DISEÑO E IMPLEMENTACION DE UN SISTEMA DE CONTROL DE PACIENTES PARA EL CONSULTORIO KARLDENTAL EN LA CIUDAD SARAGURO, AÑO 2023.

AUTOR: **KEVIN DARIO RAMON MALDONADO**

DIRECTOR: MILTON RICARDO PALACIOS MOROCHO MGTR.

INFORME DE TESIS PREVIO A LA OBTENCION DEL TITULO DE TECNÓLOGO EN ANÁLISIS DE SISTEMAS.

LOJA-ECUADOR

2022-2023

CERTIFICACIÓN

Mgtr.

Milton Ricardo Palacios Morocho

DOCENTE DEL INSTITUTO SUPERIOR TECNOLÓGICO "LOS ANDES"

CERTIFICA:

Que el presente trabajo de investigación, previo al obtener el título de Tecnólogo en Análisis de Sistemas, cuyo tema es: **DISEÑO E IMPLEMENTACION DE UN SISTEMA DE CONTROL DE PACIENTES PARA EL CONSULTORIO KARLDENTAL EN LA CIUDAD SARAGURO, EN EL PERIODO 2023** ha sido realizado bajo conducción, control y supervisión, por lo que autorizo su presentación, sustentación y defensa.

Loja, septiembre de 2023

Firmado electrónicamente por:
MILTON RICARDO
PALACIOS MOROCHO

Milton Ricardo Palacios Morocho Mgtr

DIRECTOR

AUTORÍA

Yo, **Kevin Darío Ramón Maldonado**, declaró ser el autor del presente trabajo de tesis y eximo expresamente al **Instituto Superior Tecnológico "Los Andes"** y a sus representantes jurídicos de posibles reclamos o acciones legales, por el contenido de la misma.

Adicionalmente acepto y autorizo al **Instituto Superior Tecnológico "Los Andes"**, la publicación de mi tesis en el Repositorio – Biblioteca Virtual.

KEVIN DARÍO RAMÓN MALDONADO

AUTOR

C.I.: 1150613923

AGRADECIMIENTO

Agradezco profundamente al profesor, Ing. Milton Palacios, director de este trabajo, por su invaluable paciencia y sus comentarios. A los docentes que compartieron sus conocimientos durante este periodo de educación y su personal Administrativo del Instituto Tecnológico “Los Andes”, sin olvidarme del Tnlgo. John Asanza quien estuvo también presente en este proyecto de tesis dando sus comentarios junto con sus experiencias.

También agradezco a mis compañeros, por su ayuda en las clases, las sesiones de retroalimentación nocturnas.

Por último, agradezco a mis amigos junto con mi hermana por haber estado presente en el desarrollo de este trabajo, gracias por el apoyo que me brindaron. También me gustaría agradecer a mis mascotas quienes estuvieron conmigo en cada desvelada.

KEVIN DARÍO RAMÓN MALDONADO

AUTOR

DEDICATORIA

Hay un grupo de personas con las cuales siempre estaré agradecido por haberme apoyado para terminar de realizar este proyecto de tesis, siempre estaré agradecido por la ayuda que me brindaron.

A mi hermana quien es la responsable de mi crecimiento personal como profesional, quien ha sabido apoyarme en todos los ámbitos que ella ha podido, a pesar de que no ha recibido todos los agradecimientos que se merece.

Me hubiera gustado nombrar a cada uno de los profesores como compañeros que supieron ayudar durante este trayecto.

KEVIN DARÍO RAMÓN MALDONADO

AUTOR

ÍNDICE

CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE TABLAS

RESUMEN

Cuando se inicia un negocio que está planificado a irse expandiendo, se piensa en la implementación de sistemas que permitan llevar un control de su negocio de forma más ágil y eficiente, desde pequeñas empresas, así como locales comerciales de productos de primera necesidad hasta las grandes empresas llegan a implementar un sistema de gestión.

A la hora de llevar lo que es un inventario, manejo de personal, ventas, citas a clientes, se inicia de una forma no muy ordenada. Ahí es donde nace la necesidad de ir implementado distinto tipos de sistema de control que les permita agilizar sus tareas y llevar un control ordenada de su negocio,

Es por ello por lo que se realiza el presente trabajo investigativo denominado: "**DISEÑO E IMPLEMENTACION DE UN SISTEMA DE CONTROL DE PACIENTES PARA EL CONSULTORIO KARLDENTAL EN LA CIUDAD SARAGURO, EN EL PERIODO 2023**"

Para poder llevar a cabo el presente trabajo investigativo, se plantearon objetivos claros que permitan analizar los requerimientos funcionales y no funcionales del sistema a implementar mediante la recolección de requerimientos y el análisis del consultorio KARLDENTAL para el desarrollo del sistema web.

Esto ha permitido desarrollar la arquitectura del sistema KARLDENTAL, la codificación e implementación del sistema en un servidor web que será usado por la Odont. Karla Betancourt y el utilizar la metodología KANBAN para el desarrollo del proyecto.

La metodología usada para facilitar este estudio abarca el método científico que permite plantear la hipótesis, el método analítico que facilita en análisis de la investigación general, el método cuantitativo, que permite analizar los resultados numéricos, los métodos deductivo e inductivo que ayudan a llegar a las conclusiones y recomendaciones y el uso de la metodología ágil Kanban para el desarrollo del sistema el mismo que brindó agilidad y flexibilidad en la gestión de proyecto.

Se usó además como métodos de investigación la entrevista que brindó la oportunidad de conocer los requerimientos de la Odont Karla Betancourt; la observación, que mostró a simple vista las necesidades principales del consultorio la entrevista que permitió conocer de primera mano los requerimientos y pormenores del consultorio KARLDENTAL. Así, se concluyó con la implementación del sistema funcional, que permite a la Odont Karla Betancourt, manejar de una forma más eficiente su flujo de trabajo

ABSTRACT

When starting a business with plans for expansion, one considers the implementation of systems that allow for more agile and efficient control of their business. This applies to small businesses and essential goods retail stores, all the way up to large enterprises that implement a management system.

When it comes to managing inventory, personnel, sales, and customer appointments, it often starts in a some what disorganized manner. This is where the need arises to implement various types of control systems to streamline tasks and maintain orderly control of the business.

Hence, this investigative work is carried out under the title: "DESIGN AND IMPLEMENTATION OF A PATIENT MANAGEMENT SYSTEM FOR KARLDENTAL CLINIC IN THE CITY OF SARAGURO, DURING THE YEAR 2023."

To carry out this investigative work, clear objectives were set to analyze the functional and non-functional requirements of the system to be implemented through the collection of requirements and the analysis of the KARLDENTAL clinic for the development of the web system.

This has allowed for the development of the KARLDENTAL system architecture, coding, and implementation on a web server that will be used by Dr. Karla Bentacourt, as well as the use of the KANBAN methodology for project development.

The methodology used to facilitate this study encompasses the scientific method, which allows for hypothesis formulation, the analytical method that aids in the analysis of the overall research, the quantitative method for analyzing numerical results, deductive and inductive methods to reach conclusions and recommendations, and the

use of the agile Kanban methodology for system development, providing agility and flexibility in project management.

Additionally, research methods included interviews, which provided insight into the requirements of Dr. Karla Bentacourt, and observations, which revealed the primary needs of the clinic. These interviews allowed for firsthand knowledge of the requirements and details of the KARLDENTAL clinic. As a result, the implementation of a functional system was concluded, enabling Dr. Karla Bentacourt to manage her workflow more efficiently.

INTRODUCCIÓN

El presente apartado delimitará los diversos capítulos que el lector podrá visualizar dentro de la presente investigación.

Capítulo I:

Se podrá observar los diversos los métodos y técnicas que fueron utilizados para el desarrollo de la investigación, cada proceso fue utilizado minuciosamente a fin de que se cumpla cada una de sus etapas propuestas en el trabajo, los métodos que se utilizó tanto para el desarrollo de la tesis como el documento y sistema implementado.

Capítulo II:

Se encontrará lo referente a los resultados desarrollados de la implementación del sistema es decir su codificación, sus requerimientos y funciones que se encuentran en este apartado representado mediante tablas e imágenes.

Capítulo III:

Finalmente se podrá encontrar las respectivas conclusiones y recomendaciones que son basadas en los objetivos planteados al inicio del desarrollo del proyecto.

CAPÍTULO I: DE LA METODOLOGÍA

Metodología

Métodos

Método Científico: Este método se utilizó en la redacción del problema y además se sustentó en el alcance.

Método Analítico: Se utilizó desde la descripción de la problemática, la redacción del sumario, el establecimiento de la metodología y el análisis de la información para obtener los requerimientos del sistema.

Método Inductivo: Se utilizó este método para dar un enfoque general de los patrones que se repetían en el consultorio.

Método Cualitativo: Este método se utilizó para el análisis de los resultados que sirvieron de base para plantear las recomendaciones y conclusiones.

Método Cuantitativo: Este método se utilizó en el apartado de población y muestra, donde, una cierta cantidad de personas fueron seleccionadas para recopilar información y para realizar pruebas del sistema.

Técnicas de investigación.

La Observación: Mediante esta técnica se realizó la observación de diferentes características que poseen otros sistemas que se dedican al control de pacientes como el control de historial clínico.

La Entrevista: Mediante esta técnica se realizó una recopilación de datos, que ayudaron a identificar cómo sería el diseño de la interfaz para el registro de pacientes, estos datos se recopilaron haciendo una serie de preguntas a la Doc. Karla Betancourt dueña del consultorio Kardental.

Instrumentos

Cuestionario: Este se usó en la obtención de datos que se realizaron tanto como en la encuesta y en la entrevista.

Apuntes: Este instrumento se usó en la obtención de datos a través del método de observación.

Metodología de desarrollo de software Kanban.

Kanban es una metodología de gestión de proyectos que brinda a los gerentes de proyectos total transparencia en el proceso de gestión de tareas. Consta de principios, prácticas, tableros Kanban y tarjetas Kanban (Gilibets, 2023).

Según (Martins, 2022) añade que: La metodología Kanban se implementa por medio de tableros Kanban. Se trata de un método visual de gestión de proyectos que permite a los equipos visualizar sus flujos de trabajo y la carga de trabajo. En un tablero Kanban, el trabajo se muestra en un proyecto en forma de tablero organizado por columnas.

Las fases que se debe seguir para implementar el sistema de manera correcta, ordenada y coherente son:

Fase 1: Planificación;

- Visualización del flujo de trabajo
- División de requerimientos en funcionales y no

funcionales Fase 2: Diseño

- Realización de diagrama de clases de los requerimientos funcionales
- Diseño de interfases
- Diseño de prototipo de la Api
- Diseño de prototipo del sistema

“Frontend” Fase 3: Codificación

- Se usará el lenguaje de programación Python, JavaScript
- Utilización del framework de Django REST framework
- Utilización de la librería

React Fase 4: Pruebas

- Se utilizará el cuestionario donde se elabora preguntas para el administrador del sistema valide su funcionamiento

Fase 5: Lanzamiento

- El servidor se lanzará de manera local para probar su última versión y posteriormente ponerlo en línea.

Población y Muestra

Población y muestra: En el presente proyecto, la población elegida fue un número específico pacientes que llevan el tratamiento de Ortodoncia, además de la propietaria del mismo

Muestra: Como muestra para este trabajo se tomó en cuenta los pacientes que llevan su tratamiento de manera continua

Tipo de muestreo

Muestreo no probabilístico

El autor (Westreicher, 2022) explica sobre el muestreo no probabilístico y el tipo que se usará:

El muestreo no probabilístico es aquel donde no todos los sujetos de la población estadística tienen la misma probabilidad de ser elegidos para formar parte del estudio que se está desarrollando.

Existen varios tipos de muestreo no probabilístico, pero en esta investigación se usará el tipo de muestreo

En este caso, se hace uso del muestreo no probabilístico porque se busca cumplir con una determinada condición. De modo que la muestra total tenga la misma distribución de características que se supone que existen en la población

Tabla 1: Muestra

Descripción	Total	Muestra Utilizada	Actividad
Pacientes	100	50	Recolección de información
Pacientes Gerente	100 1	25 1	Pruebas del sistema pruebas del sistema
Administrador	1	1	Pruebas de caja blanca

Autor: Kevin Ramón
Fuente: Del sistema aplicado

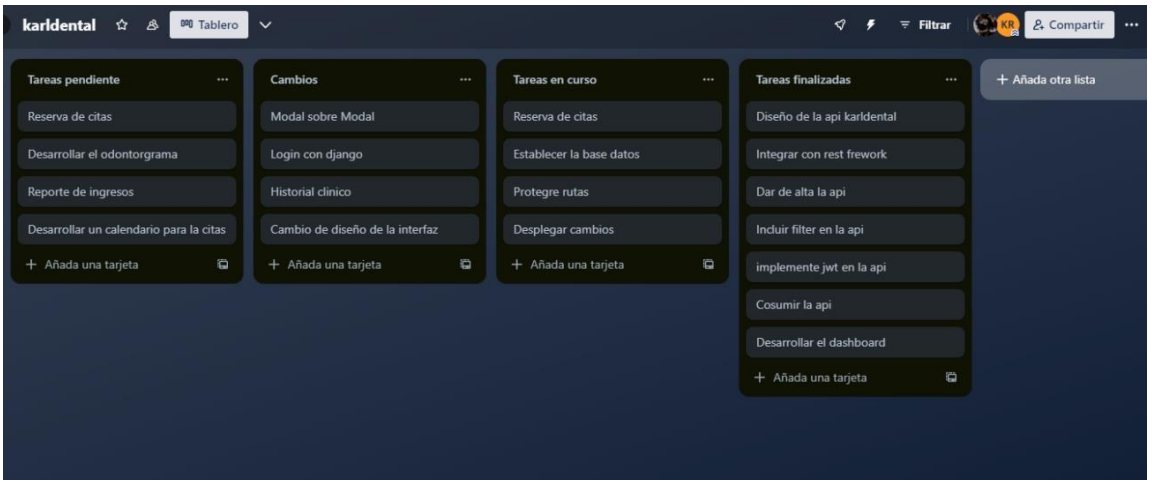
CAPÍTULO II: DESARROLLO DE LA APLICACIÓN

RESULTADOS

Resultados.

EL tablero Kanban

Ilustración 1: Tablero Kanban y tarjetas



Nota: imagen de trello.com (https://trello.com/b/oAikwoNN/karldental)

Fase de planificación

Proceso de desarrollo del sistema kardental, desarrollado bajo la metodología KANBAN.

Requerimientos funcionales

tabla 2 Requerimientos

Número de requisito	R001
Nombre de requisito	Registro del administrador-a
Tipo	☒ Requisito ☐ Restricción
Fuente del requisito	Administrador
Prioridad del requisito	☒ Alta/Esencial ☐ Media/Deseado ☐ Baja/Opcional

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Tabla 3 Requerimientos

Número de requisito	R002		
Nombre de requisito	Registro de pacientes		
Tipo	[X] Requisito	[ ] Restricción	
Fuente del requisito	Administrador		
Prioridad del requisito	[X] Alta/Esencial	[ ] Media/Deseado	[ ] Baja/Opcional

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Tabla 4 Requerimientos

Número de requisito	R003		
Nombre de requisito	Registro de historial clínico		
Tipo	[X] Requisito	[ ] Restricción	
Fuente del requisito	Administrador		
Prioridad del requisito	[X] Alta/Esencial	[ ] Media/Deseado	[ ] Baja/Opcional

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Tabla 5 Requerimientos

Número de requisito	R004		
Nombre de requisito	Registro de tratamientos		
Tipo	[X] Requisito	[ ] Restricción	
Fuente del requisito	Administrador		
Prioridad del requisito	[X] Alta/Esencial	[ ] Media/Deseado	[ ] Baja/Opcional

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Tabla 6 Requerimientos

Número de requisito	R005		
Nombre de requisito	Control de pagos		
Tipo	[X] Requisito	[ ] Restricción	
Fuente del requisito	Administrador		
Prioridad del requisito	[X] Alta/Esencial	[ ] Media/Deseado	[ ] Baja/Opcional

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Tabla 7 Requerimientos

Número de requisito	R006		
Nombre de requisito	Agendar citas		
Tipo	[X] Requisito	[ ] Restricción	
Fuente del requisito	Administrador		
Prioridad del requisito	[X] Alta/Esencial	[ ] Media/Deseado	[ ] Baja/Opcional

Autor: Kevin Ramón
Fuente: Del sistema aplicado

REQUERIMIENTO NO FUNCIONALES

Tabla 8 Requerimientos

Número de requisito	R007		
Nombre de requisito	Diseño de interfaz del Sistema (no funcional)		
Tipo	[X] Requisito	[ ] Restricción	
Fuente del requisito	Administrador		
Prioridad del requisito	[ ] Alta/Esencial	[X] Media/Deseado	[ ] Baja/Opcional

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Tabla 9 Requerimientos

Número de requisito	R008		
Nombre de requisito	Accesibilidad del Sistema (no funcional)		
Tipo	[X] Requisito	[ ] Restricción	
Fuente del requisito	Administrador		
Prioridad del requisito	[ ] Alta/Esencial	[X] Media/Deseado	[ ] Baja/Opcional

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Tabla 11. Prioridades de requerimientos

