PL_FINAL_HOSPITAL

Grado en Ingeniería Informática

Curso 2020/2021- Convocatoria Ordinaria

03210871W- García Calvo, Hugo

03208070F- Martínez Gutiérrez, David

Índice

Análisis de alto nivel

Parte 1

La primera parte de la práctica consiste en una interfaz no interactiva en la que se muestran los datos sobre las distintas salas de un hospital, así como la propia simulación de este.

Para conseguir una simulación en tiempo real hemos creado un programa concurrente haciendo uso de "Threads" (hilos) que simulan distintos elementos del hospital, como los pacientes, sanitarios, auxiliares etcétera. Este hospital dispone de 4 salas, una recepción, una sala de vacunación, una sala de observación y una sala de descanso. Las salas de vacunación y observación tienen, a su vez, 10 puestos de vacunación y 20 observación respectivamente, cada uno de ellos con un aforo máximo de 1 persona.

Al comenzar la simulación "creamos" a las personas (agentes activos) que forman parte de esta, 2 auxiliares, 10 sanitarios y 2000 pacientes. Todos estos elementos actúan individualmente y de distinta forma.

Los pacientes llegan al hospital y se añaden a una cola, en la que el orden de llegada será respetado para entrar. Entran de uno en uno y son atendidos por el primer auxiliar. La función de este auxiliar es la de asegurarse de que existe un puesto de vacunación y un puesto de observación libres para asignárselo al paciente, de esta manera nos aseguramos de que solo un paciente accede a un puesto cada vez, y también tiene la función de llevar un registro de las citas, en el que guarda si un paciente acude sin cita, o por el contrario, con que sanitario y en qué puesto se vacunará. Cada vez que el auxiliar atiende a 15 pacientes, descansa.

Una vez el paciente ha sido atendido por el auxiliar y ha recibido su puesto de vacunación y observación, se dirige a vacunarse, entrando en su puesto de vacunación correspondiente. Puede ocurrir que el paciente no tenga cita, lo cual es comprobado por el auxiliar y en tal caso el paciente abandona el hospital.

En el proceso de vacunación intervienen los pacientes, los sanitarios y el segundo auxiliar. El segundo auxiliar se encarga de suministrar vacunas a los puestos de vacunación, para que los sanitarios puedan hacer uso de ellas. Cada 20 vacunas suministradas este auxiliar descansa en la sala de descanso. Los sanitarios se encargan de vacunar al paciente, acceden al puesto de vacunación a su llegada al hospital, y esperan a que llegue un paciente. Una vez se encuentran los dos en el puesto, el sanitario comprueba si quedan vacunas disponibles y en tal caso se la administra al paciente, si no, espera a que haya una. Cada 15 vacunaciones, el sanitario se retira a la sala de descanso.

Una vez vacunado, el paciente se dirige a la sala de observación y accede a su puesto correspondiente. Ahí espera durante un tiempo, para comprobar si tiene alguna reacción por la vacuna. Un 5% de los vacunados tendrá una reacción. En caso de tener una reacción, el paciente será atendido por un sanitario, y tras esto abandonará el hospital. En caso de no tener una reacción, abandonará el hospital directamente.

El sanitario que debe acudir a atender al paciente, lo hará solo después de haber descansado en la sala de descanso. Por lo tanto, tras descansar comprobará si hay alguna reacción en la sala de vacunación y en caso de haberlo, acudirá al puesto de observación donde se encuentra y lo tratará, tras esto el sanitario accede a un puesto de vacunación para seguir vacunando.

Para transmitir esa información a nuestra interfaz gráfica relacionamos las salas directamente con esta, para que así pueda tener acceso en todo momento a los datos actualizados de las salas y poder mostrarlos en sus respectivas posiciones.

Parte 2

Para la segunda parte de la práctica, se añade la posibilidad de ver los datos del hospital en una interfaz ajena a el hospital, y esta vez sí interactiva.
Anteriormente la interfaz tenía acceso a la información del hospital directamente, ya que contenía las salas del hospital. Ahora esa forma de compartir los datos se hace por medio de una conexión TCP, en la cual se intercambia un mensaje con los datos del hospital, en un formato en concreto que es leído y mostrado posteriormente en la interfaz.
Para conseguir esto, hemos hecho uso del modelo MVC (Modelo Vista Controlador). Debido a esto tenemos un servidor, que aloja los datos del hospital para que puedan ser accedidos remotamente, un modelo cliente, que se conecta con el servidor para recibir los datos, una vista (interfaz) que muestra los datos y un controlador, el cual se encarga de comunicar el modelo cliente con la vista para así transferir la información de uno a otro.
El modelo cliente solicita la información del hospital al servidor cada segundo, por lo que la información de la interfaz se actualizará cada segundo.
Como hemos dicho en esta ocasión la interfaz es interactiva, tiene un botón por cada puesto de vacunación que permite cerrar el puesto para su limpieza. Si se pulsa cuando ningún paciente ha entrado al puesto, el puesto se cierra durante un tiempo y posteriormente el paciente que tenía asignado ese puesto se vacuna. Si se pulsa cuando el sanitario está vacunando a un paciente, una vez se termine de vacunar, el puesto será cerrado.
La comunicación entre el servidor y el modelo cliente se realiza mediante TCP, para solicitar la información, el cliente envía un mensaje al servidor. Este mensaje puede ser vacío, lo que significa que quiere recibir la información del hospital, o puede ser un número del 1 al 10, indicando el puesto que se quiere cerrar. En caso de recibir un puesto a cerrar, el servidor debe tanto cerrar el puesto como enviar la información del hospital.

Diseño general del sistema y de las herramientas de sincronización utilizados

Para la lógica del programa hemos hecho uso de las siguientes clases:

- Auxiliar1
- Auxiliar2
- Log
- Paciente
- Hospital
- PuestoVacunacion
- PuestoObservacion
- Recepción
- SalaDescanso
- SalaObservacion
- SalaVacunacion
- Sanitario.

Para la primera interfaz la clase:

- Interfaz_1

Para la segunda interfaz y la conectividad cliente-servidor:

- Interfaz_2
- Servidor
- ModeloCliente
- Controlador
- Cliente.

En cuanto a las herramientas de sincronización utilizadas para la sincronización de los hilos, vamos a dividirlas por las tareas en orden que se realizan en la ejecución del programa.

En la **clase Hospital** (main) instanciamos 2 semáforos: puestosVacunacion y puestosObservacion.

puestosVacunacion-> Inicializado a 0. Será usado posteriormente para representar el número de puestos de vacunación a los que pueden acceder los pacientes.

puestosObservacion-> Inicializado a 20. Será usado posteriormente para representar el número de puestos de observación a los que pueden acceder los pacientes.

Para el funcionamiento de la **clase recepción** hacemos uso de un semáforo propio de la clase, pacienteEnRecepcion, inicializado a 0. En esta clase existen métodos que permiten la interacción de pacientes con el auxiliar1 como comprobarPaciente y registrarse. Para sincronizarlos usamos los semáforos anteriores de la siguiente forma: cuando un paciente se quiere registrar realiza un release() del semáforo pacienteEnRecepcion, que pasa a estar a 1, indicándole al auxiliar que hay un paciente en la recepción. El auxiliar, para comprobar al paciente permancerá esperando haciendo un acquire() del semáforo pacienteEnRecepcion hasta que un paciente se registre como mencionado anteriormente. Para finalizar el registro, el paciente espera a que el paciente termine la gestión en un await de un CyclicBarrier de 2 parties el cual es un atributo del paciente.

El paciente hace el await después de llamar a registrarse() y el auxiliar hace el await cuando termina la gestión

Respecto a la sala de vacunación encontramos un CopyOnWriteArrayList de puestos de vacunacion llamado puestos. Hemos decidido usar esta colección concurrente debido a que necesitábamos algo con la funcionalidad de un ArrayList y que fuera Thread-safe debido a que a este accederían varios hilos a la vez a distintos elementos. También hemos creado 2 LinkedBlockingQueue, puestosLibresPaciente y puestosLibresSanitario. Decidimos usar colas concurrentes ya que necesitábamos un array con la funcionalidad de una cola FIFO y que fuera thread-safe. Por último creamos 2 semáforos, vacunasDisponibles inicializado a 0 y puestosVacunacionPaciente, este último pasado como parámetro desde Hospital (main) donde se instancia la sala de vacunación.

En esta clase encontramos los métodos que relacionan los pacientes, sanitarios y auxiliar2 a la hora de la vacunación. El semáforo puestosVacunacion es usado para llevar un recuento de los puestos de vacunación disponibles en cada momento. Por ello el método getPuestoVacunacionPaciente, la cual debe conseguir un puesto de vacunación para asignarlo a un paciente, tiene al comenzar un acquire(), el cual asegura que siempre que coja un puesto, lo haga con certeza y no exista la posibilidad de que se lo quiten y también asegura que no intente coger un puesto cuando no hay disponibles.

El semáforo vacunasDisponibles es usado para llevar un recuento de las vacunas disponibles. Para ello cada vez que el auxiliar añade una vacuna, realiza un release del semáforo, aumentando en 1 su valor. Por el contrario, cuando un sanitario vacuna, realizara un acquire, disminuyendo en 1 el valor del semáforo, ya que habría gastado una vacuna.

El CopyOnWriteArrayList puestos y las LinkedBlockingQueues se encargan automáticamente de sincronizar los distintos accesos.

En concreto, puestos es un CopyOnWriteArrayList de “Puestos”, los cuales guardan distintos elementos de sincronización como semáforos y exchangers. Estos son usados como una pareja para el funcionamiento del sanitario a la hora de vacunar. El sanitario espera recibir una acción (vacunar o abandonar puesto) mediante un acquire() del semáforo y posteriormente espera a recibir exactamente la tarea que tiene que hacer mediante un exchange().

El paciente hace uso de estos mismos elementos para indicar que ha llegado para ser vacunado, release(), y para indicar que se debe realizar la acción de vacunar, Exchange(). De la misma forma el método cerrarPuesto hace uso de estos dos elementos para indicar que la acción no es vacunar, si no que es abandonar el puesto.

Por ultimo las funciones salirVacunacionPaciente y salirVacunacionSanitario hacen uso del semáforo puestosVacunacionPaciente, realizando un release() que representaría la salida de la sala, aumentando en 1 unidad el valor de este para que posteriormente puedan ser adquiridos al entrar de nuevo.

En cuanto a la sala de observación encontramos de nuevo un CopyOnWriteArrayList llamado puestos, pero esta vez está formado por puestos de observación, el semáforo puestosObservacionPaciente pasado como parámetro y dos LinkedBlockingQueue de puestos de observación, colaObservacionPacientes y colaReaccionesPacientes.

Como en la sala de vacunación, aquí la función getPuestoObservacionPaciente hace uso del semáforo de puestosObservacionPaciente mediante un acquire() para asegurarse de que hay puestos disponibles a la hora de coger uno o en su defecto quedarse esperando hasta que los haya. También se realiza un poll() de la cola de pacientes, en el que se elimina el primer elemento de la cola, pero esto es gestionado automáticamente al ser thread-safe.

Para entrar a la sala de observación los pacientes hacen uso del método entrarSalaObservacion en el que se cambia la variable pacienteEnObservacion propia de cada puesto de observación mediante el CopyOnWriteArraylist puestos.

Para salir los pacientes realizan un release del semáforo puestosObservacion para dejar un hueco libre que podrá ser adquirido de nuevo por el auxiliar 1 para asignar un puesto a un paciente.

Si el paciente tiene una reacción realiza la función esperarAtencionReaccion donde se añade a la cola de reacciones y realiza un await de la CyclicBarrier de tamaño 2 del puesto de observación en el que está, con el objetivo de esperar al sanitario, el cual hará el await restante para poder ser tratado. Por último, realiza otro await más esperando a que el sanitario termine de tratarlo y haga su await para así poder marcharse.

En la sala de descanso encontramos un CopyOnWriteArraylist salaDescansoId en el que se guardan los ids de las personas que están descansando, al ser thread safe, no hemos tenido que hacer ninguna modificación más.

Clases principales que intervienen con su descripción (atributos y métodos)

Primero analizaremos los hilos: Auxiliar1, Auxiliar2, Paciente y Sanitario

Después veremos las clases compartidas: PuestoVacunacion, PuestoObservacion, Recepcion, SalaVacunacion, SalaObservacion y SalaDescanso.

En cuanto a la gestión de la estructura Cliente-Servidor creemos interesante tener en cuenta la clase Servidor y su interfaz y las clases correspondientes al modelo MVC del cliente.

Auxiliar1 y Auxiliar2

El auxiliar1 simplemente se encarga de comprobar la cita de los pacientes y el auxiliar2 se encarga de producir vacunas. No tienen ningún método relevante.

Paciente

Atributos:

- Un String para identificar al paciente
- La recepción, la sala de vacunación y la sala de observación
- Tiene asignado un PuestoVacunacion y un PuestoObservacion
- El log para registrar su paso por el hospital
- Un booleano que representa si tiene cita o no
- Un CyclicBarrier para esperar a que el auxiliar compruebe su cita

Métodos:

- CrearID(): Para formar el String del id de la forma PXXXX.
- run(): donde se describe su paso por el hospital (Entrar a la cola del hospital - Registro – Vacunación – Observación - Posible reacción – Atención de esa posible reacción – Salir del Hospital)
- getters y setters

Sanitario

Atributos:

- Un String para identificar al sanitario
- La sala de vacunación, la sala de observación y la sala de descanso
- Tiene asignado un puesto de vacunación.
- Un contador de pacientes vacunados, ya que el sanitario descansa cada 15
- El log para registrar lo que hace

Métodos:

- run(): un bucle del que se sale cuando ya no hay pacientes en el hospital recogiendo una InterruptedException. En este bucle los sanitarios vacunan hasta que lleguen a 15 pacientes vacunado o se les notifique que deben cerrar su puesto de vacunación. Una vez haya pasado alguna de estas dos cosas, el sanitario se irá a descansar. Después de descansar comprobará si hay algún paciente con reacción a la vacuna en la sala de observación, si la hay le atenderá y volverá a un puesto de vacunación libre, y si no, se irá directamente al puesto de vacunación

- getters y setters

PuestoVacunacion

Atributos:

- Un String para el sanitario que haya en el puesto y otro para el paciente.
- Un CyclicBarrier para que el paciente indique que está listo para vacunarse
- Un id, que es un entero para identificar el puesto
- Por último tenemos tres atributos que se utilizan para gestionar la acción que realiza el sanitario en el puesto de vacunación
    - Un semáforo semaforoGestionBoton que empieza en el que el sanitario hace acquire() para esperar al release() del paciente o del método de cerrar puesto.
    - Un exchanger para que, después del release(), se le indique indique de donde viene ese release() con un mensaje diferente dependiendo de si el release() lo ha hecho el paciente o el método de cerrar puesto.
    - Un booleano para indicar si el puerto está abierto.

Métodos:

- Los métodos de esta clase son todos **getters** y **setters**.

PuestoObservacion

Atributos:

- Un String para el sanitario que haya en el puesto atendiendo la reacción y otro para el paciente.
- Un CyclicBarrier para gestionar el proceso de atender una reacción (que no empiece hasta que estén paciente y sanitario en el puesto, y que el sanitario haya acabado para que el paciente se pueda ir)
- Un id, que es un entero para identificar el puesto

Métodos:

- Los métodos de esta clase son todos **getters** y **setters**.

Recepcion

Atributos:

- La recepción necesita tener la sala de vacunación y la de observación como atributos
- Un string con el id del paciente que está en la recepción en ese momento, el cual se usa para mostrarlo en la interfaz.
- Un semáforo para no comprobar un paciente que aún no ha llegado a la recepción
- Un ConcurrentLinkedQueue que será una cola de pacientes.

Métodos:

- **comprobarPaciente():** este método lo llamará el auxiliar para comprobar la cita del primer paciente que esté en la cola de pacientes. Se hará una acquire() al semáforo para esperar al paciente y una vez dentro el auxiliar tardara entre 0.5s y 1s en realizar la comprobación. Hay un 1% de posibilidades de que el paciente no tenga cita, en cuyo caso el paciente saldrá directamente del hospital al poner la variable booleana tieneCita del paciente a false. Si tiene cita el auxiliar obtendrá los puestos de vacunación y de

observación a los que tiene que ir el paciente e informará al paciente de los mismos haciendo uso de los setters correspondientes. Esta función devolverá un mensaje el cual será diferente dependiendo de si el paciente tiene cita o no.
- **registrarse():** este método lo llamará el paciente para registrarse en la recepción. Simplemente se añadirá a la cola y hará un release() del semáforo para avisar al auxiliar de que ya está en la recepción.
- **Getters** de la cola y del id del paciente que está dentro.

