Universidad Carlos III

Curso Desarrollo del Software 2021-22

Práctica Final

Fecha: **23/05/2022**

GRUPO: **80** EQUIPO: **03**

Alumnos: **Rodrigo Valderrey Tarrero / Carlos Sánchez Arroyo**

Índice

1. Primera funcionalidad

1.1 Modificaciones realizadas

En cuanto a la modificación del vaccine_manager, se ha añadido una comprobación cuando este método es llamado, para asegurar que el formato de la fecha introducida es el indicado. Si esto no sucede, se llama a VaccineManagerException para hacer un raise.

En cuanto a la clase appointment, se han modificado los posibles atributos de la misma, cambiando el número de días por "date". Además, se han tenido que hacer modificaciones en el cálculo de "appointment_date", ya que si no se recibía ninguna fecha (como es el caso en las funciones aplicadas en los métodos que ya están creados), generaba un error que causaba la imposibilidad de seguir ejecutando el código. Para asegurar el funcionamiento de todo esto, el grupo ha considerado una posible solución y ha implementado que si no se especifica la fecha, la cita pase a ser para el día en el que se está solicitando. Gracias a esto, un paciente si quiere pedir una cita para hoy mismo, no necesita especificarla.

1.2 Modificaciones en los tests

Para realizar los tests a los que esta funcionalidad afectaba, se han realizado una serie de cambios. Primero, en los tests que llegaban al final y creaban una cita, ha sido necesario llamarlos con la fecha adecuada para que se compruebe todo correctamente. Además, se han tenido que revisar los parámetros existentes en los tests que daban error debido a que no encontraban algún valor en específico, o debido a que la fecha era incorrecta.

Además, debido a la falta de pruebas en la clase "tests_get_vaccine_date_tests" que verificarán el formato de la fecha introducida para solicitar una cita, el grupo ha considerado conveniente añadir varios test que comprueben el formato y los valores del més y el día correspondientes.

Por último, esta funcionalidad ha afectado a los tests pertenecientes a la clase "test_vaccine_patient_tests", por lo que se tuvo que revisar minuciosamente el código, hasta que el grupo llegó a la conclusión de que se trataba de un problema de las modificaciones realizadas en la clase a la que pertenecen las citas. Debido a este imprevisto, el código se modificó para que se pudiese llevar a cabo esta funcionalidad sin afectar a los tests realizados.

2. Segunda funcionalidad

2.1 Explicación del código implementado

En este caso, la segunda funcionalidad ha sido completamente implementada desde cero. El fin de la misma es permitir cancelar una cita que se encuentre registrada (store_date.json) o bien modificar una cancelación de "Temporal" a "Final". Para ello se ha implementado lo siguiente:

Primeras comprobaciones (formato correcto de los datos) :

1. Sirviendonos de la clase "Parser" se lee el archivo ".json" proporcionado y se almacena en variables su contenido.
2. Se comprueba (con ayuda de una expresión regex) que "date_signature" se trata de un hexadecimal de 64 caracteres. Si no es así, se devuelve una excepción
3. Se comprueba que el tipo de cancelación sea o Temporal o Final. Si no fuese así se devuelve una excepción.
4. Nos cercioramos de que la razón tenga la longitud pedida (entre 2 y 100) caracteres.

Comprobación de que la vacuna no estuviese ya administrada:

Para ello, nos volvemos a servir de la clase "Parser" y se comprueba si dentro de "store_vaccine" hay alguna cita con una firma igual. Si es así, salta una excepción puesto que la vacuna ya habría sido administrada .

Comprobación de si la vacuna ya había sido cancelada

Para esta comprobación hemos establecido un valor booleano "cambio_temp_false" que se inicializará a False y tomará el valor True si resulta que dentro de "store_canceled.json" ya está esta cita pero con tipo "Temporal" y ahora la cancelación es "Final" (puesto que en dicho caso debe modificarse).

Volvemos a crear un objeto "Parser" gracias al que comprobamos si la cita ya fue cancelada. Si así lo fue, se devuelve un error salvo si estamos en el caso mencionado anteriormente (cambio de "Temporal" a "Final").

Comprobación de si la cita que se quiere cancelar existe

Para esta comprobación basta con recorrer el contenido de "store_date.json" y asegurar que algún "date_signature" coincide con el recibido. Si no es así se devuelve una excepción.

Comprobación de si la cita ya pasó

Se ha comprobado extrayendo la fecha actual y comparándola con la fecha de la cita en el el "store-date.json" extraida en el bucle de la comprobación anterior.

Una vez comprobado todo lo mencionado anteriormente, se debe cancelar la cita. Para ello el grupo ha decidido que se añade al "store_canceled.json", sin llegar a eliminarla del archivo de citas, ya que se ha considerado que esto corresponde a una funcionalidad que se debe realizar posteriormente. Hay dos casos posibles:

1. La cancelación pasa de ser Temporal a ser Final. (Se sabe mediante el booleano mencionado anteriormente, "cambio_temp_false").
2. La cancelación simplemente se concede (no se modifica nada).