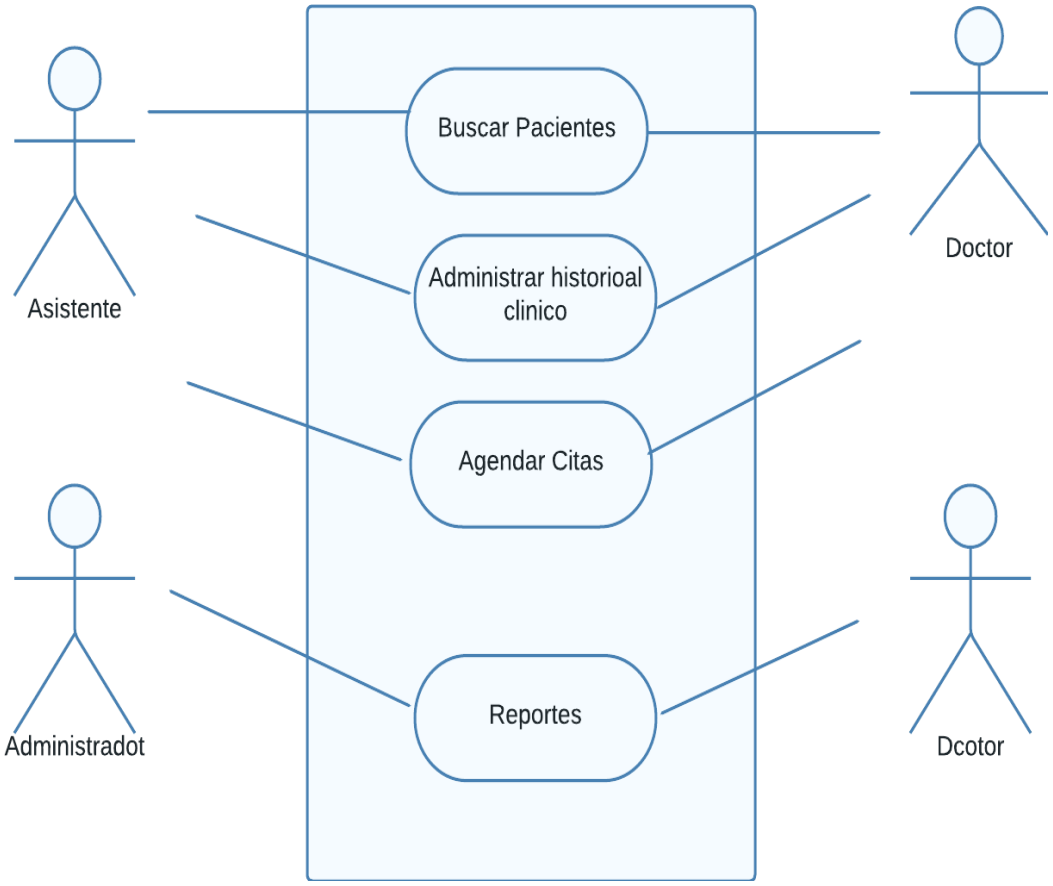
Código	Requerimiento	Prioridad
R001	Registro del administrador	1
R002	Registro de pacientes	1
R003	Registro de historial clínico	1
R004	Registro de Tratamientos	1
R005	Control de pagos	1
R006	Agendar citas	1
R007	Diseño de interfaz del Sistema (no funcional)	2
R008	Accesibilidad del Sistema (no funcional)	2

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Interacción 1:
Registro del administrador/a

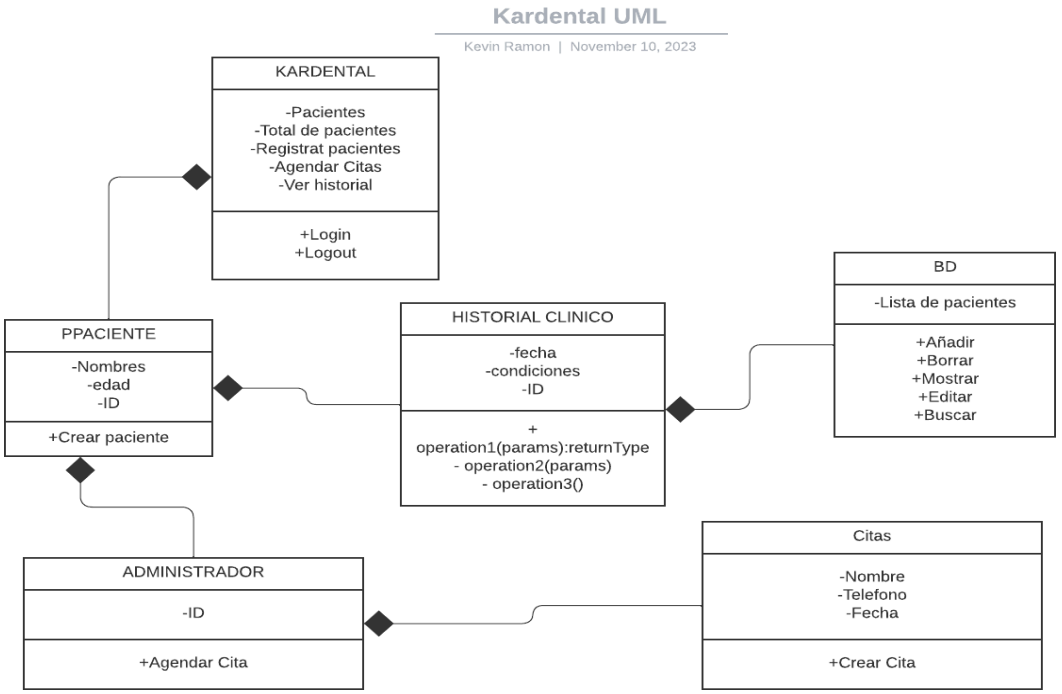
Diseño. –

Figura: 1 Diagrama Casos de Uso



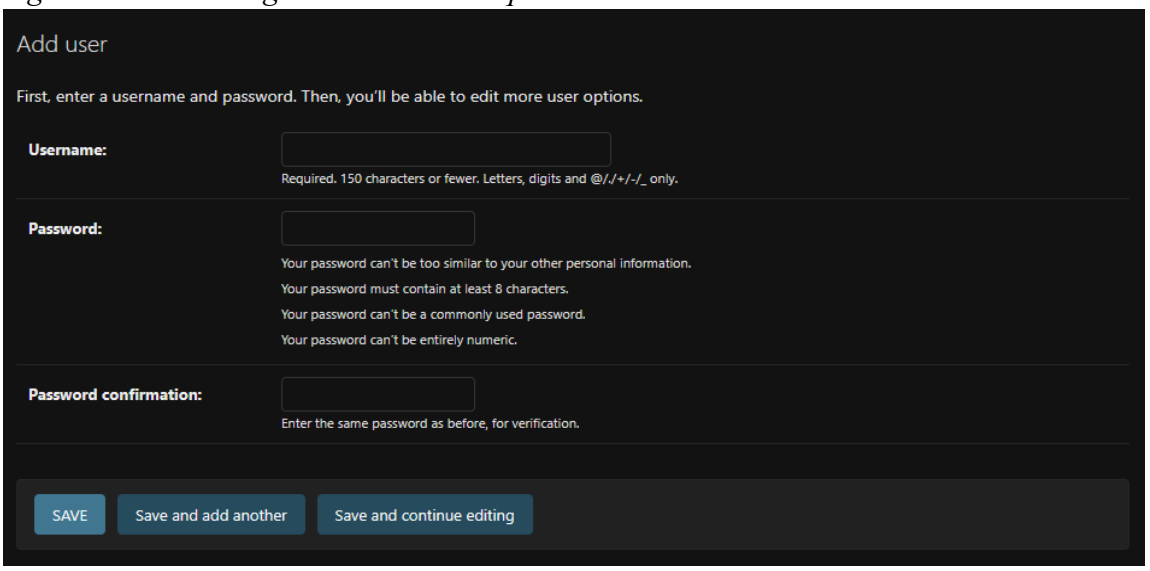
Autor: Kevin Ramón
Fuente: Del sistema aplicado

Figura 2. Diagrama de clases



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Figura 3. Vista de registro de usuarios que van a usar el sistema



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Programación. –

Se empezó por programar la API en el lenguaje de programación Python con su framework django-rest, para lo que es la base de datos se usó su propio ORM de Django por lo cual no tuvimos que crear una colección de tablas desde cero.

```
class CustomUser(BaseUserManager):
    # Agrega campos personalizados aquí
    birth_date = models.DateField(null=True, blank=True) profile_picture =
models.ImageField(upload_to='profile_pics/',  null=True, blank=True)

    def __str__(self):
        return self.username
```

Figura: 4. *Código para el registro de usuarios que van a usar el sistema*
Autor: Kevin Ramón
Fuente: Del sistema aplicado

En esta interacción la usamos el modelo dado por Django para manejar el control de usuarios

Tabla 12. Pruebas de la interacción 1

interacción 1	**Explicación**
class CustomTokenObtainPairView(TokenObtainPairView): serializer_class = CustomTokenObtainPairSerializer	Cómo estamos manejando lo que es una api con django-rest, se hace el uso de la librería JWT para permitir una autenticación controlada

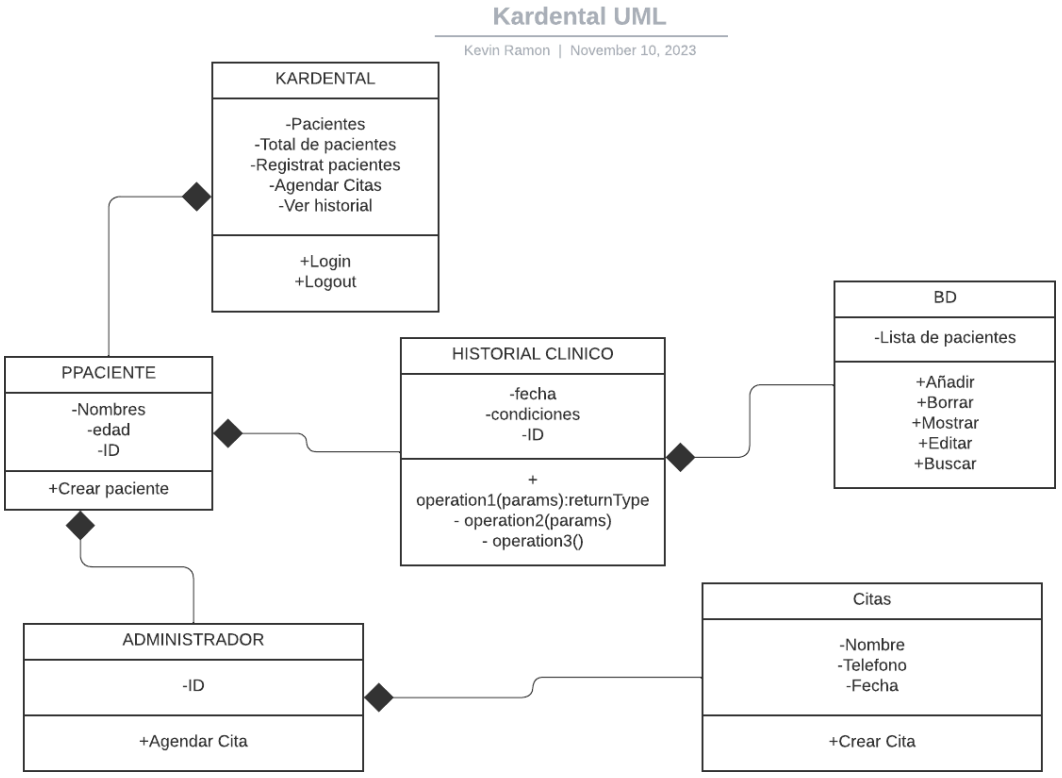
Autor: Kevin Ramón
Fuente: Del sistema aplicado

Interacción 2:

Registro de paciente al sistema

Diseño. –

Figura: 5. Diagrama de clases



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Figura: 6 Vista para poder registrar nuevos pacientes

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Programación. –

Se programó en el lenguaje de programación Python junto con su ORM, se usó el lenguaje JS con su librería React para poder desarrollar la vista.

Figura: 7. Código para el registro de pacientes
Django

```
class Patient(models.Model):
    first_name = models.CharField(max_length=100) last_name =
    models.CharField(max_length=100) address =
    models.CharField(max_length=100) card =
    models.IntegerField(blank=True) birth_date =
    models.DateField(blank=True) appointment_date =
    models.DateTimeField() notes = models.TextField(blank=True)

    def __str__(self) -> str:
        return f"Nombre: {self.first_name}, Apellido:
{self.last_name}, Nacimiento: {self.birth_date}"
```

React

```
const handleSubmit = async (event) => {
    event.preventDefault();
    const newPatient = { first_name:
        firstName, last_name:
        lastName, address: address,
        card: card, birth_date:
        birthDate,
        medical_conditions: medicalConditions, medicines:
        medicines, appointment_date: appointment_date,
        notes: notes,
    };
    try {
        await addPatient(newPatient);
        setFirstName(""); setLastName("");
        setBirthDate(""); setAddres("");
        setCard(""); setAppoinment("");
        setMedicalConditions("");
        setMedicines(""); setNotes("");
        onRequestClose(true);
    } catch (error) {
        console.log("Error adding patient: ", error);
    }
};
```

Autor: Kevin Ramón
Fuente: Del sistema aplicado

En esta interacción donde se comunica Django con React

Pruebas. –

Tabla 13. Pruebas de la interacción 2

Interacción 2	**Explicación**
export const addPatient = (patientId) => patientsApi.post("/", patientId); Con un try catch validamos la información enviada try { await addPatient(newPatient); setFirstName(""); setLastName(""); setBirthDate(""); setAddres(""); setCard(""); setAppoinment(""); setMedicalConditions(""); setMedicines(""); setNotes(""); onRequestClose(true); } catch (error) { console.log("Error adding patient: ", error); } };	Se hace una petición post con axios desde React para que django-rest pueda guardar la información mandada por el método post

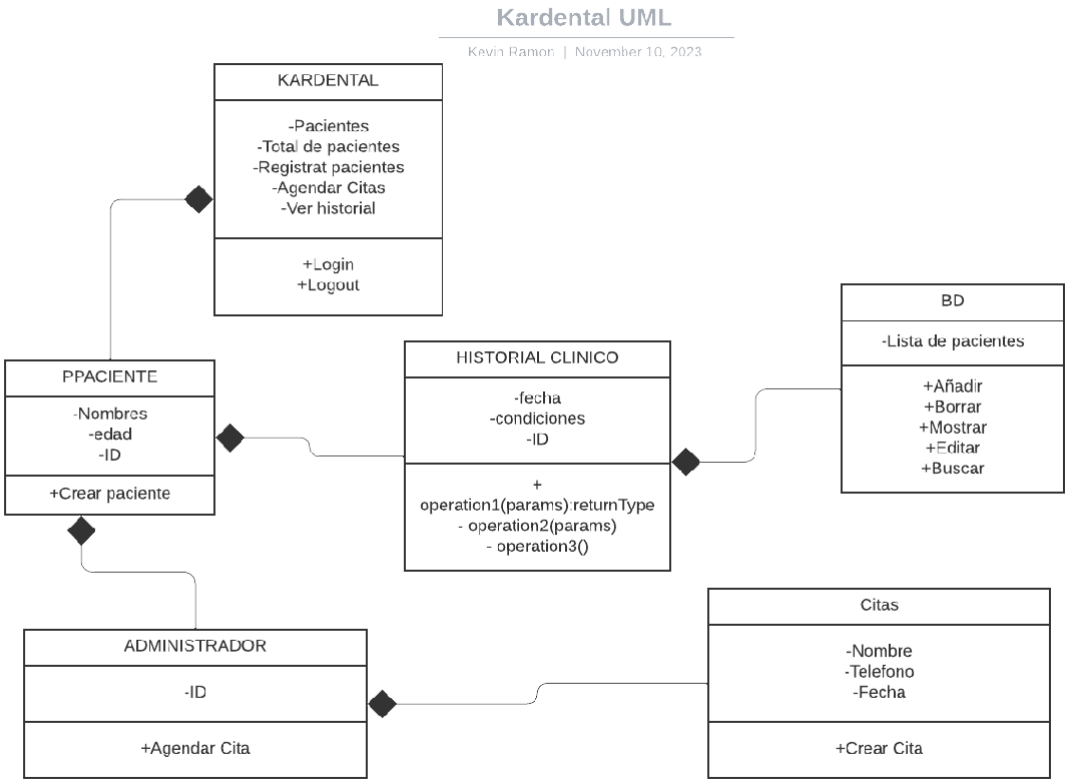
Autor: Kevin Ramón
Fuente: Del sistema aplicado

Interacción 3:

Registro de historial clínico

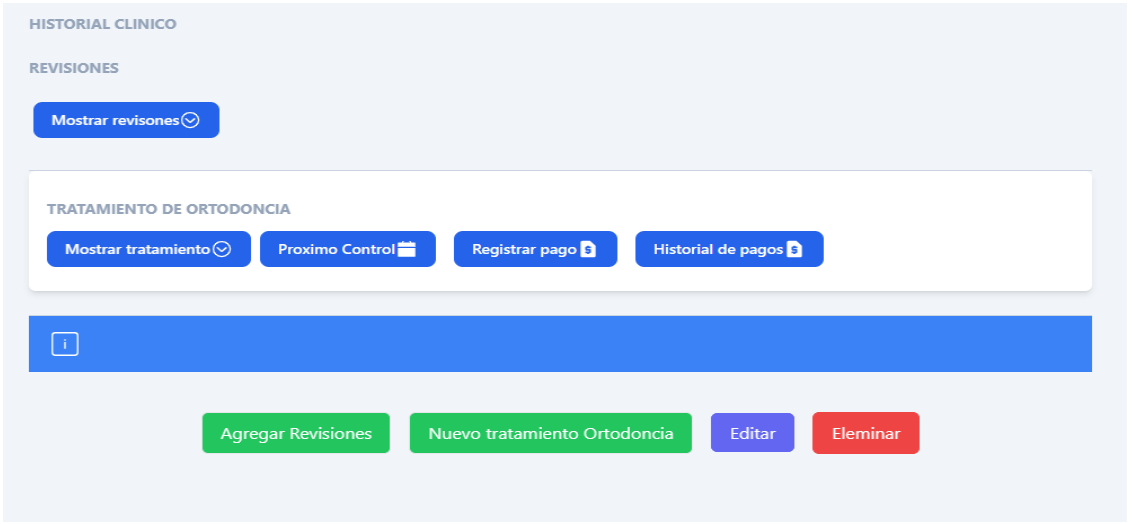
Diseño. -

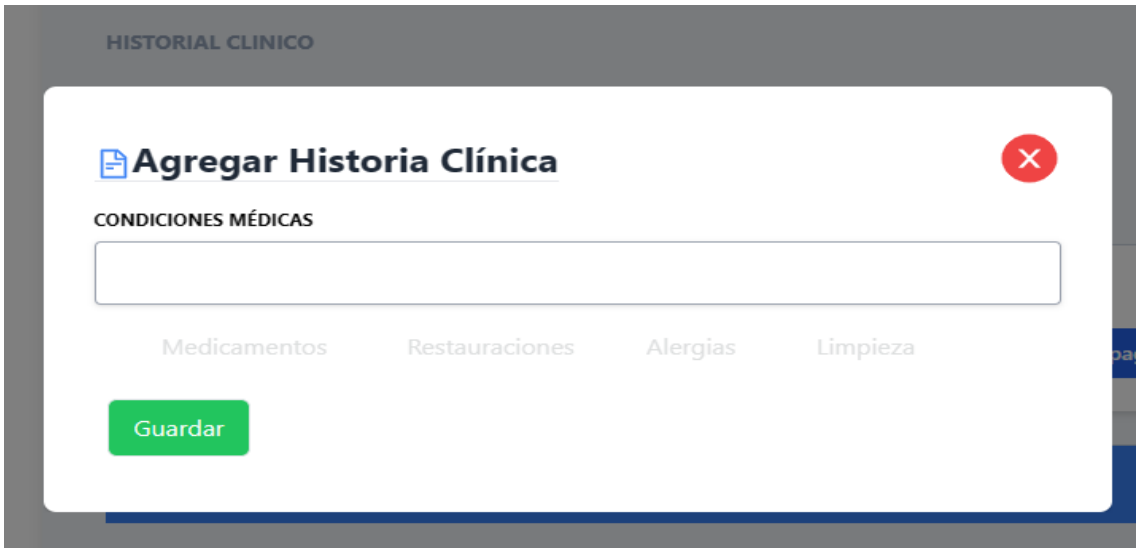
Figura: 8 Diagrama de clases



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Figura: 9 Vista para agregar historial clínico por paciente





Autor: Kevin Ramón
Fuente: Del sistema aplicado

Programación. –

Se programó en el lenguaje de programación Python junto con su ORM, se usó el lenguaje JS con su librería React para poder desarrollar la vista.

Figura: 10. Código para el registro de historial clínico

```
class ClinicHistory(models.Model): patient =
    models.ForeignKey(Patient,
on_delete=models.CASCADE)
    medical_conditions = models.TextField(blank=True) medicines =
    models.TextField(blank=True) allergies =
    models.TextField(blank=True) clean_tooth =
    models.TextField(blank=True) restoration =
    models.TextField(blank=True)
    visit_date = models.DateTimeField(auto_now_add=True)
    # Agrega campos relacionados a pagos
    payment_date = models.DateTimeField(blank=True, null=True) payment_amount =
    models.DecimalField(
        max_digits=10, decimal_places=2, blank=True, null=True
    )

        def __str_(self) -> str:
            return f"Historial clínico de {self.patient.first_name}
        {self.patient.last_name} - Fecha de Visita: {self.visit_date}"


class OrthodonticTreatment(models.Model): BRACKET_CHOICES = [
        ('metal', 'Metal'),
        ('ceramic', 'Ceramic'),
        ('lingual', 'Lingual'),
    ]
    patient = models.ForeignKey(Patient, on_delete=models.CASCADE)
    # Información sobre el tratamiento de ortodoncia
    treatment_start_date = models.DateField()
    treatment_end_date = models.DateField(blank=True, null=True) treatment_type =
    models.CharField(max_length=100,
choices=BRACKET_CHOICES)
    treatment_description = models.TextField(blank=True)

    # Información financiera
    treatment_cost = models.DecimalField(
        max_digits=10, decimal_places=2, blank=True, null=True
    )
    monthly_payment = models.DecimalField(
        max_digits=10, decimal_places=2, blank=True, null=True
```

```
    )


    # Información de pagos y revisiones mensuales
    payments_and_reviews = models.ManyToManyField( "PaymentAndReview",
        blank=True,
related_name="orthodontic_treatments"
    )

    def __str__(self):
        return f"Tratamiento de ortodoncia para
{self.patient.first_name} {self.patient.last_name}"

class NextControl(models.Model): patinet =
    models.ForeignKey(Patient,
on_delete=models.CASCADE)
    date = models.DateField(blank=True) notes =
    models.TextField(blank=True)


class Payments(models.Model):
    patient = models.ForeignKey(Patient, on_delete=models.CASCADE)
    payment_date = models.DateField(blank=True) payment_amount =
    models.IntegerField(blank=True)
```

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Figura: 11. Código de React para agregar el historial medico

```
<AddClinicHistoryModal
  isOpen={isModalOpen}
onRequestClose={() =>
setIsModalOpen(false)}
patientId={patientId}
/>
  <AddOrtodonciaModal
isOpen={isOrtodonciaModalOpen} onRequestClose={()
=> setOrtodonciaIsModalOpen(false)}
patientId={patientId}
/>
```

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Pruebas, -

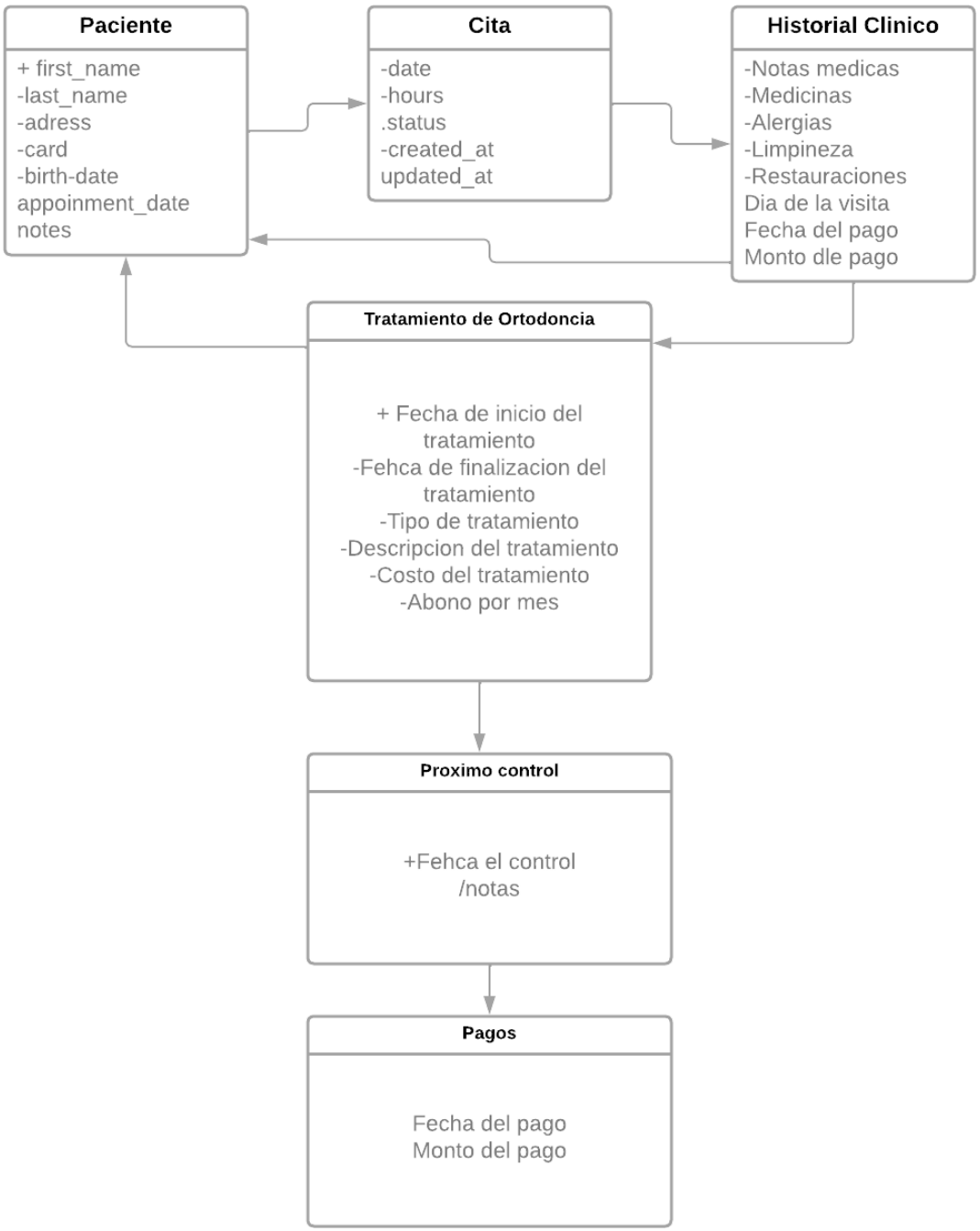
Tabla 14. Pruebas de la interacción 3

Interacción	Explicación
`export const addOrthodontic = (patient) => rthodontic.post("/", patient);` `export const nextrevison = (patientId) => extControl.post("/", patientId);` `export const addClinicHistory = (patientId) => clinicHistoryApi.post("/clinichistory/", patientId);`	Aquí se hacen solicitudes post post a la vistas de django-rest para poder guardar los campos enviados desde React

Autor: Kevin Ramón
Fuente: Del sistema aplicado

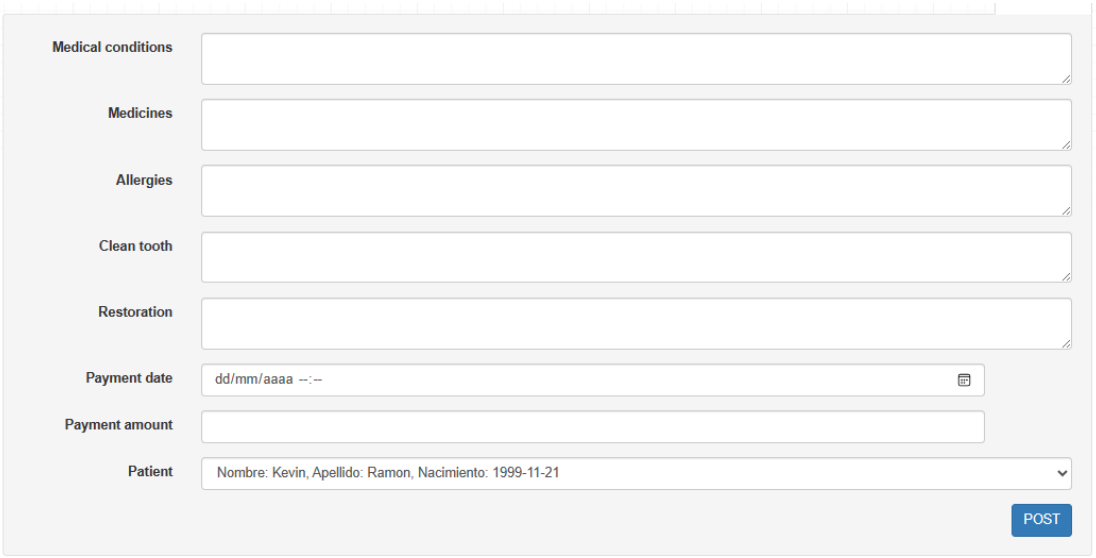
Interacción 4:
Manejo del historial clínico
Diseño

Figura: 12, Diagrama de clases



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Figura: 13. Vista manejo del historial del paciente



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Programación

Se continuó con la programación de la interacción 4, el cual nos permite gestionar lo que es el historial de cada paciente en el sistema.

Figura 14. Código del manejo del historial.

```
class ClinicHistory(models.Model): patient =
    models.ForeignKey(Patient,
on_delete=models.CASCADE)
    medical_conditions = models.TextField(blank=True) medicines =
    models.TextField(blank=True) allergies =
    models.TextField(blank=True) clean_tooth =
    models.TextField(blank=True) restoration =
    models.TextField(blank=True)
    visit_date = models.DateTimeField(auto_now_add=True)
    # Agrega campos relacionados a pagos
    payment_date = models.DateTimeField(blank=True, null=True) payment_amount =
    models.DecimalField(
        max_digits=10, decimal_places=2, blank=True, null=True
    )

    def __str__(self) -> str:
        return f"Historial clínico de {self.patient.first_name}
{self.patient.last_name} - Fecha de Visita: {self.visit_date}"
```

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Pruebas

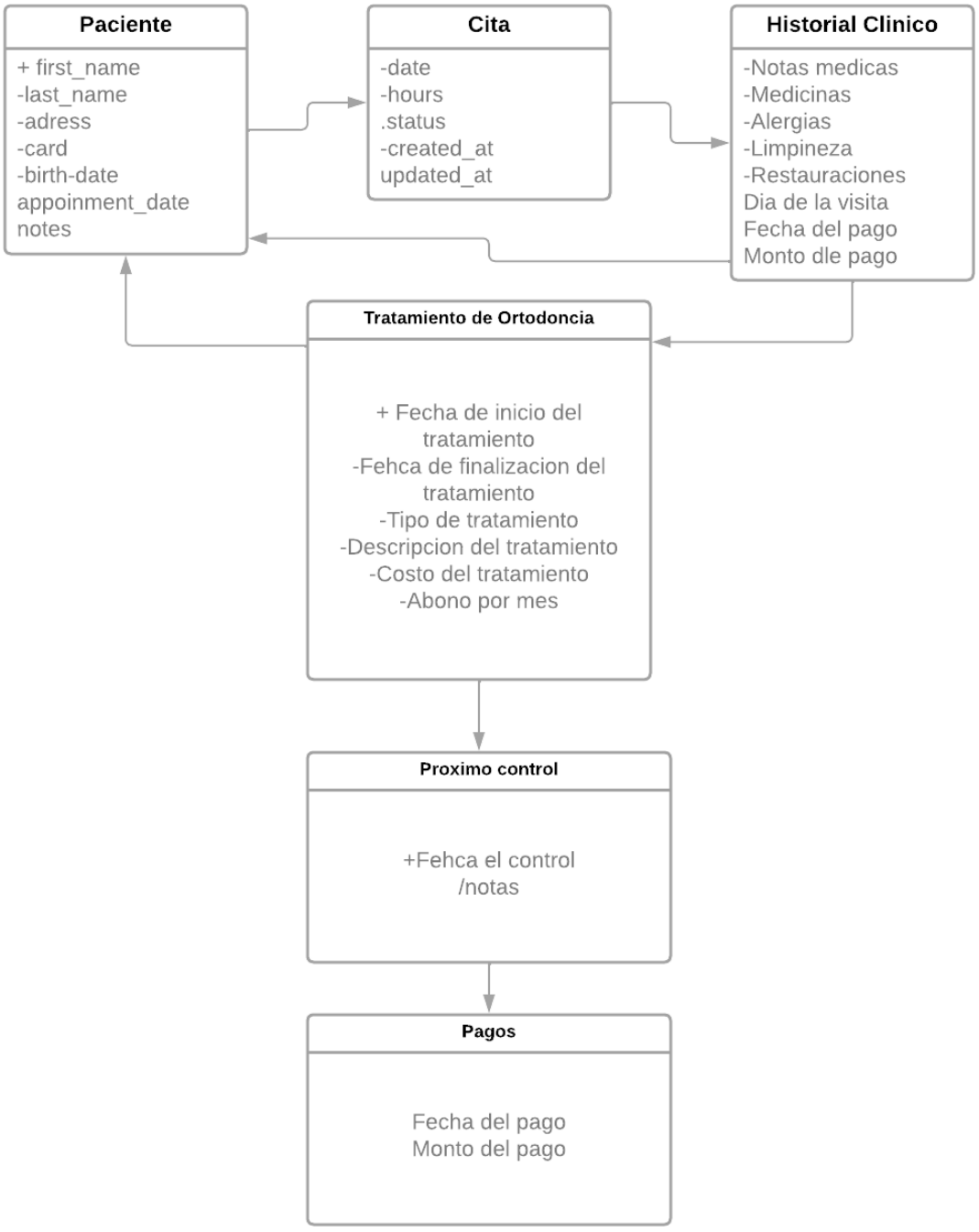
Tabla 15. Pruebas de interacción 4

Interacción	Explicación
`export const getAllClinicHistories = async (patientId) => {` `  try {` `  const response = await clinicHistoryApi.get(` `  `/patients/${patientId}/clinichistory/` `);` ` return response.data;` ` } catch (error) {` ` // If the response contains an error message, return it` ` if (error.response && error.response.data && error.response.data.message) {` ` return { error: error.response.data.message };` ` }` ` throw error; // Re-throw the error if it's not a response with a message` ` }` `};`	Para poder interactuar con el historial clínico primero comprobamos que el paciente tenga historial clínico ha ciento una petición get a nuestra api