SalaDescanso

Atributos:

- Un CopyOnWriteArrayList que será la lista de IDs que hay dentro de la sala de descanso

Métodos:

- Todos los métodos son para los descansos de los auxiliares y sanitarios. Cuando una persona descansa se le añade a la lista, y cuando termina, se elimina de la lista

SalaObservacion

Atributos:

- Un CopyOnWriteArrayList que es una lista de puestos de observación
- Dos LinkedBlockingQueue, una para los puestos libres y otra para los puestos en los que haya un paciente con reacción
- Un Semaforo puestosObservacionPaciente que gestiona el número de puestos a los que puede entrar un paciente

Métodos

- **getPuestoObservacionPaciente():** Se hace un acquire() a puestosObservacionPaciente para esperar a que haya un puesto libre en caso de que no los hubiera. Después retorna el primero puesto libre de la cola
- **atenderReaccion():** recibe un sanitario que es el que va a atender la reacción. Si la cola de reacciones está vacía no atiende nada y se va a la sala de vacunación. Si no está vacía vemos cual es el paciente que lleva mas tiempo sin ser atendido, que es el primero de la cola de reacciones. Después incluimos al sanitario en el puesto y hacemos un await en la barrera del puesto para indicar al paciente que se va a empezar a atender la reacción. El sanitario tardará entre 2 y 5 segundos en atender la reacción. Una vez atendida se hace otro await en la barrera para indicar que se ha finalizado la atención. Por ultimo se quita el sanitario del puesto.
- **entrarSalaObservacion():** recibe el numero del puesto y el paciente y lo introduce en su puesto.
- **esperarAtencionReaccion():** a esta función lo llama el paciente al que le da una reacción y recibe el número del puesto en que está el paciente. Se añade el puesto del paciente a la cola de reacciones, esperamos a que venga el sanitario a atender al paciente con un await, guardamos el id del sanitario que va a atender al paciente y después esperamos a que el sanitario termine de atender la reacción con otro await. Retornamos el id del sanitario que ha atendido la reacción

- **salirSalaObservacionPaciente():** recibe le numero del puesto y elimina al paciente de ese puesto, además de añadir el puesto a la cola de puestos libres y liberar un permit del semáforo puestosObservacionPaciente
- **getters** de los sanitarios atendiendo reacción y de los pacientes en observación

SalaVacunacion

Atributos:

- Un CopyOnWriteArrayList que es una lista de puestos de vacunación
- Dos LinkedBlockingQueue, una para los puestos libres para los pacientes y otra para los puestos para los sanitarios
- Un semáforo para indicar el numero de vacunas que hay disponibles
- Un semáforo puestosVacunacionPaciente que gestiona el número de puestos a los que puede entrar un paciente

Métodos:

- **getPuestoVacunacionPaciente():** Se hace un acquire() a puestosVacunacionPaciente para esperar a que haya un puesto libre en caso de que no los hubiera. Después retorna el primero puesto libre de la cola
- **entrarPuestoVacunacionSanitario():** recibe un sanitario. Guardamos el primer puesto libre. Se añade el sanitario en ese puesto y se añade a los puestos libres para los pacientes, ya que los pacientes no pueden entrar a un puesto hasta que haya un sanitario dentro. Se marca el puesto como abierto y se libera un permit del semáforo del numero de puestos libres.
- **vacunar():** este método recibe un numero de puesto y lo llama el sanitario que va a vacunar. Primero libera un permit del semáforo semaforoGestionBoton para esperar a que le notifiquen para hacer algo. Después recibe la acción en un exchanger. Si recibe un 1 vacuna y si recibe un 2, cierra el puesto y se va a descansar. Para vacunar primero coge una vacuna haciendo un acquire al semáforo, en caso de haber un paciente esperando vacunamos y en el caso contrario esperamos en el await de la barrera del puesto. La vacunación tardará entre 3 y 5 segundos y después se hará otro await para indicar que ya se ha terminado. Se retornara true o false al sanitario dependiendo de si ha vacunado o no.
- **esperarVacuna():** recibe un numero de puesto y un paciente que es el que va a esperar la vacuna. Primero le indicamos al sanitario que le ha llegado algo con un acquire al semáforo semaforoGestionBoton, después usamos el exchanger y le mandamos un 1 para indicarle que tiene que vacunar, luego metemos al paciente en el puesto e indicamos al sanitario que el paciente ya está en el puesto con un await de la barrera del puesto. Por último, guardamos el sanitario que vacuna y el paciente espera en otro await. Retornamos el id del sanitario.
- **salirVacunacionPaciente():** recibe un numero de puesto y elimina al paciente en ese puesto, además de añadir el puesto a los puestos libres y libera un permit del semáforo puestosVacunacionPaciente
- **salirVacunacionSanitario():** recibe un numero de puesto y elimina al sanitario en ese puesto, además de añadir el puesto a los puestos libres y adquiere un permit del semáforo puestosVacunacionPaciente

- **auxiliar2PrepararVacuna():** el auxiliar 2 llama a esta función para liberar un permit en el semáforo de vacunas disponibles. Tarda entre 0.5 y 1s
- **cerrarPuesto():** recibe una lista de puestos a cerrar. La recorre y hace un acquire al semáforo semaforoGestionBoton de cada puesto para notificar al sanitario. Le manda un 2 con el exchanger para indicarle que tiene que cerrar el puesto y marca el puesto como cerrado.
- **Getters** de los ID de pacientes y sanitarios, de las vacunas disponibles y de los puestos disponibles

Interfaz_2

Atributos:

- La interfaz del cliente recibe el controlador como parámetro en el constructor
- El resto de atributos son los componentes de la interfaz

Métodos

- **setControlador():** setter del controlador
- **hacerVisible():** hace visible la ventana y sus componentes
- **actualizar():** recibe la información del hospital, que se la manda el modelo a través del controlador, y la recorre para actualizar la interfaz
- **inicializarEventos()**: se asigna un actionCommand a cada botón y se les asigna el controlador como ActionListenner
- **habilitarBotones() y deshabilitarBotones():** para habilitar y deshabilitar los botones cuando corresponda

Servidor

Atributos:

- Un Controlador para comunicarse con la interfaz
- Un entero para representar el puerto para crear el servidor
- Un serverSocket con el que se creará el servidor
- Un socket con el que se creara la conexión
- Un ObjectInputStream y un objectOutputStream para recibir y enviar información
- Una salaObservacion para tener acceso a los datos de la sala de observacion
- Una salaVacunacion para tener acceso a los datos de la sala de vacunacion
- Una saladescanso para tener acceso a los datos de la sala de descanso
- Una recepción para tener acceso a los datos de la recepción

Métodos:

- **abrirPuerto():** Instancia el serverSocket con el puerto correspondiente
- **esperarCliente():** Espera la conexión del cliente
- **crearFlujos():** Crea los canales de comunicación de la conexión TCP con sockets

Controlador

Atributos:

- Una I_Interfaz para mandar la información
- Un ModeloCliente donde consultar la información

Métodos:

- **Arrancar():** Hace visible la interfaz e inicia la conexión.
- **ActualizarInterfaz():** Llama a la función que actualiza la interfaz de la I_Interfaz
- **HabilitarBotones():** Habilita la funcionalidad de los botones que recibe como parámetro
- **ActionPerformed():** Cuando un botón es pulsado, llama a la función del modelo que cierra un puesto

ModeloCliente

Atributos:

- Un controlador
- Un entero que representa el puerto
- Un String que representa el host
- Un socket para establecer la conexión
- Un objectInputStream y objectOutputStream para establecer los canales de comunicación
- Un CopyOnWriteArrayList de enteros para representar los puestos a cerrar
- Un ArrayList de ArrayList de objetos en el que se guarda la información del hospital

Métodos:

- **conectarseServidor():** Establece la conexión con el servidor
- **crearFlujos():** Establece los canales de comunicación con socketsTCP

Interfaz_1

Atributos:

- Una salaObservacion para tener acceso a los datos de la sala de observacion
- Una salaVacunacion para tener acceso a los datos de la sala de vacunacion
- Una saladescanso para tener acceso a los datos de la sala de descanso
- Una recepción para tener acceso a los datos de la recepción

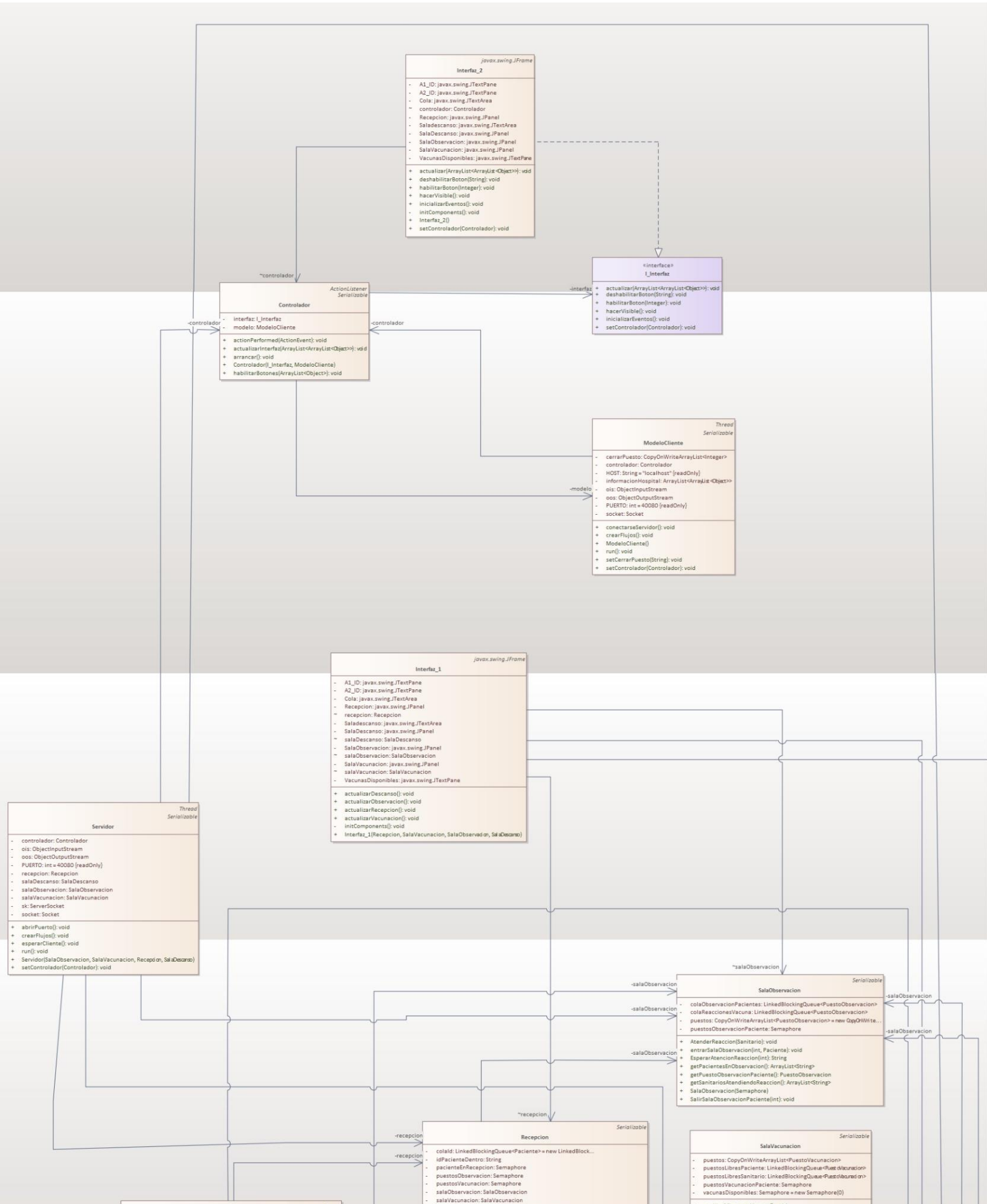
Métodos:

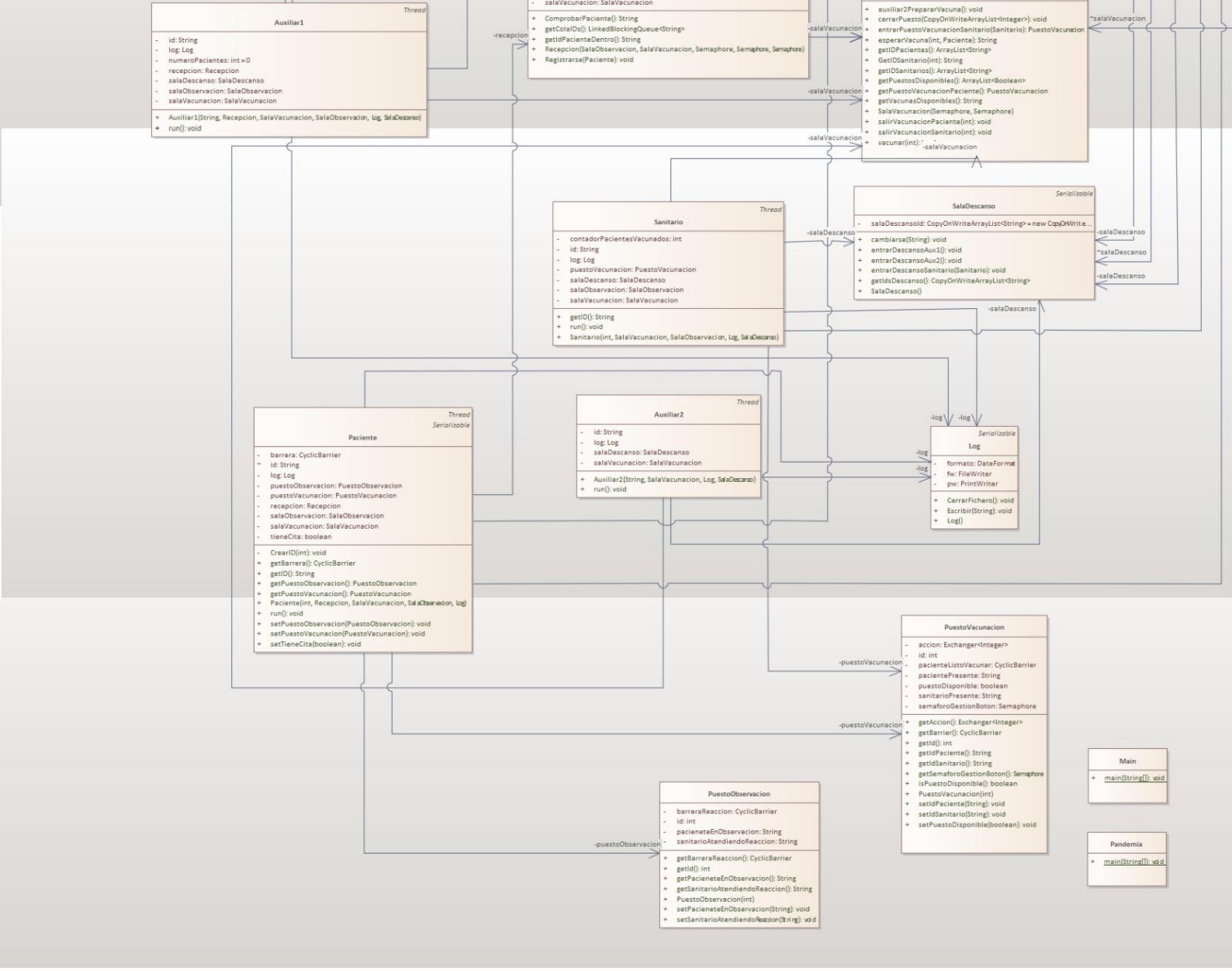
- **actualizarDescanso():** Actualiza la información de la sala de descanso
- **actualizarVacunacion():** Actualiza la información de la sala de vacunación
- **actualizarObservacion():** Actualiza la información de la sala de observación
- **actualizarRecepcion():** Actualiza la información de la sala de Recepcion

Log

En esta clase se encuentra la funcionalidad para escribir en un fichero que se encuentra en la carpeta del proyecto.

