Hospital San José St. Bonaventure (Trabajo recuperación)

María Paz Puerta Acevedo

Heily Yohana Rios Ayala

Universidad Autónoma

Programación Orientada a Objetos
Carlos Andres Rojas

13/04/2025
Manizales, Caldas, Colombia

Enunciado

El Hospital San José St. Bonaventure ha venido a la Universidad Autónoma de Manizales con el fin de solicitar a los estudiantes de la asignatura de Programación Orientada a Objetos la renovación de su sistema central, el cual está dividido por los siguientes módulos, Gestión de Empleados, Gestión de Pacientes, Gestión de Farmacia y Reportes. Para realizar el sistema debe considerar que del Hospital se almacenará su nombre, dirección, teléfono, logo, presupuesto, meta de ventas anual, fecha de fundación, estado (activo o en quiebra) y localización, para esta última es importante contemplar latitud y longitud, ya que facilitará identificar su ubicación mediante un mapa, también es importante asociar la información del gerente del cual se conoce su nombre, número de documento de identificación, edad y carrera. Esta información puede ser editada por el usuario en cualquier momento mediante un archivo de texto plano. En cuanto a cada módulo tenga en cuenta lo siguiente:

Módulo de Empleados:

Se debe permitir la gestión de empleados, teniendo en cuenta que de cada uno se almacenará su nombre, número de documento, edad, salario base, además de esto considere que existen dos tipos de empleados en el hospital, el Empleado del área de la salud del cual se debe almacenar su especialidad y número de horas trabajadas, por otro lado, se cuentan con Empleados operativos de ellos se debe almacenar el área para la cual trabajan. El proceso para calcular salario de ambos empleados es diferente; para el empleado de la salud se deberá tomar el 1.2% del salario base y multiplicarlo por el número de horas trabajadas y luego sumarlos. Al empleado operativo se le sumará el 20% al salario base para obtener el salario neto a retornar.

Debe considerar que el sistema permitirá el proceso de nómina el cual consiste en sumar el salario de todos los empleados y almacenarlos junto a la fecha de realización de la nómina y un identificador único. Tenga en cuenta que cada que se ejecuta un proceso de generación y almacenamiento de nómina se debe descontar el total de esa nómina al presupuesto global del hospital, considerar que en caso tal de que el presupuesto del hospital sea negativo se debe ejecutar un evento anómalo y manejarlo ya que esto cambia el estado del hospital de activo a en quiebra, no se debe permitir generar más procesos de nómina, para repararlo debe ejecutar una acción en el hospital denominada registrar patrocinio el cual tiene un valor y este si es superior al monto en deuda podrá cambiar el estado del hospital y permitir realizar las tareas, también tenga en cuenta que para todos estos cambios se debe informar al usuario del sistema que esto ocurrió.

Módulo de Pacientes:

Se permitirá la gestión de pacientes de los cuales se conoce su nombre, número de documento, edad, correo electrónico, teléfono, estado (saludable o crítico), una lista de enfermedades (nombre y descripción), estos serán recetables (al igual que cualquier persona) ya que podrán ejecutar la acción de curar enfermedad la cual consiste en recibir un medicamento y una enfermedad pues esta será eliminada de la lista de enfermedades del paciente y se agrega el medicamento (nombre, descripción y enfermedad que alivia) a la lista correspondiente de medicinas, en caso tal de que se recete un medicamento que ya está en la lista o se intente curar una enfermedad que no posee el paciente se ejecutará una anomalía en el sistema por lo cual se deberá informar al usuario que esto no se permite, además el paciente cambiará el estado de saludable a crítico, o cuando el paciente se encuentre sin lista de enfermedades su estado volverá a ser saludable.

