ISO II
MEMORIA

Hecho por:

Daniel Sánchez Pérez

Julián Mencías Cerro

ÍNDICE:

1. Análisis de Requisitos.

El proyecto que nos han encargado consiste en realizar un programa de gestión de vacunas para el COVID-19, donde hay que gestionar como se recogen lotes provenientes de las farmacéuticas, junto con el reparto entre las diferentes comunidades autónomas dependiendo del grado de prioridad, que se calcula a partir de los grupos prioritarios, la incidencia acumulada y la población de dicha comunidad. Finalmente, se tienen que registrar las vacunaciones a cada paciente.

Para empezar, observamos que la aplicación va a ser usada por dos tipos de usuarios: los que trabajarán en el Sistema Nacional de Salud (SNS) y los que trabajan en los sistemas regionales de salud (SRS), cuyas funciones serán distintas.

En el caso del SNS, el trabajador deberá primero poder dar de alta un lote de vacunas que reciba procedente de una farmacéutica, donde se debe conocer: su ID, que es obligatoria y única; el número de vacunas recibidas; el tipo de vacuna, que está formado por nombre y proveedor; y la fecha en la que se reciben. Una vez que se da de alta el lote, hay que repartir las vacunas entre las comunidades autónomas, pero cada una tendrá un grado de prioridad que previamente hemos definido. Por último, el trabajador podrá observar unas estadísticas a nivel nacional, o si se diese el caso a nivel de comunidades autónomas, donde se mostrarán: el número total de vacunados y el porcentaje de vacunados respecto a las vacunas entregadas.

En el caso del SRS, el trabajador debe dar de alta una entrega de vacunas procedente del SNS, cuyos datos son: su ID, la fecha en la que se recibe, el tipo de vacuna, la ID del lote del que procede, la cantidad de vacunas y el grupo prioritario para el que se hará la entrega. Además, se encargará de administrar las vacunas a sus respectivos pacientes, donde habrá que registrar los datos de este, la fecha de la vacunación, el tipo de vacuna y el número de la dosis que se le administra. Por último, podrá consultar las estadísticas de su comunidad autónoma, pudiendo revisar el número total de vacunas inoculadas, junto con el porcentaje de las vacunas recibidas.

2. Diseño del Programa.

Una vez recogidos los requisitos, nos encargamos de construir la arquitectura del proyecto. Para ello, obtuvimos el diagrama UML que nos proporcionó nuestro cliente, y con el cual generamos el esqueleto del código que implementaríamos.

3. Construcción del Programa.

3.1 Primeros pasos.

Una vez creado el esqueleto, lo transformamos en un proyecto Maven con el objetivo de gestionar el proyecto en su totalidad, ya fuera las dependencias, manejar las "build", etc.

A continuación, para poder trabajar de forma independiente en el proyecto, se creó un repositorio en GitHub, desde donde se suben las modificaciones relacionadas con el proyecto, donde participan tanto los desarrolladores, como nuestro cliente. El proyecto se realizará en código Java, usando Eclipse y Apache Netbeans.

La primera tarea que se realizó fue diseñar una presentación provisional, a partir de los requisitos y el diseño. Tras un primer contacto con el cliente, llegamos a la conclusión de que el esquema que nos daba no había que seguirlo totalmente, dándonos un margen de libertad, con lo cual empezamos a añadir ventanas y un sistema de login para los trabajadores, debido a los requisitos pedidos.

3.2 Presentación.

La primera ventana que se mostrará al ejecutar la aplicación es "Login", donde el trabajador podrá iniciar sesión aportando su identificación y contraseña, o en su defecto, registrarse, donde deberá introducir sus datos personales y donde trabaja (en la ventana de Registro). Una vez que el trabajador inicia sesión, dependiendo de donde trabaje, se abrirá un menú de gestión nacional o regional (PantallaGestionSistemaSaludNacional o PantallaGestionSistemaRegionalSalud). Ambos menús contendrán las estadísticas que nos piden en los requisitos, además de las funcionalidades para cada tipo de trabajador. En el caso del SNS, podremos tanto dar de alta un lote de vacunas (AltaLote), como repartir las vacunas entre las diferentes comunidades (Repartir). Por otro lado, en el caso del SRS, se podrá dar de alta las vacunas que lleguen a dicha comunidad (VacunasDisponibles), realizar las respectivas vacunaciones a los pacientes que haya registrados (RegistrarVacunación), y observar la lista de las vacunaciones que llevemos en ese momento (PantallaConsultaVacunados).