Diagrama de clases




Thread
Auxiliar1
- id: String
- log: Log
- numeroPacientes: int = 0
- recepcion: Recepcion
- salaDescanso: SalaDescanso
- salaObservacion: SalaObservacion
- salaVacunacion: SalaVacunacion
+ Auxiliar1(String, Recepcion, SalaVacunacion, SalaObservacion, Log, SalaDescanso)
+ run(): void
-recepcion
+ ComprobarPaciente(): String
+ getColaIDs(): LinkedBlockingQueue<String>
+ getIdPacienteDentro(): String
+ Recepcion(SalaObservacion, SalaVacunacion, Semaphore, Semaphore, Semaphore)
+ Registrarse(Paciente): void
-salaVacunacion
+ auxiliar2PrepararVacuna(): void
+ cerrarPuesto(CopyOnWriteArrayList<Integer>): void
+ entrarPuestoVacunacionSanitario(Sanitario): PuestoVacunacion
+ esperarVacuna(int, Paciente): String
+ getIDPacientes(): ArrayList<String>
+ GetIDSanitario(int): String
+ getIDSanitarios(): ArrayList<String>
+ getPuestosDisponibles(): ArrayList<Boolean>
+ getPuestoVacunacionPaciente(): PuestoVacunacion
+ getVacunasDisponibles(): String
+ SalaVacunacion(Semaphore, Semaphore)
+ salirVacunacionPaciente(int): void
+ salirVacunacionSanitario(int): void
+ vacunar(int): void
Thread
Sanitario
- contadorPacientesVacunados: int
- id: String
- log: Log
- puestoVacunacion: PuestoVacunacion
- salaDescanso: SalaDescanso
- salaObservacion: SalaObservacion
- salaVacunacion: SalaVacunacion
+ getID(): String
+ run(): void
+ Sanitario(int, SalaVacunacion, SalaObservacion, Log, SalaDescanso)
-salaDescanso
Serializable
SalaDescanso
- salaDescansoId: CopyOnWriteArrayList<String> = new CopyOnWrite...
+ cambiarse(String): void
+ entrarDescansoAux1(): void
+ entrarDescansoAux2(): void
+ entrarDescansoSanitario(Sanitario): void
+ getIdsDescanso(): CopyOnWriteArrayList<String>
+ SalaDescanso()
Thread
Serializable
Paciente
- barrera: CyclicBarrier
~ id: String
- log: Log
- puestoObservacion: PuestoObservacion
- puestoVacunacion: PuestoVacunacion
- recepcion: Recepcion
- salaObservacion: SalaObservacion
- salaVacunacion: SalaVacunacion
- tieneCita: boolean
- CrearID(int): void
+ getBarrera(): CyclicBarrier
+ getID(): String
+ getPuestoObservacion(): PuestoObservacion
+ getPuestoVacunacion(): PuestoVacunacion
+ Paciente(int, Recepcion, SalaVacunacion, SalaObservacion, Log)
+ run(): void
+ setPuestoObservacion(PuestoObservacion): void
+ setPuestoVacunacion(PuestoVacunacion): void
+ setTieneCita(boolean): void
Thread
Auxiliar2
- id: String
- log: Log
- salaDescanso: SalaDescanso
- salaVacunacion: SalaVacunacion
+ Auxiliar2(String, SalaVacunacion, Log, SalaDescanso)
+ run(): void
-log
Serializable
Log
- formato: DateFormat
- fw: FileWriter
- pw: PrintWriter
+ CerrarFichero(): void
+ Escribir(String): void
+ Log()
-puestoVacunacion
PuestoVacunacion
- accion: Exchanger<Integer>
- id: int
- pacienteListoVacunar: CyclicBarrier
- pacientePresente: String
- puestoDisponible: boolean
- sanitarioPresente: String
- semaforoGestionBoton: Semaphore
+ getAccion(): Exchanger<Integer>
+ getBarrier(): CyclicBarrier
+ getId(): int
+ getIdPaciente(): String
+ getIdSanitario(): String
+ getSemaforoGestionBoton(): Semaphore
+ isPuestoDisponible(): boolean
+ PuestoVacunacion(int)
+ setIdPaciente(String): void
+ setIdSanitario(String): void
+ setPuestoDisponible(boolean): void
-puestoObservacion
PuestoObservacion
- barreraReaccion: CyclicBarrier
- id: int
- pacienteteEnObservacion: String
- sanitarioAtendiendoReaccion: String
+ getBarreraReaccion(): CyclicBarrier
+ getId(): int
+ getPacienteteEnObservacion(): String
+ getSanitarioAtendiendoReaccion(): String
+ PuestoObservacion(int)
+ setPacienteteEnObservacion(String): void
+ setSanitarioAtendiendoReaccion(String): void
Main
+ main(String[]): void
Pandemia
+ main(String[]): void

Anexo

Clase Auxiliar1

```
package hilos;


import recursos_compartidos.Log;
import recursos_compartidos.Recepcion;
import recursos_compartidos.SalaDescanso;
import recursos_compartidos.SalaObservacion;
import recursos_compartidos.SalaVacunacion;
import java.util.Random;
import java.util.logging.Level;
import java.util.logging.Logger;


public class Auxiliar1 extends Thread{
    private String id;
    private int numeroPacientes=0;
    private Recepcion recepcion;
    private SalaVacunacion salaVacunacion;
    private SalaObservacion salaObservacion;
    private SalaDescanso salaDescanso;
    private Log log;
    public Auxiliar1(String id, Recepcion recepcion, SalaVacunacion
salaVacunacion, SalaObservacion salaObservacion, Log
log,SalaDescanso salaDescanso) {
        this.id = id;
        this.recepcion = recepcion;
        this.salaVacunacion = salaVacunacion;
        this.salaObservacion = salaObservacion;
        this.salaDescanso=salaDescanso;
        this.log = log;
    }
    /**
     * El Auxiliar 1 entrará en un bucle infinito en el cual irá
comprobando si
     * la cita de los pacientes es correcta con el metodo
ComprobarPaciente() de
     * la clase recepción. Esta función devolverá el mensaje que
depués imprimirá
     * el auxiliar por pantalla. El auxiliar descansará cada 10
pacientes comprobados
     */
    public void run() {
        try {
        while(true) {
            if (numeroPacientes < 10) {
                String mensaje = recepcion.comprobarPaciente();
                System.out.println(mensaje);
                log.escribir(mensaje);
                numeroPacientes++; //aumentamos, cuando llegue a 10
descansa
            } else {

                numeroPacientes = 0;
```

```
                log.escribir("Auxiliar A1 comienza su descanso");
                salaDescanso.entrarDescansoAux1();
            }
        }
        }catch(InterruptedException e) {
            log.escribir("Auxiliar A1 finaliza su jornada
laboral");
        }
    }


}
```

Clase Auxiliar2

```
package hilos;


import recursos_compartidos.Log;
import recursos_compartidos.SalaDescanso;
import recursos_compartidos.SalaVacunacion;
import static java.lang.Thread.sleep;
import java.util.Random;
import java.util.logging.Level;
import java.util.logging.Logger;

/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
 *
 * @author David
 */
public class Auxiliar2 extends Thread{
    private String id;
    private SalaVacunacion salaVacunacion;
    private SalaDescanso salaDescanso;
    private Log log;

    public Auxiliar2(String id, SalaVacunacion salaVacunacion, Log
log,SalaDescanso salaDescanso) {
        this.id = id;
        this.salaVacunacion = salaVacunacion;
        this.salaDescanso=salaDescanso;
        this.log = log;
    }
    /**
     * El auxiliar 2 prepara vacunas en un bucle infinito.
     * Descansa cada 20 vacunas preparadas
     */
    public void run() {
        try {
            int contadorVacunas = 0;
```

```
            while (true) {
                salaVacunacion.auxiliar2PrepararVacuna();
                contadorVacunas++;
                if (contadorVacunas == 20) {
                    contadorVacunas = 0;
                    log.escribir("Auxiliar A2 comienza su
descanso");
                    salaDescanso.entrarDescansoAux2();
                }
            }
        } catch (InterruptedException e) {
            log.escribir("Auxiliar A2 finaliza su jornada
laboral");
        }
    }


}
```

Clase Paciente

```
package hilos;


import recursos_compartidos.Log;
import recursos_compartidos.PuestoObservacion;
import recursos_compartidos.PuestoVacunacion;
import recursos_compartidos.Recepcion;
import recursos_compartidos.SalaObservacion;
import recursos_compartidos.SalaVacunacion;
import java.io.Serializable;
import java.util.Random;
import java.util.concurrent.BrokenBarrierException;
import java.util.concurrent.CyclicBarrier;
import java.util.logging.Level;
import java.util.logging.Logger;

/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
 *
 * @author David
 */
public class Paciente extends Thread implements Serializable {
    String id;
    private Recepcion recepcion;
    private SalaVacunacion salaVacunacion;
    private SalaObservacion salaObservacion;
    private PuestoVacunacion puestoVacunacion;
    private PuestoObservacion puestoObservacion;
    private Log log;
    private boolean tieneCita;
    private CyclicBarrier barrera;
```

```
    public Paciente(int id, Recepcion recepcion, SalaVacunacion
salaVacunacion, SalaObservacion salaObservacion, Log log) {
        CrearID(id);
        this.recepcion = recepcion;
        this.salaVacunacion = salaVacunacion;
        this.salaObservacion = salaObservacion;
        this.log = log;
        this.barrera = new CyclicBarrier(2);
    }
    /**
     * En este método añadimos los ceros necearios entre la 'P' y
el id pasado
     * como parametro en el bucle for de la clase 'Main' para crear
el ID del
     * paciente.
     * @param id
     */
    private void CrearID(int id) {
        this.id = "P";
        int aux = id;
        while(aux < 1000) {
            this.id += '0';
            aux *= 10;
        }
        this.id += id;
    }

    public void run() {

        recepcion.registrarse(this);
        try {
            //El paciente espera a que el auxiliar le atienda para
poder ir a vacunarse
            barrera.await();
        } catch (InterruptedException ex) {

Logger.getLogger(Paciente.class.getName()).log(Level.SEVERE, null,
ex);
        } catch (BrokenBarrierException ex) {

Logger.getLogger(Paciente.class.getName()).log(Level.SEVERE, null,
ex);
        }

        if (tieneCita) {

            //Entra a la sala de vacunacion y espera a la vacuna
            String sanitario = null;
            try {
                sanitario =
salaVacunacion.esperarVacuna(puestoVacunacion.getId(), this);
            } catch (InterruptedException ex) {

Logger.getLogger(Paciente.class.getName()).log(Level.SEVERE, null,
ex);
            }
            //Una vez vacunado, el paciente sale de la sala de
vacunacion
```

```
salaVacunacion.salirVacunacionPaciente(puestoVacunacion.getId());

            //Entra a la sala de observacion y espera por una
posible reaccion

salaObservacion.entrarSalaObservacion(puestoObservacion.getId(),
this);
            try {
                log.escribir("Paciente " + id + " vacunado en el
puesto " + puestoVacunacion.getId() + " por " + sanitario);
                sleep(10000);
            } catch (InterruptedException ex) {

Logger.getLogger(Paciente.class.getName()).log(Level.SEVERE, null,
ex);
            }

            //Tiene un 5% de tener una reaccion
            if ((new Random()).nextInt(100) < 5) {
                try {
                    //Si la tiene espera a que un sanitario le
atienda
                    log.escribir("Paciente " + id + " sufre una
reaccion y espera sanitario. Puesto: " +
puestoObservacion.getId());
                    String idSanitario =
salaObservacion.esperarAtencionReaccion(puestoObservacion.getId());
                    log.escribir("Paciente " + id + " ha sido
atendido en el puesto " + puestoObservacion.getId() + " por " +
idSanitario);
                } catch (InterruptedException ex) {

Logger.getLogger(Paciente.class.getName()).log(Level.SEVERE, null,
ex);
                }
            }

salaObservacion.salirSalaObservacionPaciente(puestoObservacion.getI
d());
            //Sale del hospital
            log.escribir("Paciente " + id + " sale del hospital");
        }
    }

    public void setTieneCita(boolean b) {
        this.tieneCita = b;
    }

    public String getID() {
        return id;
    }

    public void setPuestoVacunacion(PuestoVacunacion puesto) {
        this.puestoVacunacion = puesto;
    }
    public PuestoVacunacion getPuestoVacunacion() {
        return puestoVacunacion;
```

```
    }
    public void setPuestoObservacion(PuestoObservacion puesto) {
        this.puestoObservacion = puesto;
    }
    public PuestoObservacion getPuestoObservacion() {
        return puestoObservacion;
    }
    public CyclicBarrier getBarrera() {
        return barrera;
    }
}
```

Clase Sanitario

```
package hilos;


import recursos_compartidos.Log;
import recursos_compartidos.PuestoVacunacion;
import recursos_compartidos.SalaDescanso;
import recursos_compartidos.SalaObservacion;
import recursos_compartidos.SalaVacunacion;
import java.util.Random;
import java.util.logging.Level;
import java.util.logging.Logger;

/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author David
 */
public class Sanitario extends Thread {

    private String id;

    private SalaVacunacion salaVacunacion;
    private SalaObservacion salaObservacion;
    private SalaDescanso salaDescanso;
    private PuestoVacunacion puestoVacunacion;
    private int contadorPacientesVacunados;
    private Log log;

    public Sanitario(int id,SalaVacunacion salaVacunacion,
SalaObservacion salaObservacion, Log log,SalaDescanso salaDescanso)
{
        if (id < 10) {
            this.id = "S0" + id;
        } else {
            this.id = "S" + id;
        }
        this.salaDescanso=salaDescanso;
        this.salaVacunacion = salaVacunacion;
```

```
        this.salaObservacion = salaObservacion;
        this.log = log;
    }

    public void run() {
        try {
        salaDescanso.cambiarse(id);

        //Despues de cambiarse entra a su puesto de vacunacion
        puestoVacunacion =
salaVacunacion.entrarPuestoVacunacionSanitario(this);
        log.escribir("Sanitario " + id + " entrando a puesto " +
puestoVacunacion.getId());
        while (true) {
            //Puede pasar que el sanitario haya vacunado o que haya
cerrado su puesto
            boolean haVacunado =
salaVacunacion.vacunar(puestoVacunacion.getId());
            //Si ha vacunado se suma 1 al contador de pacientes
vacunados
            if (haVacunado) {
                contadorPacientesVacunados++;
            }
            //Si ha vacunado a 15 pacientes o no ha vacunado, es
decir, se ha cerrado el puesto
            //sale del puesto y se va a descansar
            if (contadorPacientesVacunados == 15 || !haVacunado) {

                if (contadorPacientesVacunados == 15) {
                    contadorPacientesVacunados = 0;
                }


salaVacunacion.salirVacunacionSanitario(puestoVacunacion.getId());
                log.escribir("Sanitario " + id + " se va a
descansar");
                salaDescanso.entrarDescansoSanitario(this);

                //Despues de descansar atiende una reaccion en la
sala de observaciÃ³n si la hay
                salaObservacion.atenderReaccion(this);
                //Por Ãºltimo vuelve a un puesto de vacunacion
libre
                puestoVacunacion =
salaVacunacion.entrarPuestoVacunacionSanitario(this);
                log.escribir("Sanitario " + id + " entrando a
puesto " + puestoVacunacion.getId());
            }
        }
        }catch(InterruptedException e) {
            log.escribir("Sanitario " + id + " finaliza su jornada
laboral");
        }
    }

    public String getID() {
        return id;
    }
```

```
}
```

