



DESARROLLO DE SISTEMAS
Coronel Suárez

Proyecto: Desarrollo Aplicación Web CODIV-19

Cliente: Municipalidad de Coronel Suarez



Materia: Desarrollo de Aplicaciones II

Catedra:

- Profesora Teoría: María Paula González
- Profesora Práctica: Paula González Olivera

Proyecto1: Desarrollo Aplicación Web CODIV-19

Comisión 1:

- Ignacio Coria
- Fernando Giménez
- Andrés Marino
- Agustín Suárez



INDICE:

Introducción	4
Problemática	5
Solución	6
Diseño en capas	7
Historias de Usuario	8
Mockups	12
Diagrama de Clases	17
Referencias Diagrama de Clases	18
Diagrama de Base de Datos	20
Recursos Utilizados	21
Detalles de la Implementación	23
Datos de Prueba	24
Testeos	25
Conclusión	28



Introducción:

El proyecto se realiza para la cátedra de Aplicaciones II de la carrera Tecnicatura de Universitaria en Desarrollo de Aplicaciones Web de la Universidad del Sudoeste.

Como integrantes de la Comisión 1 representamos a la empresa de desarrollo de sistemas CS S.A. que debe resolver el problema propuesto por nuestro cliente, la secretaria de Salud de la Municipalidad de Coronel Suárez, que es el desarrollo de una aplicación web para procesar la información de la campaña de vacunación Covid.



Problemática:

La Secretaría de Salud de la municipalidad de Coronel Suarez desea procesar información para llevar cierto control sobre los ciudadanos del partido que recibieron las dos dosis de alguna vacuna para el Covid. En los Vacunatorios Covid de cada centro de salud habilitado se han registrado de manera manual a los ciudadanos vacunados (DNI, nombre y apellido, domicilio, edad, grupo de riesgo (1,2,3 o 4), tipo de vacuna recibida, fecha primera dosis, fecha segunda dosis, registro R.U.P. de enfermera/enfermero).

Además, la municipalidad tiene información sobre cada Vacunatorio Covid del partido. Los mismos funcionan dentro de algunos de los centros de salud municipales, que se distinguen por un código dentro del sistema de salud de la pcia. de Buenos Aires (2030039 para el Hospital Municipal “Dr. Raúl Caccavo” de la ciudad de Coronel Suarez, 20300055 para el Hospital Municipal “Lucero del Alba” de Huanguelén, 20300098 para la Unidad Sanitaria Pueblo San José de San José, 23300063 para la Unidad Sanitaria Dr. Lew Frandzman de Santa María y 23300080 para la Unidad Sanitaria Pueblo Santa Trinidad de Santa Trinidad).

