Universidad del Valle

Problema 1. Citas Médicas Extendida

En la esquina de la calle 3 norte con 25 funciona el consultorio médico "Vivir Sano". Es un servicio pequeño, centrado en la atención ambulatoria, atendido a lo largo de la semana por tres profesionales que rotan su jornada. En el panel del consultorio constan sus datos de identificación y de contacto: Juancho Rizzo (cédula 657686, correo ChoRiZZo123@hotmail.com, médico general egresado de la Universidad del Valle), Camilo Ortaliz (cédula 454575, correo Ortalizca2022@gmail.com, médico general titulado en la Universidad Javeriana) y Anacleta Dique (cédula 243535, correo Ncltdq19@hotmail.com, médica con formación en oncología por la Universidad Libre). Esa información está disponible para consulta en todo momento, de forma que cualquier persona pueda verificar quién atiende en la institución.

En el mostrador, la recepcionista conversa con quien llama o se acerca para solicitar atención. Con cada solicitud se registran los datos del paciente —nombre, número de cédula, correo, teléfono y edad— y se elige fecha y hora para la consulta. La persona puede manifestar preferencia por un médico; cuando el profesional elegido no tiene ya una cita registrada exactamente en esa fecha y esa hora, la reserva se confirma y la cita queda registrada con los datos del paciente y del médico correspondiente. El consultorio trabaja con una agenda diaria que utiliza un catálogo de horas repetido a lo largo de la jornada (franjas regulares dentro del horario de atención), de modo que es posible conocer, para un día dado, qué horas siguen libres para cada médico. Cuando el solicitante pide una hora que ya está ocupada para el profesional elegido, el sistema informa la indisponibilidad y permite consultar las horas disponibles de ese médico para esa fecha o, si se prefiere, ver qué médicos están disponibles en ese mismo (fecha, hora) con el fin de concretar la reserva sin perder el turno.

La aplicación gira en torno a ese flujo sencillo y se ejecuta en pantalla con interacción directa. Desde el inicio, el sistema muestra la información de los médicos que trabajan en "Vivir Sano" tal y como están registrados en el consultorio; en cualquier momento se puede volver a consultar ese panel. A lo largo de la sesión, el programa admite el registro de nuevos pacientes y mantiene una lista de quienes ya han sido dados de alta. También conserva el conjunto de citas que se han ido creando, de forma que la recepcionista puede recorrer la agenda para ver todas las reservas que figuran en el día o en el periodo de trabajo. Como parte de la consulta de agenda, es posible filtrar por médico o por paciente cuando se necesita localizar más rápido la información.

Además del alta de pacientes y de la generación de nuevas reservas, la herramienta ofrece consultas de disponibilidad útiles para el trabajo diario: en cualquier momento se puede pedir al sistema los médicos disponibles en un punto concreto del calendario —una fecha y una hora del catálogo—, y también solicitar las horas que quedan libres para un médico determinado en una fecha dada. Con esa información a la vista, la recepción concreta la cita y el sistema muestra en pantalla un resumen claro de la reserva con los datos esenciales del paciente, del médico asignado y del slot (fecha y hora) confirmado.

NOTA MUY IMPORTANTE: El alcance de esta versión es deliberadamente acotado: no se contemplan especialidades distintas a las registradas en el panel del consultorio, ni listas de espera, ni cálculos de solapamientos por intervalos más allá de la regla simple que impide duplicar exactamente la misma combinación de médico, fecha y hora. El foco está en registrar correctamente los datos, consultar el panel de profesionales, listar pacientes y citas con orden legible para el usuario, y confirmar reservas asegurando que la agenda permanezca clara y consistente. Al finalizar la jornada, la sesión concluye sin otras operaciones auxiliares.

Requerimientos

R#	Descripción
R1	El sistema debe mostrar al inicio la información de los médicos registrados (nombre, cédula, correo, título).
R2	El sistema debe permitir registrar nuevos pacientes con sus datos (nombre, cédula, correo, teléfono y edad).
R3	El sistema debe permitir seleccionar una fecha, hora y médico deseada (opcional).
R4	El sistema debe validar que el médico elegido no tenga 2 citas agendadas a la misma hora y fecha.
R5	El sistema debe informar de la indisponibilidad del médico en caso de ya haber una cita en ese día y hora.
R6	El sistema debe permitir consultar las horas disponibles del médico en ese día o que otros médicos están disponibles ese día y hora.
R7	El sistema debe mantener una lista de pacientes registrados
R8	El sistema debe mantener un registro de todas las citas creadas
R9	El sistema debe permitir consultar la agenda completa
R10	El sistema debe permitir filtrar la agenda por médico o paciente
R11	El sistema debe permitir consultar los médicos disponibles en una fecha y hora específica
R12	El sistema debe mostrar un resumen con los datos del paciente, médico y horario reservado
R13	El sistema debe cerrar la sesión sin guardar información persistente

