, mediovittatus y scutellaris. El *Aedes Aegypti* es la que más abunda en las regiones tropicales y subtropicales, adaptada a la vida urbana y con un ciclo único, hombre – *Aedes Aegypti* – hombre. El ciclo de vida de este mosquito se

establece en el huevo, cuatro estadios larvales, un estadio de pupa y el de adulto. Los huevos tienen un tamaño aproximado de 1 milímetro de longitud, en forma de cigarrillo y lisos, al depositarse tienen un color blanco que luego cambia a negro brillante, suelen colocarse por encima del nivel de agua de los recipientes. En situaciones de clima con alta humedad y calor pueden ser fecundados en 48 horas, pero pueden extenderse a cinco días si hay descensos de temperatura. Pueden soportar hasta 1 año de periodos de sequía hasta volver a tener contacto con el agua, que en estos casos en 15 minutos ya pueden hacer eclosión o al ser mojados varias veces, por lo que esta característica en particular es la que dificulta su control y erradicación. El siguiente estado de Larva se desarrolla exclusivamente en el agua, el cual es la etapa de desarrollo y crecimiento, se caracterizan por permanecer en la superficie del agua casi en forma vertical y se desplazan por el agua con movimientos serpenteantes. Se distinguen por dos salientes espinas laterales del tórax y una serie lineal de siete a doce escamas del peine en el octavo segmento abdominal. El crecimiento de la larva depende de la temperatura, del alimento disponible y de la cantidad de otras larvas que comparten el depósito. En un escenario ideal, el ciclo desde el huevo a larva hasta la fase de pupa puede ser de cinco días, pero normalmente es entre 7 a 14 días. Cualquier desecho que pueda servir de recipiente para acumular agua son potenciales sitios de reproducción, como neumáticos usados, botellas, latas y en algunos casos como el exceso de lluvia que causan rebosamiento o la desecación son perturbaciones que pueden causar mortalidad en larvas y pupas. La fase de pupa también se desarrolla en el agua, en este periodo no se alimentan, ya que llevan a cabo la transición del estado de larva al de adulto, tiene la capacidad de flotar en la superficie del agua, facilitando la salida del mosquito adulto de la pupa. Este estadio tiene una duración de dos a tres días, si no existe intervención de los factores ambientales. La última fase es la del mosquito adulto, es la etapa de reproducción, se pueden identificar por su color oscuro con bandas blancas en las bases de los segmentos del torso y un peculiar diseño en forma de lira en el mesonoto o sección del tórax del mosquito. El mosquito adulto puede vivir varios meses en laboratorios, pero en la naturaleza sólo llegan a sobrevivir unas pocas semanas. Una gran cantidad de adultos suelen morir alemerger de la pupa o poco tiempo después, pero la supervivencia diaria es constante. Con una mortalidad diaria del 10%, la mitad de los mosquitos mueren en la primera semana de vida y un 95% muere al primer mes. A pesar de la alta mortandad que reduce significativamente la población, un gran número sobrevive siendo capaz de generar una epidemia del dengue (Chico Aldama, 2019). En la *Ilustración 1* podemos observar el ciclo biológico del Aedes Aegypti

Ilustración 1. Ciclo biológico del Aedes Aegypti



Fuente: Chico Aldama (2019).

Con respecto al origen del nombre de la enfermedad, no se tiene una información precisa. Una teoría refiere que proviene de la frase de la lengua Swahili “kadinga pepo”, que describe a un fantasma como causante de la enfermedad. Otra posibilidad es que la palabra en Swahili “dinga” o “dyenga”, homónimo del swahili “ki denga pepo” (cuyo significado sería ataque repentino por “un espíritu malvado”, caracterizado por calambres y fiebre), derive de la palabra

en castellano “dengue”, que significa fastidioso o cuidadoso, para describir los padecimientos de un paciente con la fiebre y los dolores de huesos que produce la enfermedad. La primera referencia a un posible caso de Dengue se describe en una enciclopedia china de medicina, de la Dinastía Jin del 265 al 420, relacionando “agua venenosa” con el vuelo de los insectos. El primer caso bien documentado de esta dolencia proviene del año 1789, realizado por Benjamín Rush, utilizando el término “fiebre rompehuesos”, por los síntomas de mialgias y artralgias que presentaban los infectados. Con respecto a la relación histórica de la presencia del dengue en el continente americano, la primera manifestación podría haberse dado en el año 1635 en Martinica y Guadalupe. En el año 1881, el autor Carlos Finlay informa que la enfermedad y el vector transmisor son autóctonos de América y el Aedes Aegypti era el principal trasmisor de la Fiebre Amarilla, permitiendo la realización de nuevos estudios sobre enfermedades transmitidas por vectores. Las primeras epidemias aparecieron prácticamente al mismo tiempo en América del Norte, Asia y África en el año 1781. La enfermedad fue individualizada y nombrada como tal en el año 1779. El primer brote pandémico del que se tiene registro ocurrió en el año 1779 en la Isla de Java, y en la ciudad de Filadelfia de los Estados Unidos de América, en el año 1780. En el año 1827 aparece la primera pandemia en la costa atlántica de los Estados Unidos y en el Caribe, y en el año 1848 la segunda pandemia expandiéndose a las ciudades de La Habana y Nueva Orleans. Una tercera pandemia ocurrió en el año 1879 en la zona del Caribe, afectando a los países de Cuba, Bermudas, Panamá, Islas Vírgenes, Puerto Rico y Venezuela. Se tiene la creencia que la Segunda Guerra Mundial fue una de las principales causas de la expansión global de la enfermedad del Dengue, por medio de los traslados de las tropas a distintas partes del mundo, llevando consigo el virus y el mosquito. Justamente desde el año 1950, después de la guerra, empezaron las pandemias con más frecuencia en varias zonas del mundo. En los años 1950 y 1975 se produjeron pandemias mundiales originadas en el sudeste de Asia, por Dengue hemorrágico, convirtiéndose en una de las principales causas de muerte de niños en las regiones afectadas. Para el año 1975, el Dengue ya se había expandido a gran parte del mundo y desde el año 1980 en adelante se convirtió en una epidemia común. Desde el año 2000 el Dengue se convirtió en la segunda enfermedad más común en ser transmitida por mosquitos, que afectan a los humanos, después de la Malaria (Lugones Botell, & Ramírez Bermúdez, 2012).

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).

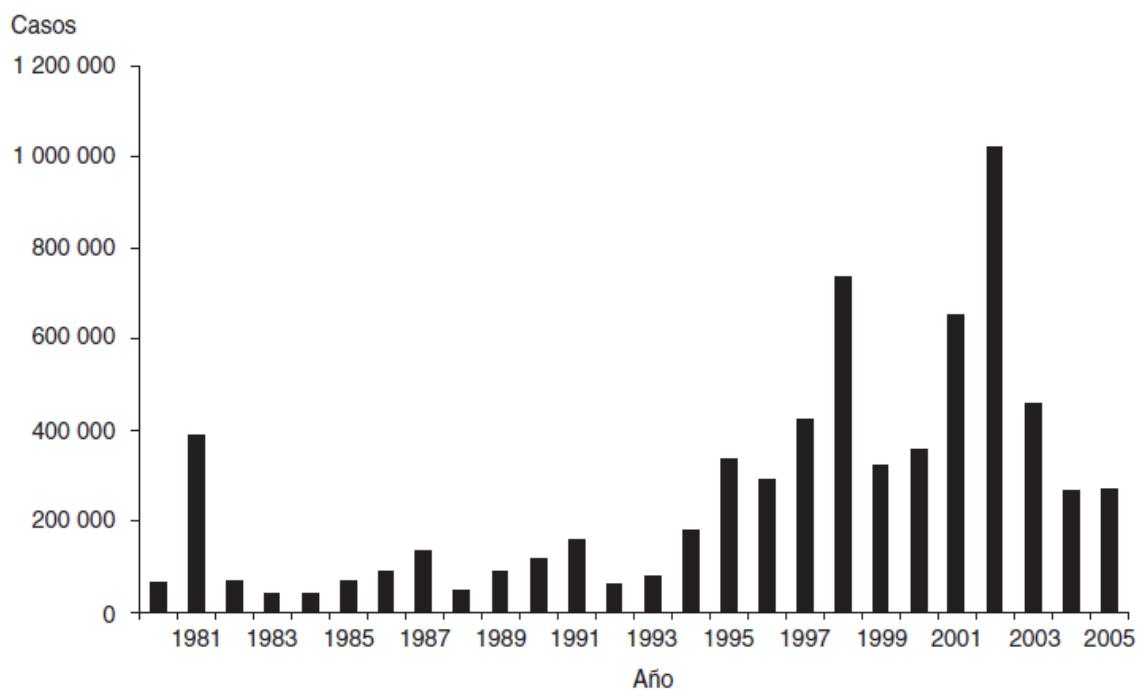
De acuerdo con la OMS (2015), presenta el informe de la secretaría en la 68<sup>a</sup> Asamblea Mundial de la Salud sobre la enfermedad del Dengue, exponiendo el riesgo a nivel mundial de esta enfermedad para la salud de la población, la situación epidemiológica de cada Región que integran la OMS, estrategias para su control y prevención, como el fortalecimiento de los sistemas de salud, control del vector transmisor e investigaciones más profundas sobre la dolencia, que puedan resultar en soluciones como vacunas contra la enfermedad. Existe una deficiencia en la notificación real de la cantidad de casos de Dengue y mayormente tienen una mala clasificación. Recientes estimaciones estiman 390 millones de casos al año, con un porcentaje de credibilidad del 95%, quedando así entre 284 a 528 millones, y de los cuales unos 96 millones, o de 67 a 136 millones tienen manifestaciones clínicas presentando diferentes niveles de gravedad de la enfermedad. Se calcula que una población de 3900 millones de 128 países está en riesgo de contraer la enfermedad. Los estados asociados a la OMS remiten periódicamente los casos detectados a la Secretaría en forma anual, de esa forma se informaron de aproximadamente 2,4 millones de casos en el año 2010, aumentando a 3 millones en el año 2013. A pesar de la incertidumbre acerca de los números reales de la enfermedad, los esfuerzos realizados para registrar de forma cierta los datos de esta dolencia evidencian el aumento pronunciado de los casos en los últimos años. Otro punto a tener en cuenta es la gravedad de las pandemias que incluyen a varios serotipos del Dengue en muchos países teniendo graves consecuencias en la salud de la población y en la economía, ya sea nacional y mundial. La expansión del vector transmisor del dengue ha continuado silenciosamente hasta llegar presente en más de 150 países. El comercio mundial de bienes y servicios que transportan los huevos secos del vector han posibilitado su propagación. Hay que tener en cuenta que además de transmitir el virus del dengue, también son vectores de transmisión de otros arbovirus como el Zika y el Chikungunya. Los países de Asia-el Pacífico son los más comprometidos con la enfermedad del Dengue, donde una población de aproximadamente 1800 millones de personas está en riesgo de contraer la enfermedad. Las epidemias crecen rápidamente y se expanden a nuevos territorios antes no afectados de las áreas urbanas a las rurales. Las epidemias suelen presentar una mortalidad elevada al principio del brote, y el rápido avance de casos graves satura los servicios de salud. Esta situación llevó a las regiones de Pacífico Occidental y Asia Sudoriental a ejecutar un plan estratégico contra el Dengue para la Región Asia-el Pacífico para los años 2008-2015 y la estrategia para enfermedades emergentes 2010. La Región del

Pacifico Occidental reportó casos de Dengue provenientes de más de 30 estados y territorios. Los países insulares son susceptibles a los brotes epidemiológicos, entre los años 2013 y 2014, en Fiji y otras islas, se reportó la circulación del serotipo 3 del Dengue, después de tres décadas de ausencia, provocando un aumento significativo de casos en una población vulnerable a este serotipo. Malasia y Singapur presentaron también brotes epidémicos sostenidos en este periodo de tiempo. A partir de los últimos meses del año 2013, solamente una pequeña cantidad de países del Pacifico han reportado pandemias de Dengue, Zika o Chikungunya. El diagnóstico y la atención médica adecuada continúan representando un problema y la lucha se concentra en el mejoramiento de la vigilancia de la enfermedad y en la erradicación del vector transmisor con el involucramiento activo de la comunidad. En el año 2014 se reportaron brotes en Japón y China. En la Región Europea, el virus se propago mediante el mosquito Aedes Albopictus, llegando a más de 25 países, acrecentando los riesgos de una potencial pandemia tras un periodo de 55 años. Los primeros casos de transmisión local notificados datan del año 2010 en Croacia y Francia, en otros países fueron detectados casos importados. En el año 2012 apareció un brote de Dengue en la Isla de Madeira, Portugal, arrojando un total de 2200 casos y en la importación de casos en otros 17 países europeos. El Dengue tiene una alta incidencia en la Región de Asia Sudoriental, aunque el impacto de la enfermedad varia tanto de país en país, como dentro de cada país. Solamente ocho países habían reportado casos hasta el año 2003. En el Año 2004 Timor Leste y Bután reportaron por primera vez casos de dengue y Nepal su primer caso local. Para el año 2009 todos los países miembros excepto la República Democrática Popular de Corea reportaron casos de Dengue. En la Región del Mediterráneo Oriental el dengue es considerado una enfermedad emergente ya que hace solo 20 años atrás empezaron las notificaciones de casos. Los casos provienen en general de países con costas en el Mar Rojo. La enfermedad está representando una situación muy problemática en países como Yemen, Arabia Saudita y Pakistán. Reportes de brotes con más frecuencia provienen de países como Djibouti, Somalia y Sudán. Omán ha reportado casos importados. En la Región de África, aunque no se tenga un panorama claro de la carga de Dengue en esta zona, se tienen datos de 22 países que reportaron brotes. El Dengue es endémico en varias zonas de África, según resultados de encuestas serológicas, en las cuales se detectan una alta cantidad de anticuerpos contra el virus. Los números registrados siguen siendo insuficientes en África debido a la poca preparación del personal médico, la proliferación de otros cuadros febriles, y la falta de insumos para la detección correcta de la enfermedad y su posterior notificación. A partir del año 2013 se registran brotes en Angola, República Unida de Tanzania y Mozambique. En la Región de las Américas durante la década de 1970 la circulación del virus del Dengue se

vio interrumpida por la campaña de erradicación del mosquito Aedes Aegypti, para la eliminación de la fiebre amarilla. A partir de los años 80 la población de los mosquitos volvió a aumentar y se empezaron a producir brotes de Dengue, debido a la relajación en los programas de control del vector. Actualmente el Caribe, Centroamérica y América del Sur están afectados por una situación de híper-endemia, afectando inclusive a las poblaciones indígenas. Debido a esta situación toda la región en colaboración con la OMS y la Organización Panamericana de la Salud (OPS), están realizando planes estratégicos integrados para la prevención y control del Dengue. Estos planes tienen la finalidad de optimizar y reforzar los sistemas de vigilancia epidemiológica, y crear una red de laboratorios para el análisis de los casos de Dengue, con los cuales tener una respuesta rápida y eficaz en casos de pandemias y escenarios críticos de la enfermedad.

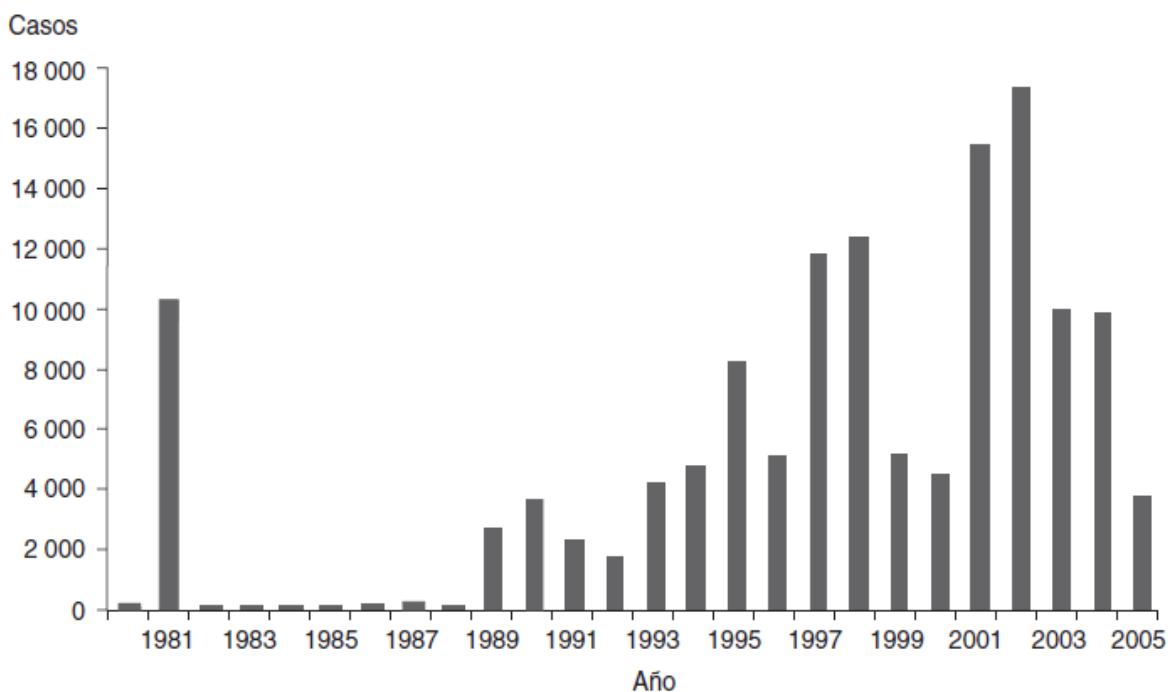
En las Américas, el Dengue tuvo un aumento sostenido desde la década de 1980, tanto en la forma clásica como en la forma más grave, el hemorrágico. En 1981 se constató en Cuba la primera epidemia de dengue hemorrágico de la región, desatado por el serotipo 2, reportándose una cantidad de más de 300 mil casos, con más de 10 mil casos graves o muy graves, con 158 muertes registradas, siendo 101 de estos fallecidos niños. Con respecto a los casos históricos de Dengue hemorrágico, se resalta esta situación en particular ocurrido en Cuba, ya que hasta 1981 solo se habían registrado 60 casos de Dengue hemorrágico en la Región de las Américas, y luego de la contención y control de la epidemia cubana, no se volvieron a registrar casos hasta la epidemia ocurrida en Venezuela en el año 1989. Algunas causas probables del aumento de casos y epidemias en las Américas serían el ingreso del serotipo 4 en el año 1981, la introducción del Aedes albopictus en el año 1985 proveniente de Asia y la reintroducción del serotipo 3 en América Central en el año 1994, extendiéndose en México, el Caribe y América del Sur. Se resalta que la cantidad de enfermos aumenta cíclicamente cada 3 a 5 años, con una tendencia incremental. En el año 2002 se registró uno de los mayores picos de la enfermedad en la Región, con un reporte de más de 1 millón de casos y alrededor de 14 mil casos de Dengue hemorrágico. En el año 2005, 27 países habían reportado casos de Dengue en sus dos formas, clásico y hemorrágico, y en 14 países con la circulación de 2 o 3 serotipos al mismo tiempo. En la *Ilustración 2* se puede visualizar los casos de Dengue en las Américas entre los años 1980 a 2005, así como en la *Ilustración 3* se pueden observar los casos de Dengue Hemorrágico en el mismo periodo de tiempo (Garcia, Guzman & Kouri, 2006).