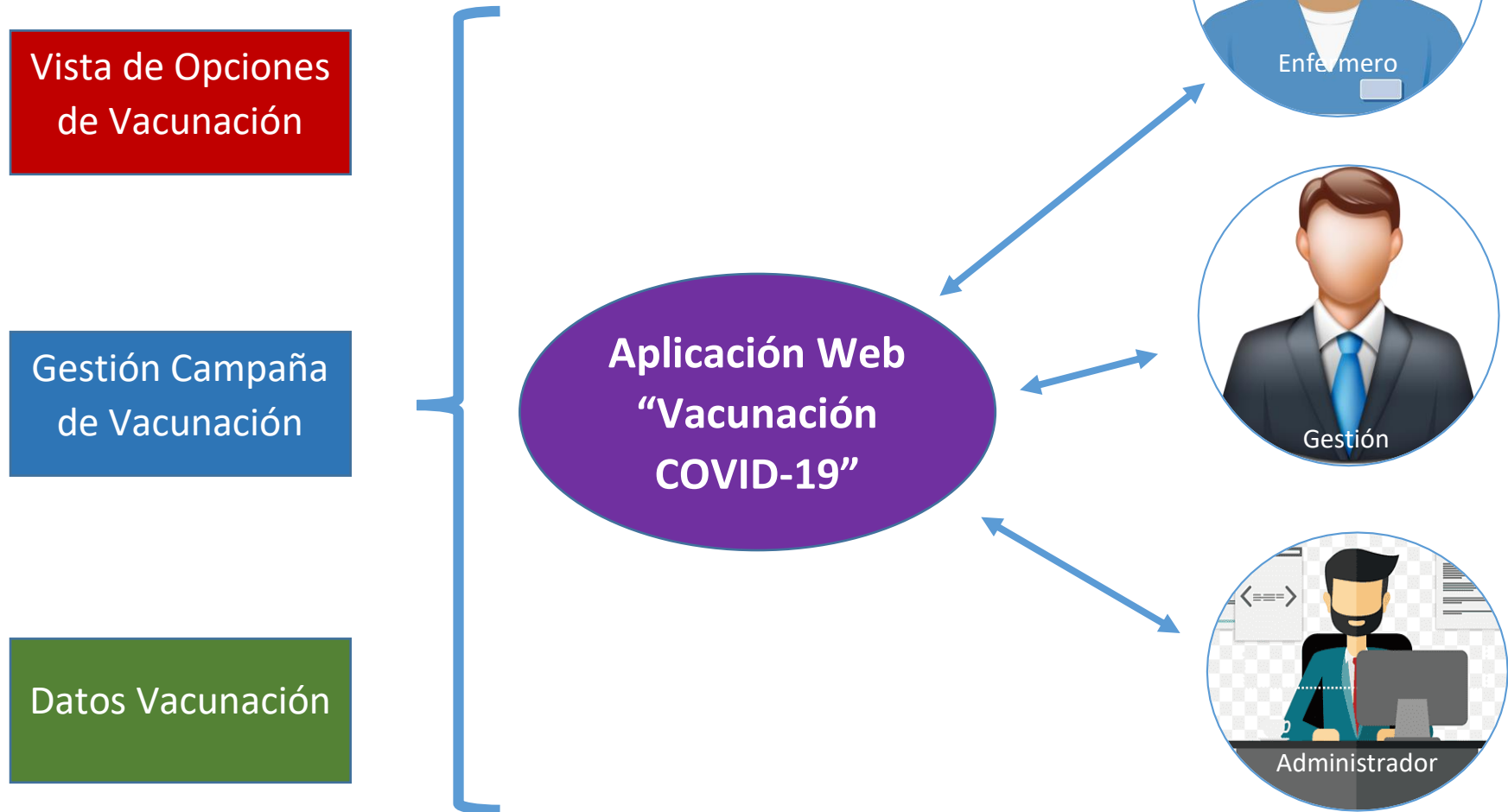
En cada centro de salud anterior funciona solo un Vacunatorio Covid. Para cada vacunatorio la municipalidad lleva registro del nombre del médico a cargo del mismo, las enfermeras que trabajan en el vacunatorio, el horario de atención y un teléfono de contacto. Cada enfermera/o que trabaja en el partido está registrada en la municipalidad con su número de registro R.U.P (Registro Único para Profesionales de la Salud), su nombre y apellido y un teléfono de contacto. Las/os enfermeras/os pueden trabajar en más de un Vacunatorio Covid.



Solución:

Se desarrollará una aplicación web para solucionar la problemática del cliente con el requerimiento de su parte de utilizar el patrón MVC, acordando con el cliente la utilización del framework Laravel.

Aplicación Web “Campaña Vacunación Covid-19”





DISEÑO DE FUNCIONALIDADES DEL SISTEMA

Historia de Usuario: Enfermero



- *Como enfermero quiero registrar las vacunas que aplico para control de la campaña.*
- *Como enfermero quiero consultar los datos del paciente a vacunar para aplicar la dosis correcta si corresponde al centro de vacunación.*

DISEÑO DE FUNCIONALIDADES DEL SISTEMA

Historia de Usuario: Administrador



- *Como administrador quiero realizar el Alta de los nuevos usuarios para controlar los accesos al sistema.*
- *Como administrador quiero realizar el Baja de los usuarios para impedir el acceso al sistema a los usuarios desafectados a la campaña.*
- *Como administrador quiero realizar el Modificación de los usuarios para mantener actualizada la información de los mismo.*
- *Como administrador quiero realizar el Alta de los tipos de vacunas para utilizar en los registros de la campaña.*
- *Como administrador quiero realizar la Baja de los tipos de vacunas para inhabilitar su uso*
- *Como administrador quiero realizar la Modificación de los tipos de vacunas para mantener actualizada la información de las mismas.*