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Interacción 5:
Detalles del paciente
Diseño

Figura: 15. Diagrama de clases



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Figura: 16. Vista de detalles del paciente

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Programación

Se continuó con la programación de la interacción 5, el cual nos permitió ver los detalles de cada paciente que haya guardado junto con su historial clínico. A continuación, se muestran las clases de la interacción 5.

```
class HistorieViewset(viewsets.ModelViewSet): queryset =
    ClinicHistory.objects.all() serializer_class =
    ClinicHistorySerializer

    def get_queryset(self):
        patient_id = self.kwargs.get("patient_id") queryset =
ClinicHistory.objects.filter(patient_id=patient_id)
        return queryset

    def get_clinicHistorie_by_patien_id(request, patient_id): clinicHistories =
ClinicHistory.objects.filter(patient_id=patient_id)

        if not clinicHistories:
            return JsonResponse({"message": "No hay historias clinicas que
mostrar"}, status=200)

        serializer = ClinicHistorySerializer(clinicHistories, many=True)
        return JsonResponse(serializer.data, status=200)
```

Figura: 17. Código de los detalles de cada paciente
Autor: Kevin Ramón
Fuente: Del sistema aplicado

Pruebas

Tabla 16. Pruebas de interacción 5

Interacción 5	Explicación
`const fetchHistories = async () => { try { const response = await getHistoriesByPatientId(patientId); setClinicHistories(response.data); } catch (error) { console.error("Error al obtener la hictoria clinica del paciente", error); } }; async function loadData() { try { const patientRes = await getPatientDetails(patientId); setPatientDetails(patientRes.data); } catch (error) { console.log("Error fetching patient details: ", error); } }`	Si tenemos historial de paciente anteriormente guardado hacemos una comprobación a la api que valide los datos para así poder presentar los datos en la vista

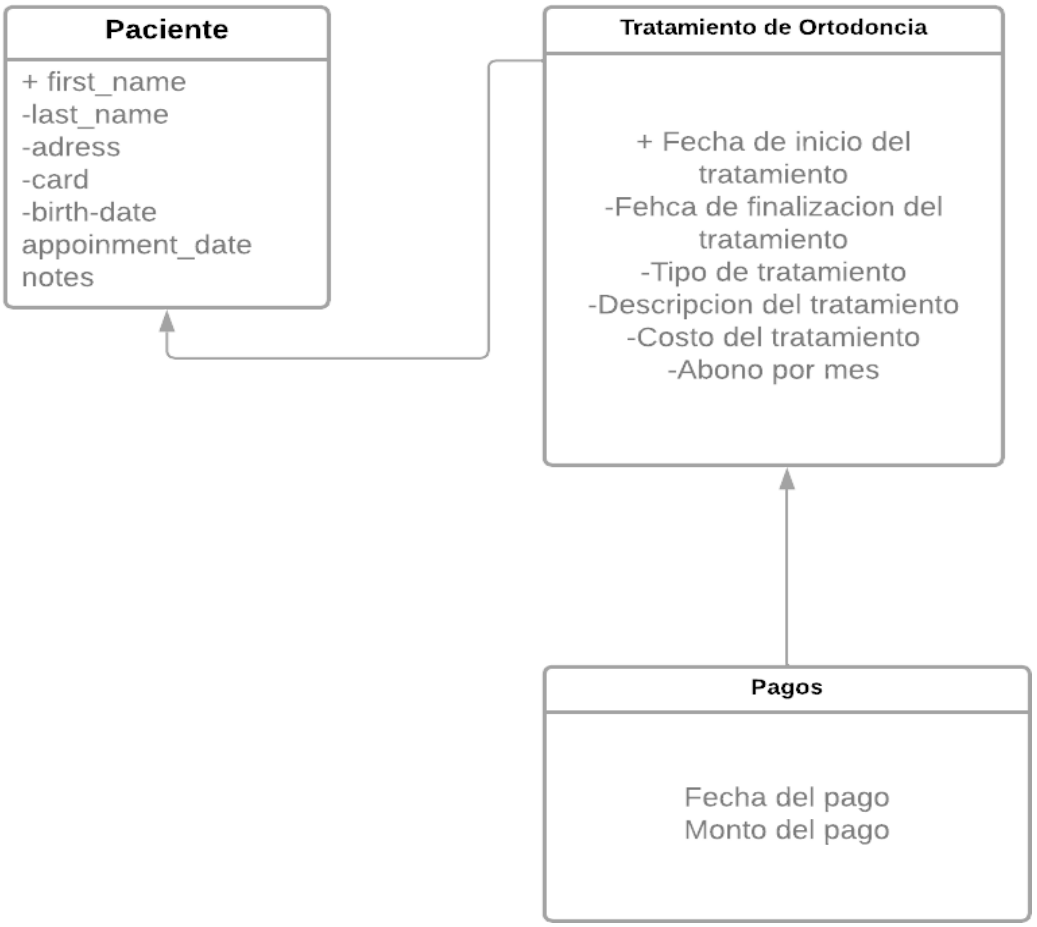
Autor: Kevin Ramón
Fuente: Del sistema aplicado

Interacción 6:

Control de pagos del tratamiento de ortodoncia

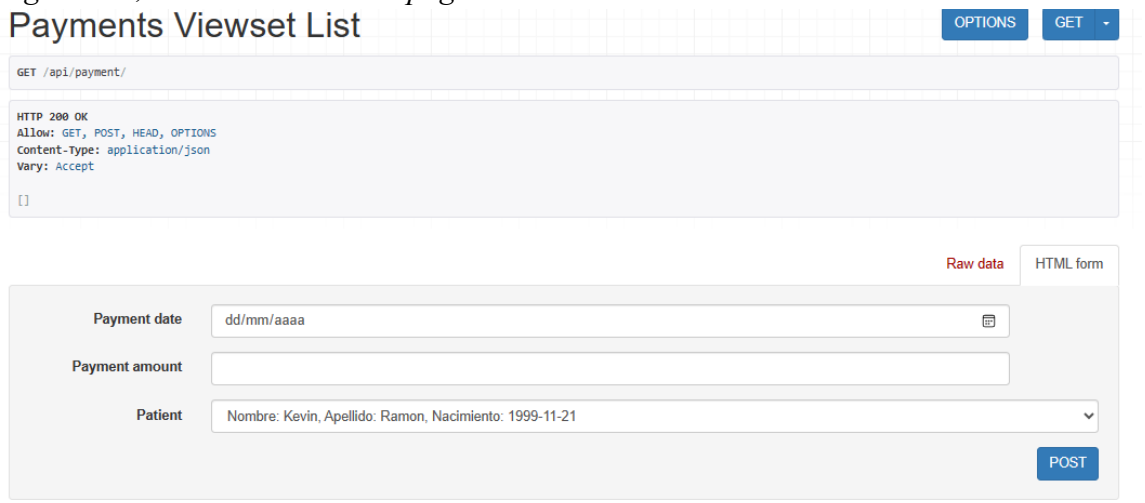
Diseño

Figura: 18. Diagrama de clases



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Figura: 19, Vista del control de pago del tratamiento de ortodoncia



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Programación

Se continuó con la programación de la interacción 6, el cual nos permite registrar los pagos que hace paciente que tenga un tratamiento de ortodoncia. A continuación, se muestra la clase de la interacción 6.

Figura 20. Programación del control de pagos del tratamiento de ortodoncia

```
class Payments(models.Model):
    patient = models.ForeignKey(Patient, on_delete=models.CASCADE)
    payment_date = models.DateField(blank=True) payment_amount =
    models.IntegerField(blank=True)
```

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Pruebas

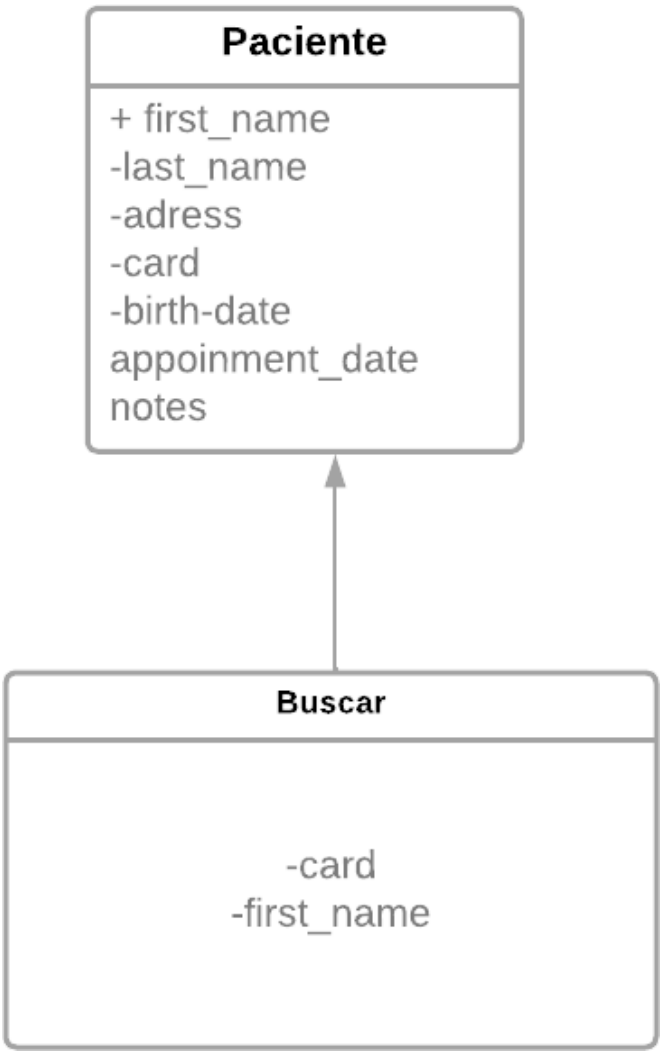
Tabla 17. Pruebas de la interacción 6

interacción 6	Explicación
class PaymentsViewset(viewsets.ModelViewSet): queryset = Payments.objects.all() serializer_class = PaymentsSerializer def get_queryset(self): patient_id = self.kwargs.get("patient_id") queryset = Payments.objects.filter(patient_id=patient_id) return queryset def get_payments_by_patient_id(request, patient_id): payments = Payments.objects.filter(patient_id=patient_id) if not payments: return JsonResponse({"message": "No hay pagos registrados para este paciente."}, status=200)	Esta vista nos permite tener un control sobre los pagos realizados donde podremos guardar los pagos y consultar los pagos que se han hecho desde react

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Interacción 7:
Buscar pacientes dentro del sistema
Diseño

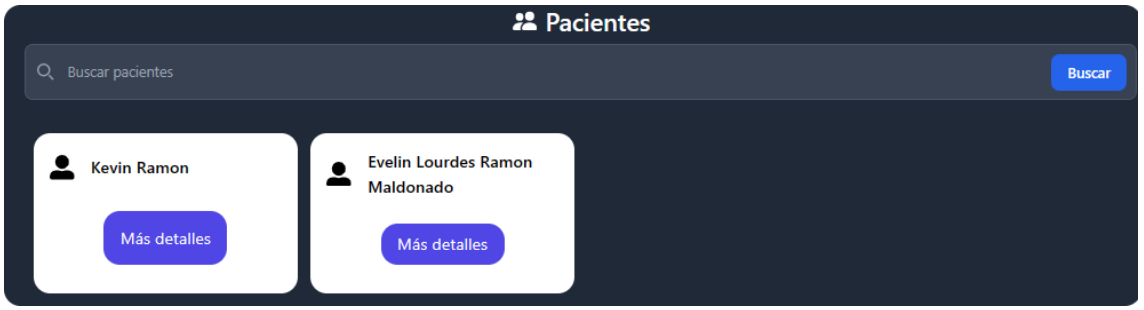
Figura 21. Diagrama de clases



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Figura: 22. Vista de buscar paciente

La vista de busqueda se la realizó directo desde React



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Programación

Se continuó con la programación de la interacción 7, el cual nos permite filtrar desde el backend los pacientes que tenemos ya guardados. A continuación, se el código de la interacción 7

```
const handleSearch = async () => {
    if (card.trim() === "") {
        // Si el campo de búsqueda está vacío, no hacemos la solicitud.
        setSearchResults([]);
        return;
    }

    try {
        // Realizamos la búsqueda cuando se hace clic en el botón. const results =
        await searchPatients(card); setSearchResults(results);
    } catch (error) {
        console.error("Error al buscar pacientes:", error);
    }
};
const handlePatientClick = (patient) => {
    setSelectedPatient(patient);
};
```

Figura: 23. Programación de búsqueda de pacientes
Autor: Kevin Ramón
Fuente: Del sistema aplicado

Pruebas

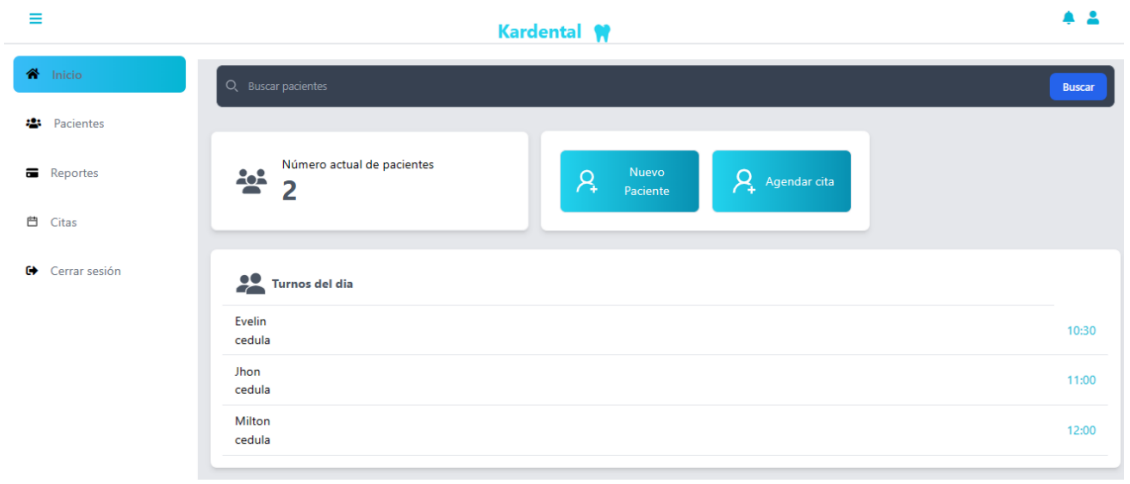
Tabla 18. Pruebas de la interacción 7

Interacción 7	**Explicación**
`` export const addClinicHistory = (patientId) => export const searchPatients = async (card) => { try { const response = await axios.get(`http://localhost:8000/api/patients/?card=${card}`); return response.data; } catch (error) { console.log("Error al buscar pacientes", error); } }; ```	Esta petición nos permite buscar en nuestro api a través de una petición get donde se realiza una búsqueda por el atributo “card” que sería el número de la cedula

Autor: Kevin Ramón
Fuente: Del sistema aplicado

Interacción 8:
Diseño de interfaz del sistema (no funcional)

Figura: 24. Vista del diseño de interfaz del sistema

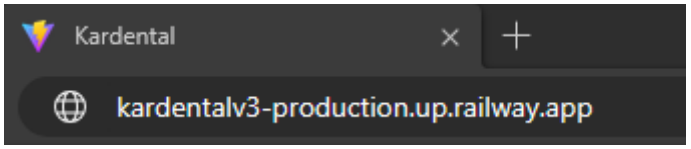


Autor: Kevin Ramón
Fuente: Del sistema aplicado

Para la interacción entre django-rest se desarrolló todo un interfaz en React donde permitía la interacción entre estos nos, donde se pueda evidenciar todo lo que corresponde del sistema el registro de paciente, historial clínico, buscar pacientes, registrar pagos, iniciar sesión en el sistema

Interacción 9:
Accesibilidad del sistema (no funcional)

Figura: 25 Vista del diseño de accesibilidad del sistema



Autor: Kevin Ramón
Fuente: Del sistema aplicado

Para esta interacción se proporcionó una IP y una URL personalizada donde tanto el asistente como el Doctor encargado puedan acceder al sistema sin ningún problema.

