Proyecto Final Programación

*Pensamiento Computacional*

Abstracción

* ¿Qué se solicita finalmente? (problema)
  + sistema de gestión para la administración de un Hospital
* ¿Qué información es relevante dado el problema anterior?

gestión de historiales médicos, programación de consultas y asignación de médicos, proporcionando una interfaz segura e intuitiva.

* ¿Cómo se agrupa la información relevante?
  + Hospital: pacientes, médicos, administradores, salas, citas, historias
  + Paciente: Datos personales, Historial clínico,
  + Medico: Horario
  + Administradores
* ¿Qué funcionalidades se solicitan?
* Funcionalidades Generales de Pacientes

● Registro y actualización de datos personales.

● Solicitud y cancelación de citas médicas.

● Consulta de historial médico.

● Recepción de notificaciones sobre citas programadas.

* Funcionalidades Generales de Médicos

● Acceso a los historiales médicos de sus pacientes.

● Registro de diagnósticos y tratamientos.

● Administración de horarios de consulta.

● Notificación de cambios en las citas.

* Administradores

● Registro, modificación y eliminación de médicos y pacientes.

● Gestión de salas y horarios de atención.

● Monitoreo de disponibilidad de médicos y asignación de pacientes.

● Generación de reportes sobre citas médicas y ocupación del hospital.

* Funcionalidad Extendida (Opcional)

● Manejo de archivos médicos: Adjuntar documentos como resultados de exámenes y recetas médicas.

● Sistema de notificaciones: Envío de recordatorios sobre citas y tratamientos por correo electrónico.

● Integración con farmacias: Conexión con farmacias para solicitar medicamentos recetados.

Descomposición

¿Cómo se distribuyen las funcionalidades?

¿Qué debo hacer para probar las funcionalidades?

Reconocimiento de patrones

¿Qué puedo reutilizar de la solución de otros problemas?

Estructura básica del programa consultorio odontologico

Codificación

¿Cómo pruebo la solución en java?

¿Cómo escribo la solución en java?

*Codigo adjunto*

*Otros*

Requisitos de entrega:

Mínimo 15 commits (cambios relevantes, modificaciones al código) por integrante

Maximo 3 integrantes, solo un integrante entrega.

Cada commit bien documentado.

Utilizar JavaFx para las interfaces, herencia, polimorfismo, encapsulamiento:y abstracción.

Commits

Cada commit tiene su identificador.

Cada commit dice que se hizo en el código .

Investigar cosas y eso sobre GitHub .