Normalización de Vocabulario del Problema

Término normalizado	Tipo	Definición	Sinonimos	¿Clase, atributo o operación candidata?
Consultorio	Objeto	Almacena las agendas,catálogo Horas,citas,slots,doctores	Clínica	clase(consultorio)
Doctor	Objeto	Almacena la información de cada uno de los tres profesionales	Médico	clase(consultorio)
Paciente	Objeto	Persona que solicita atención.	Cliente	clase(paciente)
Recepcionista	Rol	Interactúa con el sistema para registrar y agendar citas.	Operador	método(recepcionista)
Cita	Objeto	Asociación paciente-médico.	Reserva	clase(cita)
Fecha	Objeto	Día de la cita.	Dia	Clase(fecha)
Hora	Atributo	Franja horaria del catálogo.	Franja	atributo(hora)
Agenda	Objeto	Conjunto de citas por periodo.	Calendario	clase(agenda)
Disponibilidad/slot	Atributo	Estado de una hora para un médico en una fecha.	Libre/ocupado	atributo(slot)
Reserva	Acción	Proceso de crear y confirmar una cita.	Confirmación	metodo(reserva)
Panel de médicos	Método	Visualización inicial con datos de los médicos.	Listado de médicos	metodoAtributo(interfaz)
Catálogo de horas	Atributo	Lista de franjas horarias repetidas durante la jornada.	Horarios predefinidos	Atributo(catalogo)
Sesión	Proceso	Periodo de actividad del sistema.	Jornada	metodo(sesión)
Identificación	Atributo	Identificador único (médico o paciente).	ID	atributo(identificacion)
Correo electrónico	Atributo	Medio de contacto (médico o paciente).	Email	atributo(correo)
Teléfono	Atributo	Medio de contacto adicional del paciente.	Celular	atributo(Telefono)
Título profesional	Atributo	Grado o especialidad del médico.	Formación	atributo(Titulo)

Horario ocupado	Estado	Indica que un slot está reservado.	Turno lleno	metodo(ocupado)
Horario libre	Estado	Slot sin cita para un médico en una fecha.	Disponible	metodo(libre)
Filtro	Funcionalidad	Permitir ver subset de agenda o médicos.	Búsqueda	metodo(filtro)
Registro	acción	Proceso de agregar paciente o cita al sistema en la sesión.	Crear	metodo(registro)
Resumen de cita	Salida	Texto o estructura con paciente, médico y slot confirmados.	Comprobante	metodo(resumen)
Lista de pacientes registrados	Colección	Conjunto de pacientes dados de alta en la sesión.	Base de pacientes	metodo(RegistroPacientes)
Registro de citas	Colección	Conjunto de todas las citas creadas en la sesión.	Agenda de citas	metodo(RegistroCitas)

Tablas CRC

Tabla CRC	**Contenido a completar**
Clase:	Consultorio
Responsabilidades (qué hace):	-Centraliza toda la gestión del sistema. -Contiene los vectores de doctores y pacientes. -Permite agendar citas nuevas. -Permite visualizar doctores y citas existentes.
Colaboradores (a quién le pide):	-Agenda (para registrar y consultar citas). -Doctor (para mostrar disponibilidad). -Paciente (para registrar solicitud).
Mensajes clave:	agenda → verCitas() → vector<Cita> agenda → consultarDisponibilidad(medico, fecha) → vector<string> agenda → registrarCita(cita) → bool doctores → verDoctores() → vector<DatosDoctor> pacientes → mostrardatos() → bool

Tabla CRC	**Contenido a completar**
Clase:	Doctor
Responsabilidades (qué hace):	-Almacena los datos personales del médico. -Permite mostrar sus datos (nombre, cédula, correo y título). -Ofrece información de disponibilidad (indirectamente por agenda).
Colaboradores (a quién le pide):	-Cita (para ser asignado). -Consultorio (para mostrarse en panel).
Mensajes clave*:*	consultorio → verDoctores() → vector<Doctor> agenda → consultarDisponibilidad(fecha, hora) → bool