*Ilustración 2. Casos de Dengue en las Américas, desde el año 1980 hasta el 2005.*



Fuente: Garcia, Guzman & Kouri (2006).

*Ilustración 3. Casos de Dengue Hemorrágico en las Américas, desde el año 1980 hasta el 2005.*



Fuente: Garcia, Guzman & Kouri (2006).

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 los resultados de un estudio realizado en el año 2017 en el Hospital Regional de Coronel Oviedo, del departamento de Caaguazú, República del Paraguay, de un total de 22 pacientes estudiados, se llegó a la conclusión que cada paciente representa un gasto de servicios e insumos sanitarios en guaraníes de aproximadamente 5.700.000 o en dólares americanos de 1.054 (*Ilustración 4*), 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 (Rodriguez-Castro, Rolón & Rios-Gonzalez, 2019).

*Ilustración 4. Distribución según costo de paciente por Dengue.*

	<b>Por día (Gs)</b>	<b>Por 4 días (Gs)</b>	<b>Total</b>
Terapéutica (Hidratación y Medicamentos)	339.000 (61,63 USD)	2.879.096 (523,47 USD)	2.879.096 (523,47 USD)
Medios auxiliares de diagnóstico	143.000 (26 USD)	572.000 (104 USD)	572.000 (104 USD)
Recursos Humanos	466.126 (84,75 USD)	819.352 (148,97 USD)	819.352 (148,97 USD)
Aseo	380.774 (56,14 USD)	1.523.096 (276,92 USD)	1.523.096 (276,92 USD)
<b>TOTAL</b>			<b>5.793.544 (1053,53 USD)</b>

Fuente: 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. En la *Ilustración 5* se observan los costos de insumos y estudios laboratoriales por paciente, en la *Ilustración 6* se muestran los costos totales por paciente internado incluyendo los honorarios profesionales de los médicos. El costo total de pacientes figura en la *Ilustración 7*. En el caso de los pacientes ambulatorios o que no requieren de internación, los costos están reflejados en la *Ilustración 8*. En cuanto a los costos indirectos, correspondientes a subsidios por reposo y pérdida de días laborales, se aprecian en la *Ilustración 9*. Por ultimo tenemos la totalidad de los costos que resultaron de este estudio (*Ilustración 10*) (Flores, Giménez Caballero, Díaz Duba & Torales, 2015).

*Ilustración 5. Costos directos de insumos y estudios laboratoriales por paciente internado*

ITEMS	Por día (Gs.)	Por 4 días (Gs.)
<b>Insumos</b>		
Hidratación (Suero fisiológico)	26.200	104.800
Descartables	9.050	22.390
Medicación	2.930	4.430
<b>Total insumos</b>	<b>38180 (7,6 USS)</b>	<b>131.620 (26,3 USS)</b>
<b>Estudios laboratoriales</b>		
Hemograma	1890	7560
Perfil hepático	3080	6160
TP	2930	2930
Urea	770	1540
Creatinina	770	1540
Glicemia	770	1540
Electrólitos	2950	5900
Gasometría	4500	4500
Serología para dengue	.....	.....
<b>Total estudios laboratoriales</b>	<b>17.660 (3,5 USS)</b>	<b>31.670 (6,3 USS)</b>

Fuente: Flores et. al (2019).

*Ilustración 6. Costo total de pacientes internados por día y por 4 días.*

ITEMS	Por día (Gs.)	Por 4 días (Gs.)
Insumos	38.180 (7,6 USS)	131620 (26,3 USS)
Laboratorio	17.660 (3,5 USS)	31670 (6,3 USS)
Honorarios	600.000 (120 USS)	2.400.000 (480 USS)
<b>Total</b>	<b>655.840 (131,1 USS)</b>	<b>2.563.290 (512,6 USS)</b>

Fuente: Flores et. al (2019).

*Ilustración 7. Costo total de todos los pacientes internados.*

n	Adultos HC/día	Adultos HC/4 días
344	225.608.960 (45.121,7 US\$)	881.771.760 (176.354,3 US\$)
	Todos los Internados/día	4 días
1447	949.000.480 (189.800,1 US\$)	3.709.080.630 (741.816,1 US\$)

Fuente: Flores et. al (2019).

*Ilustración 8. Costos de atención ambulatoria por paciente y por el total de pacientes*

ITEMS	Tratamiento 7 días (Gs.)	
Medicación	4.938	(1 US\$)
Laboratorio	19.530	(3,9 US\$)
Honorarios	1.800.000	(360 US\$)
Total por paciente	1.824.468	(364,8 US\$)
Total casos 5.960	10.873.829.280	(2.174.765 US\$)

Fuente: Flores et. al (2019).

*Ilustración 9. Costos indirectos*

	n	Total días	Monto
Subsidios *	2024	12.552 (media: 6,2 días)	358.701.798 (71.740,3 US\$)
Pérdida Laboral	2024	12.552	829.624.276 (165.924,8)
Todos los reposos**	5960	41.720 (media: 7 días)	2.313.165.400 (462.633,1 US\$)

Los subsidios se pagan a partir del segundo día de incapacidad, el monto corresponde al 50% del salario del trabajador de los últimos 4 meses, de acuerdo a la Ley Orgánica del Instituto de Previsión Social.

\*\*Para los cálculos totales se consideró el salario mínimo del 2007 de 1.219.795 Gs.(55.445 Gs. el jornal diario)

Fuente: Flores et. al (2019).

*Ilustración 10. Costos totales*

	Costos totales
Paciente Internado (4 días)	881.771.760 Gs (176.354,3 US\$)
Ambulatorio (7 días)	10.873.829.280 Gs (2.174.765,1 US\$)
Ausentismo laboral	829.624.276 Gs (165.924,8 US\$)
<b>TOTAL</b>	<b>12.585.225.316 Gs (2.517.045,1 US\$)</b>

Fuente: Flores et al. (2019).

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 (*Ilustración 11*). 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\$ (*Ilustración 12*). Contrastando estos datos con el presupuesto anual total destinado a la salud pública en el Paraguay, que asciende a aproximadamente 800 millones de dólares americanos, en el año 2010 los costos por la enfermedad del Dengue fueron 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, Lovera, Merlo & Arbo, 2020).

*Ilustración 11. Número de casos reportados y no reportados en Paraguay entre los años 2010 al 2013*

Año	Ambulatorios		Hospitalizados	
	Casos reportados	Casos reportados y no reportados	Casos reportados	Casos reportados y no reportados
2010	20.288	182.592	1.075	1.505
2011	46.069	414.621	4.900	6.860
2012	28.676	258.084	6.061	8.485
2013	145.249	1.307.241	20.679	28.951
Total	240.282	2.162.538	32.715	45.801

Fuente: Cuellar et al. (2020).

*Ilustración 12. Promedio estimado de costos médicos directos, costos directos no médicos y costos indirectos en dólares. Paraguay 2010-2013*

Año	Costos por casos ambulatorios				Costos por casos Hospitalizados			
	Médicos directos	Médicos no directos	Indirectos	Total	Médicos directos	Médicos no directos	Indirectos	Total
2010	9.586.080	2.008.512	11.320.704	23.371.776	451.500	191.135	251.335	770.560
2011	21.767.603	4.560.831	25.706.502	53.071.488	2.058.000	871.220	1.145.620	3.512.320
2012	13.549.410	2.838.924	16.001.208	33.034.752	2.545.500	1.077.595	1.416.995	4.344.320
2013	68.630.153	14.379.651	81.048.942	167.326.848	8.685.300	3.676.777	4.834.817	14.822.912
Total	113.533.245	23.787.918	134.077.356	276.804.864	13.740.300	5.816.727	7.648.767	23.450.112

Fuente: Cuellar et al. (2020).

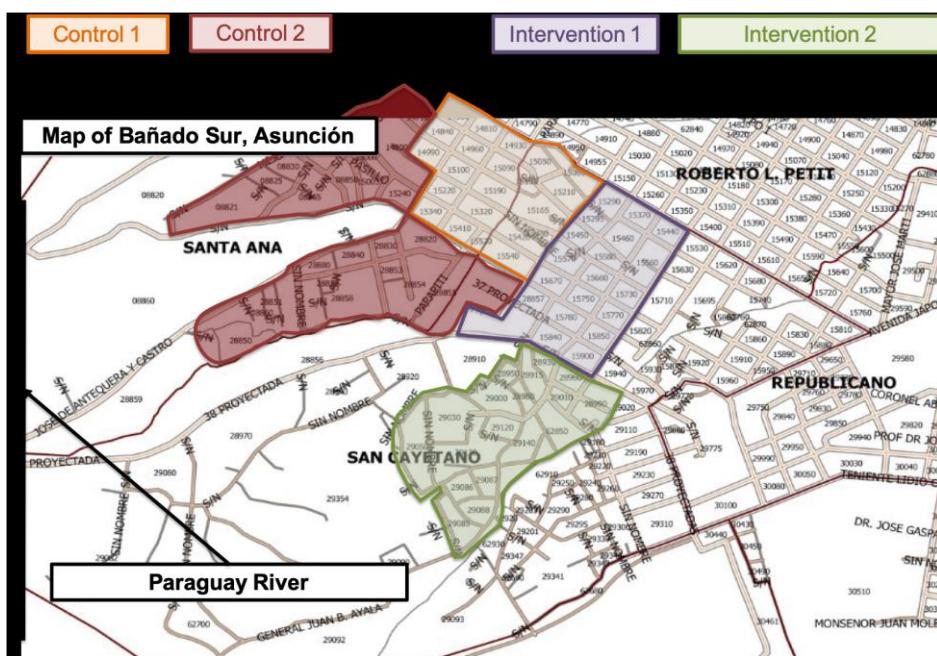
Estos números suponen una elevada carga económica y social en el sistema de atención médica y en la población en general.

## 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.

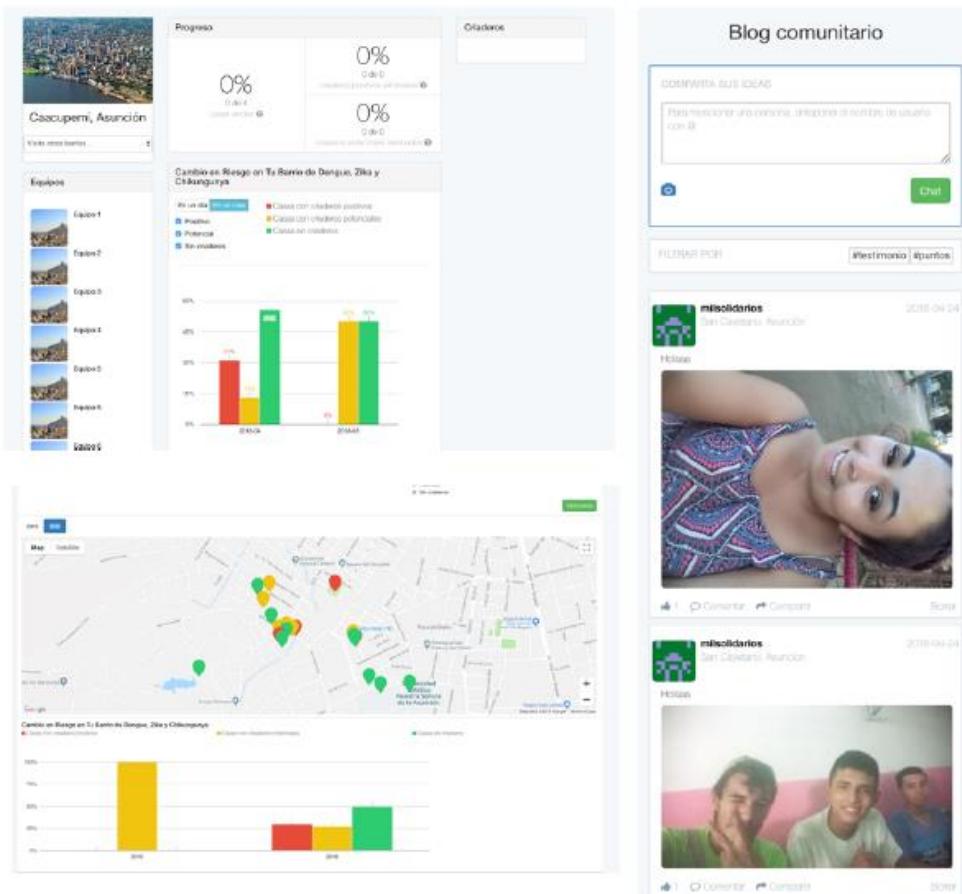
El proyecto TopaDengue, llevado a cabo por estudiantes de la Universidad Católica de la Ciudad de Asunción, Paraguay, con financiamiento del Consejo Nacional de Ciencia y Tecnología (CONACYT), tenía por finalidad reducir el alcance de las epidemias de Dengue mediante la eliminación y control de criaderos 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 (*Ilustración 13*). 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 (*Ilustración 14*), 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, México (Parra, 2019).

Ilustración 133. Mapas de las zonas de intervención y control del proyecto TopaDengue



Fuente: Parra (2019).

Ilustración 144. Componentes de datos y foro de la plataforma DengueChat

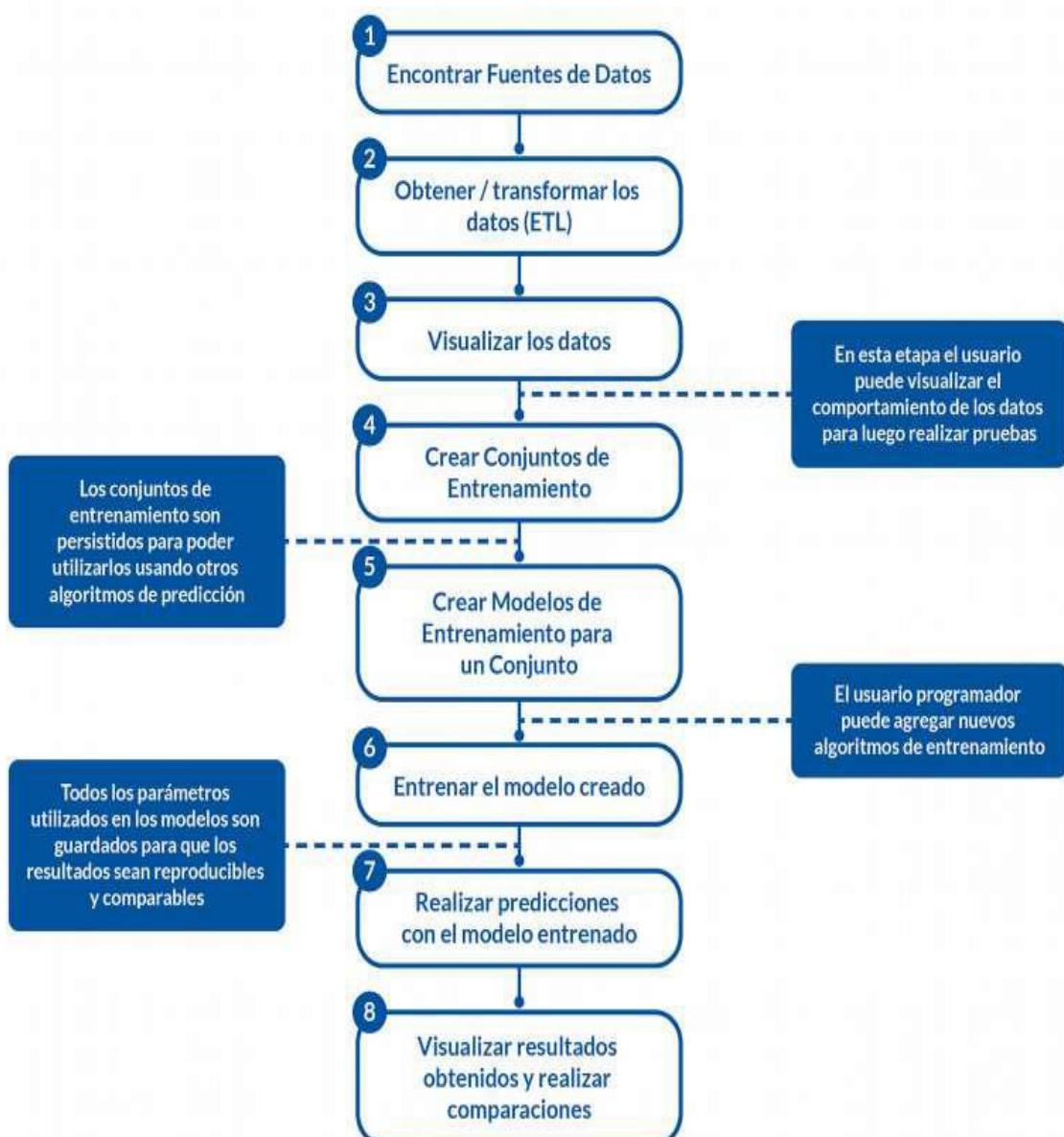


Fuente: Parra (2019).