Clase Interfaz 1

```
package interfaz_servidor;

import recursos_compartidos.Recepcion;
import recursos_compartidos.SalaDescanso;
import recursos_compartidos.SalaObservacion;
import recursos_compartidos.SalaVacunacion;
import java.util.ArrayList;
import java.util.concurrent.CopyOnWriteArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
/**
 * En esta interfaz se va actualizando la informacion de todas las
 * salas del hospital recibidas como parametro en el constructor
 * @author David
 */
public class Interfaz_1 extends javax.swing.JFrame {
    private Recepcion recepcion;
    private SalaVacunacion salaVacunacion;
    private SalaObservacion salaObservacion;
    private SalaDescanso salaDescanso;
    public Interfaz_1(Recepcion recepcion,SalaVacunacion
salaVacunacion,SalaObservacion salaObservacion,SalaDescanso
salaDescanso){
        initComponents();
        this.salaDescanso=salaDescanso;
        this.salaObservacion=salaObservacion;
        this.salaVacunacion=salaVacunacion;
        this.recepcion=recepcion;
    }
    public void actualizarDescanso(){

Saladescanso.setText(salaDescanso.getIdsDescanso().toString());

        if(salaDescanso.getIdsDescanso().contains("A1"))
            A1_ID.setText("");
        else
            A1_ID.setText("A1");
        if(salaDescanso.getIdsDescanso().contains("A2"))
            A2_ID.setText("");
        else
            A2_ID.setText("A2");
    }
    public void actualizarVacunacion(){
```

```
        ArrayList<String> idSanitarios=
salaVacunacion.getIDSanitarios();
        ArrayList<String> idPacientes=
salaVacunacion.getIDPacientes();

VacunasDisponibles.setText(salaVacunacion.getVacunasDisponibles());
        for(int i=0;i<10;i++){
            switch(i){
                case 0:
                    Puesto1.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 1:
                    Puesto2.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 2:
                    Puesto3.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 3:
                    Puesto4.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 4:
                    Puesto5.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 5:
                    Puesto6.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 6:
                    Puesto7.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 7:
                    Puesto8.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 8:
                    Puesto9.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 9:
                    Puesto10.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;

            }
        }
    }
    public void actualizarObservacion(){
        ArrayList<String> idSanitarios=
salaObservacion.getSanitariosAtendiendoReaccion();
        ArrayList<String> idPacientes=
salaObservacion.getPacientesEnObservacion();
```

```
        for(int i=0;i<20;i++){
            switch(i){
                case 0:
                    Obs1.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 1:
                    Obs2.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 2:
                    Obs3.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 3:
                    Obs4.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 4:
                    Obs5.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 5:
                    Obs6.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 6:
                    Obs7.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 7:
                    Obs8.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 8:
                    Obs9.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 9:
                    Obs10.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 10:
                    Obs11.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 11:
                    Obs12.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 12:
                    Obs13.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 13:
                    Obs14.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
```

```
                    break;
                case 14:
                    Obs15.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 15:
                    Obs16.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 16:
                    Obs17.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 17:
                    Obs18.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 18:
                    Obs19.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 19:
                    Obs20.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;

            }
        }
    }
    public void actualizarRecepcion(){
        Cola.setText(recepcion.getColaIDs().toString());
        PacienteID.setText(recepcion.getIdPacienteDentro());
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        Recepcion = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        Cola = new javax.swing.JTextArea();
        jScrollPane2 = new javax.swing.JScrollPane();
        A1_ID = new javax.swing.JTextPane();
        jScrollPane3 = new javax.swing.JScrollPane();
        PacienteID = new javax.swing.JTextPane();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        SalaVacunacion = new javax.swing.JPanel();
        jLabel6 = new javax.swing.JLabel();
        jScrollPane6 = new javax.swing.JScrollPane();
        Puesto1 = new javax.swing.JTextPane();
        jLabel7 = new javax.swing.JLabel();
        jScrollPane7 = new javax.swing.JScrollPane();
        Puesto2 = new javax.swing.JTextPane();
        jLabel8 = new javax.swing.JLabel();
```

```
jScrollPane8 = new javax.swing.JScrollPane();
Puesto3 = new javax.swing.JTextPane();
jLabel9 = new javax.swing.JLabel();
jScrollPane9 = new javax.swing.JScrollPane();
Puesto4 = new javax.swing.JTextPane();
jLabel10 = new javax.swing.JLabel();
jScrollPane10 = new javax.swing.JScrollPane();
Puesto5 = new javax.swing.JTextPane();
jLabel11 = new javax.swing.JLabel();
jScrollPane11 = new javax.swing.JScrollPane();
Puesto6 = new javax.swing.JTextPane();
jLabel12 = new javax.swing.JLabel();
jScrollPane12 = new javax.swing.JScrollPane();
Puesto7 = new javax.swing.JTextPane();
jScrollPane13 = new javax.swing.JScrollPane();
Puesto8 = new javax.swing.JTextPane();
jLabel13 = new javax.swing.JLabel();
jScrollPane14 = new javax.swing.JScrollPane();
Puesto9 = new javax.swing.JTextPane();
jLabel14 = new javax.swing.JLabel();
jScrollPane15 = new javax.swing.JScrollPane();
Puesto10 = new javax.swing.JTextPane();
jLabel15 = new javax.swing.JLabel();
jScrollPane16 = new javax.swing.JScrollPane();
A2_ID = new javax.swing.JTextPane();
jScrollPane17 = new javax.swing.JScrollPane();
VacunasDisponibles = new javax.swing.JTextPane();
jLabel16 = new javax.swing.JLabel();
jLabel17 = new javax.swing.JLabel();
jLabel18 = new javax.swing.JLabel();
SalaObservacion = new javax.swing.JPanel();
jLabel19 = new javax.swing.JLabel();
jLabel20 = new javax.swing.JLabel();
jLabel21 = new javax.swing.JLabel();
jLabel22 = new javax.swing.JLabel();
jLabel23 = new javax.swing.JLabel();
jLabel24 = new javax.swing.JLabel();
jLabel25 = new javax.swing.JLabel();
jLabel26 = new javax.swing.JLabel();
jLabel27 = new javax.swing.JLabel();
jLabel28 = new javax.swing.JLabel();
jLabel29 = new javax.swing.JLabel();
jLabel30 = new javax.swing.JLabel();
jLabel31 = new javax.swing.JLabel();
jLabel32 = new javax.swing.JLabel();
jLabel33 = new javax.swing.JLabel();
jLabel34 = new javax.swing.JLabel();
jLabel35 = new javax.swing.JLabel();
jLabel36 = new javax.swing.JLabel();
jLabel37 = new javax.swing.JLabel();
jLabel38 = new javax.swing.JLabel();
jScrollPane38 = new javax.swing.JScrollPane();
Obs1 = new javax.swing.JTextPane();
jScrollPane39 = new javax.swing.JScrollPane();
Obs2 = new javax.swing.JTextPane();
jScrollPane40 = new javax.swing.JScrollPane();
Obs3 = new javax.swing.JTextPane();
jScrollPane41 = new javax.swing.JScrollPane();
```

```
        Obs4 = new javax.swing.JTextPane();
        jScrollPane42 = new javax.swing.JScrollPane();
        Obs5 = new javax.swing.JTextPane();
        jScrollPane43 = new javax.swing.JScrollPane();
        Obs6 = new javax.swing.JTextPane();
        jScrollPane44 = new javax.swing.JScrollPane();
        Obs7 = new javax.swing.JTextPane();
        jScrollPane45 = new javax.swing.JScrollPane();
        Obs8 = new javax.swing.JTextPane();
        jScrollPane46 = new javax.swing.JScrollPane();
        Obs9 = new javax.swing.JTextPane();
        jScrollPane47 = new javax.swing.JScrollPane();
        Obs10 = new javax.swing.JTextPane();
        jScrollPane48 = new javax.swing.JScrollPane();
        Obs11 = new javax.swing.JTextPane();
        jScrollPane49 = new javax.swing.JScrollPane();
        Obs12 = new javax.swing.JTextPane();
        jScrollPane50 = new javax.swing.JScrollPane();
        Obs13 = new javax.swing.JTextPane();
        jScrollPane51 = new javax.swing.JScrollPane();
        Obs14 = new javax.swing.JTextPane();
        jScrollPane52 = new javax.swing.JScrollPane();
        Obs15 = new javax.swing.JTextPane();
        jScrollPane53 = new javax.swing.JScrollPane();
        Obs16 = new javax.swing.JTextPane();
        jScrollPane54 = new javax.swing.JScrollPane();
        Obs17 = new javax.swing.JTextPane();
        jScrollPane55 = new javax.swing.JScrollPane();
        Obs20 = new javax.swing.JTextPane();
        jScrollPane56 = new javax.swing.JScrollPane();
        Obs18 = new javax.swing.JTextPane();
        jScrollPane57 = new javax.swing.JScrollPane();
        Obs19 = new javax.swing.JTextPane();
        jLabel39 = new javax.swing.JLabel();
        SalaDescanso = new javax.swing.JPanel();
        jLabel5 = new javax.swing.JLabel();
        jScrollPane4 = new javax.swing.JScrollPane();
        Saladescanso = new javax.swing.JTextArea();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE)
;

        Recepcion.setBackground(new java.awt.Color(220, 220, 220));

Recepcion.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 13)); //
NOI18N
        jLabel1.setText("RECEPCION");

        jLabel2.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel2.setText("Cola de espera");

        Cola.setEditable(false);
        Cola.setColumns(20);
```

```
        Cola.setRows(5);
        jScrollPane1.setViewportView(Cola);

        A1_ID.setEditable(false);
        jScrollPane2.setViewportView(A1_ID);

        PacienteID.setEditable(false);
        jScrollPane3.setViewportView(PacienteID);

        jLabel3.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel3.setText("Paciente");

        jLabel4.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel4.setText("Auxiliar");

        javax.swing.GroupLayout RecepcionLayout = new
javax.swing.GroupLayout(Recepcion);
        Recepcion.setLayout(RecepcionLayout);
        RecepcionLayout.setHorizontalGroup(

RecepcionLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
            .addGroup(RecepcionLayout.createSequentialGroup()

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(RecepcionLayout.createSequentialGroup()
                        .addGap(245, 245, 245)

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.TRAILING)
                            .addComponent(jLabel2)
                            .addComponent(jLabel1)))

.addGroup(RecepcionLayout.createSequentialGroup()
                        .addGap(38, 38, 38)
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 500,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(RecepcionLayout.createSequentialGroup()
                        .addGap(49, 49, 49)
                        .addComponent(jLabel3)
                        .addGap(108, 108, 108)

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                            .addComponent(jLabel4,
javax.swing.GroupLayout.PREFERRED_SIZE, 121,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 91,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap(62, Short.MAX_VALUE))
```

```
.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                .addGroup(RecepcionLayout.createSequentialGroup()
                    .addGap(48, 48, 48)
                    .addComponent(jScrollPane3,
javax.swing.GroupLayout.PREFERRED_SIZE, 91,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap(461, Short.MAX_VALUE)))
        );
        RecepcionLayout.setVerticalGroup(

RecepcionLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
            .addGroup(RecepcionLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)
                .addGap(18, 18, 18)
                .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, 14, Short.MAX_VALUE)

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                    .addComponent(jLabel3)
                    .addComponent(jLabel4))
                .addGap(2, 2, 2)
                .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(33, 33, 33))

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
RecepcionLayout.createSequentialGroup()
                    .addContainerGap(191, Short.MAX_VALUE)
                    .addComponent(jScrollPane3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(33, 33, 33)))
        );

        SalaVacunacion.setBackground(new java.awt.Color(220, 220,
220));
```

```
SalaVacunacion.setBorder(javax.swing.BorderFactory.createLineBorder
(new java.awt.Color(0, 0, 0)));
        SalaVacunacion.setPreferredSize(new
java.awt.Dimension(1600, 320));

        jLabel6.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel6.setText("Puesto 1");

        Puesto1.setEditable(false);
        jScrollPane6.setViewportView(Puesto1);

        jLabel7.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel7.setText("Puesto 2");

        Puesto2.setEditable(false);
        jScrollPane7.setViewportView(Puesto2);

        jLabel8.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel8.setText("Puesto 3");

        Puesto3.setEditable(false);
        jScrollPane8.setViewportView(Puesto3);

        jLabel9.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel9.setText("Puesto 4");

        Puesto4.setEditable(false);
        jScrollPane9.setViewportView(Puesto4);

        jLabel10.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel10.setText("Puesto 5");

        Puesto5.setEditable(false);
        jScrollPane10.setViewportView(Puesto5);

        jLabel11.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel11.setText("Puesto 6");

        Puesto6.setEditable(false);
        jScrollPane11.setViewportView(Puesto6);

        jLabel12.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel12.setText("Puesto 7");

        Puesto7.setEditable(false);
        jScrollPane12.setViewportView(Puesto7);

        Puesto8.setEditable(false);
        jScrollPane13.setViewportView(Puesto8);
```

```
        jLabel13.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel13.setText("Puesto 8");

        Puesto9.setEditable(false);
        jScrollPane14.setViewportView(Puesto9);

        jLabel14.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel14.setText("Puesto 9");

        Puesto10.setEditable(false);
        jScrollPane15.setViewportView(Puesto10);

        jLabel15.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel15.setText("Puesto 10");

        A2_ID.setEditable(false);
        jScrollPane16.setViewportView(A2_ID);

        VacunasDisponibles.setEditable(false);
        jScrollPane17.setViewportView(VacunasDisponibles);

        jLabel16.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel16.setText("Auxiliar");

        jLabel17.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel17.setText("Vacunas disponibles");

        jLabel18.setFont(new java.awt.Font("Tahoma", 1, 13)); //
NOI18N
        jLabel18.setText("SALA DE VACUNACION");

        javax.swing.GroupLayout SalaVacunacionLayout = new
javax.swing.GroupLayout(SalaVacunacion);
        SalaVacunacion.setLayout(SalaVacunacionLayout);
        SalaVacunacionLayout.setHorizontalGroup(

SalaVacunacionLayout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
            .addGroup(SalaVacunacionLayout.createSequentialGroup()
                .addGap(48, 48, 48)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)

.addGroup(SalaVacunacionLayout.createSequentialGroup()

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel6)
                            .addComponent(jScrollPane6,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)
```

```
.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel7)
                            .addComponent(jScrollPane7,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel8)
                            .addComponent(jScrollPane8,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel9)
                            .addComponent(jScrollPane9,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel10)
                            .addComponent(jScrollPane10,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                                .addGap(309, 309, 309)
                                .addComponent(jScrollPane16,
javax.swing.GroupLayout.DEFAULT_SIZE, 75, Short.MAX_VALUE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
SalaVacunacionLayout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jLabel16)
                                .addGap(18, 18, 18)))
                        .addGap(251, 251, 251))

.addGroup(SalaVacunacionLayout.createSequentialGroup()

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel11)
                            .addComponent(jScrollPane11,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)
```

```
.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel12)
                            .addComponent(jScrollPane12,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel13)
                            .addComponent(jScrollPane13,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel14)
                            .addComponent(jScrollPane14,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jLabel15)
                            .addComponent(jScrollPane15,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                                .addGap(309, 309, 309)
                                .addComponent(jScrollPane17)
                                .addGap(251, 251, 251))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
SalaVacunacionLayout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jLabel17)
                                .addGap(231, 231, 231))))))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
SalaVacunacionLayout.createSequentialGroup()

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .addComponent(jLabel18)
                .addGap(453, 453, 453))
        );
        SalaVacunacionLayout.setVerticalGroup(
```

```
SalaVacunacionLayout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
            .addGroup(SalaVacunacionLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel18)
                .addGap(33, 33, 33)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.TRAILING)

.addGroup(SalaVacunacionLayout.createSequentialGroup()

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.BASELINE)
                            .addComponent(jLabel10)
                            .addComponent(jLabel16))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jScrollPane10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jScrollPane16,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel9)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane9,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel8)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel7)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
```

```
                        .addComponent(jScrollPane7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane6,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(47, 47, 47)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.TRAILING)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel11)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane11,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel12)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane12,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel13)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane13,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel14)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
```

```
                        .addComponent(jScrollPane14,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
SalaVacunacionLayout.createSequentialGroup()

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.BASELINE)
                            .addComponent(jLabel15)
                            .addComponent(jLabel17))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jScrollPane15,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jScrollPane17,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap(89, Short.MAX_VALUE))
        );

        SalaObservacion.setBackground(new java.awt.Color(220, 220,
220));

SalaObservacion.setBorder(javax.swing.BorderFactory.createLineBorde
r(new java.awt.Color(0, 0, 0)));

        jLabel19.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel19.setText("Puesto 1");

        jLabel20.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel20.setText("Puesto 2");

        jLabel21.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel21.setText("Puesto 3");

        jLabel22.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel22.setText("Puesto 4");

        jLabel23.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel23.setText("Puesto 5");

        jLabel24.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel24.setText("Puesto 6");
```

```
        jLabel25.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel25.setText("Puesto 7");

        jLabel26.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel26.setText("Puesto 8");

        jLabel27.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel27.setText("Puesto 9");

        jLabel28.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel28.setText("Puesto 10");

        jLabel29.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel29.setText("Puesto 11");

        jLabel30.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel30.setText("Puesto 12");

        jLabel31.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel31.setText("Puesto 13");

        jLabel32.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel32.setText("Puesto 14");

        jLabel33.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel33.setText("Puesto 15");

        jLabel34.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel34.setText("Puesto 16");

        jLabel35.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel35.setText("Puesto 17");

        jLabel36.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel36.setText("Puesto 18");

        jLabel37.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel37.setText("Puesto 19");

        jLabel38.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel38.setText("Puesto 20");

        Obs1.setEditable(false);
```

```
jScrollPane38.setViewportView(Obs1);

Obs2.setEditable(false);
jScrollPane39.setViewportView(Obs2);

Obs3.setEditable(false);
jScrollPane40.setViewportView(Obs3);

Obs4.setEditable(false);
jScrollPane41.setViewportView(Obs4);

Obs5.setEditable(false);
jScrollPane42.setViewportView(Obs5);

Obs6.setEditable(false);
jScrollPane43.setViewportView(Obs6);

Obs7.setEditable(false);
jScrollPane44.setViewportView(Obs7);

Obs8.setEditable(false);
jScrollPane45.setViewportView(Obs8);

Obs9.setEditable(false);
jScrollPane46.setViewportView(Obs9);

Obs10.setEditable(false);
jScrollPane47.setViewportView(Obs10);

Obs11.setEditable(false);
jScrollPane48.setViewportView(Obs11);

Obs12.setEditable(false);
jScrollPane49.setViewportView(Obs12);

Obs13.setEditable(false);
jScrollPane50.setViewportView(Obs13);

Obs14.setEditable(false);
jScrollPane51.setViewportView(Obs14);

Obs15.setEditable(false);
jScrollPane52.setViewportView(Obs15);

Obs16.setEditable(false);
jScrollPane53.setViewportView(Obs16);

Obs17.setEditable(false);
jScrollPane54.setViewportView(Obs17);

Obs20.setEditable(false);
jScrollPane55.setViewportView(Obs20);

Obs18.setEditable(false);
jScrollPane56.setViewportView(Obs18);

Obs19.setEditable(false);
jScrollPane57.setViewportView(Obs19);
```

```
        jLabel39.setFont(new java.awt.Font("Tahoma", 1, 13)); //
NOI18N
        jLabel39.setText("SALA DE OBSERVACION");

        javax.swing.GroupLayout SalaObservacionLayout = new
javax.swing.GroupLayout(SalaObservacion);
        SalaObservacion.setLayout(SalaObservacionLayout);
        SalaObservacionLayout.setHorizontalGroup(

SalaObservacionLayout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
            .addGroup(SalaObservacionLayout.createSequentialGroup()

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createSequentialGroup()
                        .addGap(950, 950, 950)
                        .addComponent(jScrollPane57,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaObservacionLayout.createSequentialGroup()
                        .addGap(32, 32, 32)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createSequentialGroup()

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addComponent(jScrollPane48,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addComponent(jScrollPane38,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                    .addComponent(jLabel29))
                                .addGap(32, 32, 32)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createSequentialGroup()

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addComponent(jScrollPane39,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addComponent(jScrollPane49,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                        .addGap(32, 32, 32)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addComponent(jScrollPane40,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addComponent(jScrollPane50,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addGroup(SalaObservacionLayout.createSequentialGroup()
                                        .addComponent(jLabel30)
                                        .addGap(60, 60, 60)
                                        .addComponent(jLabel31)))
                                .addGap(32, 32, 32)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createSequentialGroup()
                                        .addComponent(jLabel32)
                                        .addGap(60, 60, 60)
                                        .addComponent(jLabel33)
                                        .addGap(60, 60, 60)
                                        .addComponent(jLabel34)
                                        .addGap(60, 60, 60)
                                        .addComponent(jLabel35)
                                        .addGap(60, 60, 60)
                                        .addComponent(jLabel36)
                                        .addGap(60, 60, 60)
                                        .addComponent(jLabel37)
                                        .addGap(60, 60, 60)
                                        .addComponent(jLabel38))

.addGroup(SalaObservacionLayout.createSequentialGroup()

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addComponent(jScrollPane41,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addComponent(jScrollPane51,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                        .addGap(32, 32, 32)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)
```

```
.addComponent(jScrollPane42,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addComponent(jScrollPane52,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                        .addGap(32, 32, 32)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addComponent(jScrollPane43,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addComponent(jScrollPane53,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                        .addGap(32, 32, 32)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addComponent(jScrollPane44,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addComponent(jScrollPane54,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                        .addGap(32, 32, 32)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING, false)

.addGroup(SalaObservacionLayout.createSequentialGroup()

.addComponent(jScrollPane45,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                                .addGap(32, 32, 32)

.addComponent(jScrollPane46,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                                .addGap(32, 32, 32)

.addComponent(jScrollPane47,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaObservacionLayout.createSequentialGroup()

.addComponent(jScrollPane56,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addComponent(jScrollPane55,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                                .addGap(1, 1,
1))))))

.addGroup(SalaObservacionLayout.createSequentialGroup()
                                .addComponent(jLabel19)
                                .addGap(67, 67, 67)
                                .addComponent(jLabel20)
                                .addGap(67, 67, 67)
                                .addComponent(jLabel21)
                                .addGap(67, 67, 67)
                                .addComponent(jLabel22)
                                .addGap(67, 67, 67)
                                .addComponent(jLabel23)
                                .addGap(67, 67, 67)
                                .addComponent(jLabel24)
                                .addGap(67, 67, 67)
                                .addComponent(jLabel25)
                                .addGap(67, 67, 67)
                                .addComponent(jLabel26)
                                .addGap(67, 67, 67)
                                .addComponent(jLabel27)
                                .addGap(67, 67, 67)
                                .addComponent(jLabel28)))))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
SalaObservacionLayout.createSequentialGroup()
                .addGap(0, 0, Short.MAX_VALUE)
                .addComponent(jLabel39)
                .addGap(493, 493, 493))
        );
        SalaObservacionLayout.setVerticalGroup(

SalaObservacionLayout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
            .addGroup(SalaObservacionLayout.createSequentialGroup()
                .addGap(9, 9, 9)
                .addComponent(jLabel39)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                    .addComponent(jLabel19)
                    .addComponent(jLabel20)
                    .addComponent(jLabel21)
                    .addComponent(jLabel22)
                    .addComponent(jLabel23)
                    .addComponent(jLabel24)
```

```
                    .addComponent(jLabel25)
                    .addComponent(jLabel26)
                    .addComponent(jLabel27)
                    .addComponent(jLabel28))
                .addGap(18, 18, 18)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                    .addComponent(jScrollPane38,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane39,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane40,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane41,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane42,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane43,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane44,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane45,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane46,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane47,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)
                    .addComponent(jLabel29,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel30,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel31,
javax.swing.GroupLayout.Alignment.LEADING)
```

```
                    .addComponent(jLabel32,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel33,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel34,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel35,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel36,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel37,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel38,
javax.swing.GroupLayout.Alignment.LEADING))
                .addGap(18, 18, 18)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                    .addComponent(jScrollPane48,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane49,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane50,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane51,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane52,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane53,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane54,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane57,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane55,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane56,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
                .addContainerGap(54, Short.MAX_VALUE))
        );

        SalaDescanso.setBackground(new java.awt.Color(220, 220,
220));

SalaDescanso.setBorder(javax.swing.BorderFactory.createLineBorder(n
ew java.awt.Color(0, 0, 0)));

        jLabel5.setFont(new java.awt.Font("Tahoma", 1, 13)); //
NOI18N
        jLabel5.setText("SALA DE DESCANSO");

        Saladescanso.setEditable(false);
        Saladescanso.setColumns(20);
        Saladescanso.setRows(5);
        jScrollPane4.setViewportView(Saladescanso);

        javax.swing.GroupLayout SalaDescansoLayout = new
javax.swing.GroupLayout(SalaDescanso);
        SalaDescanso.setLayout(SalaDescansoLayout);
        SalaDescansoLayout.setHorizontalGroup(

SalaDescansoLayout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
            .addGroup(SalaDescansoLayout.createSequentialGroup()

.addGroup(SalaDescansoLayout.createParallelGroup(javax.swing.GroupL
ayout.Alignment.LEADING)

.addGroup(SalaDescansoLayout.createSequentialGroup()
                        .addGap(196, 196, 196)
                        .addComponent(jLabel5))

.addGroup(SalaDescansoLayout.createSequentialGroup()
                        .addGap(24, 24, 24)
                        .addComponent(jScrollPane4,
javax.swing.GroupLayout.PREFERRED_SIZE, 502,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        SalaDescansoLayout.setVerticalGroup(

SalaDescansoLayout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
            .addGroup(SalaDescansoLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel5)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane4,
javax.swing.GroupLayout.PREFERRED_SIZE, 182,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(19, Short.MAX_VALUE))
        );
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
            .addGroup(layout.createSequentialGroup()
                .addGap(10, 10, 10)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignm
ent.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(Recepcion,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(SalaDescanso,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                    .addComponent(SalaObservacion,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(SalaVacunacion,
javax.swing.GroupLayout.DEFAULT_SIZE, 1172, Short.MAX_VALUE))
                .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
            .addGroup(layout.createSequentialGroup()
                .addGap(11, 11, 11)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignm
ent.LEADING, false)
                    .addComponent(Recepcion,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(SalaDescanso,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addGap(9, 9, 9)
                .addComponent(SalaVacunacion,
javax.swing.GroupLayout.PREFERRED_SIZE, 292,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                .addComponent(SalaObservacion,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );

        pack();
```

```
}// </editor-fold>//GEN-END:initComponents



// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JTextPane A1_ID;
private javax.swing.JTextPane A2_ID;
private javax.swing.JTextArea Cola;
private javax.swing.JTextPane Obs1;
private javax.swing.JTextPane Obs10;
private javax.swing.JTextPane Obs11;
private javax.swing.JTextPane Obs12;
private javax.swing.JTextPane Obs13;
private javax.swing.JTextPane Obs14;
private javax.swing.JTextPane Obs15;
private javax.swing.JTextPane Obs16;
private javax.swing.JTextPane Obs17;
private javax.swing.JTextPane Obs18;
private javax.swing.JTextPane Obs19;
private javax.swing.JTextPane Obs2;
private javax.swing.JTextPane Obs20;
private javax.swing.JTextPane Obs3;
private javax.swing.JTextPane Obs4;
private javax.swing.JTextPane Obs5;
private javax.swing.JTextPane Obs6;
private javax.swing.JTextPane Obs7;
private javax.swing.JTextPane Obs8;
private javax.swing.JTextPane Obs9;
private javax.swing.JTextPane PacienteID;
private javax.swing.JTextPane Puesto1;
private javax.swing.JTextPane Puesto10;
private javax.swing.JTextPane Puesto2;
private javax.swing.JTextPane Puesto3;
private javax.swing.JTextPane Puesto4;
private javax.swing.JTextPane Puesto5;
private javax.swing.JTextPane Puesto6;
private javax.swing.JTextPane Puesto7;
private javax.swing.JTextPane Puesto8;
private javax.swing.JTextPane Puesto9;
private javax.swing.JPanel Recepcion;
private javax.swing.JPanel SalaDescanso;
private javax.swing.JPanel SalaObservacion;
private javax.swing.JPanel SalaVacunacion;
private javax.swing.JTextArea Saladescanso;
private javax.swing.JTextPane VacunasDisponibles;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel10;
private javax.swing.JLabel jLabel11;
private javax.swing.JLabel jLabel12;
private javax.swing.JLabel jLabel13;
private javax.swing.JLabel jLabel14;
private javax.swing.JLabel jLabel15;
private javax.swing.JLabel jLabel16;
private javax.swing.JLabel jLabel17;
private javax.swing.JLabel jLabel18;
private javax.swing.JLabel jLabel19;
```

```
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel20;
private javax.swing.JLabel jLabel21;
private javax.swing.JLabel jLabel22;
private javax.swing.JLabel jLabel23;
private javax.swing.JLabel jLabel24;
private javax.swing.JLabel jLabel25;
private javax.swing.JLabel jLabel26;
private javax.swing.JLabel jLabel27;
private javax.swing.JLabel jLabel28;
private javax.swing.JLabel jLabel29;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel30;
private javax.swing.JLabel jLabel31;
private javax.swing.JLabel jLabel32;
private javax.swing.JLabel jLabel33;
private javax.swing.JLabel jLabel34;
private javax.swing.JLabel jLabel35;
private javax.swing.JLabel jLabel36;
private javax.swing.JLabel jLabel37;
private javax.swing.JLabel jLabel38;
private javax.swing.JLabel jLabel39;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JLabel jLabel7;
private javax.swing.JLabel jLabel8;
private javax.swing.JLabel jLabel9;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JScrollPane jScrollPane10;
private javax.swing.JScrollPane jScrollPane11;
private javax.swing.JScrollPane jScrollPane12;
private javax.swing.JScrollPane jScrollPane13;
private javax.swing.JScrollPane jScrollPane14;
private javax.swing.JScrollPane jScrollPane15;
private javax.swing.JScrollPane jScrollPane16;
private javax.swing.JScrollPane jScrollPane17;
private javax.swing.JScrollPane jScrollPane2;
private javax.swing.JScrollPane jScrollPane3;
private javax.swing.JScrollPane jScrollPane38;
private javax.swing.JScrollPane jScrollPane39;
private javax.swing.JScrollPane jScrollPane4;
private javax.swing.JScrollPane jScrollPane40;
private javax.swing.JScrollPane jScrollPane41;
private javax.swing.JScrollPane jScrollPane42;
private javax.swing.JScrollPane jScrollPane43;
private javax.swing.JScrollPane jScrollPane44;
private javax.swing.JScrollPane jScrollPane45;
private javax.swing.JScrollPane jScrollPane46;
private javax.swing.JScrollPane jScrollPane47;
private javax.swing.JScrollPane jScrollPane48;
private javax.swing.JScrollPane jScrollPane49;
private javax.swing.JScrollPane jScrollPane50;
private javax.swing.JScrollPane jScrollPane51;
private javax.swing.JScrollPane jScrollPane52;
private javax.swing.JScrollPane jScrollPane53;
private javax.swing.JScrollPane jScrollPane54;
private javax.swing.JScrollPane jScrollPane55;
```

```
    private javax.swing.JScrollPane jScrollPane56;
    private javax.swing.JScrollPane jScrollPane57;
    private javax.swing.JScrollPane jScrollPane6;
    private javax.swing.JScrollPane jScrollPane7;
    private javax.swing.JScrollPane jScrollPane8;
    private javax.swing.JScrollPane jScrollPane9;
    // End of variables declaration//GEN-END:variables
}
```