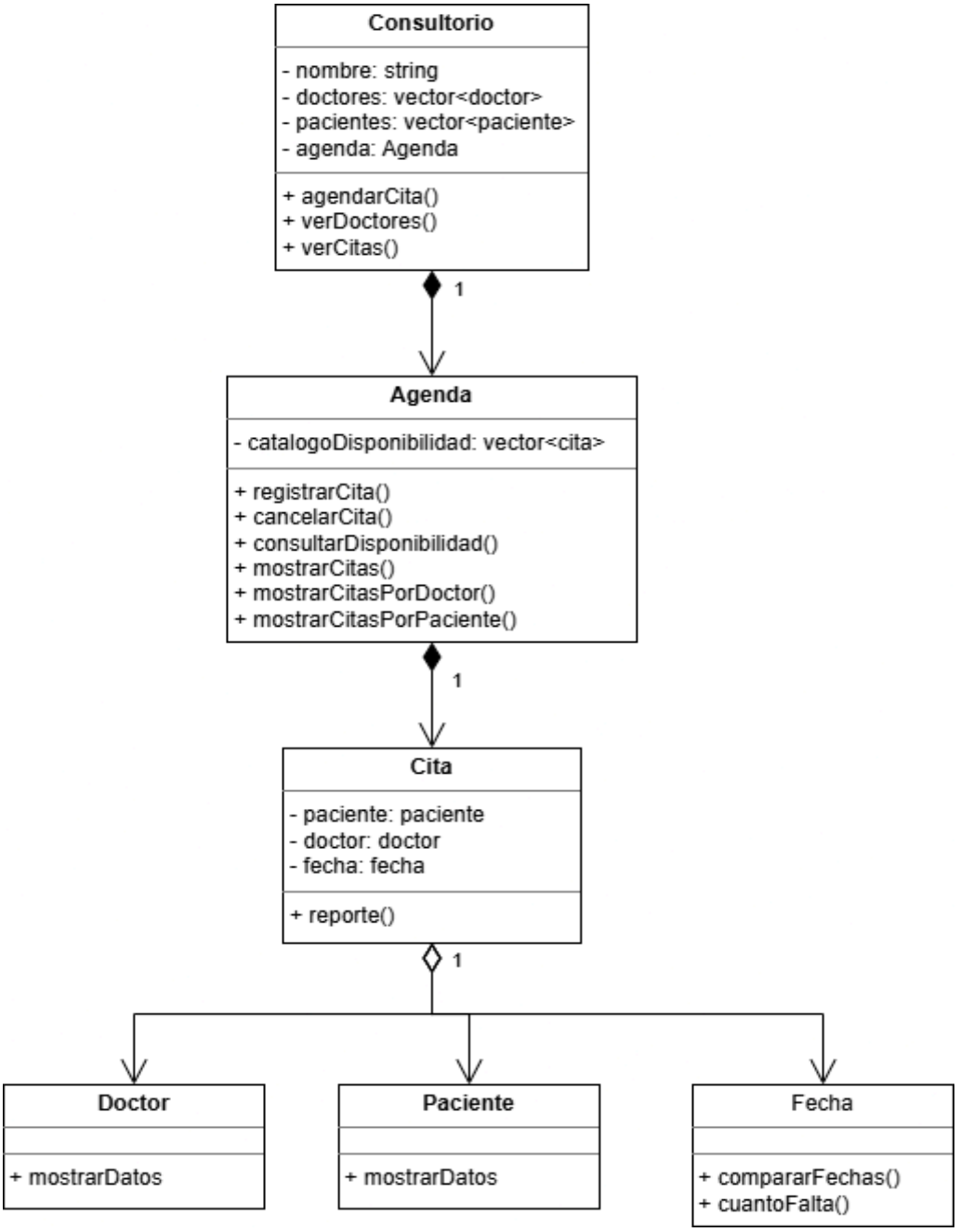
Tabla CRC	**Contenido a completar**
Clase:	Paciente
Responsabilidades (qué hace):	-Registrar los datos personales del paciente. -Permite mostrar información general. -Está asociado a una o varias citas.
Colaboradores (a quién le pide):	-Cita (para registrar la reserva). -Consultorio (para listar o crear citas).
Mensajes clave:	consultorio → registrarPaciente(nombre, cedula, correo, telefono, edad) → bool consultorio → verPacientes() → vector<Paciente> cita → asignarPaciente(paciente) → void

Tabla CRC	**Contenido a completar**
Clase:	Cita
Responsabilidades (qué hace):	-Asocia un paciente, un doctor y una fecha. -Genera reportes o resúmenes. -Valida que el horario no esté duplicado.
Colaboradores (a quién le pide):	-Doctor (para validar disponibilidad). -Paciente (para obtener datos del solicitante). -Fecha (para comparar o verificar disponibilidad).
Mensajes clave:	doctor → consultarDisponibilidad(fecha, hora) → bool paciente → getDatos() → struct DatosPaciente fecha → compararFecha(fecha) → bool agenda → registrarCita(cita) → void cita → reporte() → string