FASE DE PRUEBAS

Pruebas

Ahora se pondrá a disposición la encuesta utilizada para el doctor y el administrador sobre el uso y funcionamiento del sistema

Tabla 19. Encuesta a la Doctora del consultorio

Preguntas	Respuestas		Sugerencias
	Si	No	
¿El sistema es intuitivo al momento de usarlo?	X		
¿Los colores usados en el sistema son los adecuados?	X		
¿El sistema es adecuado para el ámbito médico?	X		
¿El sistema adapta el uso de dispositivos móviles?	X		
¿Al registrar un paciente es intuitivo?	X		
¿Cuándo se interactúa con el historial clínico del paciente es intuitivo?	X		

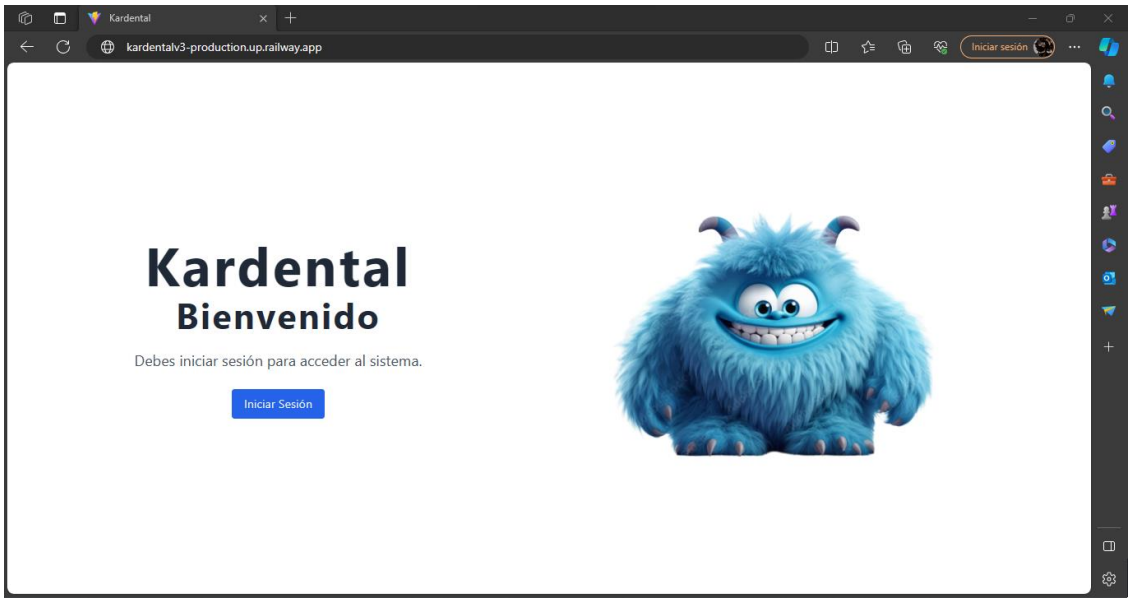
Tabla 20. Encuesta al asistente del consultorio

Preguntas	Respuestas		Sugerencias
	Si	No	
¿El sistema es intuitivo al momento de usarlo?	X		
¿Los colores usados en el sistema son los adecuados?		**X**	
¿El sistema es adecuado para el ámbito médico?	X		
¿El sistema adapta el uso de dispositivos móviles?	X		
¿Al registrar un paciente es intuitivo?	X		
¿Cuándo se interactúa con el historial clínico del paciente es intuitivo?	X		

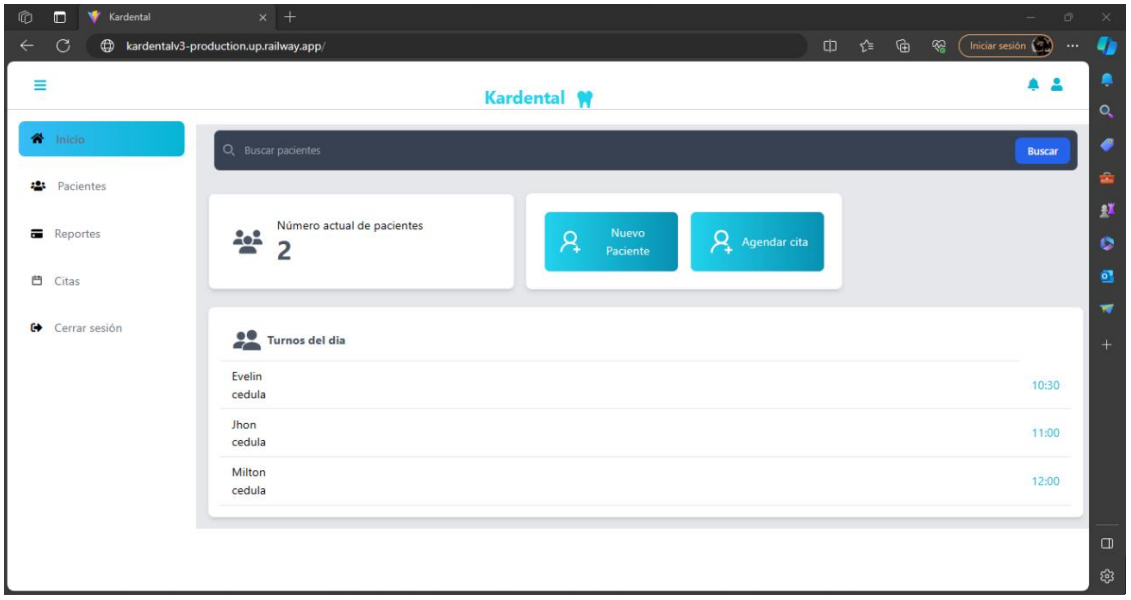
FASE DE LANZAMIENTO

El sistema se encuentra funcionando al 100%, porque es conveniente poder utilizar el sistema

Figura 26. Vista del lanzamiento del sistema



Autor: Kevin Ramón
Fuente: Del sistema aplicado



Autor: Kevin Ramón
Fuente: Del sistema aplicado

HALLAZGOS

- En el desarrollo del presente proyecto de investigación, se llevó a cabo un análisis de las necesidades que presentaba el consultorio Kardental y posteriormente los beneficios de implementar dicho sistema. Los hallazgos obtenidos proporcionan una visión reveladora sobre la automatización de los procesos, su productividad

- El sistema una vez implementado reveló que las tareas de manejo de historial de paciente como su registro se agilizaron, esos procesos eran de los más tardados el buscar un historial clínico, agregar nuevas entradas a la historia clínica del paciente era lo que más dificultad presenta

- El tiempo para la gestión de todos los procesos fue reducido en gran cantidad, ya que los procesos que se realizaban manualmente ahora son automatizados como en el caso de registrar pagos, tener su historial de pagos presentados de forma inmediata

- En temas de lo que es, el historial de pagos había que hacer sus cálculos manualmente y entregarlos por separado al paciente ahora se lo imprime directo con sus pagos realizados junto a las fechas del mismo

CONCLUSIONES

La realización del presente trabajo generó las siguientes conclusiones.

- Se concluye que para lograr un desarrollo de una aplicación de software

  efectivo del presente trabajo de investigación se aplicó una entrevista a la propietaria del negocio con el fin de conocer los requerimientos y necesidades, así mismo se utilizó la técnica de observación que fueron instrumentos fundamentales que permitió conocer las fortalezas y las debilidades del negocio y en base a ello se realizó la propuesta del sistema, para que permita gestionar las tareas más repetitivas del consultorio kardental.

- Al culminar el proyecto se determinó que la metodología de programación

  Kanban y la aplicación del lenguaje de programación Python con su framework django-rest y el lenguaje JS con su librería React contribuyeron a la realización eficaz del sistema y fueron de gran aporte, ya que tienen un proceso detallado a seguir, así mismo permitieron analizar datos automatizar operaciones y crear el sistema de maneras fiables y escalables

- Con la culminación del presente trabajo de investigación se puede concluir que

  en la fase de implementación interactiva tiene como principal actor del sistema la dueña del consultorio y como segundo actor el asistente del consultorio

RECOMENDACIONES

- Se recomienda utilizar herramientas de recolección de información para los requerimientos, que sean parte de las metodologías ágiles de desarrollo de software, estos permiten centrar los requerimientos para obtener un proceso de desarrollo de software adecuado
- Se recomienda trabajar con una metodología que permitir llevar de manera clara las tareas de desarrollo de dicho proyecto en este caso fue la metodología Kanban,
- Se recomienda usar un lenguaje que se esté familiarizado, utilizar un patrón de programación que sea familiarizado para no tener problemas en el desarrollo

BIBLIOGRAFÍA

Ekon, E. (28 de Julio de 2021). *cegid Ekon*. Obtenido de https://www.ekon.es/blog/sistemas-de-gestion-integral-para-el-funcionamiento-optimo-de-la-empresa/

EVALUANDO. (28 de Noviembre de 2019). *ERP.COM*. Obtenido de EVALUADO ERP.COM: https://www.evaluandoerp.com/software-erp/sistema-de-gestion/

Flores, F. (22 de julio de 2022). *OpenWebinars*. Obtenido de https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/#qu%C3%A9-es-visual-studio-code

Gilibets, L. (12 de Enero de 2023). *IEBS*. Obtenido de https://www.iebschool.com/blog/metodologia-kanban-agile-scrum/

LATAM, T. (13 de JULIO de 2022). *TOTVS*. Obtenido de https://es.totvs.com/blog/gestion-de-negocios/sistema-de-gestion-que-es-cuales - son-las-ventajas-y-por-que-invertir/

Martins, J. (10 de Octubre de 2022). *asana*. Obtenido de https://asana.com/es/resources/what-is-kanban

Netsoft. (2 de Marzo de 2021). Obtenido de https://netsoft.com/beneficios-de-contar-con-un-software-de-gestion-empresarial-a-tu-medida-para-impulsar-tu-negocio/

Quintero, S. D. (31 de Diciembre de 2020). *Aprende python.* Obtenido de https://aprendepython.es/_downloads/907b5202c1466977a8d6bd3a2641453f/aprendepython.pdf

Software, C. (15 de Marzo de 2021). *FITco*. Obtenido de https://blog.fitcolatam.com/blog/4-ventajas-de-tener-un-software-de-gestion-en-tu-gimnasio

Visus, A. (Octubre de 2020). *esic*. Obtenido de https://www.esic.edu/rethink/tecnologia/para-que-sirve-python

Westreicher, G. (1 de 4 de 2022). *economipedia*. Obtenido de https://economipedia.com/definiciones/muestreo-no-probabilistico.html

Westreicher, G. (1 de Abril de 2022). *economipedia*. Obtenido de https://economipedia.com/definiciones/muestreo-no-probabilistico.html

Anexo N°1: Anteproyecto

DISEÑO E IMPLEMENTACION DE UN SISTEMA DE CONTROL DE PACIENTES PARA EL CONSULTORIO KARLDENTAL EN LA CIUDAD SARAGURO, AÑO 2023.

Autor: Kevin Darío Ramon Maldonado

Director: Milton Ricardo Palacios

Morocho MGTR

ANTEPROYECTO

PREVIO A LA OBTENCION DEL TITULO DE TECNOLOGO EN ANALISIS DE SISTEMAS.

LOJA-ECUADOR

2022-2023

Glosario de términos.

Nicho: Para el marketing, un nicho de mercado es un segmento de mercado en el cual los individuos tienen características y necesidades homogéneas que no están siendo satisfechas por la oferta. Hablar de un nicho de mercado, por lo tanto, es hablar de una oportunidad que brinda la economía para desarrollar una cierta actividad comercial o productiva con elevadas posibilidades de éxito ante las condiciones del mercado.

Un nicho de mercado, en concreto, es una fracción de un segmento del mercado. Si Internet es un segmento del mercado de la tecnología, los blogs y las redes sociales, por ejemplo, son nichos de mercado.

TI: El término tecnología de la información (TI) fue creado con el propósito de hacer una distinción entre las máquinas de alcance limitado y otras con propósitos más generales. Se basa en el estudio y desarrollo de sistemas de información como aplicaciones software y hardware de computadoras. En palabras más sencillas, un TI se encarga de garantizar que las computadoras funcionen bien para el resto de las personas.

TIC: Las Tecnologías de la Información y las Comunicaciones (TIC), son el conjunto de recursos, herramientas, equipos, programas informáticos, aplicaciones, redes y medios; que permiten la compilación, procesamiento, almacenamiento, transmisión de información como: voz, datos, texto, video e imágenes.

PYMES: quiere decir Pequeñas Y Medianas Empresas. En el país (Ecuador), se llama PYMES al conjunto de pequeñas y medianas empresas, que, de acuerdo al número de trabajadores, volumen de ventas, años en el mercado, y sus niveles de producción, activos, pasivos (que representan su capital) tienen características similares en sus procesos de crecimiento.

FRAMEWORK: Un framework es un esquema o marco de trabajo que ofrece una estructura base para elaborar un proyecto con objetivos específicos, una especie de plantilla que sirve como punto de partida para la organización y desarrollo de software.

Utilizar frameworks puede simplificar (y mucho) una tarea o proceso, de ahí que se trate de una de las herramientas habituales, porque les ayuda a ser más ágiles y productivos

API REST: Una API de REST, o API de RESTful, es una interfaz de programación de aplicaciones (API o API web) que se ajusta a los límites de la arquitectura REST y permite la interacción con los servicios web de RESTful. El informático Roy Fielding es el creador de la transferencia de estado representacional (REST).

Las API son conjuntos de definiciones y protocolos que se utilizan para diseñar e integrar el software de las aplicaciones. Suele considerarse como el contrato entre el proveedor de información y el usuario, donde se establece el contenido que se necesita por parte del consumidor (la llamada) y el que requiere el productor (la respuesta).Por ejemplo, el diseño de una API de servicio meteorológico podría requerir que el usuario escribiera un código postal y que el productor diera una respuesta en dos partes: la primera sería la temperatura máxima y la segunda, la mínima.

3. Problemática

La digitalización ya no era una opción antes de la crisis originada por el COVID-19. Sin embargo, la pandemia ha supuesto un antes y un después en la implantación y el uso de las nuevas tecnologías en la empresa acelerando la transformación digital. El trabajo remoto al que muchas organizaciones se han visto obligadas a recurrir durante el periodo de confinamiento ha revelado la existencia de brechas en las infraestructuras TI (Tecnología de la Información) de las empresas.

A medida que el mercado se diversifica y que las TIC pasan a formar parte fundamental de las empresas, es necesario adecuar las herramientas y aplicaciones disponibles para que las compañías puedan ser lo más eficientes posibles al mismo tiempo que se aprovechan mejor los recursos disponibles. Esto se consigue a partir de muchos elementos distintos y, uno de ellos, es sin duda contar con un sistema de gestión empresarial, una de las soluciones cada vez más comunes en empresas grandes y pequeñas.

En el 2023, el informe señala que el número de usuarios de internet en el mundo alcanzó los 5.160 millones de personas, lo que representa al 64,4% de la población mundial. El número de internautas se incrementó un 1,9% respecto de 2022, en 98 millones de personas, un ritmo algo inferior al de los años anteriores.

Las pymes “Empresas pequeñas” ocupan un lugar muy importante dentro de la economía latinoamericana, siendo un componente fundamental del tejido empresarial en América Latina. Esta importancia se manifiesta en varias dimensiones, su participación en el número total de empresas o la creación de empleo”. En el Ecuador, las pymes se erigen como dinamizadoras de la economía pues, de acuerdo con el inec (2017), en ese país el 99% de los negocios se desempeñan bajo esta modalidad

empresarial, por lo que, según son “la fuente del desarrollo social en cuanto a producción, demanda y compra de productos o simplemente por valor agregado, lo que significa que se ha convertido en un factor indispensable para generar riqueza y empleo”.

De acuerdo con los funcionarios del sector público, la transformación de procesos manuales por sistemas informáticos ha generado que a nivel de presupuesto estatal se reduzcan los gastos corrientes, se han eliminado extensas filas de usuarios frente a ventanillas de atención al público, ya que ahora con el uso del internet se puede tener acceso a cualquier trámite a nivel de gobierno

En Ecuador el total de usuarios de internet es de 14,72 millones en Ecuador, que comparado con el total de la población representa un 81,3% de la misma. En los cambios anuales se registra un crecimiento en estos datos del 8,2%. Mientras que en el proxy de internet móvil se reporta un porcentaje del 99,3% en cuanto a los usuarios de redes sociales que acceden mediante dispositivos móviles.

En la ciudad de Saraguro el uso de páginas webs como de sistemas de control tienen mayor presencia que años atrás, tras la pandemia gran parte de nuevos locales como locales ya existentes han implementado pagos electrónicos, facturación electrónica y sistemas de control digitales pequeños negocios como de nicho como son las panaderías no tienen sistemas de control, en lo que respecta negocios relacionados con la salud y medicina empiezan a hacer uso de ciertas tecnologías que les permite ayudarse con su trabajo.

En la actualidad muchos negocios hacen uso de distintos tipos de sistemas de gestión. Un sistema de gestión es una herramienta que permite controlar, planificar, organizar y automatizar las tareas administrativas de una organización. Un sistema de

gestión analiza los rendimientos y los riesgos de una empresa. Por esto viendo la importancia de un sistema de gestión el consultorio dental ha visto conveniente actualizar sus datos clínicos haciendo uso de herramientas de digitalización que permita un mejor manejo de la información.

En el cantón Saraguro no existen negocios en el ámbito de salud que ofrezcan reserva de turnos online, lo que nos abre a la posibilidad de implementar el agendamiento de turnos online.

3. Objetivos.

3.1 Objetivo General.

Diseño e implementación de un sistema de control de pacientes para el consultorio karldental en la ciudad Saraguro, año 2023

3.2 Objetivos específicos.

- Analizar las necesidades de los usuarios de karldental para el diseño del sistema
- Desarrollar el sistema en base a las necesidades y requerimientos de los usuarios
- Implementar Tailwind CSS con la finalidad de optimizar las interfaces de usuario.

4. Alcance

El alcance del presente proyecto partirá desde el análisis de la situación actual del consultorio “Karldental” dentro de la ciudad Saraguro, diseñar y desarrollar el sistema de control de pacientes en el lenguaje de programación de Python con su framework Django utilizando el editor vscode.

Este proyecto no contará con lo que es el pago online, facturación electrónica y agendar citas online, ya que presenta demasiados conflictos al validar pagos, etc. El proyecto tendrá parte administrativa donde se podrá visualizar el total de pacientes el balance de cuentas tendrá la opción de registrar historias clínicas, registrar pagos, agendar turnos de atención.

La presente investigación que se desarrollará se llevará a cabo en el consultorio “Karldental” el cual se encuentra ubicado en la ciudad de Saraguro Loja-Ecuador, además, se presentará como tesis para el Instituto Superior Tecnológico “Los Andes”.