Clase Cliente

```
package main_cliente;

import mvc_cliente.Controlador;
import mvc_cliente.Interfaz_2;
import mvc_cliente.ModeloCliente;


/**
 * En la clase cliente se instancian los elementos del modelo MVC:
 * El modelo, la vista (interfaz) y el controlador
 * Una vez instanciados y relacionados entre sí,
 * se arranca el cliente a través del controlador.
 * @author David
 */
public class Cliente {

    public static void main(String[] args) {
        Interfaz_2 interfaz=new Interfaz_2();
        ModeloCliente modeloCliente=new ModeloCliente();
        Controlador controlador =new
Controlador(interfaz,modeloCliente);

        interfaz.setControlador(controlador);
        modeloCliente.setControlador(controlador);

        controlador.arrancar();

    }

}
```

Clase Hospital

```
package main_servidor;



import recursos_compartidos.Log;
import recursos_compartidos.Recepcion;
import recursos_compartidos.SalaDescanso;
import recursos_compartidos.SalaObservacion;
import recursos_compartidos.SalaVacunacion;
import hilos.Sanitario;
import hilos.Paciente;
import hilos.Auxiliar2;
import hilos.Auxiliar1;
import interfaz_servidor.Interfaz_1;
import servidor.Servidor;
import static java.lang.Thread.sleep;
import java.util.Random;

import java.util.concurrent.LinkedBlockingQueue;
import java.util.concurrent.Semaphore;
import java.util.logging.Level;
import java.util.logging.Logger;


public class Hospital {


    public static void main(String[] args) {
        Semaphore puestosVacunacion = new Semaphore(0, true);
        Semaphore puestosObservacion = new Semaphore(20, true);

        //Se crea las cuatro salas del hospital
        SalaDescanso salaDescanso= new SalaDescanso();
        SalaObservacion salaObservacion = new
SalaObservacion(puestosObservacion);
        SalaVacunacion salaVacunacion = new
SalaVacunacion(puestosVacunacion);
        Recepcion recepcion = new Recepcion(salaObservacion,
salaVacunacion);

        //Se crea el servidor
        Servidor servidor=new Servidor(salaObservacion,
salaVacunacion, recepcion, salaDescanso);
        servidor.start();

        //Se crea el log donde se va a escribir la informacion del
hospital
        Log log = new Log();

        //Creamos la interfaz
        Interfaz_1 interfaz = new
Interfaz_1(recepcion,salaVacunacion,salaObservacion,salaDescanso);
        Thread t = new Thread(new Runnable() {
            public void run() {
                interfaz.setVisible(true);
```

```
                while(true){

                    interfaz.actualizarDescanso();
                    interfaz.actualizarObservacion();
                    interfaz.actualizarRecepcion();
                    interfaz.actualizarVacunacion();
                }
            }
        });

        t.start();

        //Bucle para los Sanitarios
        Sanitario[] sanitarios = new Sanitario[10];
        for(int i = 1; i <= 10; i++) {
            sanitarios[i-1] = new Sanitario(i,  salaVacunacion,
salaObservacion, log,salaDescanso);
            sanitarios[i-1].start();
        }


        //Auxiliares
        Auxiliar1 auxiliar1 = new Auxiliar1("A1", recepcion,
salaVacunacion, salaObservacion,  log,salaDescanso);
        Auxiliar2 auxiliar2 = new Auxiliar2("A2", salaVacunacion,
log,salaDescanso);
        auxiliar1.start();
        auxiliar2.start();

        //Pacientes
        Paciente[] pacientes = new Paciente[2000];
        for(int i = 1; i <= 2000; i++) {
            pacientes[i-1] = new Paciente(i, recepcion,
salaVacunacion, salaObservacion, log); //Creamos Paciente
            pacientes[i-1].start(); //El paciente entra
            try {
                sleep((new Random()).nextInt(3000)+1000); //Se
espera entre 1 y 3 segundos
            } catch (InterruptedException ex) {

Logger.getLogger(Hospital.class.getName()).log(Level.SEVERE, null,
ex);
            }
        }

        //Esperamos a que acaben los pacientes
        for(int i = 0; i < 2000; i++) {
            try {
                pacientes[i].join();
            } catch (InterruptedException ex) {

Logger.getLogger(Hospital.class.getName()).log(Level.SEVERE, null,
ex);
            }
        }
```

```
        //Una vez acaban los pacientes hacemos interrupt a los
sanitarios para que
        //Cuando no haya pacientes, salte la InterruptedExxception
y termine asÃ- la ecejuciÃ³n
        //del sanitario
        for(int i = 0; i < 10; i++) {
            sanitarios[i].interrupt();
        }

        //Interrumpimos tambiÃ©n los auxiliares
        auxiliar1.interrupt();
        auxiliar2.interrupt();

        //Esperamos a que terminen su ejecuciÃ³n tanto los
auxiliares como los
        //sanitarios para cerrar el fichero log.
        try {
            auxiliar1.join();
            auxiliar2.join();
            for(int i = 0; i < 10; i++) {
                sanitarios[i].join();
            }
        } catch (InterruptedException ex) {

Logger.getLogger(Hospital.class.getName()).log(Level.SEVERE, null,
ex);
        }
        log.escribir("El hospital ha cerrado");
        log.cerrarFichero();
    }

}
```