Tabla CRC	**Contenido a completar**
Clase:	Agenda
Responsabilidades (qué hace):	-Mantiene el catálogo de disponibilidad (vector de citas). -Registra, cancela y muestra citas. -Consulta disponibilidad según médico o paciente. -Filtra citas por doctor o paciente.
Colaboradores (a quién le pide):	-Cita (para crear o eliminar registros). -Consultorio (para acceder desde la interfaz principal).
Mensajes clave*:*	citas → registrarCita(cita) → bool citas → cancelarCita(id) → bool citas → mostrarCitas(medico/paciente) → vector<Cita> citas → verCitas() → vector<Cita> doctor → consultarDisponibilidad(fecha, hora) → bool

Tabla CRC	**Contenido a completar**
Clase:	Fecha
Responsabilidades (qué hace):	-Representa el día y hora de la cita. -Permite comparar fechas (para evitar duplicados). -Calcula diferencia o tiempo faltante.
Colaboradores (a quién le pide):	-Cita (para asociarse a la reserva). -Agenda (para validar disponibilidad).
Mensajes clave:	agenda → compararFechas() → bool fecha → cuantofalta() → string

Diagrama de clase



EPS

Campo	Contenido a completar
Clase:	Consultorio
Método:	agendarCita()
Propósito (una línea)	Registrar una nueva cita entre paciente y doctor

E/P/S	
Entradas (datos requeridos)	Datos del paciente, doctor, fecha y hora
Proceso (pasos, decisiones, fórmulas si aplica)	-Verificar que el doctor esté disponible en la fecha y hora seleccionada. -Crear una instancia “cita”. -Añadir la cita a la “Agenda”. -Actualizar el estado (ocupado).
Salidas (resultado observable)	Cita creada y confirmada

Contrato PRE/POST	
PRE	El médico y el paciente deben existir en el sistema.
POST	Nueva cita registrada en la agenda del consultorio.

Campo	Contenido a completar
Clase:	Médico
Método:	mostrarDatos()
Propósito (una línea)	Mostrar la información del médico registrada en el sistema.

E/P/S	
Entradas (datos requeridos)	ninguno
Proceso (pasos, decisiones, fórmulas si aplica)	Acceder a los atributos nombre, cedula, correo, titulo y mostrarlos por pantalla o devolverlos como estructura.
Salidas (resultado observable)	Datos del medico-struct datosMedico()

Contrato PRE/POST	
PRE	El objeto Médico tiene sus atributos inicializados.
POST	Los datos se muestran correctamente o se retornan

Campo	Contenido a completar
Clase:	Paciente
Método:	mostrarDatos()
Propósito (una línea)	Mostrar la información general del paciente

E/P/S	
Entradas (datos requeridos)	ninguno
Proceso (pasos, decisiones, fórmulas si aplica)	Acceder a atributos y mostrarlos: nombre, cédula, correo, teléfono, edad.
Salidas (resultado observable)	Datos mostrados en pantalla en un struct

Contrato PRE/POST	
PRE	Paciente registrado.
POST	Información mostrada correctamente.

Campo	Contenido a completar
Clase:	Cita
Método:	
Propósito (una línea)	

E/P/S	
Entradas (datos requeridos)	
Proceso (pasos, decisiones, fórmulas si aplica)	
Salidas (resultado observable)	

Contrato PRE/POST	
PRE	
POST	

Campo	Contenido a completar
Clase:	Agenda
Método:	
Propósito (una línea)	

E/P/S	
Entradas (datos requeridos)	
Proceso (pasos, decisiones, fórmulas si aplica)	
Salidas (resultado observable)	

Contrato PRE/POST	
PRE	
POST	

Campo	Contenido a completar
Clase:	Fecha
Método:	compararFecha()
Propósito (una línea)	Comparar dos fechas y horas para verificar si son iguales

E/P/S	
Entradas (datos requeridos)	dos objetos fechas
Proceso (pasos, decisiones, fórmulas si aplica)	comparar atributos dia y hora
Salidas (resultado observable)	true si son iguales,false si no

Contrato PRE/POST	
PRE	fechas válidas
POST	resultado de comparación retornado

E/P/S	

Entradas (datos requeridos)	dos objetos fechas
Proceso (pasos, decisiones, fórmulas si aplica)	comparar atributos dia y hora
Salidas (resultado observable)	true si son iguales,false si no

Contrato PRE/POST	
PRE	fechas válidas
POST	resultado de comparación retornado

Programa en C++

Evidencia de ejecución en c++