La duración del presente proyecto de investigación tendrá alrededor de una duración de 5 meses, ya que es el tiempo pertinente para realizar este proyecto, además los gastos que se presenten durante este se asumirán por el investigador ya que, al ser una investigación de carácter educativo, es responsabilidad del postulante.

5. Sumario

CAPÍTULO 1.

SISTEMAS DE GESTIÓN

1.1 Que es un sistema de gestión.

Un sistema de gestión es una herramienta que permite controlar, planificar, organizar y automatizar las tareas administrativas de una organización, este mismo permite ve los rendimientos y los riesgos de una empresa, con el fin de otorgar un ambiente laboral más eficiente y sostenible. (EVALUANDO, 2019) Su objetivo es unificar un único software todas las operaciones de la compañía o negocio con el fin de facilitar la toma de decisiones y el análisis de datos. (Ekon, 2021)

En los últimos años, los sistemas de gestión empresarial han ido ganando popularidad como herramientas fundamentales para las organizaciones gracias a sus prestaciones cada vez más completas, especialmente en un proceso de transformación digital como el que se está viviendo actualmente. (Ekon, 2021)

1.2 Tipos de sistemas.

Existe una gran variedad de sistemas que se puede llegar a aplicar a un negocio todo va a depender de las demandas que se presenten. Los sistemas más conocidos son el CRM y el CMS que son los que más se aplican en negocios.

Tipos de sistemas de gestión.

- Planificación de recursos empresariales (ERP)

  Los ERP son los sistemas de gestión empresarial por excelencia.

  Proporciona una gestión integral de todos los procesos de la compañía.

  Además, los ERP se comunican con diferentes módulos de la empresa

tanto internos como externos, con lo que enlazan multitud de procesos empresariales y facilitan el flujo de información.

- Gestión de Relaciones con el Cliente (CRM).

  Los sistemas CRM están orientados a dar soporte a los procesos relacionados con la gestión comercial y la relación con los clientes, tanto en el ámbito de preventa como en el de posventa, Es cierto que la mayor parte de los ERP cubren también esta faceta, pero a nivel comercial existen productos orientados solamente a la gestión.

- Sistemas de Gestión de Almacenes (SGA).

  Las actividades logísticas relacionadas con la gestión de almacenes también han dado lugar a un mercado de aplicaciones orientadas a cubrir esta necesidad. La mayor parte de los sistemas ERP cubren esta función, pero algunas empresas implantan sistemas SGA especializados para conseguir un mayor nivel de automatización en la gestión de sus almacenes.

- Sistema de Gestión Documental (SGD).

  Los SGD se orientan a la gestión documental, es decir, a la creación, almacenamiento, archivado y organización de grandes volúmenes de documentos, ya sean en formato electrónico o en papel.

1.3 Beneficios de los sistemas de gestión.

La función de un sistema de gestión es optimizar las actividades operativas que componen la gestión del negocio o del propio sector. Proporciona una mayor transparencia y control tanto de las operaciones como de los datos. (LATAM, 2022)

El papel de los sistemas de gestión ha ido evolucionando hasta convertirse en el soporte del funcionamiento de cualquier compañía. A la hora de implementar un sistema dependiendo de qué tipo de sistema se incorpore este dará distintos beneficios, en negocio siempre será la parte fundamental llevar algún tipo de control de inventario.

Beneficios.

- Permite afianzar la relación cliente-empresa.
- Planificación de proyectos más ágil.
- Optimización del trabajo en equipo (Netsoft,

2021) CAPÍTULO 2.

HERRAMIENTAS POR UTILIZAR.

2.1 ¿Qué es Python?

Python es un lenguaje de programación de alto nivel creado a finales de los 80/principios de los 90 por Guido van Rossum, holandés que trabajaba por aquella época en el Centro para las Matemáticas y la Informática de los Países Bajos. Sus instrucciones están muy cercanas al lenguaje natural en inglés y se hace hincapié en la legibilidad del código. Toma su nombre de los Monty Python, grupo humorista de los 60 que gustaban mucho a Guido. Python fue creado como sucesor del lenguaje ABC. (Quintero, 2020)

Este lenguaje se caracteriza por tener una sintaxis fácil de usar, es muy utilizado para lo que es el machine learning el desarrollo web en la parte del backend tiene frameworks muy famosos como es Django, fastapi y flask.

Características de Python.

- Es totalmente gratuito. Se trata de un lenguaje open source o de código abierto, por lo que no hay que pagar ninguna licencia para utilizarlo.
- Está respaldado por una enorme comunidad. Su carácter gratuito hace que continuamente se estén desarrollando nuevas librerías y aplicaciones. Es difícil pensar en algo que no haya hecho alguien. Esto es un factor multiplicativo para los programadores, puesto que cualquier duda estará resuelta en los foros.
- Es un lenguaje multiparadigma. Esto significa que combina propiedades de diferentes paradigmas de programación, lo que permite que sea muy flexible y fácil de aprender de manera independiente de los conocimientos del interesado.
- Es un lenguaje multiparadigma. Esto significa que combina propiedades de diferentes paradigmas de programación, lo que permite que sea muy flexible y fácil de aprender de manera independiente de los conocimientos del interesado. (Visus, 2020)

2.2 ¿Para qué sirve Python?

Python se ha convertido en un lenguaje muy amplio en sus aplicaciones, no se centra en un solo sector, se han desarrollado módulos como cpython para compensar su rendimiento a bajo nivel.

2.3 ¿Qué es Django?

Django es un software que se puede utilizar para desarrollar aplicaciones web de forma rápida y eficiente. La mayoría de las aplicaciones web tienen varias funciones comunes, como la autenticación, la recuperación de información de una base de datos y la administración de cookies. Los desarrolladores tienen que codificar una funcionalidad

similar en cada aplicación web que escriban. Django facilita su trabajo al agrupar las diferentes funciones en una gran colección de módulos reutilizables, llamada marco de aplicación web. Los desarrolladores utilizan el marco web de Django para organizar y escribir su código de manera más eficiente y reducir significativamente el tiempo de desarrollo web.

2.4 ¿Qué es HTML?

HTML es el lenguaje con el que se define el contenido de las páginas web. Corresponde a las siglas en inglés de Lenguaje de Marcado de Hipertexto, básicamente son un conjunto de etiquetas que el navegador interpreta y se emplean para definir el texto y otros elementos que compondrán una página web, como imágenes, listas, tablas, vídeos, etc.

2.5 ¿Qué es CSS?

El CSS es lo que se llama un lenguaje de hojas de estilo en cascada y se utiliza para estilizar elementos escritos en un lenguaje de marcado como HTML. Separa el contenido de la representación visual del sitio.

2.6 ¿Qué es JavaScript?

JavaScript es un lenguaje de programación que los desarrolladores utilizan para hacer páginas web interactivas. Desde actualizar fuentes de redes sociales a mostrar animaciones y mapas interactivos, las funciones de JavaScript pueden mejorar la experiencia del usuario de un sitio web. Como lenguaje de scripting del lado del servidor, se trata de una de las principales tecnologías de la World Wide Web. Por ejemplo, al navegar por Internet, en cualquier momento en el que vea un carrusel de imágenes, un menú desplegable “click-to-show” (clic para mostrar), o cambien de

manera dinámica los elementos de color en una página web, estará viendo los efectos de JavaScript.

2.7 ¿Qué es Tailwind CSS?

Es un framework CSS que da prioridad a la utilidad sobre el propio estilo, pero además a diferencia de otros frameworks CSS como Bootstrap o Bulma, Tailwind no provee una serie de componentes predefinidos. En su lugar, este framework opera en un nivel inferior y te proporciona un conjunto de clases de ayuda para estructura y estilado, de forma que, usando dichas clases, puedas crear rápidamente diseños personalizados con facilidad. Además, gracias a su flexibilidad te permite crear un diseño realmente único.

2.8 Base de datos mongodb

MongoDB (del inglés humongous, "enorme") es un sistema de base de datos NoSQL orientado a documentos de código abierto y escrito en C++, que en lugar de guardar los datos en tablas lo hace en estructuras de datos BSON (similar a JSON) con un esquema dinámico. Al ser un proyecto de código abierto, sus binarios están disponibles para los sistemas operativos Windows, GNU/Linux, OS X y Solaris y es usado en múltiples proyectos o implementaciones en empresas como MTV Network, Craigslist, BCI o Foursquare.

2.9 ¿Qué es Visual Studio Code?

Visual Studio Code (VS Code) es un editor de código fuente desarrollado por Microsoft. Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS. VS Code tiene una buena integración con Git, cuenta con soporte para depuración de código, y dispone de un sinnúmero

de extensiones, que básicamente te da la posibilidad de escribir y ejecutar código en cualquier lenguaje de programación.

CAPÍTULO 3

INFORMACIÓN DEL CONSULTORIO.

3.1 Descripción General

De acuerdo con la Ortodoncista Karla Betancourt (comunicación personal, 5 de julio de 2023) el consultorio “Karldental” proporciona diferentes servicios para lo que es el tratamiento de los dientes, como tratamientos de ortodoncia, endodoncia, restauraciones y extracciones. El consultorio se encuentra ubicado en la calle Loja y Monofilo Muñoz frente a la Aktual ferretería

3.2 Información del consultorio

El consultorio karldental nace en la localidad de Saraguro como un consultorio independiente en busca de dar soluciones dentales a los ciudadanos en el ámbito de Ortodoncia en la ciudad de Saraguro.

3.3 Dirección

Se encuentra ubicado en la ciudad de Saraguro en la calle Loja frente a la Aktual ferretería

3.3 Servicios

Actualmente los servicios que ofrece son:

- Tratamientos de Ortodoncia
- Limpiezas dentales
- Restauraciones de dientes

- Prótesis dentales
- Blanqueamiento dental
- Extracción de molar

6. Metodología

Como metodología se denomina la serie de métodos y técnicas de rigor científico que se aplican sistemáticamente durante un proceso de investigación para alcanzar un resultado teóricamente válido. En este sentido, la metodología funciona como el soporte conceptual que rige la manera en que aplicamos los procedimientos en una investigación

6.1 Métodos

- **Método científico:** El método científico, por lo tanto, se refiere a la serie de etapas que hay que recorrer para obtener un conocimiento válido desde el punto de vista científico, utilizando para esto instrumentos que resulten fiables. Lo que hace este método es minimizar la influencia de la subjetividad del científico en su trabajo. El método científico está basado en los preceptos de falsabilidad (indica que cualquier proposición de la ciencia debe resultar susceptible a ser falsada) y reproducibilidad (un experimento tiene que poder repetirse en lugares indistintos y por un sujeto cualquiera) (Peréz, 2008). Este método se lo utilizara para el desarrollo de todo el proyecto

  Método analítico: El método analítico, es un proceso de investigación empírico – analítico que se encarga de descomponer diferentes partes o elementos para poder hacer la determinación de alguna causa, efecto o naturaleza. El análisis realiza estudios y examina algún hecho u objeto en

particular, dentro del campo de las ciencias naturales y las ciencias sociales (Pacheco, 2021).

Método inductivo: El método inductivo es una estrategia de razonamiento que se basa en la inducción, para ello, procede a partir de premisas particulares para generar conclusiones generales. En este sentido, el método inductivo opera realizando generalizaciones amplias apoyándose en observaciones específicas. Esto es así porque en el razonamiento inductivo las premisas son las que proporcionan la evidencia que dota de veracidad una conclusión (Coelho, 2019).

Método deductivo: El método deductivo es un método científico que considera que la conclusión se halla implícita dentro de las premisas. Esto quiere decir que las conclusiones son una consecuencia necesaria de las premisas: cuando las premisas resultan verdaderas y el razonamiento deductivo tiene validez, no hay forma de que la conclusión no sea verdadera (Pérez & Merino, 2008). Se utilizará para realizar las recomendaciones de proyecto

6.2 Técnicas

Observación: La observación es la técnica de recogida de la información que consiste básicamente, en observar, acumular e interpretar las actuaciones, comportamientos y hechos de las personas u objetos, tal y como las realizan habitualmente. En este proceso se busca contemplar en forma cuidadosa y sistemática como se desarrollan dichas características en un contexto determinado, sin intervenir sobre ellas o manipularlas. También se conoce a este término como la nota escrita que explica, aclara o corrige un dato, error o

información que puede confundir o hacer dudar. Por lo general, esta aclaratoria se encuentra en libros, textos o escritos (Mariana, 2021).

Entrevista: Se conoce como entrevista la conversación que sostienen dos o más personas que se encuentran en el rol de entrevistador y entrevistado, a fin de que el primero obtenga de la segunda información sobre un asunto particular. En toda entrevista hay dos roles: el entrevistador y el entrevistado. El entrevistador es quien formula las preguntas y conduce la conversación. Debe encargarse también de introducir el tema y hacer el cierre a la entrevista (Coelho, 2019).

6.3 Instrumentos

Cuestionario: Un cuestionario es aquel que plantea una serie de preguntas para extraer determinada información de un grupo de personas. El cuestionario permite recolectar información y datos para su tabulación, clasificación, descripción y análisis en un estudio o investigación. En este sentido, los cuestionarios pueden usarse como instrumentos de recolección de datos, o como herramienta de evaluación en el ámbito escolar. Los cuestionarios permiten medir preferencias, comportamientos o tendencias (a la manera de las encuestas), así como construir escalas que ayudan a determinar ciertas actitudes o rasgos latentes en las personas (Coelho, 2019).

Preguntas dicotómicas. Las preguntas dicotómicas son uno de los tipos de preguntas más utilizadas en encuestas y formularios. Sin embargo, sus características y ventajas para obtener información aún son desconocidas para la mayoría. Pertenecen al grupo preguntas cerradas de elección única, es decir solo ofrecen la posibilidad de elegir una opción como respuesta.

6.4 Metodología para el desarrollo del sistema Kanban

Metodología Kanban: La metodología Kanban se implementa por medio de tableros Kanban. Se trata de un método visual de gestión de proyectos que permite a los equipos visualizar sus flujos de trabajo y la carga de trabajo. En un tablero Kanban, el trabajo se muestra en un proyecto en forma de tablero organizado por columnas

Características de la metodología Kanban

- Visualización en el método Kanban
- Tareas en proceso
- Priorización de tareas en método Kanban
- Medir el tiempo

Visualización en el método Kanban

La tarjeta Kanban es el elemento clave en el tablero Kanban, cada tarjeta contiene información como el nombre de la tarea, la descripción, la persona asignada, la prioridad, comentarios enlaces externos, así como las subtareas asociadas.

Partes visibles tarjeta:

- Titulo Tarea: El nombre que le hemos dado a la tarea siempre será visible en la tarjeta Kanban
- Descripción Tarea: Aquí introduciremos los detalles de la tarea.
- Tiempo ciclo: Duración de la tarea
- Persona asignada
- Prioridad

Detalles tarjeta Kanban

- Comentarios miembros equipo
- Enlaces adjuntos
- Archivos adjuntos
- Histórico estados tarjeta

Tareas en proceso: El principio 2 de la metodología KANBAN dice que no se debería de trabajar con más de una tarjeta a la vez. Y esta tarjeta, se clasifica en uno de los tres bloques del tablero Kanban al finalizar la jornada: Por hacer, En Proceso, Pendiente revisión, Hecho. Durante la realización de las tareas, surgen dudas, o se ven fallos / mejoras. Kanban fomenta la continua modificación de las tareas, ya que se trata de un sistema de trabajo inmediato, compuesto por pequeñas tareas de corta duración. De esta forma ponemos freno a uno de los principales problemas en un entorno de trabajo: la focalización.

Priorización de tareas en método Kanban

La mayor parte de herramientas para implementar la metodología Kanban te permiten o bien ordenar con drag and drop las tareas de acuerdo con la prioridad para el equipo o dependencias o incluso asignarles un valor. Además, una buena práctica, suele ser la incorporación de filas de acuerdo con diferentes prioridades/conceptos dentro de una misma columna

Medir el tiempo: Gracias al sistema de situar las tareas en “Haciendo”, durante el tiempo que trabajamos en ellas y etiquetar las tarjetas según el tema tratado, podemos hacer un seguimiento del tiempo invertido en cada función, departamento o campo

Ventajas del método Kanban.

- Procesos innecesarios; Debido a la comunicación y las facilidades de las diferentes herramientas Kanban, evitamos procesos que reducen la eficiencia o maximizan el tiempo de trabajo.
- Conciencia sobre el desarrollo; Gracias a la comunicación y al control que se tiene en todo momento sobre el estado del producto, es mucho más sencillo mantener al equipo actualizado y con conocimiento del proceso que les envuelve.
- Equipo motivado; Debido a las características de Kanban y su metodología de trabajo, el equipo se ve más único, y con una mayor facilidad para plantear y resolver dudas al mismo tiempo que van saliendo.
- Flexibilidad; Ya que vivimos tiempos rápidos, tenemos que saber cómo enfrentarnos a los cambios, y con Kanban, podemos hacer variaciones al producto tan pronto como se detecte un error o un giro en el mercado. Estos errores se localizan en breves periodos de tiempo debido a la corta duración de las tareas asignadas
- Costoso en determinados casos. Al ser un sistema basado en tareas si lo tratamos de aplicar en departamentos de producción muy grandes o en equipos multidisciplinares puede ser imposible de aplicar puesto que acabaríamos con tableros difíciles de gestionar.
- Tipo de Proyecto. El proceso Kanban llevado a la gestión de equipos, proyectos o procesos industriales asume que se sigue un ciclo repetitivo (por hacer, en proceso, en revisión, hecho).
- Anticipación: Aplicando la metodología Kanban resulta complicado anticiparse ante cambios de un proyecto.

