DOCUMENTACIÓN
PROYECTO PRÁCTICAS.

ÍNDICE:

1. DEFINICIÓN Y ESPECIFICACIÓN DE REQUISITOS. 3

1.1. DEFINICIÓN GENERAL DEL PROYECTO SOFTWARE. 3

1.2. ESPECIFICACIÓN DE REQUISITOS DEL PROYECTO. 3

1.3. PROCEDIMIENTOS DE INSTALACIÓN Y PRUEBA. 4

2. ARQUITECTURA DEL SISTEMA. 5

2.1. DESCRIPCIÓN JERÁRQUICA. 5

2.2. DIAGRAMA DE CASOS DE USO. 5

2.3. DESCRIPCIÓN INDIVIDUAL DE LOS CASOS DE USO. 5

3. DISEÑO DEL MODELO DE DATOS. 6

4. DESCRIPCIÓN DE PROCESOS Y SERVICIOS OFRECIDOS POR EL SISTEMA. 6

1. DEFINICIÓN Y ESPECIFICACIÓN DE REQUISITOS.

1.1. DEFINICIÓN GENERAL DEL PROYECTO SOFTWARE.

Este proyecto software se basa en la gestión de pacientes con algún tipo de enfermedad infectocontagiosa, de manera que, su objetivo principal es la catalogación de dichas enfermedades, la generación de campañas informativas, la detección temprana de positivos y la realización de diversas estadísticas.

Para ello, es necesario conocer el estado en que puede encontrarse una persona y mantener una base de datos actualizada con todas las posibles enfermedades infectocontagiosas, sus síntomas y la forma de tratarlas (medicinas, vacunas, …)

En caso de tener que realizar campañas de vacunación, será necesario contar con información personal de la ciudadanía y realizar un calendario de vacunación.

Este sistema será utilizado exclusivamente por el administrador, que será el encargado de realizar todas las acciones descritas anteriormente, por lo que debe poseer un elevado nivel de experiencia y debe conocer a la perfección el funcionamiento de dicho sistema software.

1.2. ESPECIFICACIÓN DE REQUISITOS DEL PROYECTO.

Los requisitos funcionales que el sistema debe satisfacer se encuentran agrupados en casos de uso, y son los siguientes:

Caso de Uso 1: Catalogación y mantenimiento de enfermedades.

- Añadir elementos a una base de datos.
- Modificar elementos de una base de datos.
- Añadir campañas informativas.
- Modificar campañas informativas.

Caso de Uso 2: Detección y seguimiento de positivos.

- Añadir pacientes.
- Registrar estado del paciente.

Caso de Uso 3: Generación y ejecución del plan de vacunaciones.

- Registrar previsiones de aprovisionamiento de vacunas.
- Establecer calendario de vacunación.

Caso de Uso 4: Realización de estadísticas.

- Realización de diversas estadísticas.

Por otro lado, los requisitos no funcionales que debe satisfacer el sistema son aquellos relacionados con el rendimiento, la disponibilidad, la accesibilidad, la portabilidad, la seguridad, la escalabilidad, la interoperabilidad, …

El **alcance** de este sistema será el área de epidemiología de un hospital, es decir:

- Médicos y doctores encargados de la detección y catalogación de enfermedades infectocontagiosas y clasificación del estado de un paciente.
- Rastreadores, que realizan el seguimiento de los positivos.
- Enfermeros y otro personal sanitario, encargados de realizar campañas de vacunación.
- Personal de administración, encargados de la realización de estadísticas.

1.3. PROCEDIMIENTOS DE INSTALACIÓN Y PRUEBA.

Al no haber sido capaces de desarrollar el código del proyecto, no hemos podido realizar los procedimientos de instalación y pruebas. De manera que, en este apartado, describiremos la forma en que habríamos llevado a cabo dichos procedimientos en caso de haber tenido el código de la practica desarrollado.

En primer lugar, debemos destacar que el entorno de desarrollo de software es Eclipse, y el framework en el que realizaremos la fase de pruebas será JUnit.

Antes de realizar pruebas en el sistema, debemos tener un conjunto de valores de prueba, que serán aquellos valores propensos a generar fallos en el sistema, tales como valores limite o conjetura de errores. Una vez obtenidos, pasamos a realizar las pruebas.

Sería buena idea la utilización de un modelo de pruebas que nos ayudara a realizar dichas pruebas correctamente y a alcanzar un buen nivel de cobertura, por lo que utilizaremos el **modelo en V de pruebas**.

Por lo tanto, primeramente, realizaremos pruebas unitarias a cada una de las clases del código, donde utilizaremos pruebas de caja blanca para comprobar la estructura interna del sistema.