Módulo de Farmacia:

Se debe permitir la gestión de medicamentos mediante el uso de un inventario del cual se conoce su código y año de actualización, en él se almacenará la información de los medicamentos, los cuales almacenan su nombre, su descripción, su costo y su precio de venta, existen dos clases diferentes de medicamentos, los genéricos (calculan su precio de venta sumando un 10% al costo) y los de marca (almacenan el fabricante y calculan su precio de venta sumando un 25% al costo) el cálculo del precio de venta se realiza de forma automática una vez se agregue este al inventario. Cada vez que se registra un medicamento se deberá disminuir el costo al presupuesto del hospital en caso tal de que el presupuesto sea negativo se debe considerar en quiebra, y en caso de que ocurra no se debe permitir comprar más medicamentos para agregarlos al inventario (para repararlo debe realizar el flujo mencionado en el módulo de empleados) y se debe informar al usuario del sistema que esto ocurrió.

Módulo Reportes:

Se debe generar reportes asociados a los módulos anteriores en el formato de su PDF también considerar persistir en archivos de texto las acciones realizadas en el sistema con el fin de que no se pierda información cuando se cierre el sistema.

Entregables

- El modelo de clases y de excepciones (identificando claramente sus clases, atributos y métodos principales, relaciones entre clases, clases y métodos abstractos y si se presenta polimorfismo).
- Indicar en qué parte del diagrama se evidencia.

Identificación de clases

- Hospital

Atributos

- [ ] String nombre
- [ ] String direccion
- [ ] String telefono
- [ ] String logo
- [ ] double presupuesto
- [ ] double metaVentasAnual
- [ ] LocalDate fechaFundacion
- [ ] boolean activo

Métodos

- [ ] generarNomina(): void
- [ ] registrarPatrocinio(monto : double): void
- [ ] descontarDelPresupuesto(valor : double): void
- [ ] visualizarEstado() : String
- [ ] comprarMedicamento(medicamento : Medicamento, cantidad : int): void
- [ ] agregarEmpleado(empleado : Empleado) : void
- [ ] eliminarEmpleado(nombre : String) : boolean
- [ ] buscarEmpleado(nombre : String) : Empleado
- [ ] actualizarEmpleado(nombre : String, empleadoActualizado : Empleado) : void
- [ ] buscarMedicamento(nombre : String) : Medicamento
- [ ] agregarPaciente(paciente : Paciente) : void
- [ ] eliminarPaciente(nombre : String) : boolean
- [ ] buscarPaciente(nombre : String) : Paciente
- [ ] actualizarPaciente(nombre : String, pacienteActualizado : Paciente) : void
- [ ] eliminarMedicamento(nombre : String) : boolean
- [ ] guardarInformacion() : void

- ☐ mostrarInformacion() : String

- Gerente

  Atributos

  - ☐ String nombre
  - ☐ String documento
  - ☐ int edad
  - ☐ String carrera

  Métodos

- Localizacion

  Atributos

  - ☐ double latitud
  - ☐ double longitud

  Métodos

- Empleado (abstracta)

  Atributos

  - ☐ String nombre
  - ☐ String documento
  - ☐ int edad
  - ☐ double salarioBase

Métodos

- ☐ abstract double calcularSalario() : double

- EmpleadoSalud (hereda de Empleado)

  Atributos

  - ☐ String especialidad
  - ☐ int horasTrabajadas

  Métodos

  - ☐ double calcularSalario()

- EmpleadoOperativo (hereda de Empleado)

  Atributos

  - ☐ String area

  Métodos

  - ☐ double calcularSalario()

- Nomina

  Atributos

- [ ] String id
- [ ] LocalDate fecha
- [ ] double totalPagado

Métodos:

- Paciente

Atributos:

- [ ] String nombre
- [ ] String documento
- [ ] int edad
- [ ] String correo
- [ ] String telefono
- [ ] estadoPaciente : boolean

Métodos:

- [ ] curarEnfermedad(enfermedad : String, medicamento : Medicamento) : String
- [ ] agregarEnfermedad(Enfermedad enfermedad) : boolean

- Farmacia

Atributos

Métodos

- [ ] agregarMedicamento(medicamento : Medicamento, cantidad : int) : void
- [ ] eliminarMedicamento(nombre : String) : void
- [ ] obtenerMedicamento(nombre : String) : ArrayList<Medicamento>
- [ ] buscarMedicamento (nombre : String): Medicamento

- Enfermedad

  Atributos

  - ☐ String nombre
  - ☐ String descripcion

  Métodos

- Medicamento (abstracta)