Por último se devuelve la firma de la cita que se ha cancelado.

2.2 Clases de equivalencia y valores límite

Para comprobar la correcta funcionalidad de este método hemos puesto en práctica dos de las formas que hemos aprendido en la asignatura: Clases de equivalencia y valores límites; y control de flujo. Como pruebas complementarias para asegurar todas las entradas, se han decidido crear dos clases de equivalencia, una en la que el "json" no existe, y otra en la que tiene un formato incorrecto.

A pesar de que se adjunta el excel en la entrega vía github, también se adjuntan capturas del mismo a fin de agilizar la comprensión de nuestra implementación.

Identificador	Descripción
CE_V_01	date_singnature tiene 64 caracteres en hexadecimal
CE_NV_01	date_singnature tiene menos de 64 caracteres en hexadecimal
CE_NV_02	date_singnature tiene más de 64 caracteres en hexadecimal
CE_NV_03	date_singnature no es hexadecimal
CE_V_02	cancelation_type es Temporal
CE_V_03	cancelation_type es Final
CE_NV_04	cancelation_type no es ni Temporal ni Final
CE_V_04	reason tiene entre 2 y 100 caracteres
CE_NV_05	reason tiene menos de 2 caracteres
CE_NV_06	reason tiene más de 100 caracteres
CE_NV_07	el archivo json no existe
CE_NV_08	el archivo json tiene formato inválido

#	I/O	TECHNIQUE	FIELD	VALID/INVALID	ID TEST	DESCRIPTION
1	INPUT	EC		VALID	test_todo_ok	CE_V_01,CE_V_02,CE_V_04
2	INPUT	EC	date_signature	INVALID	test_date_signature_no_hex	CE_NV_03
3	INPUT	EC	date_signature	INVALID	test_date_signature_40_hex	CE_NV_01
4	INPUT	EC	date_signature	INVALID	test_date_signature_70_hex	CE_NV_02
4	INPUT	LIMIT CASE	date_signature	INVALID	test_date_signature_63_hex	CE_NV_01
5	INPUT	LIMIT CASE	date_signature	INVALID	test_date_signature_65_hex	CE_NV_02
6	INPUT	EC	cancelation_type	VALID	test_cancelation_final	CE_V_03
7	INPUT	EC	cancelation_type	INVALID	test_cancelation_otro	CE_NV_04
8	INPUT	EC	reason	INVALID	test_reason_0_caracter	CE_NV_05
9	INPUT	EC	reason	INVALID	test_reason_189_caracter	CE_NV_06
10	INPUT	LIMIT CASE	reason	VALID	test_reason_2_caracter	CE_V_04
11	INPUT	LIMIT CASE	reason	INVALID	test_reason_1_caracter	CE_NV_05
12	INPUT	LIMIT CASE	reason	VALID	test_reason_100_caracter	CE_V_04
13	INPUT	LIMIT CASE	reason	INVALID	test_reason_101_caracter	CE_NV_06
14	INPUT	EC	json	INVALID	test_invalid_json	CE_NV_07
15	INPUT	EC	json	INVALID	test_inavlid_format	CE_NV_08