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. Los usuarios de esta plataforma podrán realizar a partir del Módulo de Recolección y publicación de Datos la búsqueda de fuentes de información y crear formas para la carga, transformación y extracción de datos a la plataforma, centralizando el almacenamiento y permitiendo su libre acceso a otros usuarios interesados. El Modulo de Análisis Dinámico ofrece la posibilidad de generar diversos tipos de visualizaciones de los datos almacenados que permitan a los investigadores analizar de forma profunda los datos para una mejor comprensión y obtener información útil. Finalmente, el Módulo de Alertas Tempranas se utiliza para generar algoritmos que utilicen los datos disponibles para generar predicciones sobre posibles epidemias de Dengue. Si el usuario es un programador tiene la posibilidad de crear y entrenar con diversos tipos de algoritmos para la generación de las predicciones, y pueden ser guardados para su utilización en varias ocasiones o por otros usuarios que tengan interés en su uso. La estructura de los componentes de la plataforma se puede visualizar en la *Ilustración 15*. 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. Con respecto a las tecnologías utilizadas para el desarrollo de este proyecto podemos citar al gestor de base de datos NoSQL MongoDB para el almacenamiento de los datos, además de Elasticsearch, utilizado para el almacenamiento alternativo, para la visualización de datos se utilizó Kibana, para la autenticación y privacidad de datos se implementó Nginx, como lenguaje de programación se optó por Java con el entorno de desarrollo Springboot, con el servidor embebido Tomcat para la puesta en funcionamiento de la aplicación, utilizando el protocolo REST para la interacción con el software. Las configuraciones realizadas por los usuarios en la plataforma y los datos generados por modelos creados por estos usuarios son almacenados en una base de datos PostgreSQL. El entrenamiento de los algoritmos de predicción se realiza con la herramienta Apache Spark, debido a su capacidad de realizar pruebas de entrenamiento aprovechando un clúster de computadoras para el procesamiento de datos en paralelo. El clúster de almacenamiento Hadoop se utiliza para el cargado temporal de datos utilizado en el entrenamiento de los algoritmos. El conjunto de tecnologías y herramientas utilizadas para el

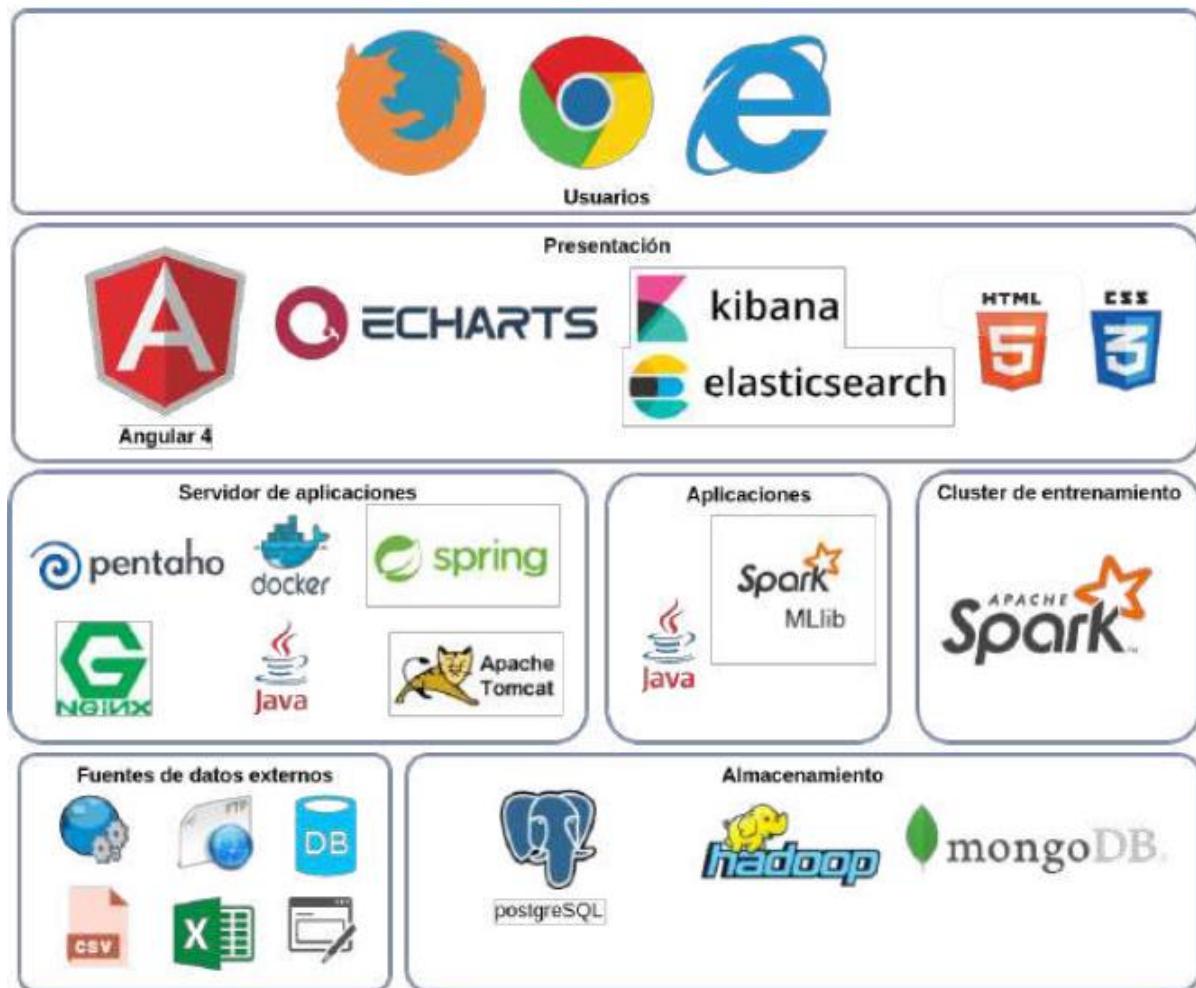
desarrollo de la Plataforma de Datos Abiertos y Alertas sobre Dengue se pueden observar en la *Ilustración 16*. Otras características a destacar son la flexibilidad para poder recolectar datos de distintas fuentes y tipos, y almacenarlas en una base de datos integrados y ofrecer un acceso de forma fácil 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).

*Ilustración 15. Componentes de la Arquitectura del proyecto Plataforma de Datos Abiertos y Alertas sobre Dengue*



Fuente: Pane et al. (2018).

*Ilustración 16. Tecnologías utilizadas para el desarrollo de la Plataforma de Datos Abiertos y Alertas sobre Dengue.*



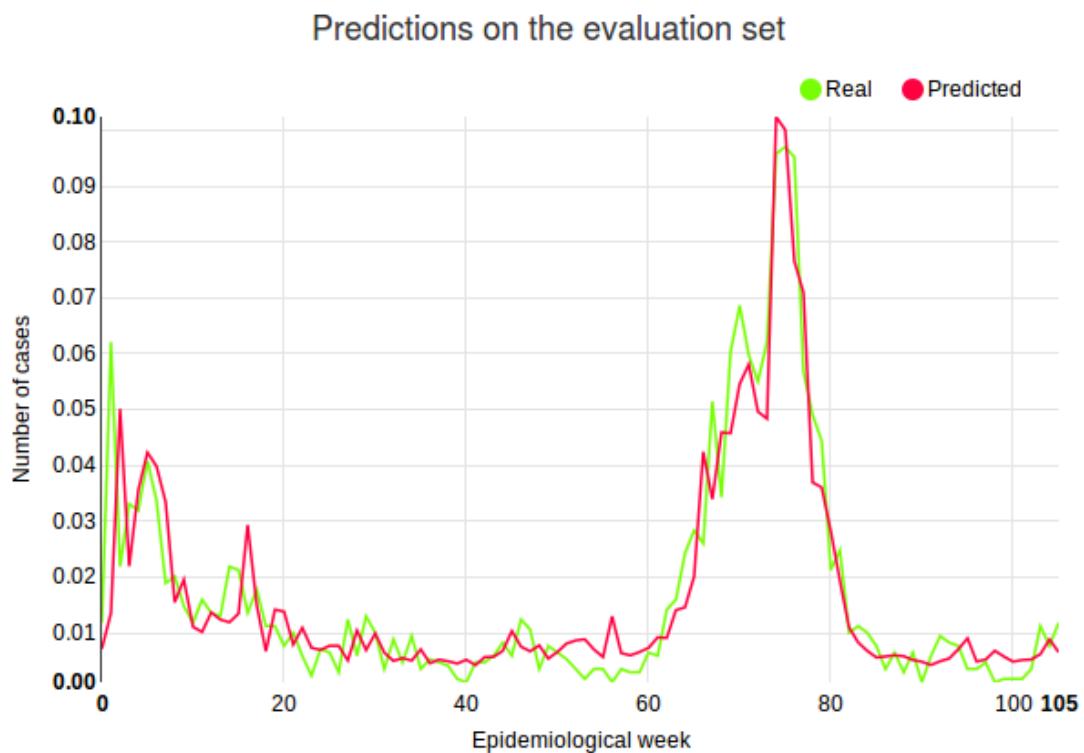
Fuente: Pane et al. (2018).

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. Un caso de estudio se realizó utilizando dos técnicas de Machine Learning, Redes Neuronales y Máquinas de Vectores de Soporte, para el diagnóstico temprano de casos de Dengue. Para la prueba se utilizó un conjunto de datos de pacientes registrados en el sistema nacional de salud de Paraguay, pertenecientes a varios establecimientos sanitarios del Departamento de Concepción, entre los años 2012 y 2016. En total fueron 4332 casos registrados los utilizados para el estudio, en donde los resultados de la aplicación de las técnicas de Machine Learning

arrojaron una alta precisión, especificidad y sensibilidad para diagnosticar casos de Dengue (Mello Román et al, 2019).

Otro estudio de implementación de Redes Neuronales para la predicción de casos de Dengue en Paraguay se realizó utilizando distintas variables para la construcción de los modelos de predicción. En este caso se tuvo en cuenta las variables medioambientales como las variaciones del nivel del Rio Paraguay, la cantidad de lluvia caída, la humedad y temperatura del ambiente y los casos de Dengue reportados. Luego de la aplicación de los algoritmos de predicción sobre los datos recolectados, los resultados tuvieron una alta precisión para anticipar los casos posibles hasta un periodo de 4 semanas de anticipación. En la *Ilustración 17* se puede observar el alto nivel de precisión entre la predicción del algoritmo, representado en la línea roja de la ilustración, con respecto a los casos reales que se registraron, representado en la línea verde de la ilustración. Asimismo, es posible aumentar la eficacia de estos algoritmos agregando más datos relacionados a la enfermedad, como ejemplo podemos citar la infestación del mosquito en las áreas estudiadas, la cantidad de larvas y criaderos, el movimiento poblacional, la distribución demográfica, la cantidad de campañas realizadas para la erradicación del vector, entre otros (Ughelli et al, 2017).

*Ilustración 17. Casos reales de Dengue en la Ciudad de Asunción comparado con la predicción de casos obtenidos por medio de algoritmos de Redes Neuronales.*



Fuente: Ughelli et al. (2017).

También encontramos proyectos como Mosquito Alert, utilizado en España, una iniciativa ciudadana sin ánimo de lucro supervisado por centros de investigación públicos con la finalidad de establecer la vigilancia, estudio y lucha contra los mosquitos causantes de enfermedades como el Dengue, fiebre amarilla, Zika o Fiebre del Nilo. A partir de la utilización de las TIC, recolectan datos sobre los vectores transmisores para su análisis y su posterior utilización para obtener información que sea de utilidad para su aplicación en estrategias relacionadas a la erradicación de estos mosquitos. A partir de una aplicación móvil los ciudadanos pueden tener una activa participación notificando con imágenes la presencia de mosquitos o criaderos, además de recibir notificaciones e informaciones útiles, tener un historial de las denuncias, y consultar en mapas la situación de la población del vector. Además cuenta con una completa página web con toda la información relacionada al proyecto, mapas interactivos para visualizar las denuncias y la situación del mosquito en el continente europeo, estadísticas, exportar datos, compartir con otras personas, entre otras posibilidades, la dirección web de este proyecto es [www.mosquitoalert.com](http://www.mosquitoalert.com). En la *Ilustración 18* podemos observar un poster promocional y en la *Ilustración 19* un díptico informativo sobre el proyecto.

*Ilustración 18. Poster informativo sobre el proyecto Mosquito Alert.*



Fuente: [www.mosquitoalert.com](http://www.mosquitoalert.com) (2021).

Ilustración 19. Díptico informativo sobre el proyecto Mosquito Alert.

## Juntos podemos eliminarlos

Las observaciones que envíes con la app Mosquito Alert se validan por un equipo de expertos y se pueden consultar des del mapa público de [www.mosquitoalert.com](http://www.mosquitoalert.com).

De esta forma colaboramos con **entidades y profesionales de la gestión de la salud pública** que se encargan del seguimiento y control de estos mosquitos vectores de enfermedades.

El objetivo es proteger la salud de las personas, minimizar las molestias que provocan estos mosquitos y reducir el riesgo de transmisión de enfermedades.



**Síguenos en:**

- [@Mosquito\\_Alert](#)
- [/mosquitoalert](#)
- [www.mosquitoalert.com](http://www.mosquitoalert.com)
- [info@mosquitoalert.com](mailto:info@mosquitoalert.com)

Licencia de uso de este documento: CC by Mosquito Alert  
Fotos portada: Jorge Mederros CC-BY, J.Gathany CC, ASPB.  
Edición 2018



## ¡Colabora en la lucha contra los mosquitos transmisores de enfermedades!



**MOSQUITO ALERT**



## Captura mosquitos con el móvil

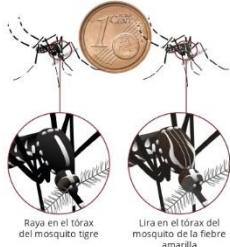
Mosquito Alert es un proyecto colaborativo de ciencia ciudadana para investigar y controlar la expansión del **mosquito tigre** y del **mosquito de la fiebre amarilla**. Estos insectos se han expandido en muchas zonas del mundo y pueden transmitir enfermedades como dengue, Chikungunya o Zika.

Con la **app Mosquito Alert** puedes ayudar al personal científico a estudiar por dónde se expanden estos mosquitos y predecir su expansión en zonas donde no ha llegado. También ayudarás al personal de gestión a detectar sus lugares de cría.

Además, entre todos podremos detectar más rápido la posible llegada del mosquito de la fiebre amarilla en España.



**¿CÓMO IDENTIFICARLOS?**



Raya en el tórax del mosquito tigre  
Lira en el tórax del mosquito de la fiebre amarilla

El mosquito tigre (*Aedes albopictus*) y el mosquito de la fiebre amarilla (*Aedes aegypti*) son muy parecidos. Ambos tienen **rayas en las patas y abdomen**.

Para diferenciarlos hay que fijarse en el **tórax**. El tigre tiene **una raya** y el de la fiebre amarilla tiene un **dibujo en forma de lira**.



Júntate a la foto de esta parte!

**EN LOS ESPACIOS PÚBLICOS**

Estos mosquitos se han adaptado a vivir en ambientes urbanos. Las hembras necesitan muy poca agua para poner huevos y en un ibornal puede haber centenares de larvas.

Si ves un **ibornal** o un espacio en la vía pública con **agua acumulada**, hazle una foto y envíala a través de Mosquito Alert.

Cuantos menos lugares de cría, ¡menos mosquitos!



Larvas y pupas.  
Foto: desmocedador.com

**EN CASA: SIGUE ESTOS CONSEJOS**



Vacia el agua de recipientes que estén en el exterior, aunque no haya larvas de mosquito.

Ponlos hacia abajo para que no se vuelvan a llenar de agua.

Tapa herméticamente los depósitos de agua con una tapa o pon una tela mosquitera.

Revisa cada 2 o 3 días el agua del plato de las macetas.

Guarda carretillas, barcas o remolques.

Evita tener neumáticos viejos al aire libre; llévalos al punto limpio.



Fuente: [www.mosquitoalert.com](http://www.mosquitoalert.com) (2021).

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 georreferencia. A partir de estos datos se cargaban en la plataforma SIG, con el cual era posible visualizar en los mapas digitales las 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).

En la *Ilustración 20* podemos observar las aplicaciones e investigaciones relacionadas con el Dengue, disponibles para el acceso público, donde se pueden consultar y visualizar datos y estadísticas de la enfermedad, como ser el serotipo, país, año región entre otros.

*Ilustración 20. Investigaciones y aplicaciones relacionadas al Dengue*

Investigaciones/aplicaciones	Variables epidemiológicas							Covariables			
	Cantidad total de casos	Serotipo	Clasificación clínica	Dimensión geográfica	Dimensión temporal	Demografía	Dimensión socio económica	Dimensión urbana	Dimensión climatológica	Dimensión ambiental	Dimensión entomológica
Denfree	*	*	*	*	*	*	*				
DengueNet				*	*	*					
Fortaleza (aplicación)	*		*	*	*						
Combining Google Earth and GIS mapping in a dengue surveillance system	*		*				*	*			*
Dengue and the World Football Cup	*		*	*	*		*		*		
Refining the Global Spatial Limits of Dengue Virus Transmission by Evidence-Based Consensus	*			*			*				*
Towards an Early Warning System to Combat Dengue	*		*		*		*		*		
The development of an early warning system for climate-sensitive disease risk	*			*	*		*	*	*	*	*
Epidemiological prediction method for dengue outbreaks	*			*	*				*	*	
The global distribution and burden of dengue	*			*	*		*		*	*	
Integrated vector management	*			*	*						*
EGI-Dengue	*			*	*						*
Total	11	1	5	10	10	1	7	2	5	3	4

Fuente: Pane et al. (2015).

La *Ilustración 21* nos muestra los diferentes mapas web donde se pueden visualizar la situación de la enfermedad en los territorios tanto de países de forma individual, como a nivel continental o global. Permiten desplegar datos como la cantidad de infectados, delimitar zonas de riesgo, y aplicar filtros a partir de varios parámetros, como edad, sexo, clasificación del tipo de dengue, entre otros.

*Ilustración 21. Mapas sobre la enfermedad del Dengue.*

Aplicación, organización o país	Descripción	URL
DengueMap	Mapa de ocurrencias de dengue a nivel mundial, junto con presencia o ausencia del virus.	<a href="http://www.healthmap.org/dengue/en/">http://www.healthmap.org/dengue/en/</a>
USGC-EEUU	Reporte de ocurrencias de dengue en los Estados Unidos de Norteamérica. Se diferencian dos mapas, uno para casos de dengue autóctono y otro para casos importados.	<a href="http://diseasemaps.usgs.gov/mapviewer/">http://diseasemaps.usgs.gov/mapviewer/</a>
Guatemala	Mapa de incidencia del dengue en las ciudades de Guatemala. Estadísticas anuales y otros datos.	<a href="http://epifichas.mspas.gob.gt/Descarga/Mapas/EpiVigila/IA/Muni/atlas.html">http://epifichas.mspas.gob.gt/Descarga/Mapas/EpiVigila/IA/Muni/atlas.html</a>
Fortaleza	Mapa de riesgo del dengue de la ciudad Fortaleza en Brasil. Permite realizar filtros por barrios específicos y otras variables de interés.	<a href="http://tc1.sms.fortaleza.ce.gov.br/simda/dengue/mapa">http://tc1.sms.fortaleza.ce.gov.br/simda/dengue/mapa</a>
Argentina	Mapa de riesgo del dengue en las ciudades de Argentina. Se muestran casos autóctonos y días de posible transmisión.	<a href="http://www.mapaeducativo.edu.ar/mapserver/aen/socioterritorial/dengue_riesgo/index.php">http://www.mapaeducativo.edu.ar/mapserver/aen/socioterritorial/dengue_riesgo/index.php</a>
DengueTrends	Aplicación de Google que provee estadísticas del dengue basadas en las búsquedas de sus usuarios alrededor del mundo.	<a href="http://www.google.org/denguetrends/">http://www.google.org/denguetrends/</a>
Epidemic and emerging disease alerts in the Pacific region	Mapa de alerta de enfermedades en la región del Pacífico. Se muestran las alertas y el estado en que se encuentran.	<a href="http://www.spc.int/phd/epidemics/">http://www.spc.int/phd/epidemics/</a>
OPS	Mapa interactivo de casos de dengue.	<a href="http://ais.paho.org/atlas/dengue/paneldengue1.html">http://ais.paho.org/atlas/dengue/paneldengue1.html</a>
	Resumen de enfermedades transmitidas por vectores, para el dengue se muestra la presencia o ausencia del virus en cada país.	<a href="http://ais.paho.org/php/viz/char_cd_vectorborn diseases.asp">http://ais.paho.org/php/viz/char_cd_vectorborn diseases.asp</a>
	Países o áreas donde se ha reportado la existencia de dengue.	<a href="http://apps.who.int/ithmap/">http://apps.who.int/ithmap/</a>
Brasil	Situación del dengue en las principales ciudades de Brasil.	<a href="http://www.dengue.org.br/dengue_mapas.html">http://www.dengue.org.br/dengue_mapas.html</a>

Fuente: 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, 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. Las metodologías tradicionales están enfocados a proyectos de software de gran escala, que requieren de un proceso de desarrollo en periodos de tiempo extenso, y equipos de trabajo que involucran a una gran cantidad de profesionales. Las metodologías de desarrollo ágil permiten 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 períodos cortos de tiempo, con equipos de trabajo pequeños, es ideal para proyectos de pequeña y mediana escala. Como ejemplo de metodologías ágiles más utilizadas podemos citar Scrum y Extreme Programming (XP). (Navarro Cadavid et. al, 2013).