  Atributos:

  - ☐ String nombre
  - ☐ String descripcion
  - ☐ double costo
  - ☐ double precioVenta
  - ☐ String enfermedadQueAlivia

  Métodos:

  - ☐ <<abstract>> + calcularPrecioVenta() : void

- MedicamentoGenerico (hereda de Medicamento)

  Métodos:

- ☐ void calcularPrecioVenta() – suma 10% al costo

- MedicamentoMarca (hereda de Medicamento)

  Atributos:

  - ☐ String fabricante

  Métodos:

  - ☐ calcularPrecioVenta() : void – suma 25% al costo

- SistemaCentral

  Atributos

  Métodos

  - ☐ agregarEmpleados(empleado : Empleado) : void
  - ☐ eliminarEmpleado(nombre : String) : boolean
  - ☐ buscarEmpleado(nombre : String) : Empleado
  - ☐ actualizarEmpleado(empleado : Empleado) : void
  - ☐ obtenerEmpleados() : ArrayList<Empleado>
  - ☐ agregarPacientes(paciente : Paciente) : void
  - ☐ eliminarPaciente(nombre : String) : boolean
  - ☐ buscarPaciente(nombre : String) : Paciente
  - ☐ actualizarPaciente(paciente : Paciente) : void
  - ☐ realizarCompra(medicamento : Medicamento, cantidad : int) : void
  - ☐ eliminarMedicamento(nombre : String) : boolean
  - ☐ obtenerMedicamentos() : ArrayList<Medicamento>
  - ☐ buscarMedicamento(nombre : String) : Medicamento
  - ☐ generarNomina() : void

- Inventario

  Atributos

  - [ ] String codigo
  - [ ] int anioActualizacion
  - [ ] List<Medicamento>Medicamento

  Métodos

  - [ ] agregarMedicamento(medicamento : Medicamento, cantidad : int) : void
  - [ ] eliminarMedicamento(nombre : String) : boolean
  - [ ] buscarPorNombre(nombre : String) : Medicamento
  - [ ] obtenerListaMedicamentos() : ArrayList<Medicamento>

- Escritor (interface)

  Métodos:

  - [ ] <<abstract>> + escribir(ArrayList<String>) : void

- Lector (interface)

  Métodos:

  - [ ] <<abstract>> + leer(String) : ArrayList<String>

- LectorTextoPlano implements Lector

  Atributos:

  - [ ] charset : Charset

  Métodos:

  - [ ] leer(localizacionArchivo : String): ArrayList<String>
  - [ ] agregarLinea(linea : String) : void
  - [ ] limpiarTexto(texto : ArrayList<String>) : void