A continuación, realizaremos las pruebas de integración, donde comprobaremos que las unidades de prueba funcionan correctamente cuando se integran junto a otras clases. En este caso usaremos pruebas de caja negra, aunque también podríamos utilizar pruebas de caja blanca si quisiéramos comprobar los flujos de comunicación entre partes del sistema.

Después, realizaremos las pruebas de sistema, que son aquellas que comprueban que el sistema se comporta correctamente en su entorno. Estas pruebas son de caja negra y suelen estar relacionadas con las características de calidad.

Por último, las pruebas de aceptación las realizará el cliente (en este caso, el profesor) para aceptar el software producido y/o rechazarlo y demandar correcciones o mejoras. Estas pruebas son siempre de caja negra y no suelen estar automatizadas.

Una vez realizadas todas estas pruebas con éxito, podríamos decir que hemos alcanzado un **nivel de cobertura**, lo que se traduce en una cierta garantía de calidad del sistema.

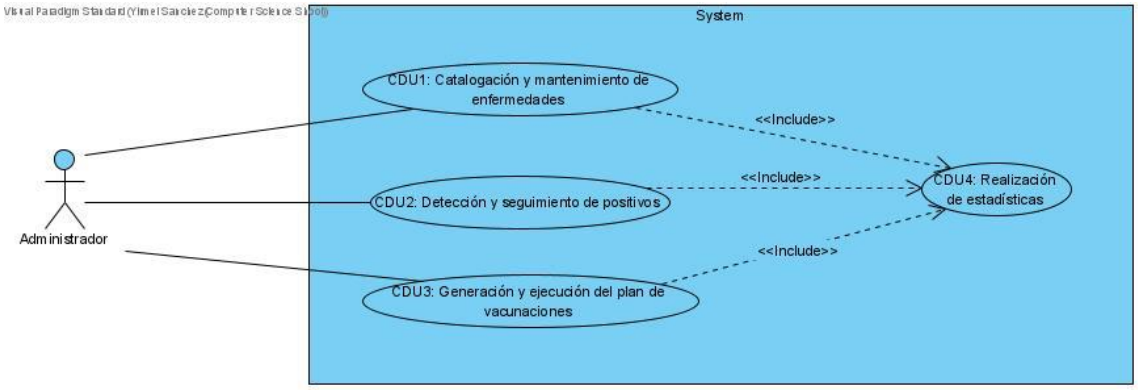
2. ARQUITECTURA DEL SISTEMA.

2.1. DESCRIPCIÓN JERÁRQUICA.

Este sistema utiliza una estructura cliente-servidor, donde las tareas se reparten entre los proveedores de servicios (servidores) y los demandantes (clientes). Los clientes son los que realizan peticiones a los servidores, que les proporcionan la respuesta.

2.2. DIAGRAMA DE CASOS DE USO.

El diagrama de casos de uso de nuestro sistema es el siguiente:



Donde observamos que el administrador es el encargado de llevar a cabo todos los casos de uso descritos anteriormente. Asimismo, podemos ver dependencias de tipo <<include>> entre los casos de uso 1, 2 y 3 con el caso de uso 4, ya que, si anteriormente no hemos catalogado enfermedades, detectado positivos o generado un plan de vacunación, no podremos realizar estadísticas referidas a estos casos de uso.

2.3. DESCRIPCIÓN INDIVIDUAL DE LOS CASOS DE USO.

Caso de Uso 1: Catalogación y mantenimiento de enfermedades. Se encarga de mantener actualizada una base de datos con todas las enfermedades infectocontagiosas, así como sus síntomas, la temporalidad de cada una de las enfermedades y las posibles formas de tratarlas (medicinas, confinamientos, vacunas y campañas de vacunación, …). Además, también se encargará de realizar campañas informativas preventivas.

Caso de Uso 2: Detección y seguimiento de positivos. Se encarga de realizar una detección temprana de los contagiados y un seguimiento de los mismos. Para ello se debe registrar el estado de cada paciente, que puede ser:

- Enfermo. Persona actualmente contagiada que puede necesitar confinamiento o atención hospitalaria.
- Enfermo en cuarentena. Persona contagiada que necesita de cuarentena.
- Sano. Persona que no está infectada.
- Sano en cuarentena. Persona sana, pero ha estado en contacto con algún enfermo.

Caso de Uso 3: Generación y ejecución del plan de vacunaciones. Se encarga de lanzar campañas de vacunación y realizar los correspondientes calendarios de vacunación, teniendo en cuenta las previsiones de aprovisionamiento de vacunas.