La *Ilustración 22* presenta las diferencias entre las metodologías tradicionales y las metodologías ágiles.

*Ilustración 22. Comparación entre las metodologías de desarrollo de software tradicional y ágil*

Metodologías tradicionales	Metodologías ágiles
Predictivos	Adaptativos
Orientados a procesos	Orientados a personas
Proceso rígido	Proceso flexible
Se concibe como un proyecto	Un proyecto es subdividido en varios proyectos más pequeños
Poca comunicación con el cliente	Comunicación constante con el cliente
Entrega de software al finalizar el desarrollo	Entregas constantes de software
Documentación extensa	Poca documentación

Fuente: Navarro Cadavid et al. (2013).

Una de las primeras metodologías de desarrollo ágil es el XP o Extreme Programming, que se caracteriza por un enfoque en la relación entre las personas como un aspecto fundamental para el logro de los objetivos y conseguir un desarrollo de software exitoso. Se basa en una relación armoniosa de todos los implicados, como los desarrolladores, los encargados del proyecto y los clientes, manteniendo una comunicación fluida y constante. Esta metodología está destinado a proyectos con requisitos imprecisos y cambiantes, que necesiten una respuesta rápida a los cambios. Entre las características principales de esta metodología se encuentran las historias de usuario, utilizado como técnica para establecer los requisitos del software, consiste en tarjetas de papel donde el cliente escribe en forma breve y sencilla las características que debe poseer el software. A partir de estas historias los programadores deben trasladar al software estos requisitos y conseguir la funcionalidad deseada. Los roles que establece XP para

definir la función de cada integrante del equipo de trabajo serían los programadores, encargado de escribir el código y realizar las pruebas, el cliente, que escribe las historias de usuario, realiza también pruebas de funcionamiento para certificar su implementación y establece la prioridad de las historias a realizarse. El encargado de pruebas o tester ejecuta los testeos de manera más profunda, ayuda a los clientes a crear pruebas funcionales, informa los resultados a los demás miembros, y es responsable de las herramientas de soporte para pruebas, el encargado de seguimiento o tracker, es responsable de la retroalimentación de los avances y errores detectados en el trabajo del equipo, para introducir mejoras en los ciclos de las actividades, el entrenador o coach, es el máximo responsable de todo el proceso y debe establecer las guías para la correcta implementación de la metodología. El consultor es un agente externo al equipo especializado en algún tema específico que pueda aportar información útil para el desarrollo correcto del proyecto y por último, tenemos al gestor o big boss que cumple la función de nexo entre el cliente y los programadores, coordinando todas las actividades para conseguir un ambiente de trabajo ideal. Otra característica muy importante de XP son los ciclos de desarrollo consistente en iteraciones estructurado en una serie de pasos comenzando con el cliente que establece el valor de negocio, continúa con el programador que realiza las estimaciones de esfuerzo para alcanzar el valor de negocio propuesto, siguiendo con el cliente que decide cuales características tendrán las prioridades más altas y el tiempo disponible para su construcción, y finalmente el programador hace posible el valor de negocio establecido, con esto se reinicia todo el proceso de vuelta. Estas iteraciones son fundamentales para que el proyecto tenga un avance sostenido, ya que en cada iteración el producto va evolucionando, agregando mejoras y corrigiendo errores, dejando un aprendizaje tanto en el cliente como en los desarrolladores, hasta llegar a la meta propuesta. XP posee una serie de prácticas que se implementan durante el proceso de desarrollo, que se detallan a continuación:

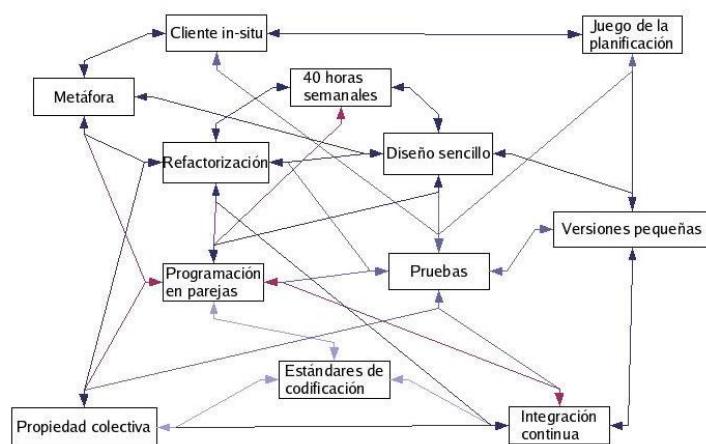
- El juego de la planificación: a partir de la comunicación entre el cliente y los desarrolladores, se establecen primeramente por parte de los programadores un cálculo estimado para la realización de las historias de usuario y los clientes toman la decisión del ámbito y cantidad de tiempo para entregar los resultados de cada iteración.
- Entregas pequeñas: Conseguir en periodos cortos de tiempo versiones del software ya con una funcionalidad operativa, si necesidad de estar completamente desarrollado, constituyendo de esta forma un resultado de valor para el cliente. Estas entregas no deben tener un periodo mayor de 3 meses.

- Metáfora: Es una representación que describe el funcionamiento teórico del sistema propuesto, creado por el cliente en conjunto con el equipo de programadores.
- Diseño simple: se debe buscar siempre la solución más simple que otorgue la funcionalidad requerida.
- Pruebas: son establecidas previamente por el cliente, y son constantemente aplicadas al código realizado, de acuerdo a las modificaciones introducidas.
- Refactorización: es una reestructuración constante del código para evitar la duplicación, ser más legible y comprensible, tener mayor flexibilidad para posteriores cambios y optimizar la simplificación de la estructura.
- Programación en parejas: la escritura de código se realiza con dos programadores trabajando conjuntamente, resultando en un mejor aprovechamiento de los recursos humanos, como también una menor tasa de errores, diseño optimizado, mayor satisfacción de los desarrolladores, entre otros.
- Propiedad colectiva del Código: El código es accesible y modifiable por todos los programadores del equipo, en cualquier parte, esto aumenta la implicación y el sentido de responsabilidad de todo el equipo.
- Integración continua: Cada parte del código producido una vez listo es integrado al sistema, pudiendo ser modificado y vuelto a integrar varias veces en un mismo día
- 40 horas semanales: se establecen 40 horas de trabajo a la semana como tope máximo. Se toleran horas extras solamente en situaciones muy particulares. Si existen demasiadas horas extras puede deberse a algún problema que debe atenderse y solucionarse rápidamente. El trabajo extra puede derivar en la desmotivación del equipo.
- Cliente in situ: El equipo desarrollador debe tener contacto directo con el cliente, el cual debe estar presente físicamente en el mismo lugar de trabajo. Es el principal factor de éxito de la metodología XP. La comunicación oral y constante del cliente con los programadores tiene una alta efectividad para conducir a obtener un mayor valor de negocio y poder resolver los problemas y dudas de la manera más rápida posible.

- Estándares de programación: XP pone énfasis a establecer reglas y normas claras en la escritura de código, estableciendo el código como un medio de comunicación entre los programadores, por lo tanto, es indispensable el establecimiento de estándares para mantener el código legible.

Estas prácticas de XP si son utilizadas en forma conjunta y equilibrada, llevan a los proyectos a alcanzar las metas propuestas y tener éxito. En la Ilustración 23 podemos ver las prácticas XP interconectadas entre sí (Canós, Letelier & Panadés, 2003).

*Ilustración 23. Conjunto de prácticas XP relacionadas entre sí*



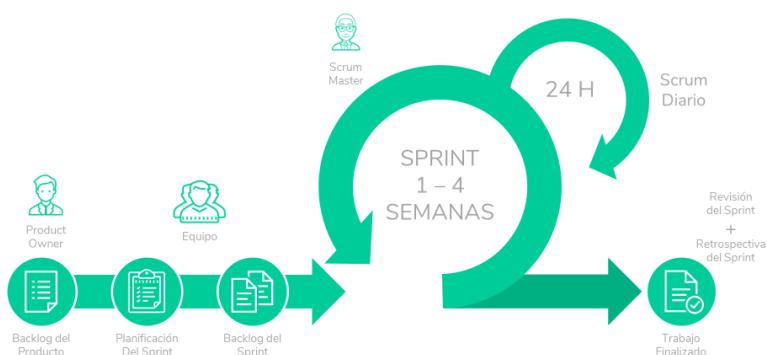
Fuente: Canos et al. (2003).

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 último tenemos el equipo de desarrollo, que se encarga de realizar todas las labores requeridas para obtener el producto final. En cuanto a los elementos principales que componen la metodología Scrum, se destacan los eventos como el Sprint, definido como un periodo específico 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 (Trigás Gallego, 2012).

La *Ilustración 23* muestra los componentes principales de la metodología SCRUM, como los artefactos, eventos y roles de los integrantes.

*Ilustración 24. Esquema de Roles, elementos y eventos que componen la Metodología ágil SCRUM.*



Fuente: Sarango Yunga (2020).

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. Así 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, 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 Hypertext Transfer Protocol. Existen dos especificaciones de APIs web más utilizadas actualmente, denominadas SOAP y REST. Las APIs basadas en SOAP, del inglés “Simple Object Access 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 Transfer”, que traducido al español sería “Transferencia de Estado Representacional”, se resalta que puede implementar como formato de trasferencia de datos JSON, del inglés “JavaScript 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. 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 (Sala, 2003).

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

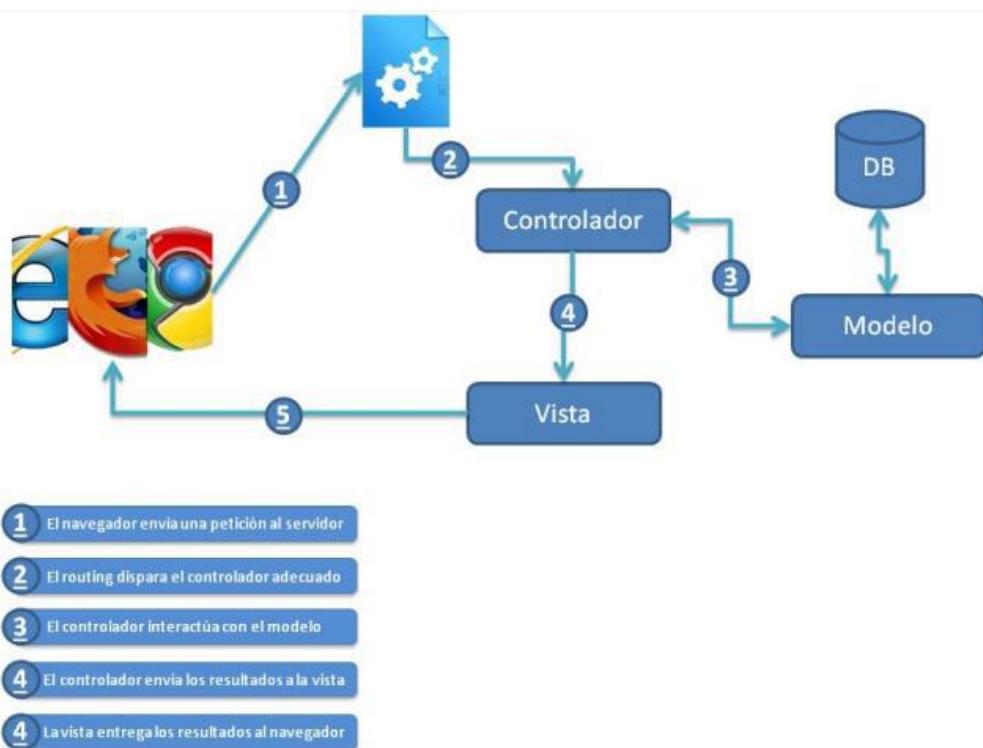
comunidad de desarrolladores se ha consolidado como referencia en la elección de los programadores (Díaz & Banchoff Tzancoff, 2000).

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 (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 (*Ilustración 24*) se basa en tres componentes que interactúan entre sí 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

comprende 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. Además posee utilidades adicionales incorporadas que facilitan el desarrollo del software, como ejemplo podemos nombrar a Eloquent, una librería encargada del Mapeo Objeto Relacional (ORM, de las siglas en inglés Object Relational Mapping), que facilita la comunicación y la conexión con las distintas bases de datos. También dispone de una gran cantidad de paquetes de librerías que se pueden agregar al proyecto para desarrollar nuevas funciones e incorporar características de una manera fácil y rápida. 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).

*Ilustración 25. Arquitectura de software Modelo Vista Controlador.*



Fuente: 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 SGDB 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 SGDB sería no garantizar una integridad completa de los datos. Como ejemplo podemos mencionar a Mongo DB, Redis, Cassandra. Los SGDB 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 SGDB 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 SGDB 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 SGDB 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.

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.

## **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 de investigación es no experimental cualitativo, ya que no se realizan manipulaciones deliberadas de las variables recolectadas, en este caso se aplican sobre los datos recogidos las prácticas, métodos y procesos ya existentes que ya fueron comprobados y validados anteriormente en otros estudios e investigaciones. Se tiene un enfoque cualitativo porque se busca la verificación del funcionamiento correcto y comportamiento deseado de la totalidad de las funcionalidades establecidas (Hernández Sampieri, Fernández Collado & Baptista Lucio, 2014).

### **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

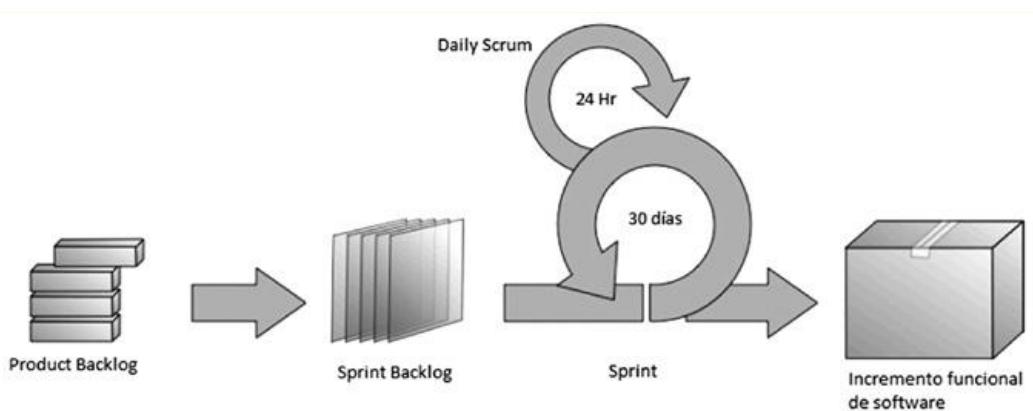
Se realizó una revisión bibliográfica con la finalidad de establecer las herramientas de desarrollo más adecuadas al software proyectado y a partir de la utilización de una metodología de desarrollo de software ágil denominado SCRUM se aplicaron las herramientas para definir las funcionalidades y requerimientos 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 26*).

*Ilustración 26. 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 se 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

## 4.7. Validación de los datos

Siguiendo los lineamientos de la metodología SCRUM, las pruebas de funcionamiento de las funcionalidades se aplicaron durante todo el proceso de desarrollo del proyecto, de forma constante de acuerdo a los avances concretados, de forma práctica y ágil. Por medio de testeos unitarios se verificaron el comportamiento deseado de las características del software, se utilizaron formularios en las cuales se registraban los detalles y los puntos analizados en cada prueba (*Ilustración 27*).

*Ilustración 27. Formulario de pruebas de funcionalidades del software.*

FORMULARIO DE TESTEO DE FUNCIONALIDAD		
Responsable:		
Fecha:		
Formulario Nº:		
Modulo:	API REST Principal	
Funcionalidad:	Registro de personas infectadas	
Procesos	SI	NO
Crea registro		
Modifica registro		
Elimina registro		
Valida correctamente todos los campos del registro		
Genera el registro de auditoria		
Genera la respuesta de confirmación de cada proceso		
Observaciones:		

Fuente: Elaboración propia.

## 5. RESULTADOS Y DISCUSIÓN

### 5.1. 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. De esta manera Scrum se vuelve la metodología adecuada para afrontar estos tipos de proyectos.

### 5.2. Establecimiento del modelo de negocio del módulo.

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. Estos, integrados al proceso provisto por Scrum, compondrán las funcionalidades requeridas por la metodología.

### 5.3. Definición del entorno tecnológico.

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 (versión 7.4): lenguaje de programación de código abierto con enfoque al entorno web

Laravel (versión 7.24): es un entorno de desarrollo para crear aplicaciones y servicios web, utilizando el lenguaje PHP

Composer (versión 1.10.7): Es un administrador de librerías y dependencias para PHP, utilizado en la instalación de proyectos realizados en el entorno de desarrollo Laravel.

PostgreSQL (versión 12.4): 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.

#### 5.4. 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 módulo API se establecieron cuatro columnas, una columna destinada para el product backlog, con las tareas pendientes a realizarse, otra columna representaba las tareas establecidas para el Sprint, una columna para las tareas en proceso de realización y por último una columna con las tareas finalizadas (*Ilustración 28*).

Ilustración 28. Tablero Trello con las tareas del product backlog del módulo API.

Columna	Tareas
Lista de tareas	<ul style="list-style-type: none"> <li>Programación de ABM reporte de casos de dengue (vencimiento: 22 de oct.)</li> <li>Programación de ABM reportes ciudadanos (vencimiento: 22 de oct.)</li> <li>Programación de características de seguridad (vencimiento: 22 de oct.)</li> <li>Pruebas de funcionamiento general (vencimiento: 19 de nov.)</li> <li>Documentación del proyecto (vencimiento: 19 de nov.)</li> </ul>
Sprint 1	+ Añada una tarjeta
En proceso	<ul style="list-style-type: none"> <li>Programación de ABM Usuarios (vencimiento: 24 de sep.)</li> </ul>
Hecho	<ul style="list-style-type: none"> <li>Programación de ABM Entidad (vencimiento: 24 de sep.)</li> <li>Programación de ABM roles (vencimiento: 24 de sep.)</li> <li>Configuración básica del proyecto (vencimiento: 24 de sep.)</li> </ul>

Fuente: Elaboración propia.

## 5.5. 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 29*).

Ilustración 29. Panel de control de la aplicación XAMPP.