- EscritorTextoPlano implements Escritor

  Atributos:
  - ☐ final charset : Charset
  - ☐ filePath : String

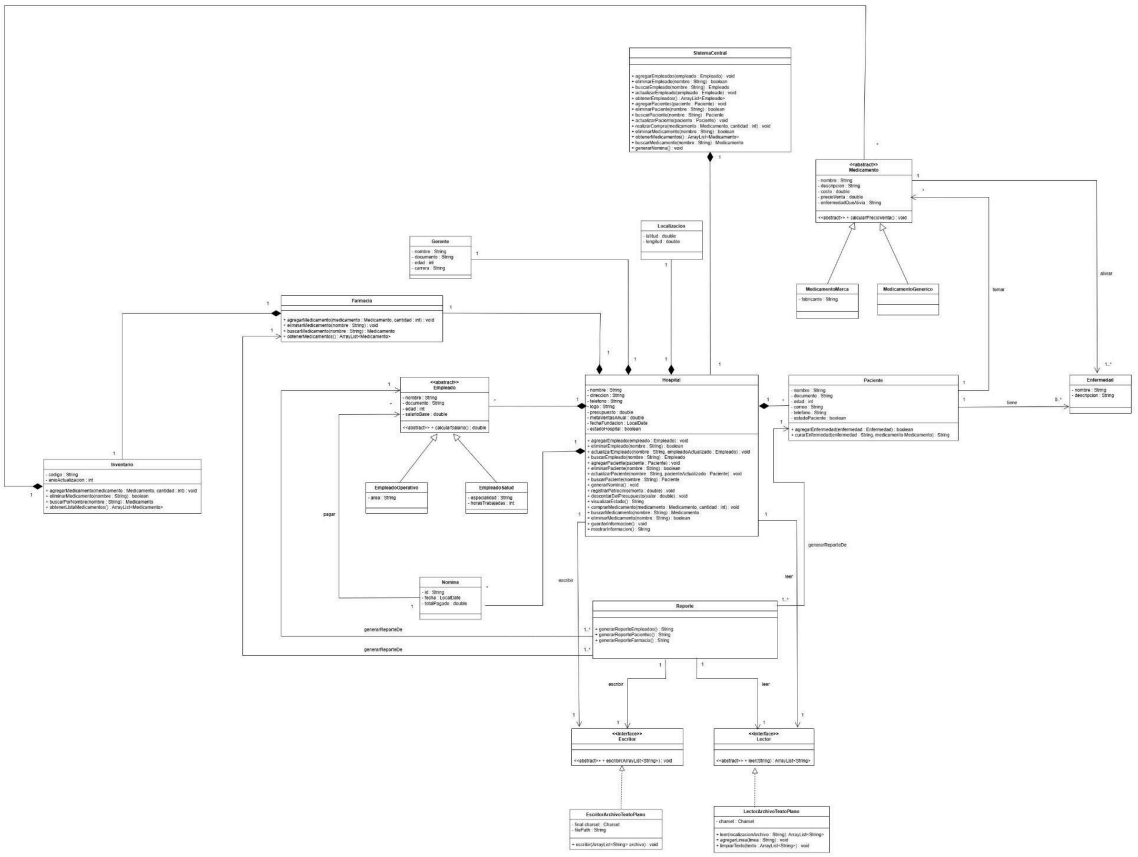
  Métodos:
  - ☐ escribir(ArrayList<String> archivo) : void

- Reporte

  Métodos:

  - ☐ generarReporteEmpleados(): String
  - ☐ generarReportePacientes(): String
  - ☐ generarReporteFarmacia(): String