Clase Controlador

```
package mvc_cliente;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.concurrent.CopyOnWriteArrayList;

/**
 * La clase controlador es la que comunica la interfaz y el modelo
del cliente
 * @author David
 */
public class Controlador implements ActionListener, Serializable{
    private I_Interfaz interfaz;
    private ModeloCliente modelo;
    public Controlador(I_Interfaz interfaz, ModeloCliente modelo) {
        this.interfaz=interfaz;
        this.modelo=modelo;
    }
    /**
     * En este metodo se hace visible la interfaz y se inicializan
sus eventos.
     * También se realizan todas las acciones necesarias para
conseguir una conexion TCP
     * entre el cliente y el servidor usando métodos del modelo
     */
    public void arrancar(){
        interfaz.hacerVisible();
        interfaz.inicializarEventos();
        System.out.println("Conectando con servidor...");
        modelo.conectarseServidor();
        modelo.crearFlujos();
        System.out.println("Conectado");
        modelo.start();
    }

    /**
     * Se le dice a la interfaz que se actualice con la informacion
del hospital
     * y además habilitamos los botones que estaban dehabilitados
siempre que haya
     * vuelto un sanitario al puesto
     * @param informacionHospital
     */
    public void actualizarInterfaz(ArrayList<ArrayList<Object>>
informacionHospital) {
        interfaz.actualizar(informacionHospital);
        habilitarBotones(informacionHospital.get(8)); //Sanitarios
en Puestos de vacunacion
    }
    @Override
    public void actionPerformed(ActionEvent e) {
        //Se pasa el String del ActionCommand del botón pulsado al
modelo
        modelo.setCerrarPuesto(e.getActionCommand());
```

```
        //Se dehabilita el botón
        interfaz.deshabilitarBoton(e.getActionCommand());

    }

    public void habilitarBotones(ArrayList<Object> puestos) {
        for(int i = 0; i < puestos.size(); i++)
            if((Boolean)puestos.get(i) == true) {
                interfaz.habilitarBoton(i + 1);
            }
    }
}
```

```
package mvc_cliente;

import mvc_cliente.Controlador;
import java.util.ArrayList;


public interface I_Interfaz {
    public void setControlador(Controlador controlador);

    public void hacerVisible();

    public void actualizar(ArrayList<ArrayList<Object>>
informacionHospital);

    public void inicializarEventos();

    public void deshabilitarBoton(String actionCommand);

    public void habilitarBoton(Integer get);

}
```