3.3 Persistencia.

Para poder conectar la persistencia del proyecto con la base de datos, construimos los elementos DAO. Lo primero que hacemos, es construir la propia base de datos, que será embebida (con Apache Derby), teniendo 5 tablas de datos. Con "ConsultarEstadísticasDAO", obtendremos los datos que el cliente nos pide, tanto para el ámbito nacional como para el regional, ya sea la población total, junto con el número de las personas vacunadas (para SNS) o las vacunas inoculadas (para SRS). "EntregaDAO" tiene como función insertar en la base de datos un número determinado de vacunas procedentes del SNS, indicando el lote con su respectiva cantidad y la farmacéutica, la región, y el grupo de prioridad, además de poder seleccionarlas. Por otro lado, para poder insertar usuarios o comprobar si existen con su respectiva identificación y contraseña, tenemos "LoginDAO". Como el SNS se encarga de dar de alta lotes de vacunas, se usa "LoteVacunasDAO", donde se insertarán dichos lotes en la base de datos, además de poder seleccionarlos para que nos lo muestre en la pantalla de las estadísticas del SNS, coger un lote en concreto mediante su id para poder repartirlo entre las comunidades y, por último, eliminar un lote una vez se ha repartido. Como último elemento, tenemos "VacunaciónDAO", donde se insertan las vacunaciones a los pacientes además de seleccionar sus respectivos datos.

3.4 Dominio.

En la parte de dominio de la aplicación, se encuentran los gestores, que son los que se van a encargar de realizar las funciones principales con ayudas de los DAO. Para empezar, tenemos "GestorEstadísticas", que se encarga de recuperar de la base de datos tanto a nivel nacional como regional: el número total de vacunados, el porcentaje de vacunados con respecto a la población correspondiente y, por último, en el caso del nivel regional, el número total de vacunas inoculadas junto con el porcentaje de cuáles han sido primera o segunda dosis (suponiendo que por ahora se inocularán dos dosis, aunque se dé la posibilidad de que haya más). El siguiente gestor que tenemos es "GestorLogin", que se encarga de comprobar si el dni y contraseña que introduce el usuario al logearse es correcto. El gestor "GestorRepartoVacunas", se va a encargar de dos funciones primordiales, la primera va a ser dar de alta un lote de vacunas. La segunda función, será repartir un número de vacunas entre las regiones en función de su prioridad. Para calcular la prioridad de cada comunidad, nos basamos en el cálculo de la relación vacunados/población, y una vez la tenemos realizamos la inversa, así la comunidad que menos relación tenga, más vacunas obtendrá. Por último, tenemos "GestorVacunación", donde registraremos una vacunación para un paciente, donde primero se comprueba que la vacunación no esté repetida, es decir, una misma persona no se puede vacunar con la misma dosis. Una vez comprobado, en caso de que no esté repetida se insertará la vacunación en la base de datos.

4. Calidad del Software.

Para comprobar la calidad de nuestro software, hemos usado SonarCloud. Este sistema nos analiza el código evaluando su calidad, y para ello se basa en múltiples factores.

El primero de ellos son los “code smells”, o también llamado código maloliente, que son trozos de código que, aunque haga su función, son más difícil de mantener. Otro factor que nos mira el Sonar son las exposiciones de seguridad, donde nos indica si tenemos deficiencias de seguridad. Por último, se comprueba por todo el código que no haya partes repetidas o bugs (errores).

Para que el Sonar nos evaluase el código, enlazamos el repositorio a un nuevo proyecto de Sonar, y añadimos nuestro programa. Acto seguido lo configuramos y, para que tuviésemos un análisis continuo cada vez que actualizábamos el repositorio, añadimos las propiedades del Sonar al fichero pom del proyecto, de forma que una vez que se realiza el build en el Github, también analiza su calidad en el Sonar.

5. Pruebas de Software.

En este apartado del proyecto, nos encargamos de realizar las pruebas unitarias a todas las clases de nuestro proyecto. Lo primero que hemos hecho, ha sido integrar Junit en nuestro proyecto. JUnit es un conjunto bibliotecas que se usan para realizar las pruebas anteriormente mencionadas, para evaluar si el funcionamiento de los métodos de cada una de las clases se comporta como debe. Al igual que con Sonar, integramos en el pom el JUnit, y una vez hecho, tenemos la posibilidad de crear las pruebas de las clases de dominio y persistencia de forma sencilla. Cuando se crea una clase de pruebas unitarias, aparecen todos los métodos que vamos a evaluar con sus respectivos parámetros, los cuales les podremos dar valores en función de los Casos de Prueba que hayamos calculado en un Excel.

Con JUnit creamos las clases de prueba, pero también necesitamos generar informes de dichas pruebas, y para ello necesitamos configurar en nuestro fichero pom los informes de Surefiles y Jacoco. El Surefile se usa durante la fase de pruebas para crear un informe de estas, y el Jacoco nos sirve para evaluar la cobertura de sentencia, que se integra con el JUnit.

6. Mantenimiento de Software.

Durante el periodo de mantenimiento de nuestro programa, nos hemos limitado a quitar errores y a añadir nuevas funcionalidades solicitadas por el cliente.