Caso de Uso 4: Realización de estadísticas. Se encarga de la realización de determinadas estadísticas, como el numero de enfermedades recogidas, el número de campañas informativas realizadas, la evolución de los pacientes en el tiempo, la relación de personas vacunadas e infectadas en un periodo de tiempo, …

Este último caso de uso, como hemos visto antes, depende de los 3 anteriores ya que necesita los datos que estos van a ir registrando en la base de datos para poder realizar estadísticas.

3. DISEÑO DEL MODELO DE DATOS.

La entidad involucrada en el sistema será el administrador, que es el encargado de añadir, modificar o eliminar elementos de la base de datos del sistema.

Por un lado, los datos de entrada del sistema será aquella información que el administrador añada a la base de datos del sistema, tal como tipos de enfermedades infectocontagiosas, síntomas de cada tipo de enfermedad, estado del paciente, planes de vacunación, …

Por otro lado, los datos de salida serán las diversas estadísticas realizadas sobre los datos de entrada, tales como la previsión de olas de contagio por cada una de las enfermedades, estadísticas de vacunación por cada enfermedad, tipos de enfermedades catalogadas, …

4. SERVICIOS OFRECIDOS POR EL SISTEMA.

El sistema ofrece una serie de servicios, como:

- Asistencia especializada en función del tipo de paciente, enfermedad, nivel de vulnerabilidad o tratamiento.
- Procedimientos diagnósticos.
- Procedimientos informativos.
- Hospitalización domiciliaria.
- Proveer tratamientos especializados.

5. DOCUMENTACIÓN TÉCNICA.

En este apartado detallaremos el contenido de cada una de las clases del sistema:

- ***Agente_BBDD.java*** -> contiene los siguientes métodos:
    - *insertarEnfermedad()*
    - *cargarEnfermedades()*
    - *actualizarEnfermedades()*
    - *borrarEnfermedad()*
    - *insertarSintoma()*
    - *cargarSintomas()*
    - *actualizarSintoma()*
    - *borrarSintoma()*
    - *añadirVacuna()*
    - *cargarVacunas()*
    - *actualizarVacuna()*
    - *borrarVacuna()*
    - *añadirMedicina()*
    - *cargarMedicinas()*
    - *actualizarMedicina()*
    - *borrarMedicina()*
    - *añadirCampañaVacuna()*
    - *actualizarCampañaVacuna()*
    - *borrarCampañaVacuna()*
- ***CampañaInfo.java*** -> Contiene información acerca campaña de vacunación tal como la enfermedad, la temporalidad, la efectividad y el coste.
- ***CampañaVacuna.java*** -> Posee información de la vacuna y la fecha de inicio de vacunación. Además, contiene métodos como:
    - *añadirCampañaVacunacion()*
    - *actualizarCampañaVacunacion()*
    - *borrarCampañaVacunacion()*
- ***Enfermedad.java*** -> Esta clase contiene información acerca de la enfermedad como la temporalidad, si es necesario confinamiento, las medicinas necesarias para tratarla, las vacunas, los síntomas que tiene o su propio nombre. Además, contiene los métodos:
    - *crearEnfermedad()*
    - *cargarEnfermedades()*
    - *actualizarEnfermedad()*
    - *borrarEnfermedad()*
- ***GestorCampañas.java*** -> Pose los siguientes métodos:
    - *obtenerCampañasVacunacion()*
    - *añadirCampañaVacunacion()*
    - *actualizarCampañaVacunacion()*
    - *borrarCampañaVacunacion()*
- ***GestorEnfermedades.java*** -> Tiene los siguientes métodos:
    - *insertarEnfermedad()*
    - *cargarEnfermedades()*
    - *actualizarEnfermedades()*

    - *borrarEnfermedad()*
    - *insertarSintoma()*
    - *cargarSintomas()*
    - *actualizarSintoma()*
    - *borrarSintoma()*
    - *añadirVacuna()*
    - *cargarVacunas()*
    - *actualizarVacuna()*
    - *borrarVacuna()*
    - *añadirMedicina()*
    - *cargarMedicinas()*
    - *actualizarMedicina()*
    - *borrarMedicina()*
- ***IU_Campañas.java*** -> Tiene métodos como:
    - *mostrarCampañasVacunacion()*
    - *introducirDatosCampaña()*
    - *actualizarCampaña()*
    - *borrarCampaña()*
    - *mostrarCampaña()*
- ***IU_Enfermedades.java*** -> Tiene los siguientes métodos:
    - *insertarEnfermedad()*
    - *cargarEnfermedades()*
    - *actualizarEnfermedades()*
    - *borrarEnfermedad()*
    - *insertarSintoma()*
    - *cargarSintomas()*
    - *actualizarSintoma()*
    - *borrarSintoma()*
    - *añadirVacuna()*
    - *cargarVacunas()*
    - *actualizarVacuna()*
    - *borrarVacuna()*
    - *añadirMedicina()*
    - *cargarMedicinas()*
    - *actualizarMedicina()*
    - *borrarMedicina()*
- ***Medicina.java*** -> Contiene el nombre de la medicina y los siguientes métodos:
    - *añadirMedicina()*
    - *cargarMedicinas()*
    - *actualizarMedicina()*
    - *borrarMedicina()*
- ***Síntoma.java*** -> Contiene la gravedad de los síntomas y los siguientes métodos:
    - *insertarSintoma()*
    - *cargarSintomas()*
    - *actualizarSintoma()*
    - *borrarSintoma()*