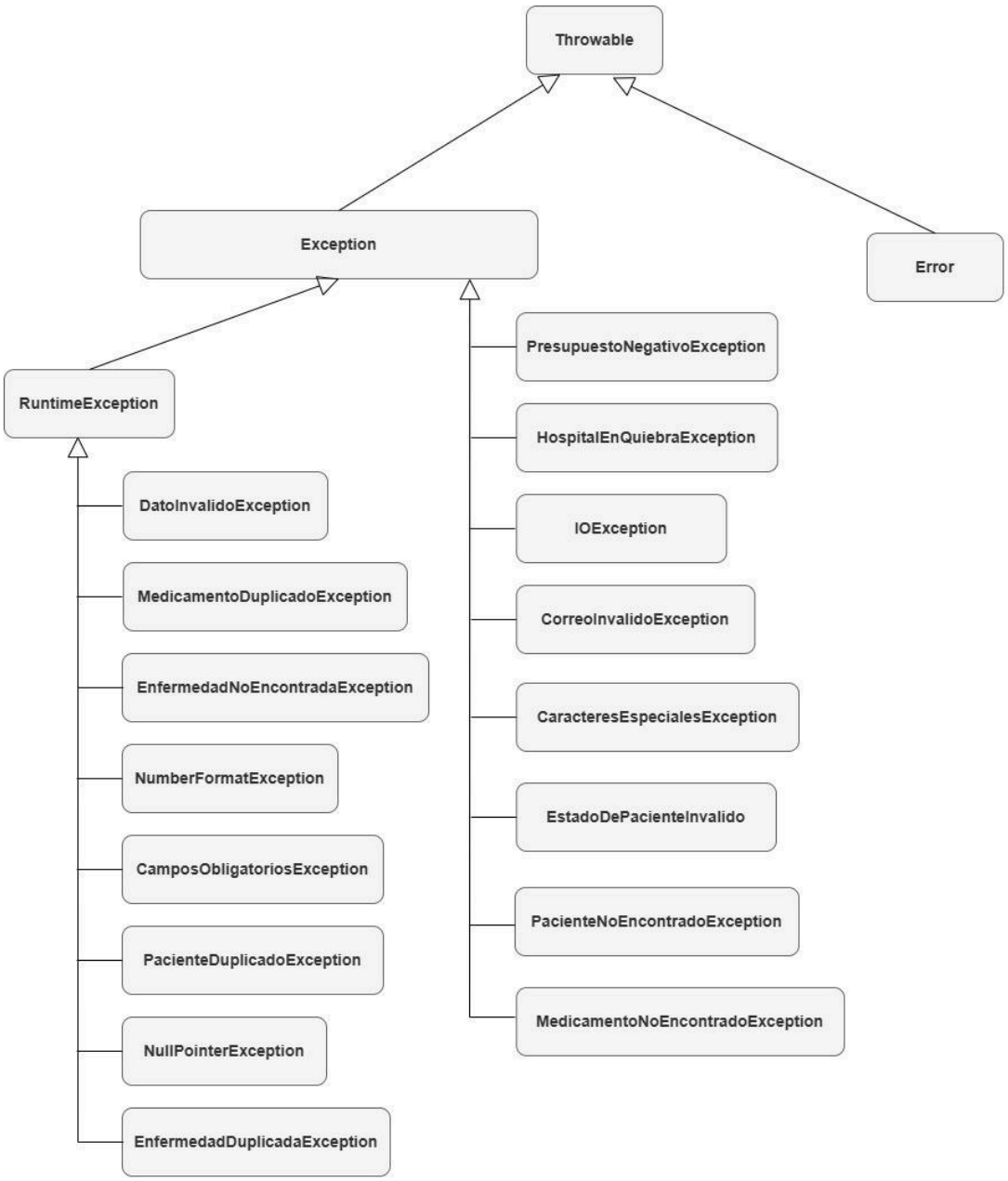
Diagrama de clases



Lista de excepciones

- **PresupuestoNegativoException:** Se lanzará esta excepción cuando el presupuesto sea negativo, en este caso se declarará en quiebra el hospital.
- **DatoInvalidoException:** Se lanzará esta excepción cuando por ejemplo, el costo, salario base y las horas trabajadas sea negativo.
- **CorreoInvalidoException:** Se lanzará cuando el correo no tenga un " @".
- **CarecteresEspecialesException:** Cuando el usuario intenta registrar solo caracteres especiales.
- **HospitalEnQuiebraException:** Se lanzará cuando el hospital esté en quiebra, en donde no se podrá procesar la nómina de los empleados.
- **MedicamentoDuplicadoException:** Se lanzará cuando se intente añadir un medicamento que ya está en la lista del paciente.
- **MedicamentoNoEncontradoException:** Se lanzará cuando no se encuentre un medicamento en la lista de medicamentos.
- **EnfermedadNoEncontradaException:** Se lanzará cuando se intente curar una enfermedad que no esté en la lista del paciente.
- **IOException:** Se lanzará cuando no se pueda leer o escribir el archivo de manera correcta.
- **EstadoDePacienteInvalido:** Se lanzará cuando se seleccione un estado del paciente inválido donde solo puede haber dos que son saludable y crítico.
- **PacienteNoEncontradoException:** Se lanzará si no se encuentre la persona en la lista de pacientes.
- **NumberFormatException:** Se lanzará cuando haya una mala conversión de tipos de datos en el programa.
- **CamposObligatoriosException:** Se lanzará cuando no se llenen los campos obligatorios en el programa.
- **PacienteDuplicadoException:** Se lanzará cuando ya haya un paciente con los mismos datos de otro paciente que ya esté en la lista.
- **NullPointerException:** Se lanzará cuando no se encuentre un objeto en una lista.
- **EnfermedadDuplicadaException:** Se lanzará cuando se intente añadir una enfermedad que ya está en la lista del paciente.

Diagrama de excepciones



Enlace GitHub: https://github.com/heily0118/HospitalApp.git