Service	Module	PID(s)	Port(s)	Actions
	Apache			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>
	MySQL			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>
	FileZilla			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>
	Mercury			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>
	Tomcat			<button>Start</button> <button>Admin</button> <button>Config</button> <button>Logs</button>

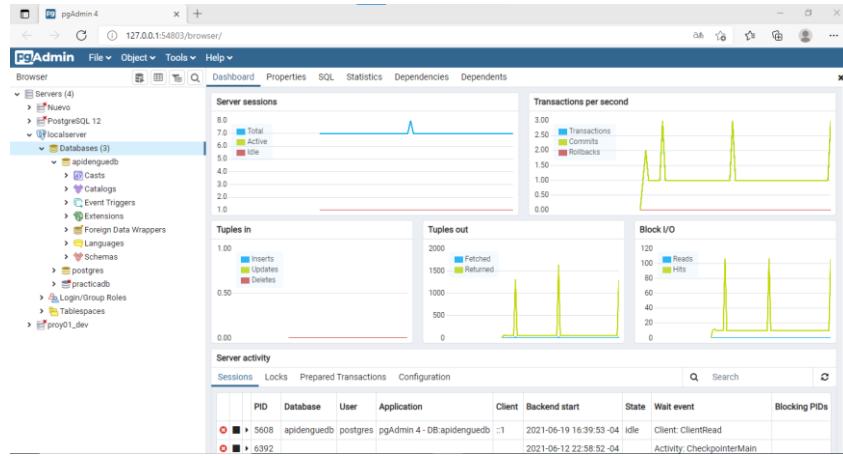
16:26:16 [main] there will be a security dialogue or things will break! So think about running this application with administrator rights!
   
 16:26:16 [main] XAMPP Installation Directory: "c:\xampp\"
   
 16:26:16 [main] Checking for prerequisites
   
 16:26:21 [main] All prerequisites found
   
 16:26:21 [main] Initializing Modules
   
 16:26:21 [main] Starting Check-Timer
   
 16:26:21 [main] Control Panel Ready

Fuente: elaboración propia

## 5.6. 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 30).

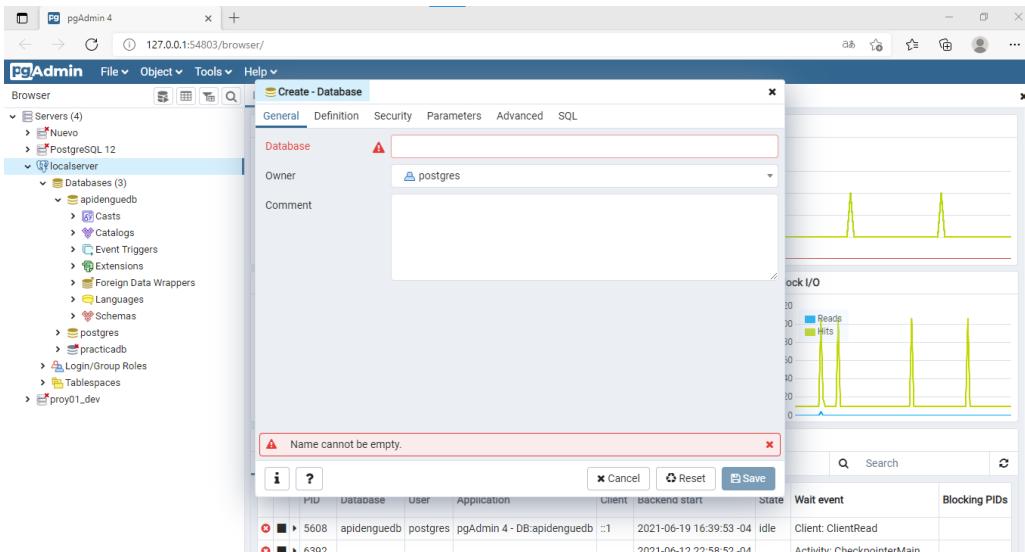
Ilustración 30. Panel principal de PgAdmin.



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 31*).

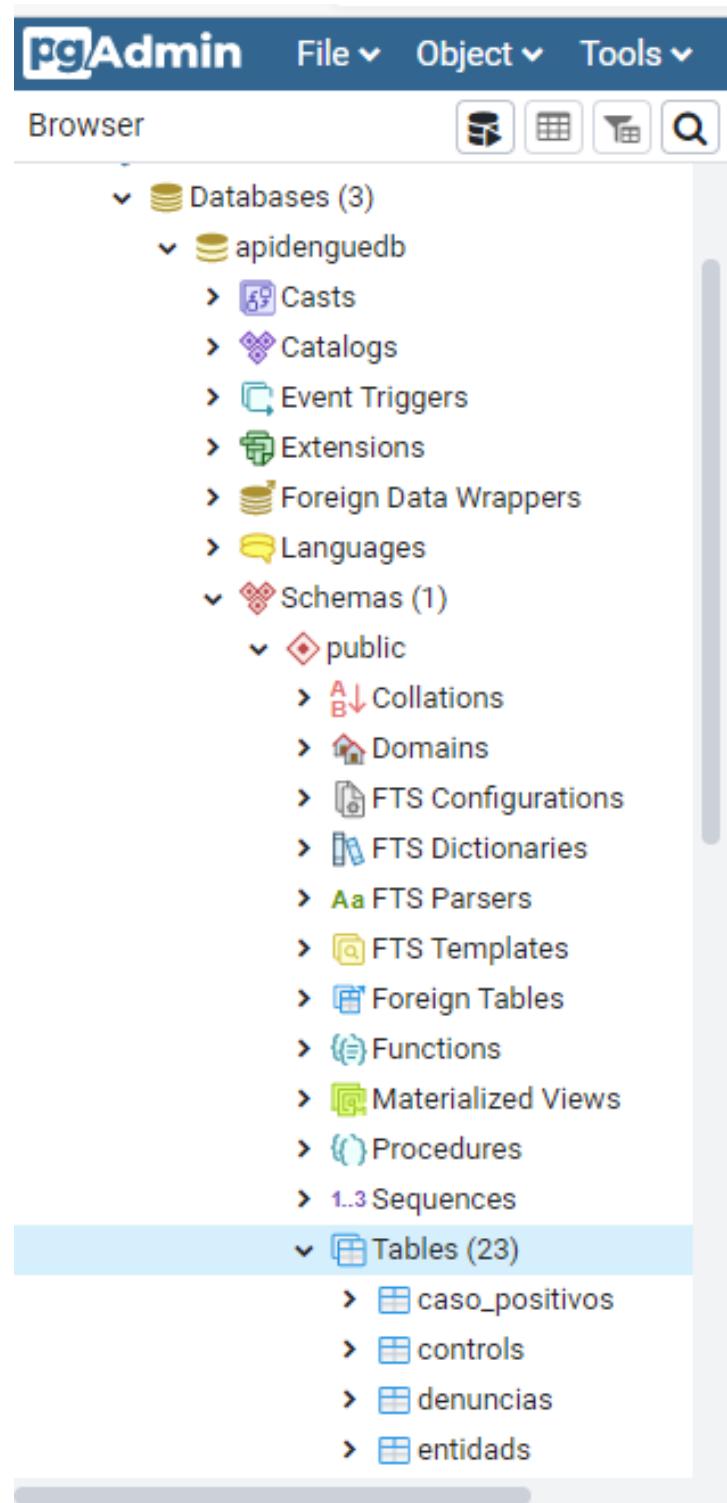
Ilustración 31. Menú de PgAdmin para la creación de una base de datos.



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 32*).

*Ilustración 32. Visualización de la base de datos creada para el proyecto*

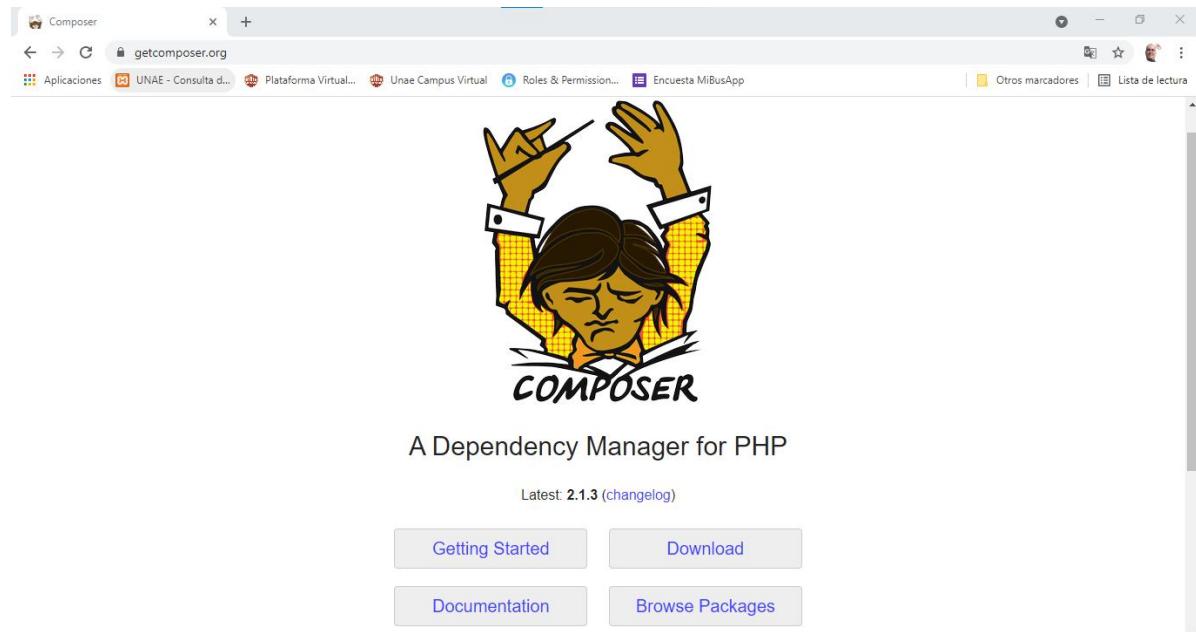


Fuente: Elaboración propia

## 5.7. Instalación de Composer

Como requisito 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 33*).

*Ilustración 33. Página web del gestor de dependencias para PHP Composer.*



Fuente: Elaboración propia.

## 5.8. Creación del proyecto Laravel

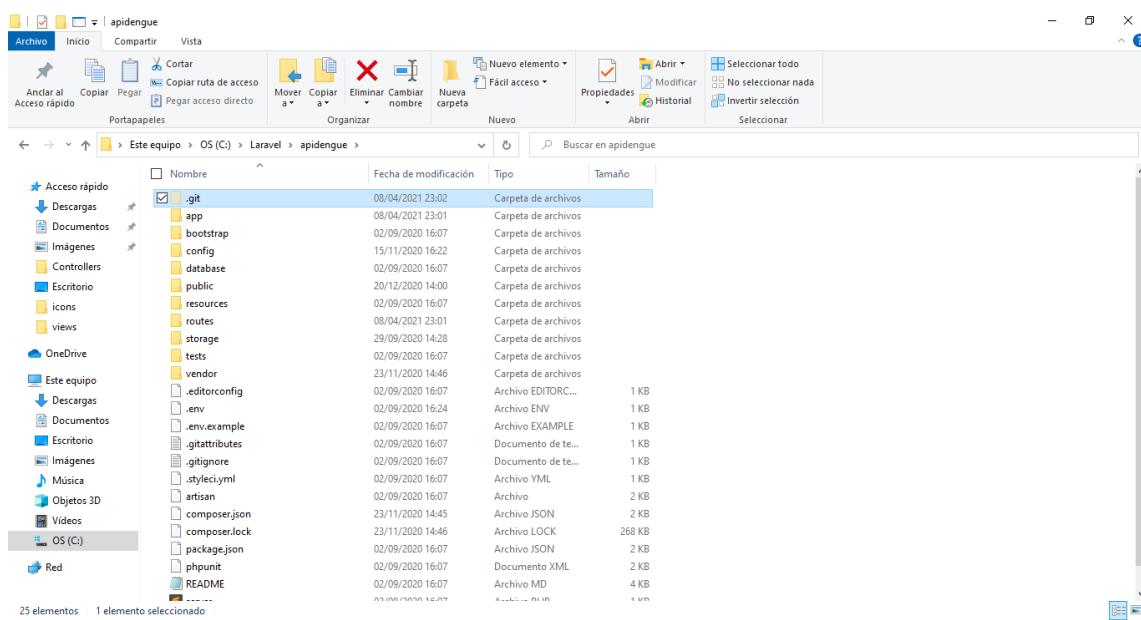
Para la creación del proyecto Laravel se cumplieron con los requisitos necesarios, como la instalación del gestor de dependencias Composer, y de las librerías del lenguaje PHP, así como la creación de una base de datos destinado para el módulo.

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 (*Ilustración 34*)

Ilustración 34. Directorio del proyecto Laravel



Fuente: Elaboración propia.

## 5.9. 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 35*).

Ilustración 35. Configuración del archivo “.env” para establecer la conexión con la base de datos

```

.env — C:\xampp\htdocs\apidengue — Atom
File Edit View Selection Find Packages Help
Project .env
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:izxtYL3ySCM20P4spGSDASArmpvDRfhFx5nrbcng0=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=pgsql
10 DB_HOST=127.0.0.1
11 DB_PORT=5432
12 DB_DATABASE=apidenguedb
13 DB_USERNAME=postgres
14 DB_PASSWORD=12345
15
16 BROADCAST_DRIVER=log
17 CACHE_DRIVER=file
18 QUEUE_CONNECTION=sync
19 SESSION_DRIVER=file
20 SESSION_LIFETIME=120
21
22 REDIS_HOST=127.0.0.1
23 REDIS_PASSWORD=null
24 REDIS_PORT=6379
25
26 MAIL_MAILER=smtp
27 MAIL_HOST=smtp.mailtrap.io
28 MAIL_PORT=2525
29 MAIL_USERNAME=null
30 MAIL_PASSWORD=null

```

Fuente: elaboración propia

La configuración tiene los siguientes parámetros:

- Pgsql: establece que la base de datos es postgresSQL
- 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 36*).

*Ilustración 36. Configuración del archivo “.env” para la conexión con la base de datos.*

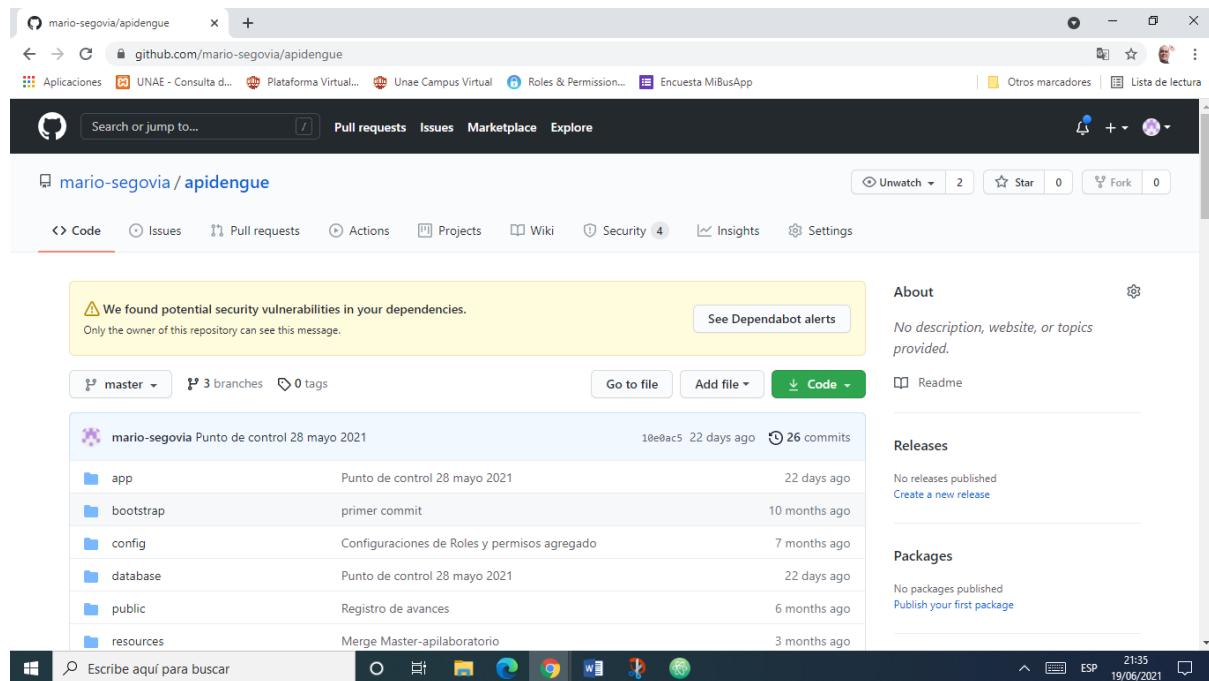
```
DB_CONNECTION=pgsql
DB_HOST=127.0.0.1
DB_PORT=5432
DB_DATABASE=apidenguedb
DB_USERNAME=postgres
DB_PASSWORD=12345
```

Fuente: Elaboración propia.

#### 5.10. Creación del 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 37*).

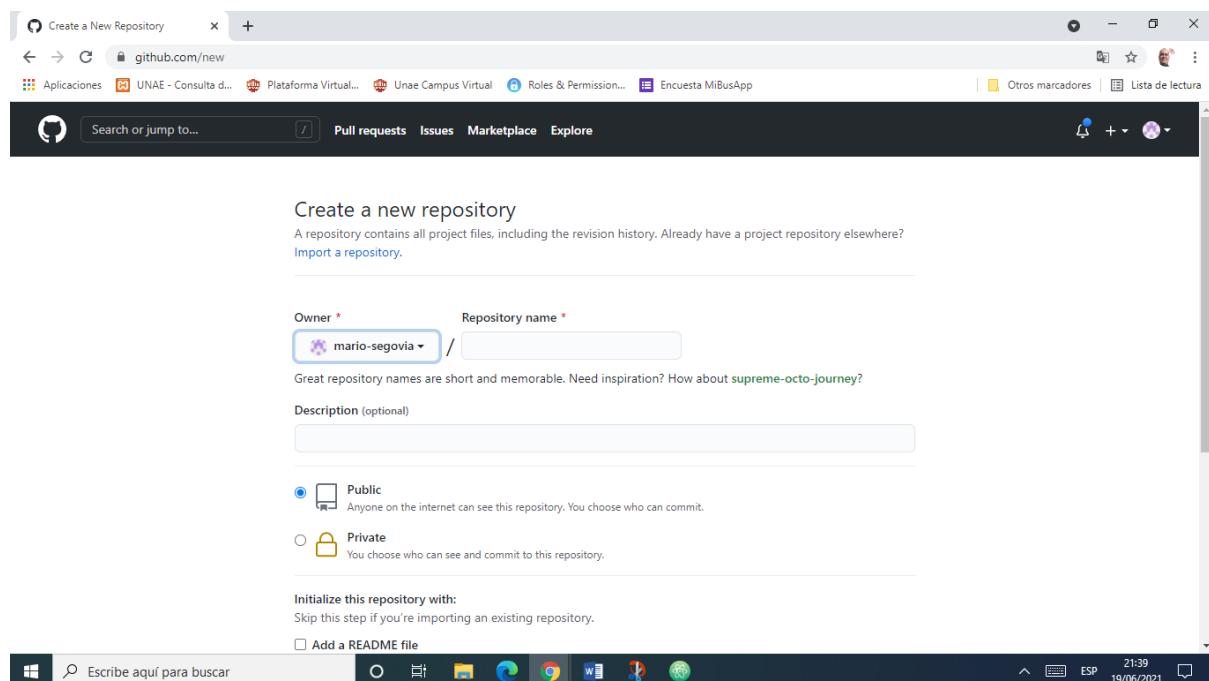
*Ilustración 37. Repositorio del módulo API en Github.*



Fuente: Elaboración propia

Los pasos realizados fueron primeramente la creación de un nuevo repositorio en la página web de Github (*Ilustración 38*).