DISEÑO DE FUNCIONALIDADES DEL SISTEMA

Historia de Usuario: Administrador



- *Como administrador quiero realizar el Alta de los nuevos centros de vacunación para incorporarlos a la campaña de vacunación.*
- *Como administrador quiero realizar la Baja de los centros de vacunación desafectarlo de la campaña*
- *Como administrador quiero realizar la Modificación de los centros de vacunación para mantener actualizada la información de los mismos.*
- *Como administrador quiero realizar el Alta de los nuevos vacunatorios para incorporarlos a la campaña de vacunación.*
- *Como administrador quiero realizar la Baja de los vacunatorios para desafectarlo de la campaña*
- *Como administrador quiero realizar la Modificación de los vacunatorios para mantener actualizada la información de los mismos.*



DISEÑO DE FUNCIONALIDADES DEL SISTEMA

Historia de Usuario: Gestión Municipal



- *Como personal de gestión quiero consultar todas las vacunas disponibles*
- *Como personal de gestión quiero consultar todos los vacunatorios disponibles*
- *Como personal de gestión quiero consultar todos los centros disponibles*
- *Como personal de gestión quiero consultar información de los usuarios*



Capa de Presentación:

Mockups.

1- Login:

DNI

Contraseña

[¿Olvidaste tu contraseña?](#)

Iniciar Sesión

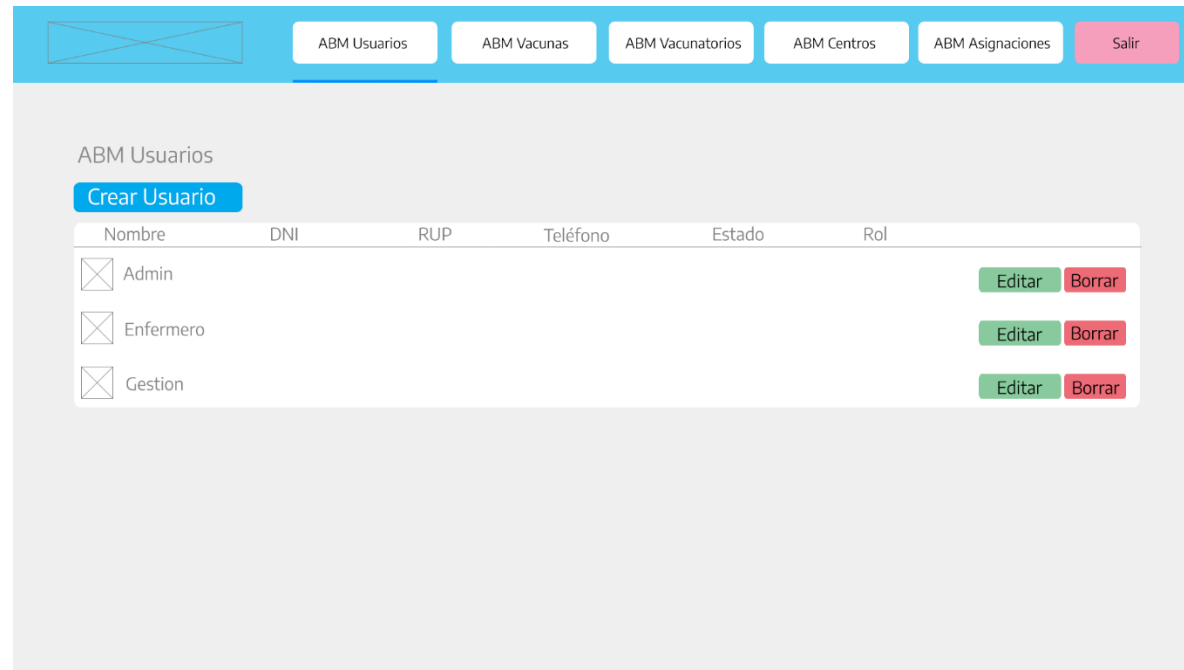


2- Menú Inicio:



Para los demás menús se utilizó el mismo mockup, pero cambiándole los botones en la barra de navegación

3- Menú Funcionalidades:




The mockup shows a web interface for 'ABM Usuarios'. At the top, there is a navigation bar with a placeholder icon and buttons for 'ABM Usuarios', 'ABM Vacunas', 'ABM Vacunatorios', 'ABM Centros', 'ABM Asignaciones', and 'Salir'. Below the navigation bar, the main content area is titled 'ABM Usuarios' and contains a 'Crear Usuario' button. A table lists three users: 'Admin', 'Enfermero', and 'Gestion'. Each user entry has a checkbox on the left and 'Editar' and 'Borrar' buttons on the right.