Fase 1: Planificación

- Listado de Requerimientos
- División de requerimientos en funcionales y no funcionales Fase 2: Diseño

- Realización de diagrama de clases de los requerimientos funcionales
- Diseño de interfases
- Diseño de un api rest.

Fase 3: Codificación

- Se usará el lenguaje de programación Python
- Utilización del Framework de Django

Fase 4: Pruebas

- Se utilizará el cuestionario donde se elabora preguntas para que el usuario final valide el funcionamiento del sistema

Fase 5: Lanzamiento

- El servidor se lanzará primeramente de manera local para dar una última revisión y continuamente ponerlo en línea finalmente.

7. Población y Muestra

Población: Para marcar una cantidad específica de población del consultorio Karldental, se contará los pacientes que consten ya con historia clínica previamente registrada, y además del propietario de este.

El número de pacientes con expedientes es de 175.

Muestra: Como muestra para este trabajo se tomará en cuenta a los pacientes recurrentes del consultorio.

Número de pacientes recurrentes 30.

7.1 Tipo de Muestreo

El muestreo no probabilístico es aquel donde no todos los sujetos de la población estadística tienen la misma probabilidad de ser elegidos para formar parte del estudio que se está desarrollando. Es decir, este tipo de muestreo implica que el encuestador o investigador no selecciona aleatoriamente o al azar, entre toda la población, a los individuos que forman parte de la muestra sobre la que trabaja. (Westreicher, 2022)

Existen varios tipos de muestreos no probabilísticos, pero, en esta investigación se usará el tipo de muestreo discrecional.

Discrecional, opinativo o intencional: El investigador, según su juicio, elige a un grupo de personas que considera que son idóneas para el estudio. Por ejemplo, si se desea realizar una encuesta a los doctores que trabajan en un hospital, el investigador podría seleccionar aquellos a quienes considera que son más idóneos porque confía en su criterio.

8. Presupuesto y financiamiento

RECURSOS HUMANOS				
Cantidad	**Recursos**	**Descripción**	**Valor Unitario**	**Valor Total**
1	Director de Tesis	Docente Asignado	$150.00	$150.00
1	Tesista	Kevin Darío Ramon Maldonado	$10.00	$120.00
1	Propietaria	Karla Betancourt	$0.00	$0.00
20	Clientes	Pacientes recurrentes	$0.00	$0.00
Cantidad	**Recursos**	**Descripción**	**Valor Unitario**	**Valor Total**
1	Computadora Portátil	HP 15-dy	$650.00	$650.00
1	Teléfono Móvil	Redmi 10	$210.00	$210.00
1	Internet	Para consultar información	$30.00	$120.00
1	Python	Lenguaje de Programación	$0.00	$0.00
1	Django	Framework	$0.00	$0.00
1	Bootstrap	Biblioteca de código	$0.00	$0.00
1	Tailwind	Biblioteca de código	$0.00	$0.00

	HTML	Lenguaje de marcado web	$0.00	$0.00
	CSS	Lenguaje de diseño gráfico	$0.00	$0.00
1	Dominio	URL para la página	$10.00	$10.00
3	Hosting	Alojamiento Web	$5.00	$20.00
15	Transporte	Medio de transporte por la ciudad	$5.00	$15.00
1	Imprevistos 3%	Margen de error en los precios		66.66
TOTAL				$1.361.66

El financiamiento del proyecto será asumido el 100% por el tesista

ACTIVIDADES	**AÑO 2023**											
	JUNIO				**JULIO**				**AGOSTO**			
	1	2	3	4	1	2	3	4	1	2	3	4
Definir el tema	**X**											
Elaboración del Ante proyecto			**X**									
Aplicación de Instrumentos				**X**								
Tabulación y representación gráfica de resultados						**X**						
Elaboración del borrador						**X**						
Correcciones							**X**					
Disertación Privada								**X**				
Correcciones									**X**			
Disertación Pública										**X**		
Graduación											**X**	

9.BIBLIOGRAFÍA

Ekon, E. (2021, Julio 28). *cegid Ekon*. Retrieved from https://www.ekon.es/blog/sistemas-de-gestion-integral-para-el-funcionamiento-optimo-de-la-empresa/

EVALUANDO. (2019, Noviembre 28). *ERP.COM*. Retrieved from EVALUADO ERP.COM: https://www.evaluandoerp.com/software-erp/sistema-de-gestion/

Flores, F. (2022, julio 22). *OpenWebinars*. Retrieved from https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/#qu%C3%A9-es-visual-studio-code

LATAM, T. (2022, JULIO 13). *TOTVS*. Retrieved from https://es.totvs.com/blog/gestion-de-negocios/sistema-de-gestion-que-es-cuales - son-las-ventajas-y-por-que-invertir/

Netsoft. (2021, Marzo 2). Retrieved from https://netsoft.com/beneficios-de-contar-con-un-software-de-gestion-empresarial-a-tu-medida-para-impulsar-tu-negocio/

Quintero, S. D. (2020, Diciembre 31). *Aprende python.* Retrieved from https://aprendepython.es/_downloads/907b5202c1466977a8d6bd3a2641453f/aprendepython.pdf

Software, C. (2021, Marzo 15). *FITco*. Retrieved from https://blog.fitcolatam.com/blog/4-ventajas-de-tener-un-software-de-gestion-en-tu-gimnasio

Visus, A. (2020, Octubre). *esic*. Retrieved from https://www.esic.edu/rethink/tecnologia/para-que-sirve-python

Westreicher, G. (2022, 4 1). *economipedia*. Retrieved from https://economipedia.com/definiciones/muestreo-no-probabilistico.html

10.Anexos

10.1 Cuestionario de la Entrevista

1. ¿Considera que el consultorio necesita un sistema?

2. ¿Cuántos pacientes tiene mensualmente?

3. ¿Considera añadir más servicios?

4. Tiene facilidad de pago?

5. ¿Realiza tratamientos odontológicos fuera de la ciudad de Saraguro?

6. Cuanto tiempo les dedica a los tratamientos odontológicos?

7. ¿Cuáles son los métodos de pago?

8. Qué servicios ofrece?

10.2 Cuestionario de la Encuesta

1. ¿Considera que los precios son accesibles?

2. ¿Le gustaría brindar otros servicios odontológicos?

3.¿Cree que el tiempo empleado en los tratamientos es el adecuado?

4. Cómo conoció el sistema?

5. La atención brindada la considera buena?

6. Los tratamientos cumplen con lo prometido?

KARDENTAL

Manual del programador

Sistema kadental

Clase

Las clases en Python hacen referencia a lo que es la programación orientada en objetos donde podemos volver a reutilizar las clases en distintos archivos .py, en este caso como se esta usando django_rest_framework que esta basad en Python. Las clases las usamos para definir el orm de django que permite ya crear las instancias de la base de datos, también nos permite usarlas en las vistas de django, tambien en los serializers.py que nos la funcionalidad de llamar los objetos de las clases para poder llamarlos en las vistas y así tener una api con django_rest_framework

Ejemplo models.py

Clase Paciente “Patient”

```python
# models para agregar un paciente
class Patient(models.Model):
    first_name = models.CharField(max_length=100)
    last_name = models.CharField(max_length=100)
    address = models.CharField(max_length=100)
    card = models.IntegerField(blank=True)
    birth_date = models.DateField(blank=True)
    appointment_date = models.DateTimeField()
    notes = models.TextField(blank=True)

    def __str__(self) -> str:
        return f"Nombre: {self.first_name}, Apellido: {self.last_name}, Nacimiento: {self.birth_date}"
```

Serializers

```python
class PatientSerializer(serializers.ModelSerializer):

    class Meta:
        model = Patient
        fields = "__all__"
```

En este caso para el serializer de paciente “patient” llamamos todos sus objetos pero esto va depender de las necesidades de cada programador, donde puede llamar algo en especifico de la clase Paciente.

Vistas

En este caso en las vistas de django “views.py” nos va somos en las vistas por clases en ves de funciones

```
class PatientViewset(viewsets.ModelViewSet):
    queryset = Patient.objects.all()
    serializer_class = PatientSerializer

    @action(detail=True, methods=["GET"])
    def clinichistory(self, request, pk=None):
        patient = self.get_object()
        try:
            clinic_history = ClinicHistory.objects.get(patient=patient)
            serializer = ClinicHistorySerializer(clinic_history)
            return Response(serializer.data)
        except ClinicHistory.DoesNotExist:
            return Response(
                {"message": "Este paciente no tiene historia clínica."},
                status=status.HTTP_404_NOT_FOUND,
            )
```

Ya teniendo nuestras vistas, modelos y serializer pasamos al punto uno de los puntos mas importante donde definmos nuestras urls.py las cuales usaremos para poder llamarlas a nuestro cliente que en este caso es React

```
from django.urls import path, include
from rest_framework.documentation import include_docs_urls
from rest_framework.routers import DefaultRouter
from rest_framework_simplejwt.views import (
```

```
    TokenObtainPairView,
    TokenRefreshView,
    TokenVerifyView,
)
from django.views.decorators.csrf import csrf_exempt
from . import views
from .views import (
    PatientViewset,
    AppoinmentViewset,
    ClinicHistoryViewset,
    OrthodonticTreatmentViewset,
    DataViewset,
    PaymentsViewset,
    NextControlViewset,
    HistorieViewset,
    CustomTokenObtainPairView,
)

router = DefaultRouter()
router.register(r"patients", PatientViewset)
router.register(r"appoinment", AppoinmentViewset)
router.register(r"clinichistory", ClinicHistoryViewset)
router.register(r"orthodontictreatment", OrthodonticTreatmentViewset)
router.register(r"payment", PaymentsViewset)
router.register(r"nextcontrol", NextControlViewset)
router.register(r"data", DataViewset)


urlpatterns = [
    path("", include(router.urls)),
    path("docs/", include_docs_urls("Karldental Api")),
    path(
        "payments/<int:patient_id>/",
        views.PaymentsViewset.as_view({"get": "list"}),
        name="payments-list",
    ),
    path("histories/<int:patient_id>/", views.HistorieViewset.as_view({"get": "list"}), name="histories-list"),
    path("token/", TokenObtainPairView.as_view(), name="token_obtain_pair"),
    path("token/refresh/", TokenRefreshView.as_view(), name="token_refresh"),
    path("token/verify/", TokenVerifyView.as_view(), name="token_verify"),
    path("home/", views.HomeView.as_view(), name="home"),
    path("logout/", views.LogoutView.as_view(), name="logout"),
    path("login/", views.login_view, name="login"),
]
```

Podemos definir nuestra propias urls según nuestras necesidades o bien podemos usar el router.register de django_rest que se encarga establecer las url por nosotros.

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "patients": "http://localhost:8000/api/patients/",
    "appoinment": "http://localhost:8000/api/appoinment/",
    "clinichistory": "http://localhost:8000/api/clinichistory/",
    "orthodontictreatment": "http://localhost:8000/api/orthodontictreatment/",
    "payment": "http://localhost:8000/api/payment/",
    "nextcontrol": "http://localhost:8000/api/nextcontrol/",
    "data": "http://localhost:8000/api/data/"
}
```

El lado del cliente donde usamos nuestra api

Para este proyecto para el lado del cliente se uso la librería React.

Nuestro árbol de trabajo en el lado del cliente

├── dist

├── index.html

├── node_modules

├── package-lock.json

├── package.json

├── postcss.config.js

├── public

├── src

├── tailwind.config.js

└── vite.config.js

Todo nuestro código lo tenemos en el directorio de “src”

.

├── App.css

├── App.jsx

├── api

│ └── patients.api.js

├── assets

├── components

│ ├── AddPatientOrto.jsx

│ ├── EditPatient.jsx

│ ├── MedicalNotes.jsx

│ ├── Modals

| ├── Navigation.jsx
| ├── PatientCard.jsx
| ├── PatientList.jsx
| ├── PatientsDeEdi.jsx
| ├── Patientsdetail.jsx
| ├── Readme.md
| └── search.jsx
├── index.css
├── main.jsx
├── pages
| ├── Home.jsx
| ├── Sidenav.jsx
| ├── auth
| ├── css
| ├── dashboard.jsx
| ├── images

```
|   └── js
└── static
    ├── JS
    └── modal.css
```

Para este proyecto se hizo de los modal para tener ventanas emejergentes dentro del dashboard

Tenemos en el directorio pages/dashboard.jsx

```
  <div className="bg-white p-4 rounded-lg xs:mb-4 max-w-full shadow-md lg:w-[65%]flex-auto">

          <div className="flex flex-wrap justify-between h-full">
            <div className="flex-1 bg-gradient-to-r from-cyan-400 to-cyan-600 rounded-lg flex flex-row items-center justify-center p-4 space-x-2 border border-gray-200 m-2">
              <AiOutlineUserAdd className="text-white" size={40} />
              <button className="text-white" onClick={handleAddPatientClick}>
                Nuevo Paciente
              </button>
            </div>
            {isAddPatientFormOpen && (
              <AddPatientForm
                isOpen={true}
                onRequestClose={handleClosePatientForm}
              />
            )}
            <div className="flex-1 bg-gradient-to-r from-cyan-400 to-cyan-600 rounded-lg flex flex-row items-center justify-center p-4 space-x-2 border border-gray-200 m-2">
              <AiOutlineUserAdd className="text-white" size={40} />
              <button onClick={handleAddCitieClick} className="text-white">
                Agendar cita
              </button>
              {isAddCitie && (
                <AddCitie
                  isOpen={true}
                  onRequestClose={handleCloseAddCitie}
                />
              )}
            </div>
          </div>
        </div>
```

```
      </div>
```

que es nuestra pantalla de bienvenida
donde se importa otras paginas como la pagina sidenav.jsx

```
<div className="flex justify-between items-center px-9">

        <button
          onClick={() => {
            setIsMenuVisible(!isMenuVisible);
          }}
        >
          <FaBars className="text-cyan-500 text-lg" />
        </button>
        <div className="ml-1">
          <h1 className=" text-2xl font-bold text-cyan-400 mt-8">
            Kardental <FaTooth className="inline-block ml-2" />
          </h1>
        </div>
        <div className="space-x-4">
          <button>
            <i className="fas fa-bell text-cyan-500 text-lg"></i>
          </button>
          {/* User menu options */}
          <button
            onClick={() => {
              setIsUserMenuVisible(!isUserMenuVisible);
            }}
          >
            <i className="fas fa-user text-cyan-500 text-lg"></i>
          </button>
          {isUserMenuVisible && (
            <div className="absolute right-0 mt-2 bg-white border border-gray-300 rounded shadow-lg">
              <button
                href="#"
                className="block px-4 py-2 hover:bg-gray-100"
                onClick={() => {
                  handleViewChange("cerrarSesion");
                  setIsUserMenuVisible(false);
                  {
                    handleLogout()
                  } // Cierra el menú después de hacer clic
                }}
              >
                Cerrar sesión
```

```
            </button>
          </div>
        )}
      </div>
    </div>
```

Recomendación seria separar al sidenav por componentes.

En los componentes tenemos esta estructura

components

├── AddPatientOrto.jsx

├── EditPatient.jsx

├── MedicalNotes.jsx

├── Modals

│   ├── AddCitie.jsx

│   ├── AddPatientForm.jsx

│   ├── Cities.jsx

│   ├── DeletePatienModal.jsx

│   ├── Historialpayments.jsx

│   ├── JS

│   │   ├── css

│   │   │   └── file.css

```
|   |   └── print.js
|   ├── NextcontrolModal.jsx
|   ├── NotesModal.jsx
|   ├── Ortodoncia.jsx
|   ├── PatientsDetailModal.jsx
|   ├── Registerpayment.jsx
|   ├── Revision.jsx
|   └── css
|       └── tasb.css
├── Navigation.jsx
├── PatientCard.jsx
├── PatientList.jsx
├── PatientsDeEdi.jsx
├── Patientsdetail.jsx
├── Readme.md
└── search.jsx
```

Tenemos el modal AddPatienform.jsx

Donde hacemos un petición post a la api de django para guardar los datos de nuestro paciente

```
const handleSubmit = async (event) => {

  event.preventDefault();
  const newPatient = {
    first_name: firstName,
    last_name: lastName,
    address: address,
    card: card,
    birth_date: birthDate,
    medical_conditions: medicalConditions,
    medicines: medicines,
    appointment_date: appointment_date,
    notes: notes,
  };
  try {
    await addPatient(newPatient);
    setFirstName("");
    setLastName("");
    setBirthDate("");
    setAddres("");
    setCard("");
    setAppoinment("");
    setMedicalConditions("");
    setMedicines("");
    setNotes("");
    onRequestClose(true);
  } catch (error) {
    console.log("Error adding patient: ", error);
  }
};
```

Tenemos al compoennte Patientsdetail.jsx, donde hacemos uso de la mayoría de los modals como de los componentes.

En este mismo hacer peticiones post y get a api

```
const navigate = useNavigate();


const handleDelete = () => {
  setIsDeleteDialogOpen(true);
};

const confirmDelete = async () => {
  if (isDeleteDialogOpen) {
```

```
      try {
        await deletePatient(patientId);
        toast.success("Paciente eliminado exitosamente", {
          position: "bottom-right",
          style: {
            background: "#101010",
            color: "#fff",
          },
        });
        navigate("/");
      } catch (error) {
        console.error("Error al eliminar el paciente:", error);
        toast.error("Error al eliminar el paciente");
      } finally {
        setIsDeleteDialogOpen(false);
      }
    }
  };
  const cancelDelete = () => {
    setIsDeleteDialogOpen(false);
  };

  const toggleDropdown = () => {
    setIsDropdownOpen(!isDropdownOpen);
  };

  const toggleHistorieDropdown = () => {
    setisHistoriDropdownOpen(!isHistoriDropdownOpen);
  };

  useEffect(() => {
    async function fetchNextControl() {
      try {
        const nextControlRes = await getnextrevison(patientId);
        setNextControlData(nextControlRes.data);
      } catch (error) {
        console.error("Error al obtener los datos del próximo control:", error);
      }
    }
    async function fetchOrtodonciaTreatment() {
      try {
        const ortodonciaData = await getOrtodonciaTreatment(patientId);
        setOrtodonciaTreatmentData(ortodonciaData.data);
      } catch (error) {
        console.error("Error fetching ortodoncia treatment:", error);
```