Clase Interfaz_2

```
package mvc_cliente;

import mvc_cliente.I_Interfaz;
import mvc_cliente.Controlador;
import java.util.ArrayList;

/**
 * Esta clase incluye métodos para actualizar la interfaz y para
habilitar y
 * deshabilitar los botones
 * los puestos
 * @author David
 */
public class Interfaz_2 extends javax.swing.JFrame implements
I_Interfaz  {

    private Controlador controlador;

    public Interfaz_2() {
        initComponents();
    }
```

```
    public void setControlador(Controlador controlador){
        this.controlador=controlador;
    }

    @Override
    public void hacerVisible() {
        this.setVisible(true);
    }

    @Override
    public void actualizar(ArrayList<ArrayList<Object>>
informacionHospital) {


        //RECEPCION
        String pacienteEnRecepcion =
(String)informacionHospital.get(0).get(0);
        PacienteID.setText(pacienteEnRecepcion);
        //PacienteID.setText(pacienteEnRecepcion);
        ArrayList<Object> colaRecepcion =
informacionHospital.get(1);
        Cola.setText(colaRecepcion.toString());



        //SALA VACUNACION
        String vacunasDisponibles=
(String)informacionHospital.get(2).get(0);
        ArrayList<Object> pacientesVacunacion =
informacionHospital.get(3);
        ArrayList<Object> sanitariosVacunacion =
informacionHospital.get(4);
        VacunasDisponibles.setText(vacunasDisponibles) ;

        for(int i=0;i<10;i++){
            switch(i){
                case 0:
                    Puesto1.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;
                case 1:
                    Puesto2.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;
                case 2:
                    Puesto3.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;
                case 3:
                    Puesto4.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;
                case 4:
                    Puesto5.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;
                case 5:
```

```
                    Puesto6.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;
                case 6:
                    Puesto7.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;
                case 7:
                    Puesto8.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;
                case 8:
                    Puesto9.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;
                case 9:
                    Puesto10.setText(pacientesVacunacion.get(i)+" |
"+sanitariosVacunacion.get(i));
                    break;

            }
        }

        //Paceintes en puestos de observacion
        ArrayList<Object> idPacientes = informacionHospital.get(5);
        //Sanitarios en puestos de observacion
        ArrayList<Object> idSanitarios =
informacionHospital.get(6);
        for(int i=0;i<20;i++){
            switch(i){
                case 0:
                    Obs1.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 1:
                    Obs2.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 2:
                    Obs3.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 3:
                    Obs4.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 4:
                    Obs5.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 5:
                    Obs6.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
                case 6:
                    Obs7.setText(idPacientes.get(i)+" |
"+idSanitarios.get(i));
                    break;
```

```
                case 7:
                    Obs8.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 8:
                    Obs9.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 9:
                    Obs10.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 10:
                    Obs11.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 11:
                    Obs12.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 12:
                    Obs13.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 13:
                    Obs14.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 14:
                    Obs15.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 15:
                    Obs16.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 16:
                    Obs17.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 17:
                    Obs18.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 18:
                    Obs19.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
                case 19:
                    Obs20.setText(idPacientes.get(i) + " | " +
idSanitarios.get(i));
                    break;
            }

        //Personas en sala de descanso
        ArrayList<Object> personasDescansando =
informacionHospital.get(7);
        Saladescanso.setText(personasDescansando.toString());
```

```
    if(personasDescansando.contains("A1"))
        A1_ID.setText("");
    else
        A1_ID.setText("A1");
    if(personasDescansando.contains("A2"))
        A2_ID.setText("");
    else
        A2_ID.setText("A2");

}
}

public void inicializarEventos() {
    Cerrar1.setActionCommand("CERRAR1");
    Cerrar1.addActionListener(controlador);

    Cerrar2.setActionCommand("CERRAR2");
    Cerrar2.addActionListener(controlador);

    Cerrar3.setActionCommand("CERRAR3");
    Cerrar3.addActionListener(controlador);

    Cerrar4.setActionCommand("CERRAR4");
    Cerrar4.addActionListener(controlador);

    Cerrar5.setActionCommand("CERRAR5");
    Cerrar5.addActionListener(controlador);

    Cerrar6.setActionCommand("CERRAR6");
    Cerrar6.addActionListener(controlador);

    Cerrar7.setActionCommand("CERRAR7");
    Cerrar7.addActionListener(controlador);

    Cerrar8.setActionCommand("CERRAR8");
    Cerrar8.addActionListener(controlador);

    Cerrar9.setActionCommand("CERRAR9");
    Cerrar9.addActionListener(controlador);

    Cerrar10.setActionCommand("CERRAR10");
    Cerrar10.addActionListener(controlador);

}

@Override
public void deshabilitarBoton(String e) {
    int boton = Integer.parseInt(e.substring(6));

    switch(boton) {
        case 1:
            Cerrar1.setEnabled(false);
            break;
        case 2:
            Cerrar2.setEnabled(false);
            break;
        case 3:
            Cerrar3.setEnabled(false);
```

```
            break;
        case 4:
            Cerrar4.setEnabled(false);
            break;
        case 5:
            Cerrar5.setEnabled(false);
            break;
        case 6:
            Cerrar6.setEnabled(false);
            break;
        case 7:
            Cerrar7.setEnabled(false);
            break;
        case 8:
            Cerrar8.setEnabled(false);
            break;
        case 9:
            Cerrar9.setEnabled(false);
            break;
        case 10:
            Cerrar10.setEnabled(false);
            break;
    }
}

@Override
public void habilitarBoton(Integer boton) {
    switch(boton) {
        case 1:
            Cerrar1.setEnabled(true);
            break;
        case 2:
            Cerrar2.setEnabled(true);
            break;
        case 3:
            Cerrar3.setEnabled(true);
            break;
        case 4:
            Cerrar4.setEnabled(true);
            break;
        case 5:
            Cerrar5.setEnabled(true);
            break;
        case 6:
            Cerrar6.setEnabled(true);
            break;
        case 7:
            Cerrar7.setEnabled(true);
            break;
        case 8:
            Cerrar8.setEnabled(true);
            break;
        case 9:
            Cerrar9.setEnabled(true);
            break;
        case 10:
            Cerrar10.setEnabled(true);
            break;
```

```
        }
    }
```

```
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">//GEN-BEGIN:initComponents
    private void initComponents() {

        Recepcion = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        Cola = new javax.swing.JTextArea();
        jScrollPane2 = new javax.swing.JScrollPane();
        A1_ID = new javax.swing.JTextPane();
        jScrollPane3 = new javax.swing.JScrollPane();
        PacienteID = new javax.swing.JTextPane();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        SalaVacunacion = new javax.swing.JPanel();
        jLabel6 = new javax.swing.JLabel();
        jScrollPane6 = new javax.swing.JScrollPane();
        Puesto1 = new javax.swing.JTextPane();
        Cerrar1 = new javax.swing.JButton();
        jLabel7 = new javax.swing.JLabel();
        jScrollPane7 = new javax.swing.JScrollPane();
        Puesto2 = new javax.swing.JTextPane();
        Cerrar2 = new javax.swing.JButton();
        jLabel8 = new javax.swing.JLabel();
        jScrollPane8 = new javax.swing.JScrollPane();
        Puesto3 = new javax.swing.JTextPane();
```

```
Cerrar3 = new javax.swing.JButton();
jLabel9 = new javax.swing.JLabel();
jScrollPane9 = new javax.swing.JScrollPane();
Puesto4 = new javax.swing.JTextPane();
Cerrar4 = new javax.swing.JButton();
jLabel10 = new javax.swing.JLabel();
jScrollPane10 = new javax.swing.JScrollPane();
Puesto5 = new javax.swing.JTextPane();
Cerrar5 = new javax.swing.JButton();
jLabel11 = new javax.swing.JLabel();
jScrollPane11 = new javax.swing.JScrollPane();
Puesto6 = new javax.swing.JTextPane();
Cerrar6 = new javax.swing.JButton();
jLabel12 = new javax.swing.JLabel();
jScrollPane12 = new javax.swing.JScrollPane();
Puesto7 = new javax.swing.JTextPane();
Cerrar7 = new javax.swing.JButton();
jScrollPane13 = new javax.swing.JScrollPane();
Puesto8 = new javax.swing.JTextPane();
Cerrar8 = new javax.swing.JButton();
jLabel13 = new javax.swing.JLabel();
jScrollPane14 = new javax.swing.JScrollPane();
Puesto9 = new javax.swing.JTextPane();
Cerrar9 = new javax.swing.JButton();
jLabel14 = new javax.swing.JLabel();
jScrollPane15 = new javax.swing.JScrollPane();
Puesto10 = new javax.swing.JTextPane();
jLabel15 = new javax.swing.JLabel();
Cerrar10 = new javax.swing.JButton();
jScrollPane16 = new javax.swing.JScrollPane();
A2_ID = new javax.swing.JTextPane();
jScrollPane17 = new javax.swing.JScrollPane();
VacunasDisponibles = new javax.swing.JTextPane();
jLabel16 = new javax.swing.JLabel();
jLabel17 = new javax.swing.JLabel();
jLabel18 = new javax.swing.JLabel();
SalaObservacion = new javax.swing.JPanel();
jScrollPane18 = new javax.swing.JScrollPane();
Obs11 = new javax.swing.JTextPane();
jLabel19 = new javax.swing.JLabel();
jLabel20 = new javax.swing.JLabel();
jLabel21 = new javax.swing.JLabel();
jLabel22 = new javax.swing.JLabel();
jLabel23 = new javax.swing.JLabel();
jLabel24 = new javax.swing.JLabel();
jLabel25 = new javax.swing.JLabel();
jLabel26 = new javax.swing.JLabel();
jLabel27 = new javax.swing.JLabel();
jLabel28 = new javax.swing.JLabel();
jLabel29 = new javax.swing.JLabel();
jLabel30 = new javax.swing.JLabel();
jLabel31 = new javax.swing.JLabel();
jLabel32 = new javax.swing.JLabel();
jLabel33 = new javax.swing.JLabel();
jLabel34 = new javax.swing.JLabel();
jLabel35 = new javax.swing.JLabel();
jLabel36 = new javax.swing.JLabel();
jLabel37 = new javax.swing.JLabel();
```

```
        jLabel38 = new javax.swing.JLabel();
        jScrollPane19 = new javax.swing.JScrollPane();
        Obs1 = new javax.swing.JTextPane();
        jScrollPane20 = new javax.swing.JScrollPane();
        Obs2 = new javax.swing.JTextPane();
        jScrollPane21 = new javax.swing.JScrollPane();
        Obs3 = new javax.swing.JTextPane();
        jScrollPane22 = new javax.swing.JScrollPane();
        Obs4 = new javax.swing.JTextPane();
        jScrollPane23 = new javax.swing.JScrollPane();
        Obs5 = new javax.swing.JTextPane();
        jScrollPane24 = new javax.swing.JScrollPane();
        Obs6 = new javax.swing.JTextPane();
        jScrollPane25 = new javax.swing.JScrollPane();
        Obs7 = new javax.swing.JTextPane();
        jScrollPane26 = new javax.swing.JScrollPane();
        Obs8 = new javax.swing.JTextPane();
        jScrollPane27 = new javax.swing.JScrollPane();
        Obs9 = new javax.swing.JTextPane();
        jScrollPane28 = new javax.swing.JScrollPane();
        Obs10 = new javax.swing.JTextPane();
        jScrollPane29 = new javax.swing.JScrollPane();
        Obs12 = new javax.swing.JTextPane();
        jScrollPane30 = new javax.swing.JScrollPane();
        Obs13 = new javax.swing.JTextPane();
        jScrollPane31 = new javax.swing.JScrollPane();
        Obs14 = new javax.swing.JTextPane();
        jScrollPane32 = new javax.swing.JScrollPane();
        Obs15 = new javax.swing.JTextPane();
        jScrollPane33 = new javax.swing.JScrollPane();
        Obs16 = new javax.swing.JTextPane();
        jScrollPane34 = new javax.swing.JScrollPane();
        Obs17 = new javax.swing.JTextPane();
        jScrollPane35 = new javax.swing.JScrollPane();
        Obs18 = new javax.swing.JTextPane();
        jScrollPane36 = new javax.swing.JScrollPane();
        Obs19 = new javax.swing.JTextPane();
        jScrollPane37 = new javax.swing.JScrollPane();
        Obs20 = new javax.swing.JTextPane();
        jLabel39 = new javax.swing.JLabel();
        SalaDescanso = new javax.swing.JPanel();
        jLabel5 = new javax.swing.JLabel();
        jScrollPane4 = new javax.swing.JScrollPane();
        Saladescanso = new javax.swing.JTextArea();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE)
;
        setBackground(new java.awt.Color(255, 255, 255));
        setResizable(false);
        setSize(new java.awt.Dimension(1200, 900));

        Recepcion.setBackground(new java.awt.Color(220, 220, 220));

Recepcion.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));
```

```
        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 13)); //
NOI18N
        jLabel1.setText("RECEPCION");

        jLabel2.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel2.setText("Cola de espera");

        Cola.setEditable(false);
        Cola.setColumns(20);
        Cola.setRows(5);
        jScrollPane1.setViewportView(Cola);

        A1_ID.setEditable(false);
        jScrollPane2.setViewportView(A1_ID);

        PacienteID.setEditable(false);
        jScrollPane3.setViewportView(PacienteID);

        jLabel3.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel3.setText("Paciente");

        jLabel4.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel4.setText("Auxiliar");

        javax.swing.GroupLayout RecepcionLayout = new
javax.swing.GroupLayout(Recepcion);
        Recepcion.setLayout(RecepcionLayout);
        RecepcionLayout.setHorizontalGroup(

RecepcionLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
            .addGroup(RecepcionLayout.createSequentialGroup()

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(RecepcionLayout.createSequentialGroup()
                        .addGap(245, 245, 245)

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.TRAILING)
                            .addComponent(jLabel2)
                            .addComponent(jLabel1)))

.addGroup(RecepcionLayout.createSequentialGroup()
                        .addGap(38, 38, 38)
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 500,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(RecepcionLayout.createSequentialGroup()
                        .addGap(49, 49, 49)
                        .addComponent(jLabel3)
                        .addGap(108, 108, 108)
```

```
.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                            .addComponent(jLabel4,
javax.swing.GroupLayout.PREFERRED_SIZE, 121,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 91,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addContainerGap(62, Short.MAX_VALUE))

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                .addGroup(RecepcionLayout.createSequentialGroup()
                    .addGap(48, 48, 48)
                    .addComponent(jScrollPane3,
javax.swing.GroupLayout.PREFERRED_SIZE, 91,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap(461, Short.MAX_VALUE)))
        );
        RecepcionLayout.setVerticalGroup(

RecepcionLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
            .addGroup(RecepcionLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel1)
                .addGap(18, 18, 18)
                .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, 14, Short.MAX_VALUE)

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                    .addComponent(jLabel3)
                    .addComponent(jLabel4))
                .addGap(2, 2, 2)
                .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(33, 33, 33))

.addGroup(RecepcionLayout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
RecepcionLayout.createSequentialGroup()
                    .addContainerGap(191, Short.MAX_VALUE)
```

```
                    .addComponent(jScrollPane3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(33, 33, 33)))
        );

        SalaVacunacion.setBackground(new java.awt.Color(220, 220,
220));

SalaVacunacion.setBorder(javax.swing.BorderFactory.createLineBorder
(new java.awt.Color(0, 0, 0)));
        SalaVacunacion.setPreferredSize(new
java.awt.Dimension(1600, 320));

        jLabel6.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel6.setText("Puesto 1");

        Puesto1.setEditable(false);
        jScrollPane6.setViewportView(Puesto1);

        Cerrar1.setText("Cerrar");

        jLabel7.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel7.setText("Puesto 2");

        Puesto2.setEditable(false);
        jScrollPane7.setViewportView(Puesto2);

        Cerrar2.setText("Cerrar");

        jLabel8.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel8.setText("Puesto 3");

        Puesto3.setEditable(false);
        jScrollPane8.setViewportView(Puesto3);

        Cerrar3.setText("Cerrar");

        jLabel9.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel9.setText("Puesto 4");

        Puesto4.setEditable(false);
        jScrollPane9.setViewportView(Puesto4);

        Cerrar4.setText("Cerrar");

        jLabel10.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel10.setText("Puesto 5");

        Puesto5.setEditable(false);
        jScrollPane10.setViewportView(Puesto5);
```

```
        Cerrar5.setText("Cerrar");

        jLabel11.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel11.setText("Puesto 6");

        Puesto6.setEditable(false);
        jScrollPane11.setViewportView(Puesto6);

        Cerrar6.setText("Cerrar");

        jLabel12.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel12.setText("Puesto 7");

        Puesto7.setEditable(false);
        jScrollPane12.setViewportView(Puesto7);

        Cerrar7.setText("Cerrar");

        Puesto8.setEditable(false);
        jScrollPane13.setViewportView(Puesto8);

        Cerrar8.setText("Cerrar");

        jLabel13.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel13.setText("Puesto 8");

        Puesto9.setEditable(false);
        jScrollPane14.setViewportView(Puesto9);

        Cerrar9.setText("Cerrar");

        jLabel14.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel14.setText("Puesto 9");

        Puesto10.setEditable(false);
        jScrollPane15.setViewportView(Puesto10);

        jLabel15.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel15.setText("Puesto 10");

        Cerrar10.setText("Cerrar");

        A2_ID.setEditable(false);
        jScrollPane16.setViewportView(A2_ID);

        VacunasDisponibles.setEditable(false);
        jScrollPane17.setViewportView(VacunasDisponibles);

        jLabel16.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel16.setText("Auxiliar");
```

```
        jLabel17.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel17.setText("Vacunas disponibles");

        jLabel18.setFont(new java.awt.Font("Tahoma", 1, 13)); //
NOI18N
        jLabel18.setText("SALA DE VACUNACION");

        javax.swing.GroupLayout SalaVacunacionLayout = new
javax.swing.GroupLayout(SalaVacunacion);
        SalaVacunacion.setLayout(SalaVacunacionLayout);
        SalaVacunacionLayout.setHorizontalGroup(

SalaVacunacionLayout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
            .addGroup(SalaVacunacionLayout.createSequentialGroup()
                .addGap(48, 48, 48)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)

.addGroup(SalaVacunacionLayout.createSequentialGroup()

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel6)
                            .addComponent(jScrollPane6)
                            .addComponent(Cerrar1,
javax.swing.GroupLayout.DEFAULT_SIZE, 83, Short.MAX_VALUE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel7)
                            .addComponent(jScrollPane7)
                            .addComponent(Cerrar2,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel8)
                            .addComponent(jScrollPane8)
                            .addComponent(Cerrar3,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel9)
                            .addComponent(jScrollPane9)
                            .addComponent(Cerrar4,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)
```

```
.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel10)
                            .addComponent(jScrollPane10)
                            .addComponent(Cerrar5,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                                .addGap(309, 309, 309)
                                .addComponent(jScrollPane16,
javax.swing.GroupLayout.DEFAULT_SIZE, 83, Short.MAX_VALUE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
SalaVacunacionLayout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jLabel16)
                                .addGap(18, 18, 18)))
                        .addGap(251, 251, 251))

.addGroup(SalaVacunacionLayout.createSequentialGroup()

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel11)
                            .addComponent(jScrollPane11)
                            .addComponent(Cerrar6,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel12)
                            .addComponent(jScrollPane12)
                            .addComponent(Cerrar7,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel13)
                            .addComponent(jScrollPane13)
                            .addComponent(Cerrar8,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel14)
```

```
                            .addComponent(jScrollPane14)
                            .addComponent(Cerrar9,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel15)
                            .addComponent(jScrollPane15)
                            .addComponent(Cerrar10,
javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                                .addGap(309, 309, 309)
                                .addComponent(jScrollPane17)
                                .addGap(251, 251, 251))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
SalaVacunacionLayout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jLabel17)
                                .addGap(231, 231, 231))))))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
SalaVacunacionLayout.createSequentialGroup()

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .addComponent(jLabel18)
                .addGap(453, 453, 453))
        );
        SalaVacunacionLayout.setVerticalGroup(

SalaVacunacionLayout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
            .addGroup(SalaVacunacionLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel18)
                .addGap(33, 33, 33)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.TRAILING)

.addGroup(SalaVacunacionLayout.createSequentialGroup()

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.BASELINE)
                            .addComponent(jLabel10)
                            .addComponent(jLabel16))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
```

```
.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                                .addComponent(jScrollPane10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(jScrollPane16,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                            .addComponent(Cerrar5))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                            .addComponent(jLabel9)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                            .addComponent(jScrollPane9,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                            .addComponent(Cerrar4))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                            .addComponent(jLabel8)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                            .addComponent(jScrollPane8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                            .addComponent(Cerrar3))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                            .addComponent(jLabel7)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                            .addComponent(jScrollPane7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                            .addComponent(Cerrar2))
```

```
.addGroup(SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane6,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                        .addComponent(Cerrar1)))
                .addGap(18, 18, 18)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.TRAILING)

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel11)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane11,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                        .addComponent(Cerrar6))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel12)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane12,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                        .addComponent(Cerrar7))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel13)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane13,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                        .addComponent(Cerrar8))

.addGroup(SalaVacunacionLayout.createSequentialGroup()
                        .addComponent(jLabel14)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                        .addComponent(jScrollPane14,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                        .addComponent(Cerrar9))

.addGroup(SalaVacunacionLayout.createSequentialGroup()

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.BASELINE)
                            .addComponent(jLabel15)
                            .addComponent(jLabel17))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)

.addGroup(SalaVacunacionLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                            .addComponent(jScrollPane15,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jScrollPane17,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
)
                        .addComponent(Cerrar10)))
                .addContainerGap(60, Short.MAX_VALUE))
        );

        SalaObservacion.setBackground(new java.awt.Color(220, 220,
220));

SalaObservacion.setBorder(javax.swing.BorderFactory.createLineBorde
r(new java.awt.Color(0, 0, 0)));

        Obs11.setEditable(false);
        jScrollPane18.setViewportView(Obs11);

        jLabel19.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel19.setText("Puesto 1");
```

```
        jLabel20.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel20.setText("Puesto 2");

        jLabel21.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel21.setText("Puesto 3");

        jLabel22.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel22.setText("Puesto 4");

        jLabel23.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel23.setText("Puesto 5");

        jLabel24.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel24.setText("Puesto 6");

        jLabel25.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel25.setText("Puesto 7");

        jLabel26.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel26.setText("Puesto 8");

        jLabel27.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel27.setText("Puesto 9");

        jLabel28.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel28.setText("Puesto 10");

        jLabel29.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel29.setText("Puesto 11");

        jLabel30.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel30.setText("Puesto 12");

        jLabel31.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel31.setText("Puesto 13");

        jLabel32.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel32.setText("Puesto 14");

        jLabel33.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel33.setText("Puesto 15");
```

```
        jLabel34.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel34.setText("Puesto 16");

        jLabel35.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel35.setText("Puesto 17");

        jLabel36.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel36.setText("Puesto 18");

        jLabel37.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel37.setText("Puesto 19");

        jLabel38.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel38.setText("Puesto 20");

        Obs1.setEditable(false);
        jScrollPane19.setViewportView(Obs1);

        Obs2.setEditable(false);
        jScrollPane20.setViewportView(Obs2);

        Obs3.setEditable(false);
        jScrollPane21.setViewportView(Obs3);

        Obs4.setEditable(false);
        jScrollPane22.setViewportView(Obs4);

        Obs5.setEditable(false);
        jScrollPane23.setViewportView(Obs5);

        Obs6.setEditable(false);
        jScrollPane24.setViewportView(Obs6);

        Obs7.setEditable(false);
        jScrollPane25.setViewportView(Obs7);

        Obs8.setEditable(false);
        jScrollPane26.setViewportView(Obs8);

        Obs9.setEditable(false);
        jScrollPane27.setViewportView(Obs9);

        Obs10.setEditable(false);
        jScrollPane28.setViewportView(Obs10);

        Obs12.setEditable(false);
        jScrollPane29.setViewportView(Obs12);

        Obs13.setEditable(false);
        jScrollPane30.setViewportView(Obs13);

        Obs14.setEditable(false);
        jScrollPane31.setViewportView(Obs14);
```

```
        Obs15.setEditable(false);
        jScrollPane32.setViewportView(Obs15);

        Obs16.setEditable(false);
        jScrollPane33.setViewportView(Obs16);

        Obs17.setEditable(false);
        jScrollPane34.setViewportView(Obs17);

        Obs18.setEditable(false);
        jScrollPane35.setViewportView(Obs18);

        Obs19.setEditable(false);
        jScrollPane36.setViewportView(Obs19);

        Obs20.setEditable(false);
        jScrollPane37.setViewportView(Obs20);

        jLabel39.setFont(new java.awt.Font("Tahoma", 1, 13)); //
NOI18N
        jLabel39.setText("SALA DE OBSERVACION");

        javax.swing.GroupLayout SalaObservacionLayout = new
javax.swing.GroupLayout(SalaObservacion);
        SalaObservacion.setLayout(SalaObservacionLayout);
        SalaObservacionLayout.setHorizontalGroup(

SalaObservacionLayout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
            .addGroup(SalaObservacionLayout.createSequentialGroup()

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createSequentialGroup()
                        .addGap(33, 33, 33)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addGroup(SalaObservacionLayout.createSequentialGroup()
                                .addComponent(jScrollPane18,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(32, 32, 32)
                                .addComponent(jScrollPane29,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaObservacionLayout.createSequentialGroup()
                                .addComponent(jScrollPane19,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(32, 32, 32)
                                .addComponent(jScrollPane20,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)))
```

```
                        .addGap(30, 30, 30)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)
                            .addComponent(jScrollPane21,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jScrollPane30,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(28, 28, 28)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)
                            .addComponent(jScrollPane31,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jScrollPane22,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(30, 30, 30)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)
                            .addComponent(jScrollPane23,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jScrollPane32,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jScrollPane26,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaObservacionLayout.createSequentialGroup()
                                .addGap(29, 29, 29)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)
                                    .addComponent(jScrollPane24,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(jScrollPane33,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addGap(38, 38, 38)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
```

```
                                    .addComponent(jScrollPane25,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(jScrollPane34,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(jScrollPane35,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                            .addGap(28, 28, 28)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)
                                .addComponent(jScrollPane27,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(jScrollPane36,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGap(29, 29, 29))

.addGroup(SalaObservacionLayout.createSequentialGroup()
                            .addGap(49, 49, 49)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addGroup(SalaObservacionLayout.createSequentialGroup()

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                                    .addComponent(jLabel19)
                                    .addComponent(jLabel29))
                                .addGap(60, 60, 60)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                                    .addComponent(jLabel20)
                                    .addComponent(jLabel30))
                                .addGap(60, 60, 60)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                                    .addComponent(jLabel21)
                                    .addComponent(jLabel31))
                                .addGap(60, 60, 60)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                                    .addComponent(jLabel22)
                                    .addComponent(jLabel32))
                                .addGap(60, 60, 60)
```

```
.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                                    .addComponent(jLabel23)
                                    .addComponent(jLabel33))
                                .addGap(60, 60, 60)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                                    .addComponent(jLabel24)
                                    .addComponent(jLabel34)))

.addGroup(SalaObservacionLayout.createSequentialGroup()
                                .addComponent(jLabel39)
                                .addGap(7, 7, 7)))
                        .addGap(60, 60, 60)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                            .addComponent(jLabel25)
                            .addComponent(jLabel35))
                        .addGap(60, 60, 60)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                            .addComponent(jLabel26)
                            .addComponent(jLabel36))
                        .addGap(60, 60, 60)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                            .addComponent(jLabel27)
                            .addComponent(jLabel37))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED
, 38, Short.MAX_VALUE)))

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                        .addComponent(jScrollPane28,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jScrollPane37,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(SalaObservacionLayout.createSequentialGroup()
                        .addGap(10, 10, 10)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                            .addComponent(jLabel38)
                            .addComponent(jLabel28))))
                .addGap(22, 22, 22))
```

```
        );
        SalaObservacionLayout.setVerticalGroup(

SalaObservacionLayout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
            .addGroup(SalaObservacionLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel39)
                .addGap(13, 13, 13)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                    .addComponent(jLabel19)
                    .addComponent(jLabel20)
                    .addComponent(jLabel21)
                    .addComponent(jLabel22)
                    .addComponent(jLabel23)
                    .addComponent(jLabel24)
                    .addComponent(jLabel25)
                    .addComponent(jLabel26)
                    .addComponent(jLabel27)
                    .addComponent(jLabel28))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)
                    .addComponent(jScrollPane19,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane20,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane21,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane22,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane23,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane24,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane25,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane26,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane27,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane28,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(21, 21, 21)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
SalaObservacionLayout.createParallelGroup(javax.swing.GroupLayout.A
lignment.BASELINE)
                        .addComponent(jLabel29)
                        .addComponent(jLabel30))
                    .addComponent(jLabel31,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel32,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel33,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel34,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel35,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel36,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel37,
javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel38,
javax.swing.GroupLayout.Alignment.LEADING))
                .addGap(18, 18, 18)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.LEADING)

.addGroup(SalaObservacionLayout.createParallelGroup(javax.swing.Gro
upLayout.Alignment.TRAILING)
                                .addComponent(jScrollPane18,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(jScrollPane29,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
                            .addComponent(jScrollPane37,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addComponent(jScrollPane30,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jScrollPane31,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(jScrollPane33,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane32,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane34,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane35,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane36,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(54, Short.MAX_VALUE))
        );

        SalaDescanso.setBackground(new java.awt.Color(220, 220,
220));

SalaDescanso.setBorder(javax.swing.BorderFactory.createLineBorder(n
ew java.awt.Color(0, 0, 0)));

        jLabel5.setFont(new java.awt.Font("Tahoma", 1, 13)); //
NOI18N
        jLabel5.setText("SALA DE DESCANSO");

        Saladescanso.setEditable(false);
        Saladescanso.setColumns(20);
        Saladescanso.setRows(5);
        jScrollPane4.setViewportView(Saladescanso);

        javax.swing.GroupLayout SalaDescansoLayout = new
javax.swing.GroupLayout(SalaDescanso);
        SalaDescanso.setLayout(SalaDescansoLayout);
        SalaDescansoLayout.setHorizontalGroup(

SalaDescansoLayout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
            .addGroup(SalaDescansoLayout.createSequentialGroup()
```

```
.addGroup(SalaDescansoLayout.createParallelGroup(javax.swing.GroupL
ayout.Alignment.LEADING)

.addGroup(SalaDescansoLayout.createSequentialGroup()
                        .addGap(196, 196, 196)
                        .addComponent(jLabel5))

.addGroup(SalaDescansoLayout.createSequentialGroup()
                        .addGap(24, 24, 24)
                        .addComponent(jScrollPane4,
javax.swing.GroupLayout.PREFERRED_SIZE, 502,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        SalaDescansoLayout.setVerticalGroup(

SalaDescansoLayout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
            .addGroup(SalaDescansoLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel5)
                .addGap(18, 18, 18)
                .addComponent(jScrollPane4,
javax.swing.GroupLayout.PREFERRED_SIZE, 182,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(19, Short.MAX_VALUE))
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
            .addGroup(layout.createSequentialGroup()
                .addGap(10, 10, 10)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignm
ent.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(Recepcion,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(SalaDescanso,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                    .addComponent(SalaObservacion,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(SalaVacunacion,
javax.swing.GroupLayout.DEFAULT_SIZE, 1180, Short.MAX_VALUE))
                .addContainerGap())
```

```
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
            .addGroup(layout.createSequentialGroup()
                .addGap(11, 11, 11)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignm
ent.LEADING, false)
                    .addComponent(Recepcion,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(SalaDescanso,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addGap(9, 9, 9)
                .addComponent(SalaVacunacion,
javax.swing.GroupLayout.PREFERRED_SIZE, 292,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELAT
ED)
                .addComponent(SalaObservacion,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents



    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JTextPane A1_ID;
    private javax.swing.JTextPane A2_ID;
    private javax.swing.JButton Cerrar1;
    private javax.swing.JButton Cerrar10;
    private javax.swing.JButton Cerrar2;
    private javax.swing.JButton Cerrar3;
    private javax.swing.JButton Cerrar4;
    private javax.swing.JButton Cerrar5;
    private javax.swing.JButton Cerrar6;
    private javax.swing.JButton Cerrar7;
    private javax.swing.JButton Cerrar8;
    private javax.swing.JButton Cerrar9;
    private javax.swing.JTextArea Cola;
    private javax.swing.JTextPane Obs1;
    private javax.swing.JTextPane Obs10;
    private javax.swing.JTextPane Obs11;
    private javax.swing.JTextPane Obs12;
    private javax.swing.JTextPane Obs13;
    private javax.swing.JTextPane Obs14;
    private javax.swing.JTextPane Obs15;
    private javax.swing.JTextPane Obs16;
```

```
private javax.swing.JTextPane Obs17;
private javax.swing.JTextPane Obs18;
private javax.swing.JTextPane Obs19;
private javax.swing.JTextPane Obs2;
private javax.swing.JTextPane Obs20;
private javax.swing.JTextPane Obs3;
private javax.swing.JTextPane Obs4;
private javax.swing.JTextPane Obs5;
private javax.swing.JTextPane Obs6;
private javax.swing.JTextPane Obs7;
private javax.swing.JTextPane Obs8;
private javax.swing.JTextPane Obs9;
private javax.swing.JTextPane PacienteID;
private javax.swing.JTextPane Puesto1;
private javax.swing.JTextPane Puesto10;
private javax.swing.JTextPane Puesto2;
private javax.swing.JTextPane Puesto3;
private javax.swing.JTextPane Puesto4;
private javax.swing.JTextPane Puesto5;
private javax.swing.JTextPane Puesto6;
private javax.swing.JTextPane Puesto7;
private javax.swing.JTextPane Puesto8;
private javax.swing.JTextPane Puesto9;
private javax.swing.JPanel Recepcion;
private javax.swing.JPanel SalaDescanso;
private javax.swing.JPanel SalaObservacion;
private javax.swing.JPanel SalaVacunacion;
private javax.swing.JTextArea Saladescanso;
private javax.swing.JTextPane VacunasDisponibles;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel10;
private javax.swing.JLabel jLabel11;
private javax.swing.JLabel jLabel12;
private javax.swing.JLabel jLabel13;
private javax.swing.JLabel jLabel14;
private javax.swing.JLabel jLabel15;
private javax.swing.JLabel jLabel16;
private javax.swing.JLabel jLabel17;
private javax.swing.JLabel jLabel18;
private javax.swing.JLabel jLabel19;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel20;
private javax.swing.JLabel jLabel21;
private javax.swing.JLabel jLabel22;
private javax.swing.JLabel jLabel23;
private javax.swing.JLabel jLabel24;
private javax.swing.JLabel jLabel25;
private javax.swing.JLabel jLabel26;
private javax.swing.JLabel jLabel27;
private javax.swing.JLabel jLabel28;
private javax.swing.JLabel jLabel29;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel30;
private javax.swing.JLabel jLabel31;
private javax.swing.JLabel jLabel32;
private javax.swing.JLabel jLabel33;
private javax.swing.JLabel jLabel34;
private javax.swing.JLabel jLabel35;
```

```
    private javax.swing.JLabel jLabel36;
    private javax.swing.JLabel jLabel37;
    private javax.swing.JLabel jLabel38;
    private javax.swing.JLabel jLabel39;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane10;
    private javax.swing.JScrollPane jScrollPane11;
    private javax.swing.JScrollPane jScrollPane12;
    private javax.swing.JScrollPane jScrollPane13;
    private javax.swing.JScrollPane jScrollPane14;
    private javax.swing.JScrollPane jScrollPane15;
    private javax.swing.JScrollPane jScrollPane16;
    private javax.swing.JScrollPane jScrollPane17;
    private javax.swing.JScrollPane jScrollPane18;
    private javax.swing.JScrollPane jScrollPane19;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JScrollPane jScrollPane20;
    private javax.swing.JScrollPane jScrollPane21;
    private javax.swing.JScrollPane jScrollPane22;
    private javax.swing.JScrollPane jScrollPane23;
    private javax.swing.JScrollPane jScrollPane24;
    private javax.swing.JScrollPane jScrollPane25;
    private javax.swing.JScrollPane jScrollPane26;
    private javax.swing.JScrollPane jScrollPane27;
    private javax.swing.JScrollPane jScrollPane28;
    private javax.swing.JScrollPane jScrollPane29;
    private javax.swing.JScrollPane jScrollPane3;
    private javax.swing.JScrollPane jScrollPane30;
    private javax.swing.JScrollPane jScrollPane31;
    private javax.swing.JScrollPane jScrollPane32;
    private javax.swing.JScrollPane jScrollPane33;
    private javax.swing.JScrollPane jScrollPane34;
    private javax.swing.JScrollPane jScrollPane35;
    private javax.swing.JScrollPane jScrollPane36;
    private javax.swing.JScrollPane jScrollPane37;
    private javax.swing.JScrollPane jScrollPane4;
    private javax.swing.JScrollPane jScrollPane6;
    private javax.swing.JScrollPane jScrollPane7;
    private javax.swing.JScrollPane jScrollPane8;
    private javax.swing.JScrollPane jScrollPane9;
    // End of variables declaration//GEN-END:variables



}
```