Nombre	DNI	RUP	Teléfono	Estado	Rol
<input type="checkbox"/> Admin					Editar Borrar
<input type="checkbox"/> Enfermero					Editar Borrar
<input type="checkbox"/> Gestion					Editar Borrar

Para las otras funcionalidades del sistema se utilizó el mismo mockup



4- Menú Editar Perfil:



Editar perfil

Nombre y apellido:

DNI:

Email:


Contraseña Actual:

Nueva Contraseña:

Confirmar Contraseña:



5- Menú Registrar Vacunado:



Registrar Vacunado

Salir

Registrar Vacunado

DNI:

Buscar

Nombre y Apellido:

Edad:

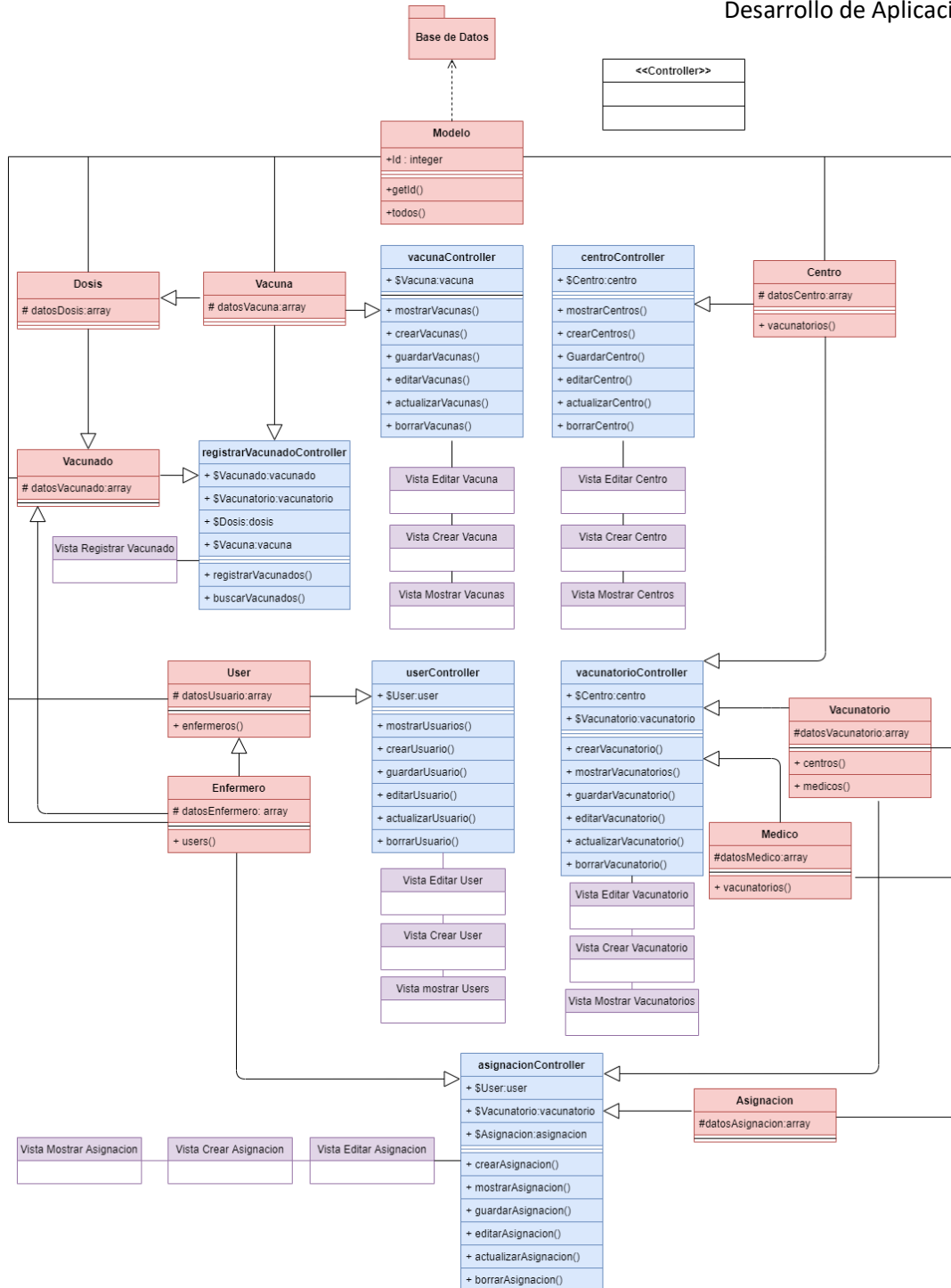
Domicilio:

Grupo de riesgo:

Tipo de vacuna:

Registrar Vacunado

Diagrama de Clases:



Referencias del Diagrama de clases:

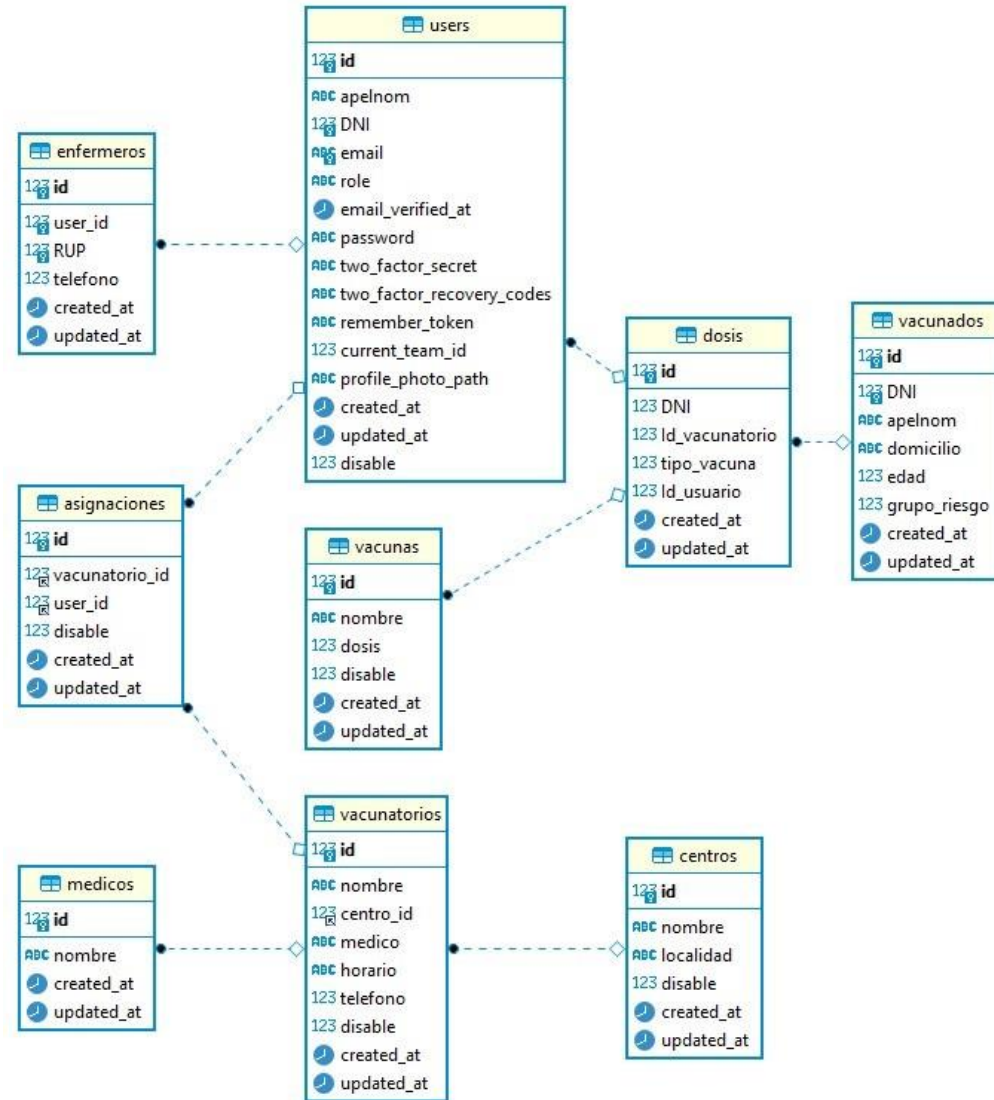
- En los modelos: Vacuna, Centro, Vacunatorio, Usuario, Enfermero, Medico, Asignación, Vacunados y Dosis. La variable llamada **datos(nombreModelo)** hace referencia a la variable \$fillable en el código ejecutable.
- En el modelo vacunaController los métodos hacen referencia a:
 - MostrarVacunas() : método index (muestra todas las vacunas).
 - CrearVacunas() : método create(muestra el formulario para cargar los datos de la vacuna).
 - guardarVacunas(): método store(inserta los datos en la bd).
 - editarVacunas(): método edit(muestra el formulario con los campos a editar).
 - actualizarVacunas(): método update(actualiza los datos en la bd).
 - borrarVacunas(): método destroy(elimina la vacuna de la bd).
- En el modelo centroController los métodos hacen referencia a:
 - MostrarCentros() : método index (muestra todos los centros).
 - CrearCentros() : método create(muestra el formulario para cargar los datos de los centros).
 - guardarCentro(): método store(inserta los datos en la bd).
 - editarCentro(): método edit(muestra el formulario con los campos a editar).
 - actualizarCentro(): método update(actualiza los datos en la bd).
 - borrarCentro(): método destroy(elimina los centros de la bd).
- En el modelo useController los métodos hacen referencia a:
 - MostrarUsuario() : método index (muestra todos los usuarios).
 - CrearUsuario() : método create(muestra el formulario para cargar los datos del nuevo usuario).
 - guardarUsuario(): método store(inserta los datos en la bd).