```
    }
  }

  const fetchHistories = async () => {
    try {
      const response = await getHistoriesByPatientId(patientId);
      setClinicHistories(response.data);
    } catch (error) {
      console.error("Error al obtener las hictorias clinicas", error);
    }
  };
  async function loadData() {
    try {
      const patientRes = await getPatientDetails(patientId);
      setPatientDetails(patientRes.data);
    } catch (error) {
      console.log("Error fetching patient details: ", error);
    }
  }

  loadData();
  fetchOrtodonciaTreatment();
  fetchNextControl();
  // historiesData();
  fetchHistories();
}, [patientId]);

if (!patientDetails) {
  return <p className="">Cargando...</p>;
}
```

Tenemos un directorio api donde tenemos el archivo patients.api.js

En ese archvi es donde tenemos las peticiones que le hacemos a nuestra api

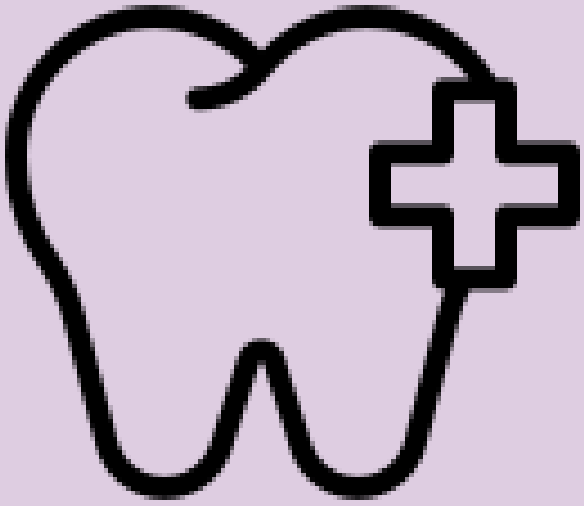
```
export const getAllPatients = () => patientsApi.get("/");


export const getPatient = (patientId) => patientsApi.get(`/${patientId}/`);

export const addPatient = (patientId) => patientsApi.post("/", patientId);

export const updatePatient = (patientId, patient) =>
  patientsApi.put(`/${patientId}/`, patient);

export const deletePatient = (id, patient) =>
  patientsApi.delete(`/${id}/`, patient);
```

```
export const addOrthodontic = (patient) => orthodontic.post("/", patient);

export const getOrtodonciaTreatment = (patientId) =>
  orthodontic.get(`/${patientId}/`);

export const getPatientDetails = (patientId) =>
  patientsApi.get(`/${patientId}/`);
```

Estas peticiones las usamos en cada uno de nuestros modals como compoentes según necesitemos ya sea para guardar datos enviandos desde el cliente o traer los mismos datos que fueron guardados por el cliente.

Kardental

Manual de usuario



Sistema Kardental

Bienvenido al sistema “KARDENTAL”

Me siento privilegiado que haya escogido el sistema web kardenal; por medio de este podrá hacer uso de los servicios de control de pacientes como el control de los historiales cincos de sus pacientes registrados.

KARDENTAL

¿Como puedo conseguir el sistema dental?

Para conseguir un sistema para cedió al de kardental debe ponerse en contacto con el desarrollador del sistema kardental “Kevin Ramon”

kartierdev

¿Tiene algún costo?

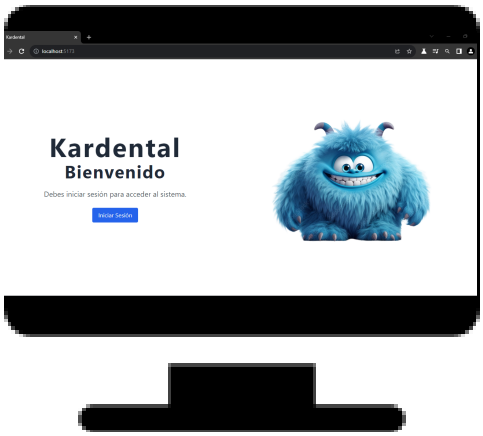
Por el momento el sistema se encuentra en su versión 0.3 beta, no esta disponible para otros consultorios

Objetivo

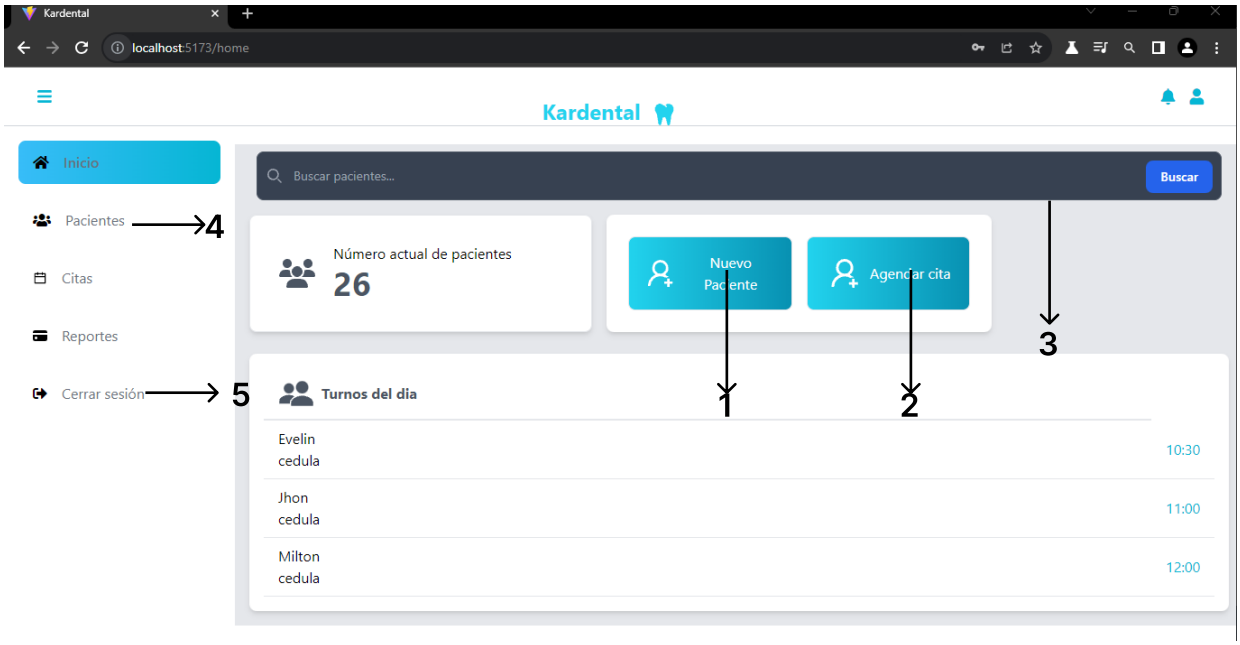
Este sistema a pesar de estar un su fase beta tiene como objetivo ofrecer servicios profesionales en el ámbito de la salud en este caso en lo que es el tema dental. como manejar pacientes citas, historial clínico de los pacientes

Empezando

Una vez que se le de acceso al dominio del sistema con su usuario y contraseña respectiva podrá hacer uso del mismo-

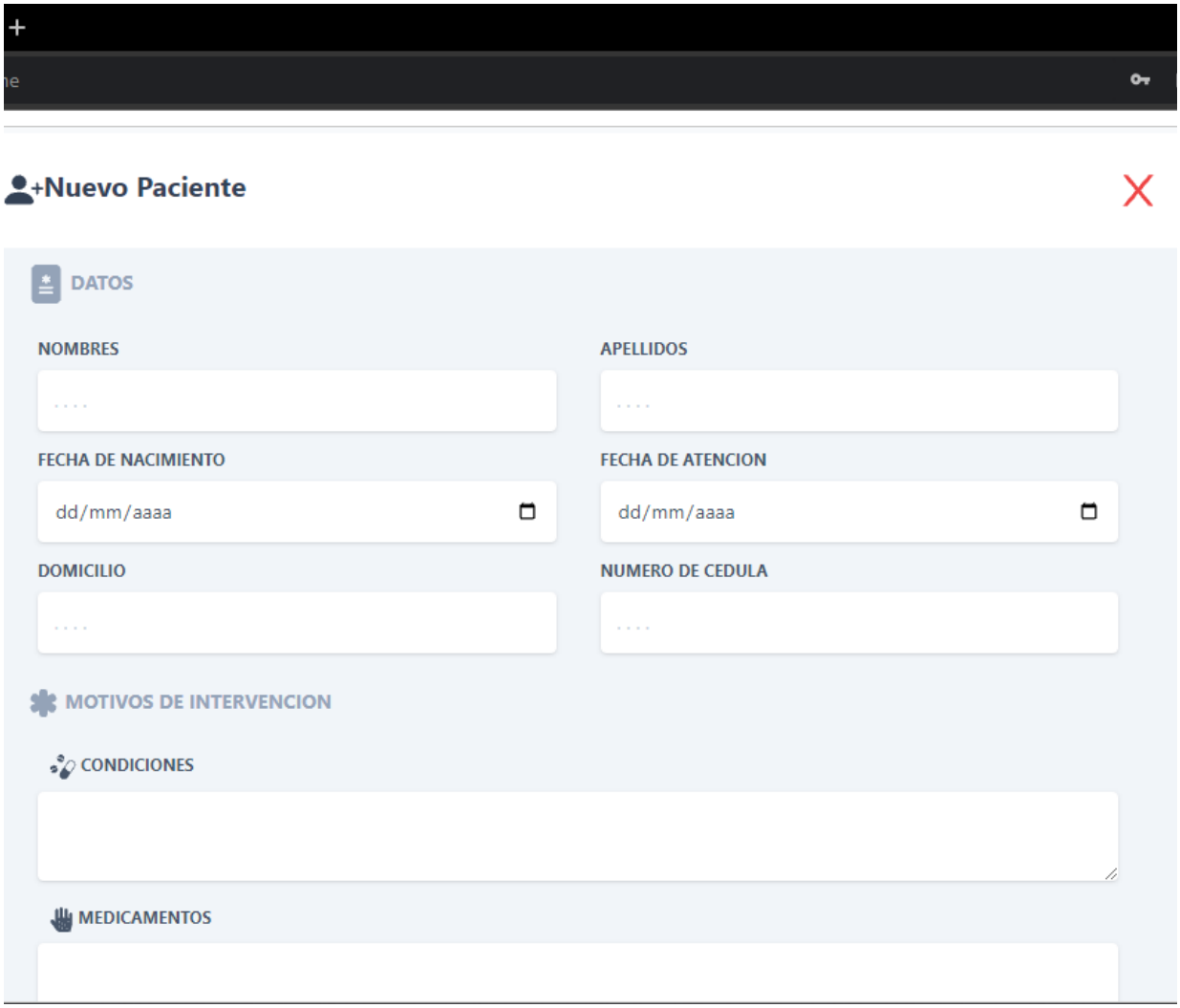


IDEA GENERAL DEL DASHBOARD



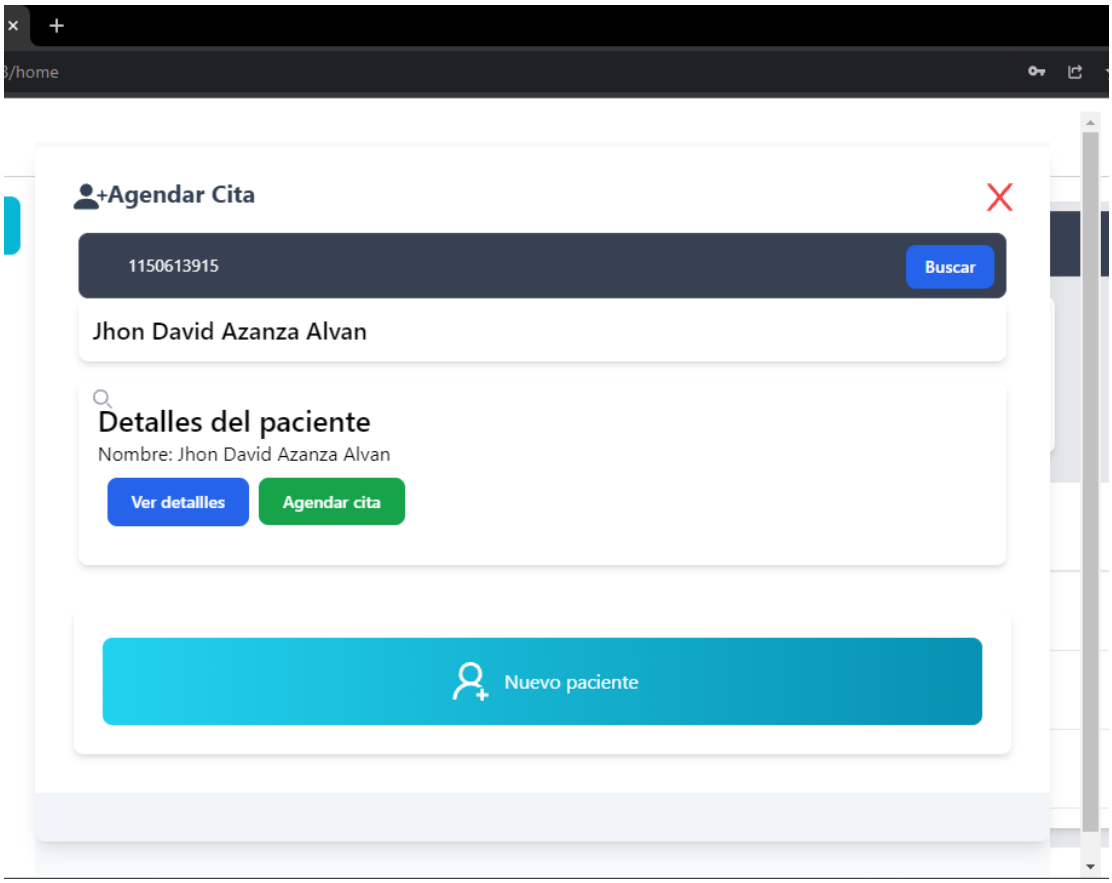
REGISTRAR PACIENTE

Una vez ya logeado en la aplicación nos encontramos con el dashboard. Donde vemos el botón registrar paciente donde se no abrirá un formulario para poder llenar con los datos del paciente



AGENDAR CITA

Ya que estamos logedo aparte del formulario de registro, también ternemos la posibilidad de agendar citas

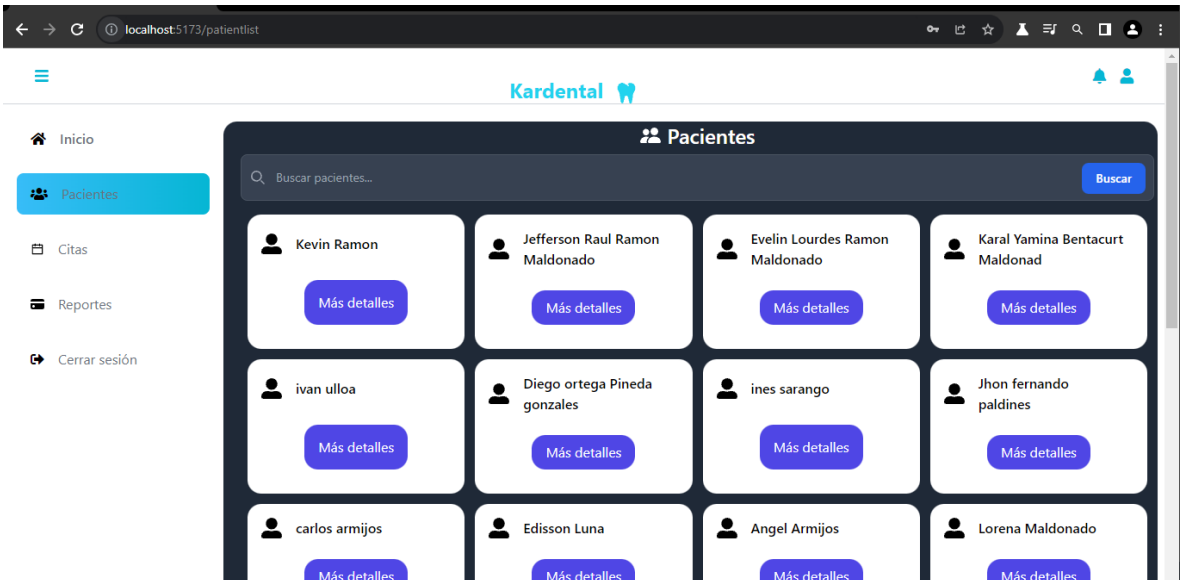


BUSCAR ENTRE TODOS LOS PACIENTE

Se cuenta con una barra de búsqueda general donde se busca entre todos los pacientes, por el momento la barra de búsqueda esta habilitada para buscar solo por el numero de cedula

PACIENTES

También existe la opción de poder ver todos los pacientes que hemos registrado



Historial del paciente

Ya sea que busquemos al paciente o estemos en la opción donde vemos todos los pacientes, podemos ver su historial clinico.