- ***Vacuna.java*** -> Contiene información como el nombre de la vacuna y su eficacia. Además, tiene los siguientes métodos:
    - *añadirVacuna()*
    - *cargarVacunas()*
    - *actualizarVacuna()*
    - *borrarVacuna()*
- ***Contagiado.java*** -> Tiene información acerca del nombre y apellidos de las personas contagiadas, su DNI, una lista de contactos cercanos, el número de teléfono y si esta persona debe hacer cuarentena. Además, contiene una serie de métodos como son:
    - *añadirContagiado()*
    - *consultarContagiados()*
    - *actualizarContagiados()*
    - *borrarContagiado()*
- ***Enfermo.java***
- ***Gestor_Contagiados.java*** -> Contiene los siguientes métodos:
    - *obtenerDatos()*
    - *añadirContagiado()*
    - *consultarContagiados()*
    - *actualizarContagiados()*
    - *borrarContagiado()*
- ***IU_Contagiados.java*** -> Contiene estos métodos:
    - *añadirContagiado()*
    - *consultarContagiados()*
    - *actualizarContagiados()*
    - *borrarContagiado()*
- ***CalendarioVacunacion.java*** -> Contiene información relativa a las fechas de inicio y fin de riesgos, tanto para los pacientes como para los sanitarios, así como sus métodos get y set expuestos a continuación:
    - *getFechaInicioPRiesgo()*
    - *setFechaInicioPRiesgo()*
    - *getFechaFinPRiesgo()*
    - *setFechaFinPRiesgo()*
    - *getFechaInicioSanitarios()*
    - *setFechaInicioSanitarios()*
    - *getFechaFinSanitarios()*
    - *setFechaFinSanitarios()*
    - *getFechaInicioPersonas()*
    - *setFechaInicioPersonas()*
    - *getFechaFinPersonas()*
    - *setFechaFinPersonas()*
- ***Previsión.java*** -> Contiene como atributos un identificador de la vacuna, su nombre y la fecha y el número de vacunas; además de sus métodos get y set correspondientes:
    - *getId()*
    - *setId()*
    - *getNombre()*
    - *setNombre()*

    - *getNumeroVacunas()*
    - *setNumeroVacunas()*
- ***IU_Prevision.java*** -> Posee un registro de aprovisionamiento de vacunas.
- ***IU_Vacunacion.java*** -> Posee los siguientes métodos:
    - *añadirCalVacunacion()*
    - *modificarCalVacunacion()*
    - *borrarCalVacunacion()*
- ***gestorCalVacunacion.java*** -> Contiene información relativa a las fechas de inicio y fin de riesgos, tanto para los pacientes como para los sanitarios, así como estos métodos:
    - *añadirCalVacunacion()*
    - *modificarCalVacunacion()*
    - *borrarCalVacunacion()*
- ***gestorPrevisiones.java*** -> Posee parámetros como el nombre, la fecha o el numero de vacunas. Además, contiene los siguientes métodos:
    - *añadirPrevisión()*
    - *modificarPrevisión ()*
    - *borrarPrevisión ()*
- ***gestorEstadisticas.java*** -> Contiene los siguientes métodos:
    - *consultarEnfermedadesYPersonas()*
    - *consultarNEnfermedades()*
    - *consultarEnfermedades()*
    - *consultarEvolucion()*
    - *consultarEstimacion()*
    - *consultarNCampañasInfo()*
    - *consultarDatosVacunacion()*
    - *consultarRelacion()*
- ***IU_Estadisticas.java*** -> Posee los siguientes métodos:
    - *mostrarListaContagiados()*
    - *mostrarNumeroEnfermedades()*
    - *mostrarListaYGraficaTemporal()*
    - *mostrarListaEnfermedades()*
    - *mostrarEstimacion()*
    - *mostrarMensajesNumeroCampaña()*
    - *mostrarInformacionVacunacion()*
    - *mostrarMensajeRelacion()*