- `editarUsuario()`: método `edit`(muestra el formulario con los campos a editar).
 - `actualizarUsuario()`: método `update`(actualiza los datos en la bd).
 - `borrarUsuario()`: método `destroy`(elimina el usuario de la bd).
- En el modelo `vacunatorioController` los métodos hacen referencia a:
 - `MostrarVacunatorio()` : método `index` (muestra todos los vacunatorios).
 - `CrearVacunatorio()` : método `create`(muestra el formulario para cargar los datos del vacunatorio).
 - `guardarVacunatorio()`: método `store`(inserta los datos en la bd).
 - `editarVacunatorio()`: método `edit`(muestra el formulario con los campos a editar).
 - `actualizarVacunatorio()`: método `update`(actualiza los datos en la bd).
 - `borrarVacunatorio()`: método `destroy`(elimina el vacunatorio de la bd).
- En el modelo `AsignacionController` los métodos hacen referencia a:
 - `MostrarAsignaciones()` : método `index` (muestra todas las asignaciones)
 - `CrearAsignacion()` : método `create`(muestra el formulario para crear las asignaciones).
 - `guardarAsignacion()`: método `store`(inserta los datos en la bd).
 - `editarAsignacion()`: método `edit`(muestra el formulario con los campos a editar).
 - `actualizarAsignacion()`: método `update`(actualiza los datos en la bd).
 - `borrarAsignacion()`: método `destroy`(elimina la asignación de la bd).

Diseño Base de Datos

Capa de Persistencia:

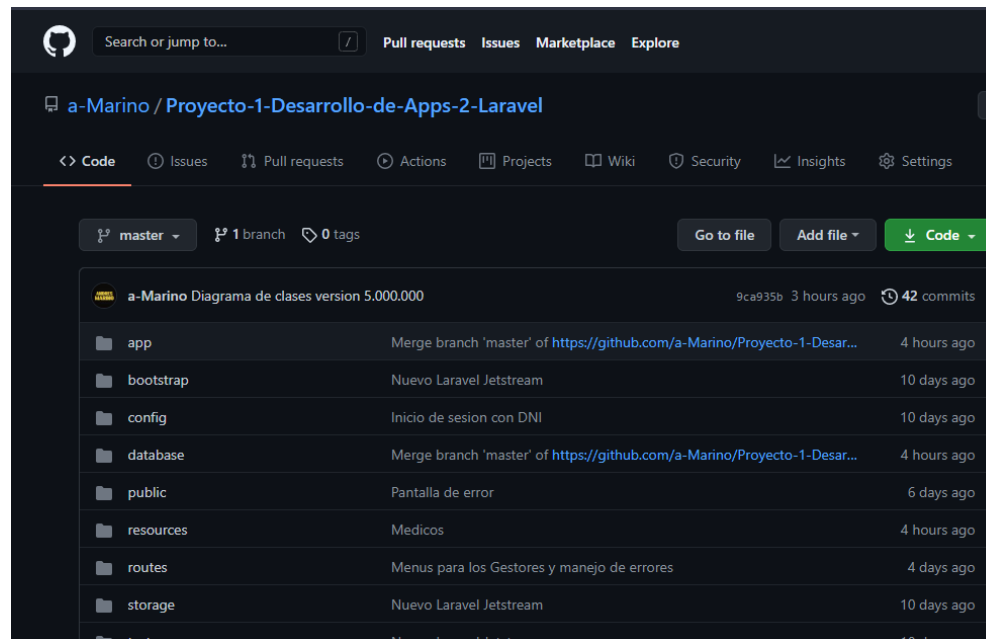




Recursos utilizados:

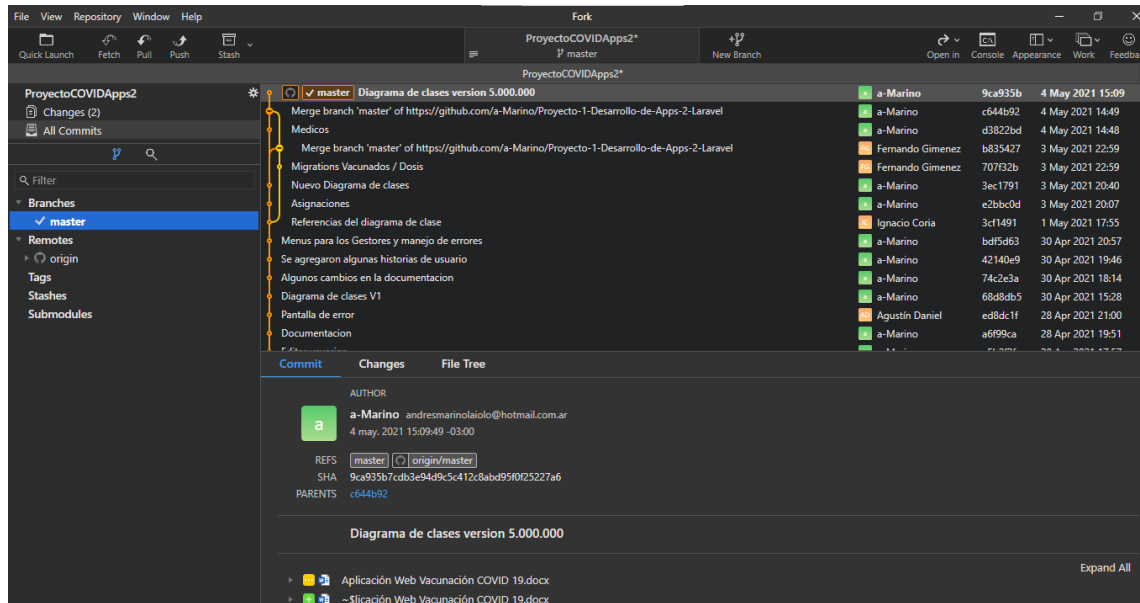
Para la codificación de la Aplicación se utilizó Laravel en su versión 8 y como manejador de Base de datos MySQL.

Con los integrantes del grupo, se armó un GitHub, lo cual fue de gran ayuda ya que podíamos ir guardando los cambios que hacíamos durante el día, y que los otros integrantes tuvieran la posibilidad de accederlos desde su pc.



<https://github.com/a-Marino/Proyecto-1-Desarrollo-de-Apps-2-Laravel>

Para esto también se utilizó un manejador de versiones, en este caso se utilizó uno llamado Fork.



Durante la realización del trabajo se fueron haciendo reuniones periódicas por Zoom, para comunicar como iba cada uno con su parte, para acordar nuevas tareas, aclarar conceptos, etc.

También se utilizó Adobe Illustrator para la realización de los iconos del sistema y Adobe XD para realizar los mockups.



Detalles de implementación:

El framework Laravel trabaja con una arquitectura de carpetas avanzada, de modo que promueve la separación de los archivos con un orden correcto y definido, que guía a todos los integrantes del equipo de trabajo y será un estándar a lo largo de los distintos proyectos. Su estilo arquitectónico es MVC.

El Modelo Vista Controlador (MVC) en laravel está implementado de la siguiente manera.

Las clases del controlador se encuentran en la carpeta ***App\Http\Controllers***

Las clases del modelo se encuentran en la carpeta ***App\Models***

Las clases de la vista se encuentran en la carpeta ***Resources\Views***

Esta estructura de carpetas es provista por el Framework.