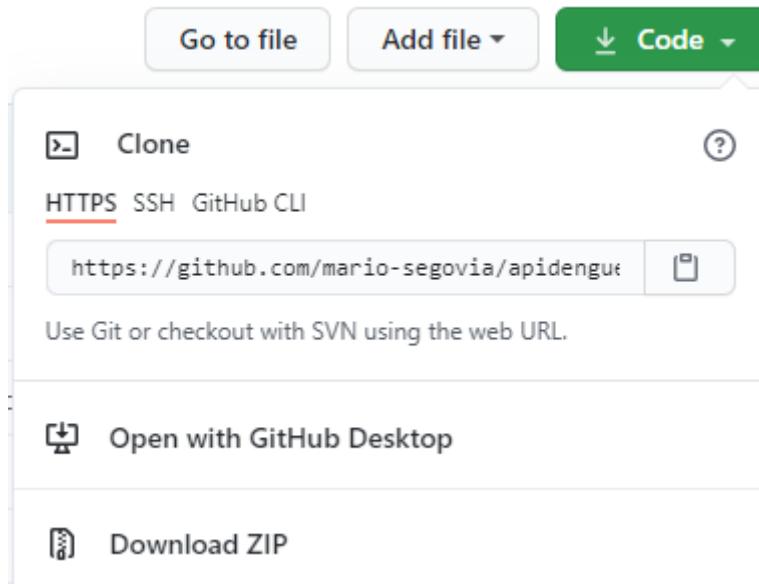
*Ilustración 38. Formulario para la creación de un repositorio nuevo en Github.*



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 39*).

*Ilustración 39. 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 modificaciones 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.

## 5.11. 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, 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.

#### 5.11.1. 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, así como los procedimientos y las relaciones que establecerá con los otros modelos.

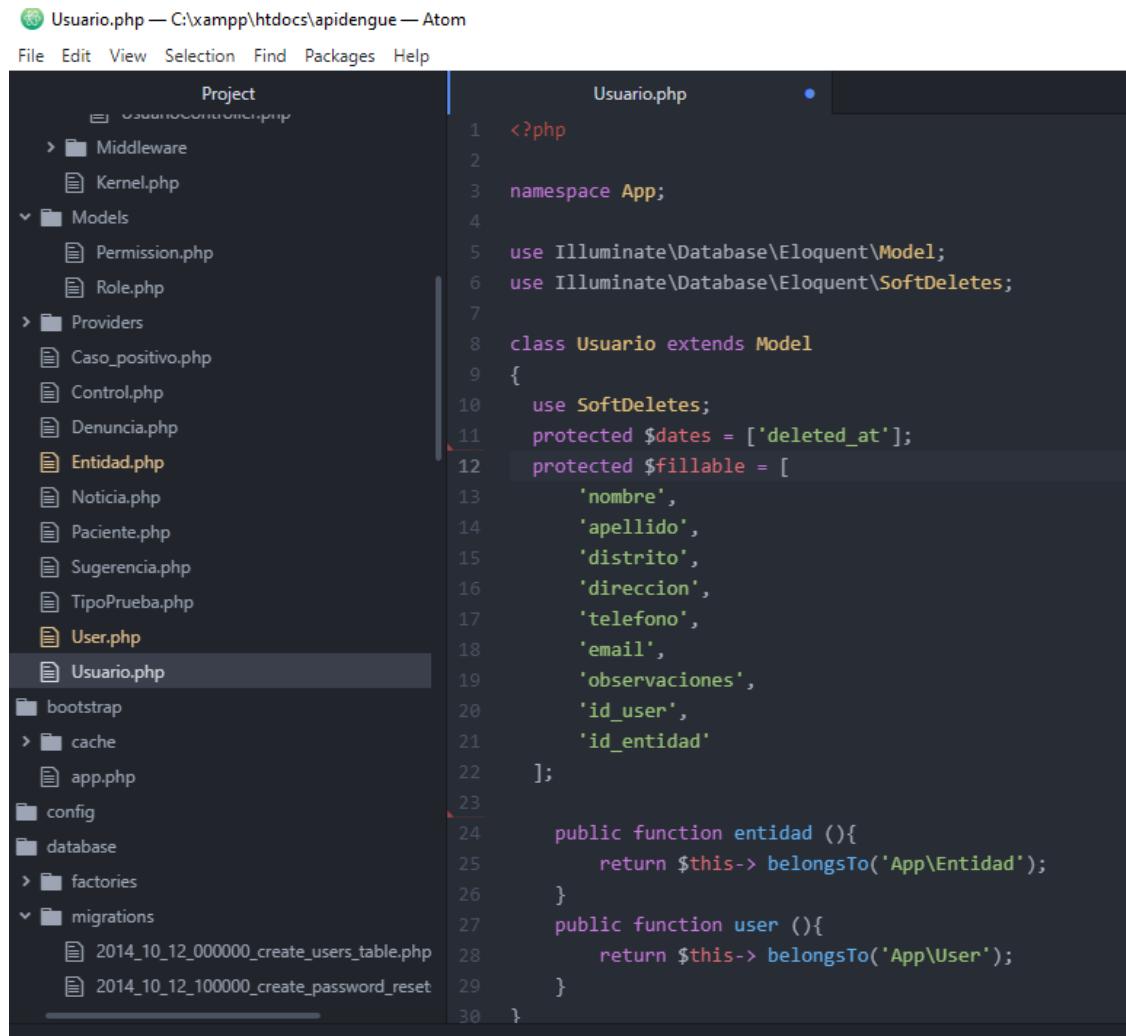
En este proyecto para la creación de los Modelos se utilizó el siguiente comando:

- `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 40* podemos ver un ejemplo de un Modelo en Laravel.

*Ilustración 40. Ejemplo de Modelo de Laravel.*



The screenshot shows the Atom code editor interface. The title bar says "Usuario.php — C:\xampp\htdocs\apidengue — Atom". The menu bar includes File, Edit, View, Selection, Find, Packages, and Help. On the left is a sidebar titled "Project" showing the directory structure of a Laravel application:

- Project
- ↳ Middleware
- Kernel.php
- Models
  - Permission.php
  - Role.php
- Providers
  - Caso\_positivo.php
  - Control.php
  - Denuncia.php
  - Entidad.php
  - Noticia.php
  - Paciente.php
  - Sugerencia.php
  - TipoPrueba.php
  - User.php
  - Usuario.php
- bootstrap
- cache
- app.php
- config
- database
- factories
- migrations
  - 2014\_10\_12\_000000\_create\_users\_table.php
  - 2014\_10\_12\_100000\_create\_password\_reset

The main editor area displays the "Usuario.php" file content:

```
<?php  
namespace App;  
  
use Illuminate\Database\Eloquent\Model;  
use Illuminate\Database\Eloquent\SoftDeletes;  
  
class Usuario extends Model  
{  
    use SoftDeletes;  
    protected $dates = ['deleted_at'];  
    protected $fillable = [  
        'nombre',  
        'apellido',  
        'distrito',  
        'direccion',  
        'telefono',  
        'email',  
        'observaciones',  
        'id_user',  
        'id_entidad'  
    ];  
  
    public function entidad (){  
        return $this-> belongsTo('App\Entidad');  
    }  
    public function user (){  
        return $this-> belongsTo('App\User');  
    }  
}
```

Fuente: Elaboración propia.

La *Ilustración 41* nos muestra la configuración de los campos de datos que componen el Modelo.

*Ilustración 41. Atributos o campos de datos que componen el Modelo.*

```
protected $fillable = [  
    'nombre',  
    'apellido',  
    'distrito',  
    'direccion',  
    'telefono',  
    'email',  
    'observaciones',  
    'id_user',  
    'id_entidad'  
];
```

Fuente: Elaboración propia.

En la *Ilustración 42* podemos visualizar las relaciones del Modelo con los demás Modelos, con los cuales se crean los vínculos para interactuar entre ellos.

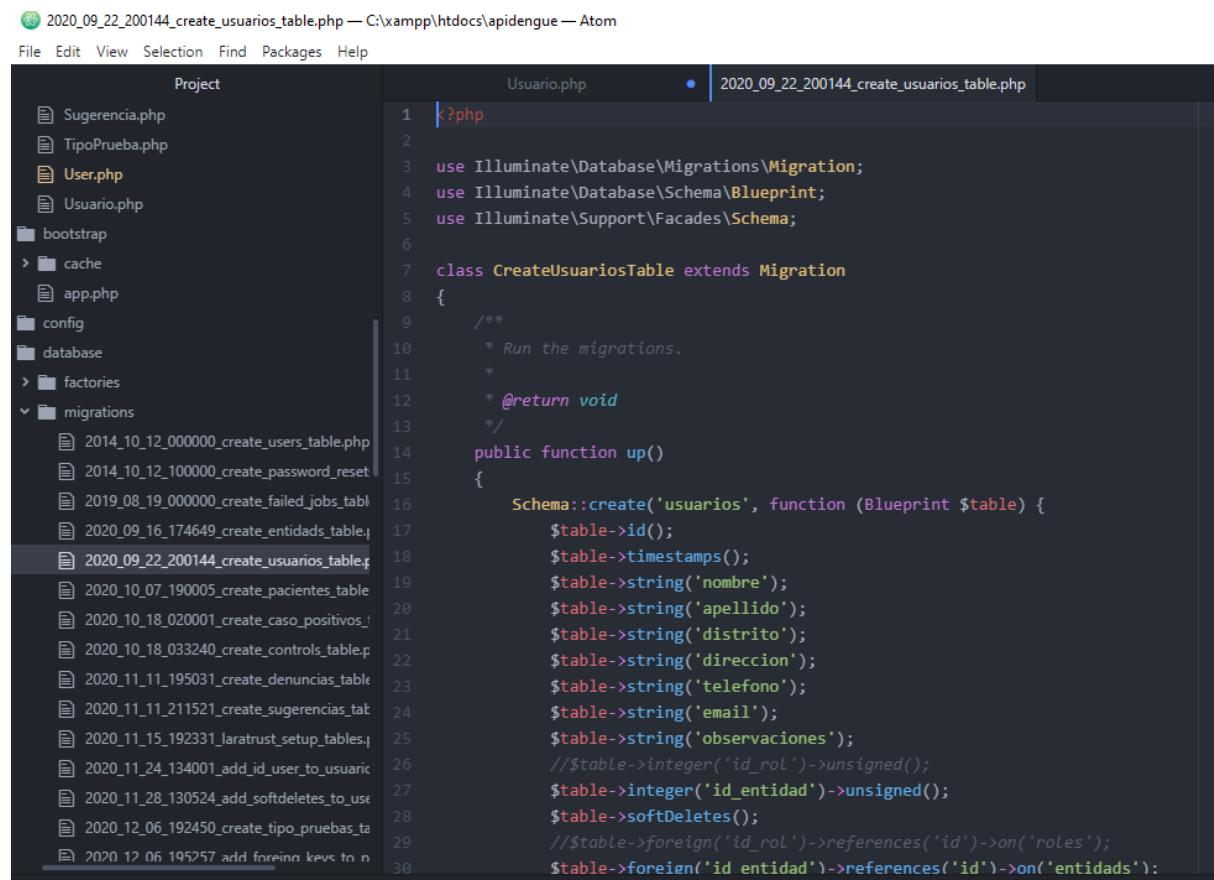
*Ilustración 42. Relaciones de dependencia con otros modelos*

```
public function entidad (){
    return $this-> belongsTo('App\Entidad');
}
public function user (){
    return $this-> belongsTo('App\User');
}
```

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 43*).

*Ilustración 43. Archivo de migración.*



The screenshot shows a screenshot of the Atom code editor. The title bar says "2020\_09\_22\_200144\_create\_usuarios\_table.php — C:\xampp\htdocs\apidengue — Atom". The menu bar includes File, Edit, View, Selection, Find, Packages, and Help. On the left is a project tree with files like Sugerencia.php, TipoPrueba.php, User.php, Usuario.php, bootstrap, cache, app.php, config, database, factories, migrations (containing 2014\_10\_12\_000000\_create\_users\_table.php, 2014\_10\_12\_100000\_create\_password\_reset.php, 2019\_08\_19\_000000\_create\_failed\_jobs\_table.php, 2020\_09\_16\_174649\_create\_entidades\_table.php, 2020\_09\_22\_200144\_create\_usuarios\_table.php, 2020\_10\_07\_190005\_create\_pacientes\_table.php, 2020\_10\_18\_020001\_create\_caso\_positivos\_table.php, 2020\_10\_18\_033240\_create\_controls\_table.php, 2020\_11\_11\_195031\_create\_denuncias\_table.php, 2020\_11\_11\_211521\_create\_sugerencias\_table.php, 2020\_11\_15\_192331\_jarastrust\_setup\_tables.php, 2020\_11\_24\_134001\_add\_id\_user\_to\_usuario.php, 2020\_11\_28\_130524\_add\_softdeletes\_to\_usuario.php, 2020\_12\_06\_192450\_create\_tipo\_pruebas\_table.php, 2020\_12\_06\_195257\_add\_foreign\_keys\_to\_n.php), and 2020\_12\_06\_195257\_add\_foreign\_keys\_to\_n.php. The main editor area shows the code for "CreateUsuariosTable" migration:

```
?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
class CreateUsuariosTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('usuarios', function (Blueprint $table) {
            $table->id();
            $table->timestamps();
            $table->string('nombre');
            $table->string('apellido');
            $table->string('distrito');
            $table->string('direccion');
            $table->string('telefono');
            $table->string('email');
            $table->string('observaciones');
            // $table->integer('id_rol')->unsigned();
            $table->integer('id_entidad')->unsigned();
            $table->softDeletes();
            // $table->foreign('id_rol')->references('id')->on('roles');
            $table->foreign('id_entidad')->references('id')->on('entidades');
        });
    }
    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('usuarios');
    }
}
```

Fuente: Elaboración propia.

En la migración se deben especificar todos los campos de datos que contendrá la tabla en la base de datos, así como el tipo de datos, como texto, numero, booleano, relaciones con otras tablas, entre otros (*Ilustración 44*).

*Ilustración 44. Campos de datos de la migración.*

```
Schema::create('usuarios', function (Blueprint $table) {
    $table->id();
    $table->timestamps();
    $table->string('nombre');
    $table->string('apellido');
    $table->string('distrito');
    $table->string('direccion');
    $table->string('telefono');
    $table->string('email');
    $table->string('observaciones');
    // $table->integer('id_rol')->unsigned();
    $table->integer('id_entidad')->unsigned();
    $table->softDeletes();
```

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.

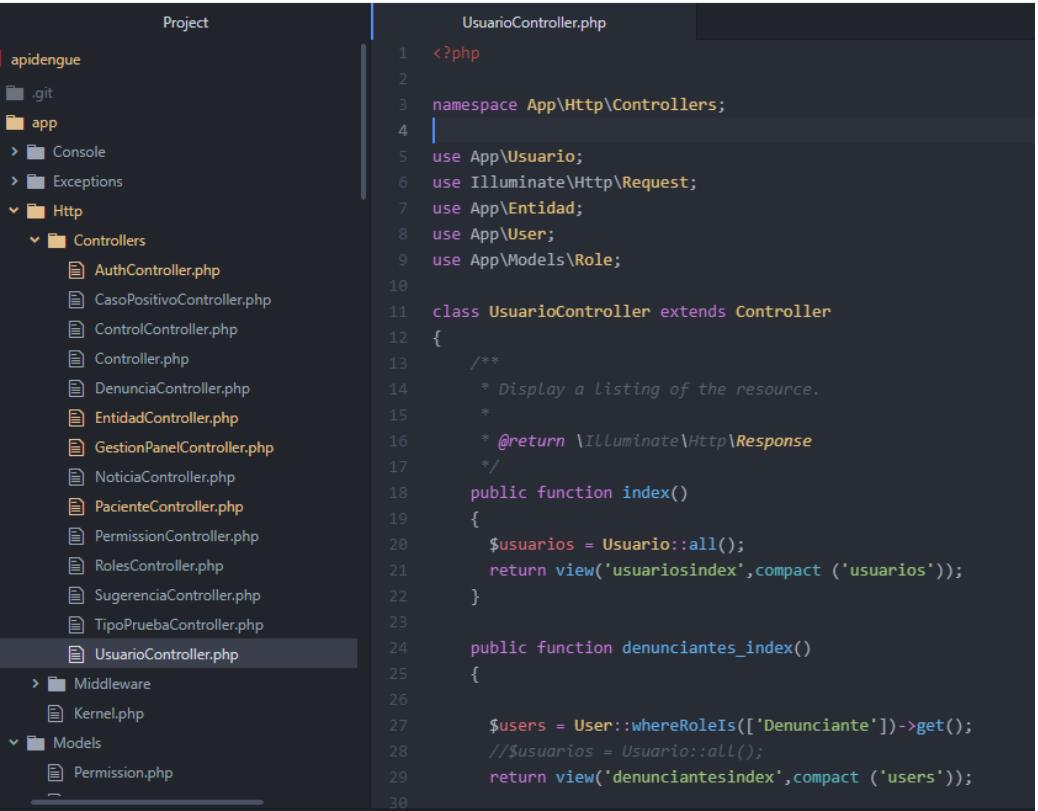
### 5.11.2. Programación de Controladores en el proyecto Laravel.

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 45*).

*Ilustración 45. Controlador de Laravel.*



The screenshot shows the Atom code editor interface. On the left is a sidebar titled "Project" showing the directory structure of the "apidengue" project. It includes ".git", "app" (which contains "Console", "Exceptions", and "Http" folders), "Http" (which contains "Controllers" folder with files like AuthController.php, CasoPositivoController.php, etc.), "Middleware", "Kernel.php", and "Models" (with Permission.php). The main editor area on the right displays the "UsuarioController.php" file. The code is as follows:

```
<?php  
namespace App\Http\Controllers;  
  
use App\Usuario;  
use Illuminate\Http\Request;  
use App\Entidad;  
use App\User;  
use App\Models\Role;  
  
class UsuarioController extends Controller  
{  
    /**  
     * Display a listing of the resource.  
     *  
     * @return \Illuminate\Http\Response  
     */  
    public function index()  
    {  
        $usuarios = Usuario::all();  
        return view('usuariosindex',compact ('usuarios'));  
    }  
  
    public function denunciantes_index()  
    {  
        $users = User::whereRoleIs(['Denunciante'])->get();  
        // $usuarios = Usuario::all();  
        return view('denunciantesindex',compact ('users'));  
    }  
}
```

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 46*).