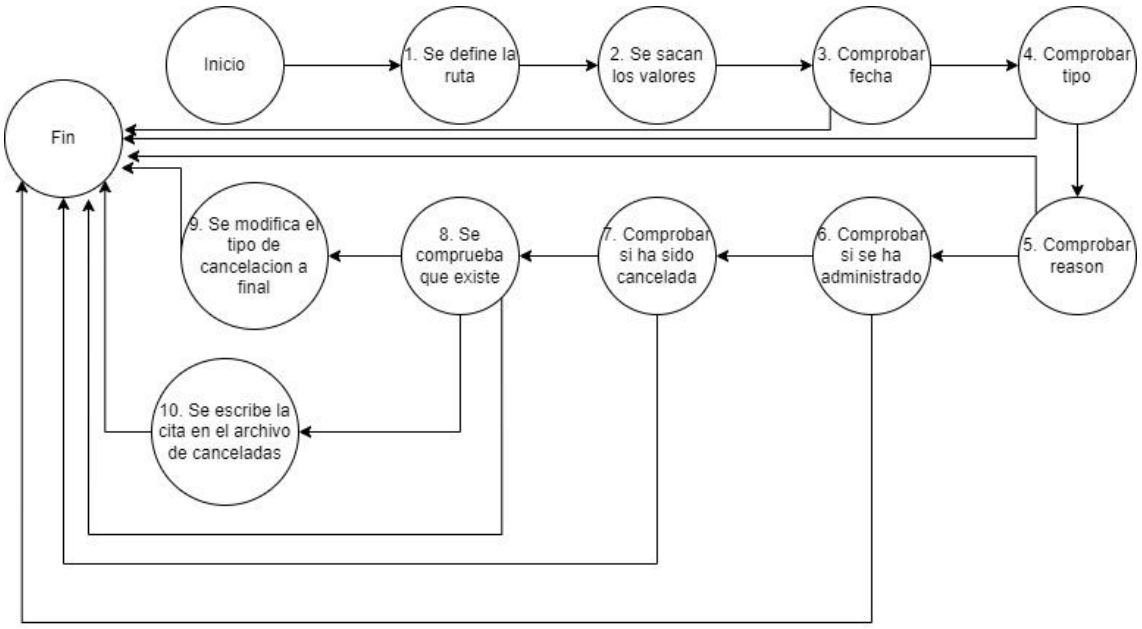
DATE_SIGNATURE	CANCELATION_TYPE	REASON
AAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Temporal	No puedo asistir ya que juega el Betis
YYYYYYYYEEEE34024932849034843YYYYYFFBJDSLKDLKNFNSLDJNFJLKSNLKDNS	Final	No puedo asistir ya que juega el Betis
aaaaaabaaafaaa2aaa7aaaaaaa3aaaaaaaa6aaaa	Final	No puedo asistir ya que juega el Betis
aaaaaabaaafaaa2aaa7aaaaaaa3aaaaaaaa6aaaaaaaaaaaaaaaddddddddddccccccccc	Final	No puedo asistir ya que juega el Betis
AA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Final	No puedo asistir ya que juega el Betis
AAAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Final	No puedo asistir ya que juega el Betis
AAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Final	No puedo asistir ya que juega el Betis
AAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Hasbullah	No puedo asistir ya que juega el Betis
AAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Temporal	
AAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Temporal	Esto es una razón para anular la cita demasiado larga que no sería aceptada por contener más caracteres c
AAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Temporal	No
AAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Temporal	E
AAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Temporal	Esta razon tiene que contener exactamente 100 caracteres ni uno mas ni uno menos. Así se comprueba e
AAA0953d112ab693b83d1ced965fcc670b558235361b9d1bd62536769a1efa3b	Temporal	Esta razon tiene que contener exactamente 100 caracteres ni uno mas ni uno menos. Así se comprueba eo

Todos estos test comprueban que los formatos sean correctos. Aquellas comprobaciones que no se ven contempladas con este tipo de pruebas se evaluarán mediante el control del flujo.

2.3 Control de flujo

Además de comprobar los posibles valores que se pueden dar a las variables pertenecientes al fichero que se introduce como parámetro a través de las clases de equivalencia y casos límite, el grupo ha decidido que es necesario realizar más pruebas. Se ha considerado necesario realizar un control de flujo del código, ya que así se puede asegurar que todas las posibles funcionalidades del código funcionan como se espera, por lo que tras estas se puede afirmar que su funcionamiento es correcto.

El grafo de flujo resultante del código implementado es el siguiente:



Tras haber realizado este diagrama sobre el funcionamiento implementado, se han establecido en el excel los posibles caminos que se pueden recorrer para llegar del inicio a fin. Las distintas posibilidades a las que se han llegado son las siguientes:

#	PATH	ID TEST	DESCRIPTION	EXPECTED RESULT	OBSERVATIONS
1	1_2_3	test_formato_de_cita_incorrecto	Formato de cita incorrecto	exception	Salta una excepción
2	1_2_3_4	test_tipo_incorrecto	Tipo incorrecto	exception	Salta una excepción
3	1_2_3_4_5	test_reason_incorrecta	Reason incorrecta	exception	Salta una excepción
4	1_2_3_4_5_6	test_se_ha_administrado_previamente	Se ha administrado previamente	exception	Salta una excepción
5	1_2_3_4_5_6_7	test_se_ha_cancelado_como_final_previamente	Se ha cancelado como final previamente	exception	Salta una excepción
6	1_2_3_4_5_6_7_8	test_no_existe_la_cita_en_el_archivo	No existe la cita en el archivo	exception	Salta una excepción
7	1_2_3_4_5_6_7_8_9	test_se_modifica_el_tipo	Se modifica el tipo	Se devuelve la cita cuya cancelación ha sido modificda de temporal a final	Se ejecuta correctamente
8	1_2_3_4_5_6_7_8_10	test_se_añade_la_cita_a_las_canceladas	Se añade la cita a las canceladas	Se devuelve la cita que ha sido cancelada	Se ejecuta correctamente

Tras haber realizado este proceso de identificación de los posibles recorridos, se han implementado los correspondientes tests que comprueban si esto funciona. Se ha podido comprobar que el funcionamiento de todos es el correcto, por lo que el código funciona correctamente.