Configuración Capa de Persistencia

La clase Modelo y la Conexión a la base de datos NO se implementaron manualmente, se utilizaron las provistas por el Framework.

El **.env** archivo predeterminado de Laravel contiene algunos valores de configuración comunes que pueden diferir según si su aplicación se ejecuta localmente o en un servidor web de producción. Estos valores luego se recuperan de varios archivos de configuración de Laravel dentro del **config** directorio usando la env función de Laravel.



Datos de Prueba

Se le proveerá con el sistema 4 usuarios precargados.

Una vez hecha la conexión con la base de datos y haber migrado las tablas, ejecutar por consola para ejecutar los seeder del sistema el siguiente comando: `php artisan db:seed`.

Estos Perfiles precargados son:

- **Administrador:** DNI: 100 / Password: 100
- **Enfermero 1:** DNI: 101 / Password: 101
- **Enfermero 2:** DNI: 102 / Password: 102
- **Gestion:** DNI: 103 / Password: 103



Testeos:

Se testearon los siguientes casos:

- Si el usuario ingresa un DNI o contraseña incorrecto:

**BUENOS AIRES
VACUNATE**

Whoops! Algo salio mal.

- Estas credenciales no coinciden con nuestros registros.

DNI

100

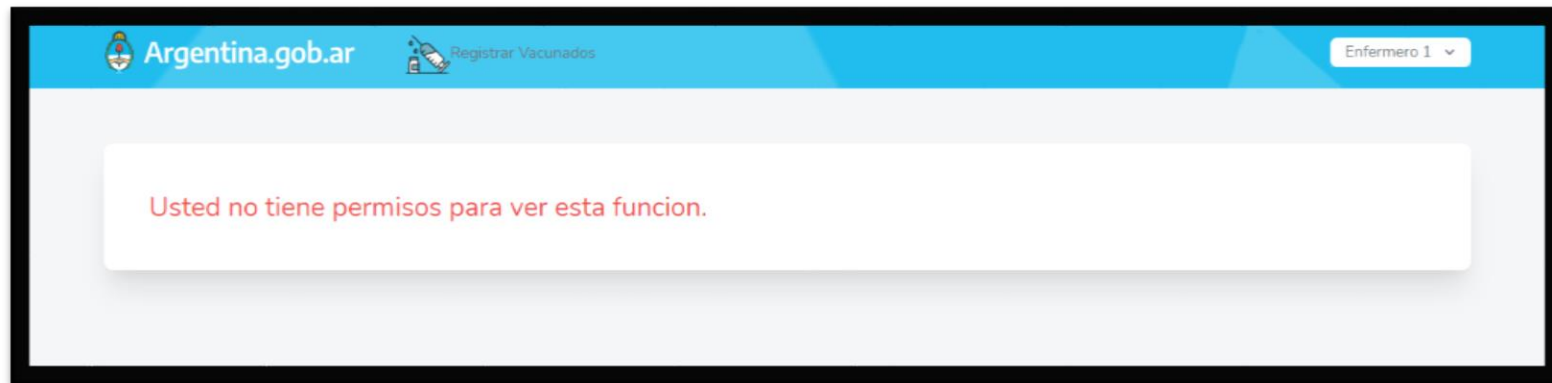
Password

☐ Remember me

[Olvidaste tu Contraseña?](#) **INICIAR SESIÓN**



- Si un usuario intenta acceder a una función que no le corresponde a su rol:



- Si ocurre algún error en la conexión con la Base de datos:





Conclusión:

El desarrollo del proyecto utilizando el patrón MVC, nos brindó no solo una forma de organizar el código, sino que nos permitió organizar el trabajo en equipo, donde cada integrante desarrollo una parte del proyecto independiente del resto, pero orgánicamente respetando el diseño establecido previamente.

Si bien comenzamos, por un error de interpretación, no planteando el desarrollo desde un comienzo con el patrón de diseño MVC pudimos adaptar el proyecto a este requerimiento.

Tuvimos que implementar un plan de contingencia donde evaluamos los diferentes riesgos en cuanto a las soluciones que planteábamos y al tiempo que demandaría su implementación.

Optamos por Laravel, ya que cumplía con los requerimientos y por la experiencia del equipo en el framework.

La anticipación con la cual comenzamos el proyecto y que el trabajo previo realizado pudo ser reutilizado en su gran mayoría, nos permitió cumplir así con los tiempos establecidos entregando un proyecto completamente funcional.