*Ilustración 46. Método index de un controlador.*

```
/*
public function index()
{
    $usuarios = Usuario::all();
    return view('usuariosindex',compact ('usuarios'));
}
```

Fuente: Elaboración propia.

- show: método para mostrar un registro específico de un modelo dado (*Ilustración 47*).

*Ilustración 47. Método show de un controlador*

```
public function show($id)
{
    $paciente = Paciente::find($id);
    return $paciente;
}
```

Fuente: Elaboración propia.

- store: método utilizado para guardar un nuevo registro de un modelo específico (*Ilustración 48*).

*Ilustración 48. Método store de un controlador*

```
public function store(Request $request)
{
    $paciente = Paciente::create($request->all());
    return response(['mensaje' => 'Paciente creado exitosamente']);
}
```

Fuente: Elaboración propia.

- update: método para guardar los cambios de un registro del modelo (*Ilustración 49*).

*Ilustración 49. Método update de un controlador*

```
/*
public function update(Request $request, $id)
{
    $paciente = Paciente::find($id);
    $paciente->update($request->all());
    return response(['mensaje' => 'Paciente actualizado exitosamente']);
}
```

Fuente: Elaboración propia.

- destroy: método para eliminar un registro del modelo (*Ilustración 50*).

*Ilustración 50. Método destroy de un controlador.*

```
public function destroy( $id )
{
    $paciente = Paciente::find($id);
    $paciente->delete();
    return response(['mensaje' => 'Paciente eliminado exitosamente']);
}
```

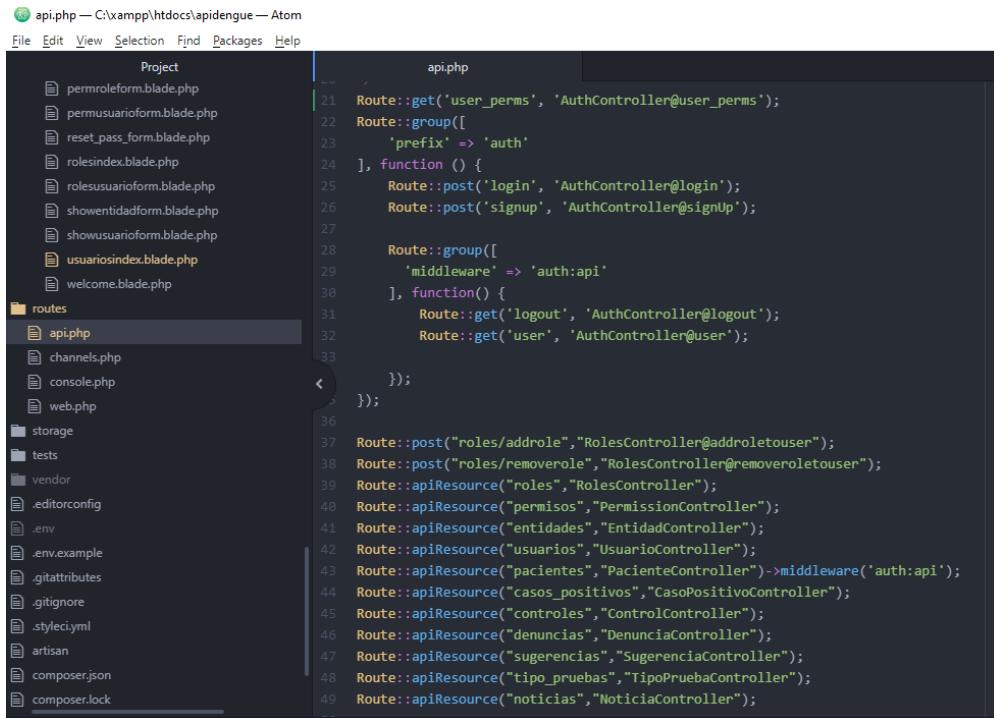
Fuente: Elaboración propia.

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

#### 5.11.3. 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 51*).

*Ilustración 51. Archivo “api”, para la configuración de rutas del módulo.*



The screenshot shows the Atom code editor with the file 'api.php' open. The file is located in the 'routes' directory of a project. The code defines several API routes using the Route class. It includes route groups for authentication and roles, and specific routes for user permissions, roles, and various entities like pacientes, casos\_positivos, controles, denuncias, sugerencias, tipo\_pruebas, and noticias. The code also demonstrates middleware usage.

```
api.php — C:\xampp\htdocs\apidengue — Atom
File Edit View Selection Find Packages Help
Project
  permoleform.blade.php
  permusuariiform.blade.php
  reset_pass_form.blade.php
  rolesindex.blade.php
  rolesusuariiform.blade.php
  showentidadform.blade.php
  showusuariiform.blade.php
  usuariosindex.blade.php
  welcome.blade.php
routes
  api.php
  channels.php
  console.php
  web.php
storage
tests
vendor
.editorconfig
.env
.env.example
.gitattributes
.gitignore
.styleci.yml
artisan
composer.json
composer.lock
api.php
Route::get('user_perms', 'AuthController@user_perms');
Route::group([
    'prefix' => 'auth'
], function () {
    Route::post('login', 'AuthController@login');
    Route::post('signup', 'AuthController@signUp');

    Route::group([
        'middleware' => 'auth:api'
    ], function() {
        Route::get('logout', 'AuthController@logout');
        Route::get('user', 'AuthController@user');
    });
});
Route::post("roles/addrole", "RolesController@addroletouser");
Route::post("roles/removerole", "RolesController@removeroletouser");
Route::apiResource("roles", "RolesController");
Route::apiResource("permisos", "PermissionController");
Route::apiResource("entidades", "EntidadController");
Route::apiResource("usuarios", "UsuarioController");
Route::apiResource("pacientes", "PacienteController")->middleware('auth:api');
Route::apiResource("casos_positivos", "CasoPositivoController");
Route::apiResource("controles", "ControlController");
Route::apiResource("denuncias", "DenunciaController");
Route::apiResource("sugerencias", "SugerenciaController");
Route::apiResource("tipo_pruebas", "TipoPruebaController");
Route::apiResource("noticias", "NoticiaController");
```

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 52*).

*Ilustración 52. Rutas de acceso a los recursos del módulo.*

```
Route::post("roles/addrole","RolesController@addroletouser");
Route::post("roles/removerole","RolesController@removeroletouser");
Route::apiResource("roles","RolesController");
Route::apiResource("permisos","PermissionController");
Route::apiResource("entidades","EntidadController");
Route::apiResource("usuarios","UsuarioController");
Route::apiResource("pacientes","PacienteController")->middleware('auth:api');
Route::apiResource("casos_positivos","CasoPositivoController");
Route::apiResource("controles","ControlController");
Route::apiResource("denuncias","DenunciaController");
Route::apiResource("sugerencias","SugerenciaController");
Route::apiResource("tipo_pruebas","TipoPruebaController");
Route::apiResource("noticias","NoticiaController");
```

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.

## 5.12. 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 53*).

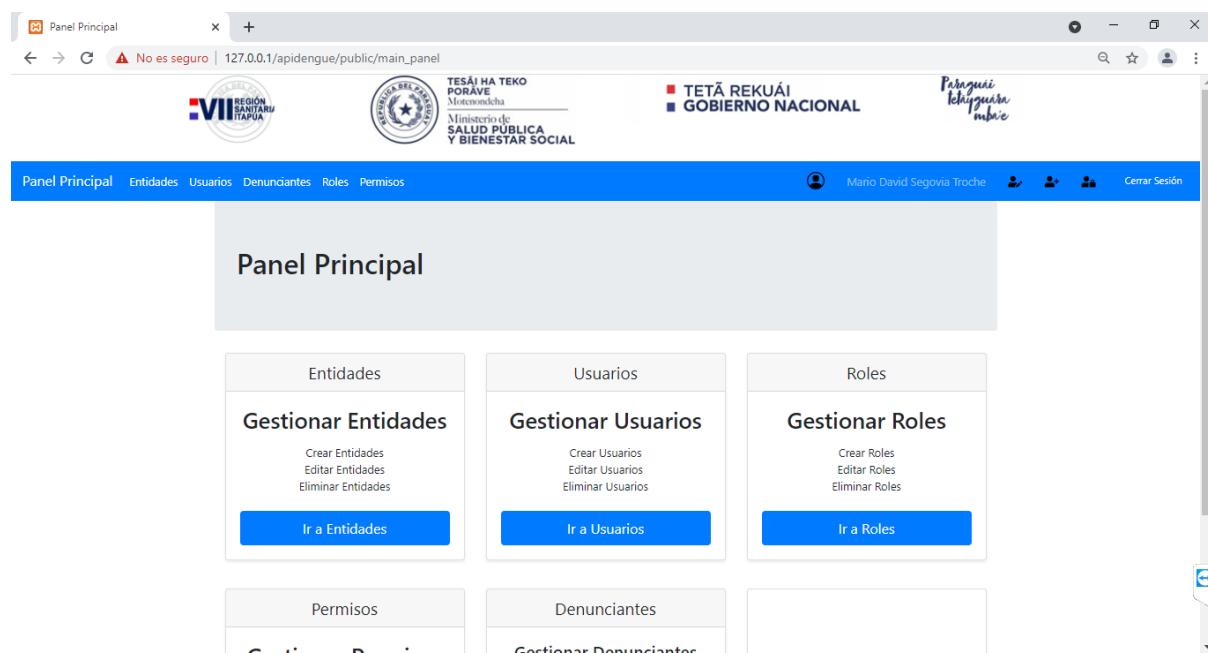
*Ilustración 53. Página de ingreso al módulo de gestión de usuarios.*



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 54*).

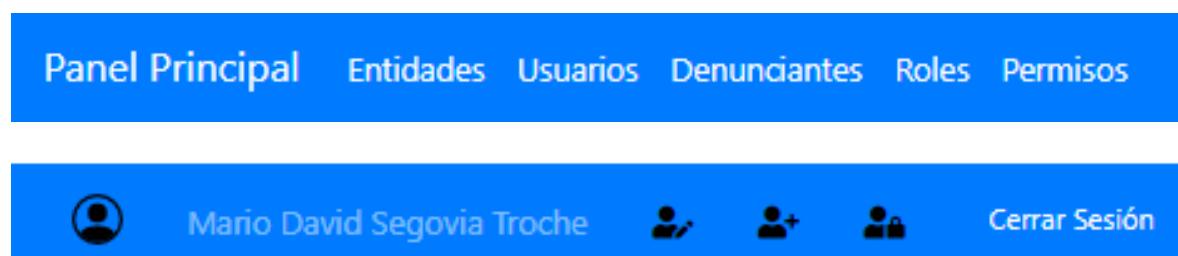
*Ilustración 54. Panel principal del módulo de gestión de usuarios.*



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 55*).

*Ilustración 55. 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 56*).

*Ilustración 56. Vistas de lista de entidades, nueva entidad y datos de entidad.*

The screenshots illustrate the Entity management section of the system:

- Screenshot 1: List of Entities**  
Shows a table with two rows of entity data:
 

Nombre de Entidad	Correo Electronico	direccion	Ver datos	Editar	Eliminar
Hospital Pediatrico	hosp.pediatrico@gmail.com	Barrio Nueva Esperanza	<a href="#">Ver Datos</a>	<a href="#">Editar</a>	<a href="#">Borrar</a>
Septima Region Sanitaria	7msecretariageneral@gmail.com	Memmel esq. Dr. Honorio Gonzalez	<a href="#">Ver Datos</a>	<a href="#">Editar</a>	<a href="#">Borrar</a>
- Screenshot 2: New Entity Form**  
A form for creating a new entity with fields for:
  - Nombre:
  - Localidad:
  - Direccion:
  - Telefono:
  - Email:
  - Observaciones:
 Buttons: Guardar, Cancelar.
- Screenshot 3: Entity Data Form**  
A form displaying the details of an entity with fields for:
  - Nombre: Hospital Pediatrico
  - Localidad: Encarnacion
  - Direccion: Barrio Nueva Esperanza
  - Telefono: 0995230250
  - Email: hosp.pediatrico@gmail.com
  - Observaciones: Ninguna
 Buttons: Volver.

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 57*), ver o editar los datos de los usuarios (*Ilustración 58*), 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 59*).

*Ilustración 57 Vistas de lista de usuario y nuevo usuario.*

**Lista de Usuarios**

Nombre(s)	Apellido(s)	Correo Electronico	Roles	Permisos	Ver datos	Editar	Cambiar Contraseña	Eliminar
Gabriel	Sotelo	soteweb1@gmail.com	<a href="#">Roles</a>	<a href="#">Permisos</a>	<a href="#">Ver Datos</a>	<a href="#">Editar</a>	<a href="#">Cambiar Contraseña</a>	<a href="#">Borrar</a>
Marcos	Bordon	marcosbordon3@gmail.com	<a href="#">Roles</a>	<a href="#">Permisos</a>	<a href="#">Ver Datos</a>	<a href="#">Editar</a>	<a href="#">Cambiar Contraseña</a>	<a href="#">Borrar</a>
Mario David	Segovia Troche	madasetro@hotmail.com	<a href="#">Roles</a>	<a href="#">Permisos</a>	<a href="#">Ver Datos</a>	<a href="#">Editar</a>	<a href="#">Cambiar Contraseña</a>	<a href="#">Borrar</a>
Rafael	Escobar	rep141998@gmail.com	<a href="#">Roles</a>	<a href="#">Permisos</a>	<a href="#">Ver Datos</a>	<a href="#">Editar</a>	<a href="#">Cambiar Contraseña</a>	<a href="#">Borrar</a>
Romina	Patiño	romipatiño@gmail.com	<a href="#">Roles</a>	<a href="#">Permisos</a>	<a href="#">Ver Datos</a>	<a href="#">Editar</a>	<a href="#">Cambiar Contraseña</a>	<a href="#">Borrar</a>

Mostrando registros del 1 al 5 de un total de 5 registros

**Nuevo Usuario**

Nombre:

Apellido:

distrito:

Direccion:

Telefono:

Email:

Entidad a la que pertenece:

Observaciones:

Fuente: Elaboración propia.

Ilustración 58. Vistas de datos de usuario y editar usuario.

The screenshots show the following details:

**Nuevo Usuario View (Top Screenshot):**

- Form fields:
  - Nombre: Gabriel
  - Apellido: Sotelo
  - distrito: Encarnacion
  - Direccion: San Pedro
  - Telefono: 0985 758031
  - Email: soteweb1@gmail.com
  - Entidad a la que pertenece: Septima Region Sanitaria

**Editar Usuario View (Bottom Screenshot):**

- Form fields:
  - Nombre: Gabriel
  - Apellido: Sotelo
  - distrito: Encarnacion
  - Direccion: San Pedro
  - Telefono: 0985 758031
  - Email: soteweb1@gmail.com
  - Entidad a la que pertenece: Septima Region Sanitaria

Fuente: Elaboración propia.

*Ilustración 59. Vistas para cambiar contraseña, roles y permisos de usuario*

The image consists of three vertically stacked screenshots of a web-based application interface, likely a dashboard for managing user roles and permissions. Each screenshot shows a header with the logo of the VII Región Sanitaria (Tetá Rekuái Gobierno Nacional), the date (22/06/2021), and the time (10:01 p.m.). The top screenshot shows a 'Cambiar Contraseña' (Change Password) form with fields for Nombre (Name), Email, Contraseña (Password), and Repetir Contraseña (Repeat Password). The middle screenshot shows a 'Roles de Usuario' (User Roles) form for a user named Gabriel Sotelo, with fields for Nombre, Apellido, Email, and a list of assigned roles (admin). The bottom screenshot shows a 'Permisos de Usuario' (User Permissions) form for the same user, with fields for Nombre, Apellido, Email, and a list of assigned permissions.

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 60*).

*Ilustración 60. Vistas de lista de denunciantes, crear y editar un registro.*

The screenshots illustrate the user interface for managing denunciants in a web application. The top screenshot shows a list of denunciants with two entries: Ana Brun and Liz Lugo. The middle screenshot shows a 'Nuevo Denunciante' (New Denunciante) form with fields for Nombre, Email, Contraseña, and Repetir Contraseña. The bottom screenshot shows an 'Editar Denunciante' (Edit Denunciante) form for Ana Brun, showing the same fields as the new form.

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 61*), editar o borrar un rol y agregar o quitar permisos (*Ilustración 62*).

*Ilustración 61. Vista de la lista de roles y crear un nuevo rol.*

The screenshot displays two consecutive screenshots of a web application interface for managing roles.

**Screenshot 1: Lista de Roles (List of Roles)**

- Header:** Roles, No es seguro | 127.0.0.1/apidengue/public/roles
- Logos:** VII Región de Tapachula, TESÁI HA TEKO PORÁVE Motonameha, Ministerio de SALUD PÚBLICA Y BIENESTAR SOCIAL, TETÁ REKUÁI GOBIERNO NACIONAL, Parque Ichiguan mb'ie.
- Navigation:** Panel Principal, Entidades, Usuarios, Denunciantes, Roles, Permisos. User: Mario David Segovia Troche, Cerrar Sesión.
- Title:** Lista de Roles
- Table:**| Rol | Nombre Rol | Descripción de Rol | Permisos | Editar | Eliminar |
| --- | --- | --- | --- | --- | --- |
| admin | Administrador | Administrador con acceso total | Permisos | Editar | Borrar |
| Denunciante | Denunciante | Ciudadanos denunciantes de criaderos de Dengue | Permisos | Editar | Borrar |
| lab\_user | Usuario de laboratorio | Usuario de laboratorio | Permisos | Editar | Borrar |
| res\_stock | Responsable de Stock | Responsable de Stock | Permisos | Editar | Borrar |
- Footer:** Mostrando registros del 1 al 4 de un total de 4 registros, Anterior, 1, Siguiente.

**Screenshot 2: Nuevo Rol (New Role) Creation Form**

- Header:** Roles, No es seguro | 127.0.0.1/apidengue/public/roles/create
- Logos:** VII Región de Tapachula, TESÁI HA TEKO PORÁVE Motonameha, Ministerio de SALUD PÚBLICA Y BIENESTAR SOCIAL, TETÁ REKUÁI GOBIERNO NACIONAL, Parque Ichiguan mb'ie.
- Navigation:** Panel Principal, Entidades, Usuarios, Denunciantes, Roles, Permisos. User: Mario David Segovia Troche, Cerrar Sesión.
- Title:** Nuevo Rol
- Form Fields:**
  - Role:
  - Name Role:
  - Description:
- Buttons:** Guardar (Save), Cancelar (Cancel).

Fuente: Elaboración propia.

*Ilustración 62. Vistas para editar rol y agregar o quitar permisos a los roles.*

The image consists of two vertically stacked screenshots of a web-based administrative interface. Both screenshots are framed by a dark blue header bar containing navigation links: 'Panel Principal', 'Entidades', 'Usuarios', 'Denunciantes', 'Roles', and 'Permisos'. On the right side of the header are user profile icons and a 'Cerrar Sesión' (Logout) button.

**Screenshot 1: Editar Rol (Edit Role) screen**

- Header:** Shows the title 'Roles' and a URL '127.0.0.1/apidengue/public/roles/1/edit'.
- Content:**
  - Role Selection:** A dropdown menu labeled 'Rol:' with the value 'admin'.
  - Name:** A text input field labeled 'Nombre Rol:' with the value 'Administrador'.
  - Description:** A text input field labeled 'Descripción:' with the value 'Administrador con acceso total'.
  - Buttons:** 'Guardar' (Save) and 'Cancelar' (Cancel).

**Screenshot 2: Permisos de Rol (Role Permissions) screen**

- Header:** Shows the title 'Permisos de Rol' and a URL '127.0.0.1/apidengue/public/role\_perms/1'.
- Content:**
  - Role Selection:** A dropdown menu labeled 'Rol:' with the value 'Administrador'.
  - Description:** A text input field labeled 'Descripción:' with the value 'Administrador con acceso total'.
  - Granted Permissions:** A section titled 'Permisos asignados:' with a note '(Marque los Permisos asignados si desea desvincularlos del Rol)'. It contains three checkboxes:
    - Modificar paciente
    - Nuevo paciente
    - Eliminar paciente
  - Add Permissions:** A section titled 'Agregar Permisos:' with five checkboxes:
    - Listar pacientes
    - Crear caso positivo
    - Modificar caso positivo
    - Eliminar caso positivo
    - Listar casos positivos

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 63*).

Ilustración 63. Vistas de lista de permisos, crear y editar permisos.

**Lista de Permisos**

Permiso	Nombre Permiso	Descripción de Permiso	Editar	Eliminar
create_caso_pos	Crear caso positivo	Crear caso positivo	Editar	Borrar
create_control	Nuevo control	Crear nuevo control	Editar	Borrar
create_denuncia	Nueva denuncia	Nueva denuncia	Editar	Borrar
create_noticia	Nueva Noticia	Nueva noticia	Editar	Borrar
create_paciente	Nuevo paciente	Nuevo paciente	Editar	Borrar
create_sugerencia	Nueva sugerencia	Nueva sugerencia	Editar	Borrar
create_tipo_prueba	Crear tipo de prueba	Crear tipo de prueba	Editar	Borrar
del_caso_pos	Eliminar caso positivo	Eliminar caso positivo	Editar	Borrar

**Nuevo Permiso**

Permiso:

Nombre Permiso:

Descripción:

**Guardar** **Cancelar**

**Editar Permiso**

Permiso:

Nombre Permiso:

Descripción:

**Guardar** **Cancelar**

Fuente: Elaboración propia.

### 5.13. 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.

- Modulo de Auditoría.

Función:

Control, registro y seguimiento del movimiento de la información dentro de la base de datos de la plataforma y visualización de los eventos en una interfaz web. Opción de emitir reportes del historial de cambios de la información almacenada.

Modelos creados: Auditoría.

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

- Módulo de aplicación móvil de alertas sobre el Dengue.

Función:

Aplicación móvil para teléfonos inteligentes para generar alertas sobre casos de Dengue en cercanías a la posición del dispositivo mediante la comparación de la geolocalización por GPS y los datos de georreferencia de casos de Dengue almacenados en la plataforma web.

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

- Módulo de Gestión de Tareas destinado al SENEPA.

Función:

Gestión de tareas para responsables del SENEPA, para registrar y administrar las actividades realizadas para el control y erradicación del Dengue, como fumigaciones, verificación de criaderos, entre otros.

Modelos creados: Actividades, Asignaciones

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

#### 5.14. 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 64*).

*Ilustración 64. Ruta protegida por token de seguridad.*

```
Route::apiResource("pacientes", "PacienteController")->middleware('auth:api');
```

Fuente: Elaboración propia.

#### 5.15. 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.

De esta manera se culmina el desarrollo de todas las funcionalidades establecidas por el proceso de desarrollo mediante SCRUM.

## 5.16. Pruebas de funcionamiento

El testeo de las funcionalidades se realizaron con pruebas de caja negra, que se fueron aplicando según los avances del proyecto. Las pruebas se registraron en formularios en las cuales se verificaron los procesos realizados (*Ilustración 65*).

*Ilustración 65. Formulario de Verificación de funcionalidades de la API.*

FORMULARIO DE TESTEO DE FUNCIONALIDAD		
Responsable:		
Fecha:		
Formulario Nº:		
Modulo:	API REST Principal	
Funcionalidad:	Registro de personas infectadas	
	Procesos	SI
Crea registro		
Modifica registro		
Elimina registro		
Valida correctamente todos los campos del registro		
Genera el registro de auditoria		
Genera la respuesta de confirmación de cada proceso		
Observaciones:		

Fuente: Elaboración propia.

Para la realización de las pruebas se utilizó la aplicación Postman, que permite ejecutar peticiones REST a una API y verificar el correcto funcionamiento de del módulo (*Ilustración 66*).

*Ilustración 66. Aplicación Postman para el testeo de APIs.*

The screenshot shows the Postman application interface. At the top, there's a toolbar with 'NEW', 'Runner', 'Import', and other icons. Below the toolbar, a navigation bar includes 'Builder', 'Team Library', 'Sign In', and sync status. A message about Chrome apps being deprecated is displayed. The main area has tabs for 'History' (selected), 'Collections', and 'Clear all'. On the left, a sidebar shows a tree view of requests grouped by date: 'April 11' and 'February 20'. Under 'February 20', several POST requests are listed with URLs like 'http://laravel1.soteweb.com/api/estudiantes' and 'http://localhost/proyecto1/public/api/estudiantes'. The right side of the screen shows a detailed view of a selected POST request for 'http://127.0.0.1/apidengue/public/api/auth/login'. It includes fields for 'Authorization', 'Headers (3)', 'Body (x)', 'Pre-request Script', 'Tests', and 'Code'. Below this, a large text area says 'Hit the Send button to get a response.' with a progress bar below it. At the bottom, there are buttons for 'Share', 'Mock', 'Monitor', and 'Document'.

Fuente: Elaboración propia.

La aplicación Postman permite realizar los distintos tipos de peticiones que se utilizan en el protocolo REST (*Ilustración 67*).

*Ilustración 67. Métodos REST para la prueba de APIs.*

The screenshot shows the Postman interface with the following details:

- Method:** POST (selected)
- URL:** http://127.0.0.1/apidengue/public/api/auth/login
- Params:** (empty)
- Send:** (blue button)
- Save:** (button with dropdown)
- Header Section:**
  - GET leaders (3) Body Pre-request Script Tests
  - POST (disabled)
  - PUT (disabled)
  - PATCH (disabled)
  - DELETE (disabled)
- Auth Section:** No Auth (dropdown menu)

Fuente: Elaboración propia.

Permite agregar los parámetros necesarios al encabezado de una petición REST (*Ilustración 68*).

*Ilustración 68. Parámetros para los encabezados de peticiones REST.*

Key	Value	Description	...	Bulk Edit	Presets
<input checked="" type="checkbox"/> Content-Type	application/json				
<input checked="" type="checkbox"/> X-Requested-With	XHttpRequest				
<input type="checkbox"/> Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJhd...				
New key	Value	Description			

Fuente: Elaboración propia.

Se pueden agregar datos en el cuerpo de las peticiones REST para el testeo del correcto almacenamiento de información en la base de datos de la API (*Ilustración 69*).

*Ilustración 69. Ejemplo de datos adjuntos al cuerpo de una petición REST.*

The screenshot shows the Postman interface with the following details:

- Headers:** Authorization, Headers (3), Body (selected), Pre-request Script, Tests
- Body Type:** JSON (application/json)
- Body Content:**

```

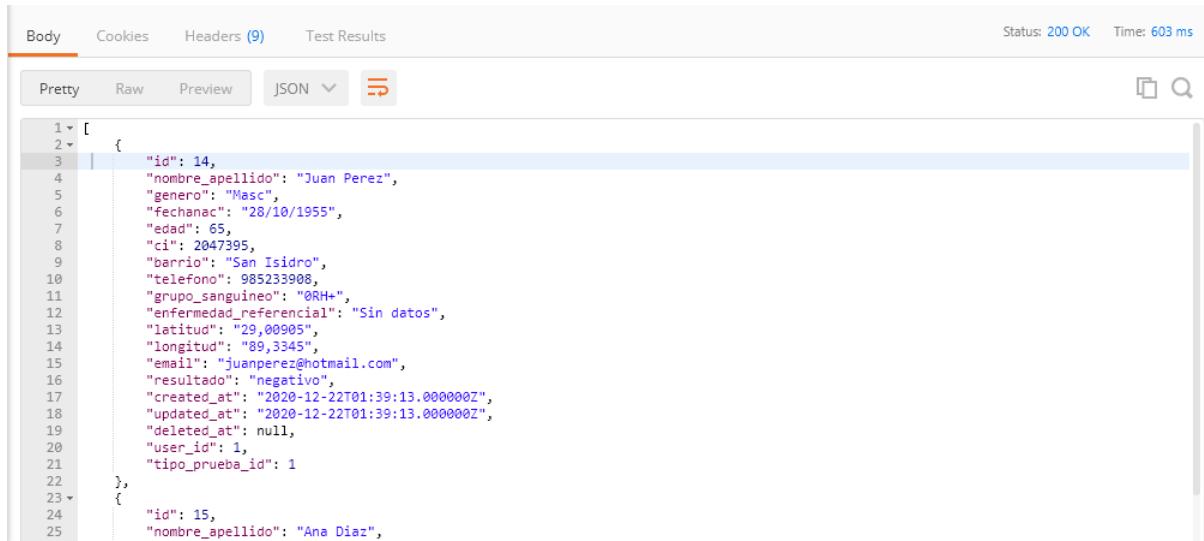
1 [
2   "nombre_apellido": "Isabel Gutierrez",
3   "genero": "Fem",
4   "fechanac": "21/05/200",
5   "edad": 20,
6   "ci": 6743761,
7   "barrio": "Chairepe",
8   "telefono": 995645632,
9   "grupo_sanguineo": "0+",
10  "enfermedad_referencial": "Sin datos",
11  "latitud": "87,00905",
12  "longitud": "123,3345",
13  "email": "isatierrez@gmail.com",
14  "resultado": "negativo",
15  "user_id": 1,
16  "tipo_prueba_id": 1
17 ]

```

Fuente: Elaboración propia.

Se pueden verificar las respuestas a las peticiones realizadas si son correctas y en el formato deseado (*Ilustración 70*).

*Ilustración 70. Respuesta en formato JSON de los datos requeridos en una petición GET.*



The screenshot shows a REST API response in JSON format. The response status is 200 OK and the time taken is 603 ms. The JSON data is displayed in a pretty-printed format with line numbers on the left. It contains two user records, each represented by a JSON object. The first record has an id of 14 and the second has an id of 15. Both records include fields such as nombre\_apellido, genero, fechanac, edad, c1, barrio, telefono, grupo\_sanguineo, enfermedad\_referencial, latitud, longitud, email, resultado, created\_at, updated\_at, deleted\_at, user\_id, and tipo\_prueba\_id.

```
1 [  
2 {  
3     "id": 14,  
4     "nombre_apellido": "Juan Perez",  
5     "genero": "Masculino",  
6     "fechanac": "28/10/1955",  
7     "edad": 65,  
8     "c1": 2047395,  
9     "barrio": "San Isidro",  
10    "telefono": 985233908,  
11    "grupo_sanguineo": "0RH+",  
12    "enfermedad_referencial": "Sin datos",  
13    "latitud": "29,00905",  
14    "longitud": "89,3345",  
15    "email": "juanperez@hotmail.com",  
16    "resultado": "negativo",  
17    "created_at": "2020-12-22T01:39:13.000000Z",  
18    "updated_at": "2020-12-22T01:39:13.000000Z",  
19    "deleted_at": null,  
20    "user_id": 1,  
21    "tipo_prueba_id": 1  
22 },  
23 {  
24     "id": 15,  
25     "nombre_apellido": "Ana Diaz",  
26 }
```

Fuente: Elaboración propia.

De esta forma se validaron todas las funcionalidades que componen el modulo API, con la comprobación y verificación del correcto funcionamiento y comportamiento de las características proyectadas, logrando una funcionalidad óptima mediante la cobertura de todas las funcionalidades propuestas y sin errores por corregir.

## 6. CONCLUSIÓN

Como se expuso en este trabajo de investigación, el Dengue es actualmente una de las enfermedades que más impacto generan en los servicios sanitarios y en la población en general en grandes regiones del planeta. Varias organizaciones de salud e instituciones sanitarias de muchos países del mundo resaltan la importancia de establecer campañas y estrategias para luchar contra esta enfermedad y su erradicación. De las distintas estrategias utilizadas en la lucha contra este flagelo, la utilización de las Tecnologías de la Información y Comunicación, así como las relacionadas a las ciencias de la informática y la computación, en iniciativas, proyectos y campañas contra la enfermedad del Dengue, dieron resultados exitosos. Por estos motivos, surge el proyecto de una plataforma para el registro de personas infectadas con el virus del Dengue, con la finalidad de ser una herramienta tecnológica de utilidad para la institución denominada Séptima Región Sanitaria del Departamento de Itapúa, Paraguay.

La plataforma desarrollada se diseñó en una estructura modular, donde se definieron las funcionalidades de cada módulo y en este documento se describe el proceso de desarrollo del módulo API REST, cuya función es la de servir de núcleo de la transferencia de información y datos con los demás módulos y ofrecer el control y almacenamiento seguro de éstos.

Para el desarrollo del módulo API REST se utilizó una metodología de desarrollo ágil SCRUM, siendo la opción más adecuada para este proyecto. Esta metodología ofrece todas las herramientas, métodos, procesos y artefactos necesarios para lograr los objetivos propuestos. Debido a su gran flexibilidad, se logró planificar exitosamente todas las actividades necesarias para cumplir con las metas trazadas en el periodo de tiempo establecido.

El desarrollo de la API REST se llevó a cabo de forma exitosa completando todas las funcionalidades que fueron previstas. Con la planificación de las actividades de acuerdo a la metodología SCRUM se establecieron las funcionalidades requeridas, las etapas de trabajo necesarias, los periodos de tiempo estimados y de acuerdo a los avances del proyecto, los ajustes en función al progreso de las actividades. Las herramientas tecnológicas utilizadas, como el lenguaje de programación PHP, el entorno de desarrollo Laravel, el sistema de gestión de bases de datos PostgreSQL, entre otros, brindaron todos los recursos necesarios para completar las características establecidas para el módulo, teniendo en cuenta que son open source o de uso libre, con lo cual no suponen un costo monetario su utilización e implementación.

La comprobación del correcto funcionamiento de las funcionalidades del módulo API REST se realizó utilizando las guías proveídas por la metodología SCRUM, por medio de pruebas constantes aplicadas durante el proceso de desarrollo de los trabajos.

Como recomendaciones de trabajos a futuro se puede ampliar este proyecto a otras dolencias que afecten a la población, ya que este trabajo está dirigido a la enfermedad del Dengue, pero ofrece una posibilidad de expansión gracias a su estructura y herramientas tecnológicas utilizadas. Esta plataforma tiene el potencial de crecer de una solución para un área regional en un sistema integrado a un nivel nacional.

## 7. REFERENCIAS BIBLIOGRÁFICAS.

- Arbo, A. (2019). *Dengue: pesada carga para la salud pública del Paraguay*. <http://dx.doi.org/10.18004/imt/20191411-2>
- Canós, J. H., Letelier, P., & Penadés, M. C. (2003). Metodologías ágiles en el desarrollo de software. Universidad Politécnica de Valencia, Valencia, 1-8.
- Chico Aldama, P. (2019). Ciclo de vida del Aedes aegypti y manifestaciones clínicas del dengue. <http://repositorio.pediatria.gob.mx:8180/handle/20.500.12103/1532>
- Condor Camara, D., Nolasco Cardenas, O. P., Carrasco Escobar, G. & Egoavil Ayala, M. (2018). *Sistema de Información basado en Tecnologías de Información y Comunicación para geolocalización de Zika, Dengue, Chikungunya y Malaria*.
- 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.
- 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>
- Flores, L., Giménez Caballero, E., Díaz Duba, S. & Torales, J. (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](https://dx.doi.org/10.18004/Mem.iics/1812-9528/2015.013(02)78-085)
- Guzmán, M. G., García, G., & Kourí, G. (2006). El dengue y el dengue hemorrágico: prioridades de investigación. *Revista panamericana de salud pública*, 19, 204-215. <http://coloquioenfermeria2018.sld.cu/index.php/coloquio/2018/paper/view/440/96>
- Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, P. (2014). Metodología de la investigación. Sexta edición. México DF McGraw Hill.
- Lage, R. J., Herrera Graña, T., Simpson Johnson, B. & Zulueta Torres, Z. (2015). *Aspectos actualizados sobre dengue*. <https://dialnet.unirioja.es/servlet/articulo?codigo=6027633>

Lugones Botell, M., & Ramírez Bermúdez, M. (2012). Dengue. *Revista Cubana de Medicina General Integral*, 28(1), 123-126.

Marín, R. (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>

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

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/>>

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

Navarro Cadavid, A., Fernández Martínez, J. D., & Morales Vélez, J. (2013). *Revisión de metodologías ágiles para el desarrollo de software*. <https://www.redalyc.org/pdf/4962/496250736004.pdf>

Organización Mundial de la Salud (2015). *Dengue: prevención y control: informe de la Secretaría* (No. A68/29). Organización Mundial de la Salud.

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

Pane, J., Ojeda V. & Valdez, N. (2015). *Dengue Open Data*. <https://idatosabiertos.org/wp-content/uploads/2015/10/7.Dengue-Pane-Ojeda-Valdez.pdf>

Pane, J., Paciello, J., Lisnichuk, Y., Martínez, H., Valdez, S. & Durañona, N. (2018). *DATOS ABIERTOS Y ALERTAS SOBRE DENGUE*. <https://publicaciones.fctunca.edu.py/jspui/handle/123456789/77>

Parra, C. (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>

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](https://www.researchgate.net/publication/334279550_Participation_in_Public_Health_Programs_from_the_Community_the_TopaDengue_case)

Revuelta Arribas, A. (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](http://oa.upm.es/58093/1/TFG_ALBERTO_REVUELTA_ARRIBAS.pdf)

Rodríguez-Castro, A. I., Rolón, J., & Ríos-González, C. M. (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>

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>

Sarango Yunga, D. X. (2020). *Desarrollo de plataforma web para la evaluación de software basado en la metodología SCRUM* (Bachelor's thesis, Machala: Universidad Técnica de Machala). <http://repositorio.utmachala.edu.ec/handle/48000/15665>

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>

Trigás Gallego, M. (2012). *Metodología Scrum*.

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>