Clase ModeloCliente

```
package mvc_cliente;

import mvc_cliente.Controlador;
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.OutputStream;
import java.io.Serializable;
import java.net.Socket;
import java.util.ArrayList;
import java.util.concurrent.CopyOnWriteArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;

public class ModeloCliente extends Thread implements Serializable {
    private Controlador controlador;
    private final int PUERTO=40080;
    private final String HOST="localhost";
    private Socket socket;
    private ObjectInputStream ois;
    private ObjectOutputStream oos;
    private CopyOnWriteArrayList<Integer> cerrarPuesto;
    private ArrayList<ArrayList<Object>> informacionHospital;
    public ModeloCliente() {
    }

    public void setControlador(Controlador controlador) {
        this.controlador=controlador;
        this.cerrarPuesto = new CopyOnWriteArrayList<Integer>();
    }


    public void conectarseServidor() {

        try {
            socket= new Socket(HOST,PUERTO);

        } catch (IOException ex) {

Logger.getLogger(ModeloCliente.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }


    public void crearFlujos() {
        try {
            oos= new ObjectOutputStream(socket.getOutputStream());
            ois= new ObjectInputStream(socket.getInputStream());

        } catch (IOException ex) {
```

```
Logger.getLogger(ModeloCliente.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }
    public void setCerrarPuesto(String n){
        cerrarPuesto.add(Integer.parseInt(n.substring(6)));
    }

    public void run() {

        while(true){
            try {
                //Se va a pedir informacion al servidor cada
segundo
                Thread.sleep(1000);
                //Al servidor le mandamos una lista con los puestos
que queremos cerrar
                oos.writeObject(cerrarPuesto);
                oos.reset();
                cerrarPuesto.clear();

                //El servidor nos manda toda la informacion del
hospital
                informacionHospital =
(ArrayList<ArrayList<Object>>) ois.readObject();

                //Actualizamos la interfaz a traves del controlador

controlador.actualizarInterfaz(informacionHospital);
                informacionHospital.clear();


            }  catch (InterruptedException ex) {

Logger.getLogger(ModeloCliente.class.getName()).log(Level.SEVERE,
null, ex);
            } catch (IOException ex) {

Logger.getLogger(ModeloCliente.class.getName()).log(Level.SEVERE,
null, ex);
            } catch (ClassNotFoundException ex) {

Logger.getLogger(ModeloCliente.class.getName()).log(Level.SEVERE,
null, ex);
            }

        }

    }




}
```

Clase Log

```
package recursos_compartidos;


import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.io.Serializable;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.logging.Level;
import java.util.logging.Logger;


/**
 * La clase Log servirá para escribir toda lo que vaya pasando en
el hospital
 * en el fichero informacionHospital.txt que estará dentro del
proio proyecto
 * @author David
 */
public class Log implements Serializable{

    private FileWriter fw;
    private PrintWriter pw;
    private DateFormat formato;

    public Log() {
        try {
            fw = new FileWriter("./evolucionHospital.txt");
            pw = null;
            formato = new SimpleDateFormat("dd/MM/yyyy
HH:mm:ss.SSS");
        } catch (IOException ex) {
            Logger.getLogger(Log.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }
    public void escribir(String mensaje) {
        pw = new PrintWriter(fw);
        pw.println(formato.format(new Date()) + " " + mensaje);
        pw.flush();
    }

    public void cerrarFichero() {
        try {
            pw.close();
            fw.close();
        } catch (IOException ex) {
            Logger.getLogger(Log.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }
```

```
}


/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
```

Clase PuestoObservacion

```
package recursos_compartidos;

import java.util.concurrent.CyclicBarrier;

/**
 * Clase para representar un puesto de la sala de observación
 * @author David
 */
public class PuestoObservacion {

    private int id;
    private String sanitarioAtendiendoReaccion;
    private String pacieneteEnObservacion;
    //barrera para gestionar el proceso de atender una reaccion
    private CyclicBarrier barreraReaccion;
    public PuestoObservacion(int id) {
        this.id = id;
        this.barreraReaccion = new CyclicBarrier(2);
        this.sanitarioAtendiendoReaccion = "";
        this.pacieneteEnObservacion = "            ";
    }

    public int getId() {
        return id;
    }

    public CyclicBarrier getBarreraReaccion() {
        return barreraReaccion;
    }

    public String getSanitarioAtendiendoReaccion() {
        return sanitarioAtendiendoReaccion;
    }

    public void setSanitarioAtendiendoReaccion(String
sanitarioAtendiendoReaccion) {
        this.sanitarioAtendiendoReaccion =
sanitarioAtendiendoReaccion;
    }

    public String getPacieneteEnObservacion() {
        return pacieneteEnObservacion;
    }
```

```
    public void setPacieneteEnObservacion(String
pacieneteEnObservacion) {
        this.pacieneteEnObservacion = pacieneteEnObservacion;
    }


}
```

Clase PuestoVacunacion

```
package recursos_compartidos;

import java.util.concurrent.CyclicBarrier;
import java.util.concurrent.Exchanger;
import java.util.concurrent.Semaphore;
/**
 * Clase para representar un puesto de la sala de observación
 * @author David
 */
public class PuestoVacunacion {
    private String sanitarioPresente,pacientePresente;
    //barrera para el proceso de vacunacion
    private CyclicBarrier pacienteListoVacunar;
    private int id;

    //Para la gestion de los botones para cerrar puestos
    private Exchanger<Integer> accion;
    private Semaphore semaforoGestionBoton;
    private boolean puestoAbierto;

    public PuestoVacunacion(int id) {
        this.pacientePresente="            ";
        this.sanitarioPresente="";
        this.pacienteListoVacunar = new CyclicBarrier(2);
        this.accion = new Exchanger<Integer>();
        this.semaforoGestionBoton = new Semaphore(0, true);
        this.puestoAbierto = false;
        this.id = id;
    }

    public Exchanger<Integer> getAccion() {
        return accion;
    }

    public Semaphore getSemaforoGestionBoton() {
        return semaforoGestionBoton;
    }

    public boolean isPuestoAbierto() {
        return puestoAbierto;
    }

    public void setPuestoAbierto(boolean puestoAbierto) {
        this.puestoAbierto = puestoAbierto;
    }

    public String getIdSanitario(){
```

```
        return sanitarioPresente;
    }

    public CyclicBarrier getBarrier(){
        return pacienteListoVacunar ;
    }

    public void setIdPaciente(String pacientePresente) {
        this.pacientePresente = pacientePresente;
    }

    public void setIdSanitario(String sanitarioPresente) {
        this.sanitarioPresente = sanitarioPresente;
    }
    public String getIdPaciente(){
        return pacientePresente;
    }

    public int getId() {
        return id;
    }
}
```

Clase Recepcion

```
package recursos_compartidos;

import recursos_compartidos.SalaObservacion;
import recursos_compartidos.SalaVacunacion;
import hilos.Paciente;
import java.io.Serializable;
import static java.lang.Thread.sleep;
import java.util.ArrayList;
;
import java.util.Random;
import java.util.concurrent.BrokenBarrierException;
import java.util.concurrent.CopyOnWriteArrayList;
import java.util.concurrent.LinkedBlockingQueue;

import java.util.concurrent.Semaphore;
import java.util.logging.Level;
import java.util.logging.Logger;



public class Recepcion implements Serializable {

    private SalaVacunacion salaVacunacion;
    private SalaObservacion salaObservacion;
    private String idPacienteDentro;
    private Semaphore pacienteEnRecepcion;

    private LinkedBlockingQueue<Paciente> cola = new
LinkedBlockingQueue<Paciente>();

    public Recepcion(SalaObservacion salaObservacion,
SalaVacunacion salaVacunacion) {
```

```
        this.salaObservacion = salaObservacion;
        this.salaVacunacion = salaVacunacion;
        this.pacienteEnRecepcion = new Semaphore(0);
    }
    /**
     * En esta funcion se gestiona el la comprobacion del paciente
(si tiene cita o no)
     * por parte del auxiliar 1.
     * Se utiliza un semáforo que empieza en 0 y que liberará el
paciente cuando
     * llegue a la recepción para que el auxiliar no atienda cuando
aún no hay paciente
     * Se utiliza un CyclicBarrier para indicarle al paciente que
ya ha sido atendido y que puede ir a vacunarse
     * @return mensaje que indica si el paciente no tiene cita, o,
en caso contrario,
     * en que puesto y con qué sanitario se vacunará
     * @throws InterruptedException
     */
    public String comprobarPaciente() throws InterruptedException {
        //Semaforo para empezar solo cuando haya un paciente en la
recepción
        pacienteEnRecepcion.acquire();

        //Entra el primer paciente de la cola
        Paciente paciente = cola.poll();
        //Se guarda el paciente que está actualmente en la
recepción
        this.idPacienteDentro = paciente.getID();
        String mensaje = "";
        //El auxiliar tarda entre 0.5 y 1s en realizar la
comprobacion
        sleep((new Random()).nextInt(500) + 500);

        //Hay un 1% de posibilidad de que el paciente no tenga cita
        int porcentaje = (new Random()).nextInt(100);
        if (porcentaje == 1) {
            paciente.setTieneCita(false);
            mensaje = "Paciente " + paciente.getID() + " ha acudido
sin cita";

        } else {
            //Si tiene cita se le asigna un puesto de vacunación y
de observación.
            //Además se ve que sanitario le vacunará para mostrarlo
en el registro del auxiliar 1

paciente.setPuestoVacunacion(salaVacunacion.getPuestoVacunacionPaci
ente());
            String idSanitario =
paciente.getPuestoVacunacion().getIdSanitario();
            mensaje = "Paciente " + paciente.getID() + " será
vacunado en el puesto " + paciente.getPuestoVacunacion().getId() +
" por " + idSanitario;

paciente.setPuestoObservacion(salaObservacion.getPuestoObservacionP
aciente());
            paciente.setTieneCita(true);
```

```
        }
        try {
            //Cuando sale el paciente se vacía idPacienteDentro y
se hace await() a la barrera
            //para indicar al paciente que ya puede irse
            this.idPacienteDentro = "";
            paciente.getBarrera().await();
        } catch (BrokenBarrierException ex) {

Logger.getLogger(Recepcion.class.getName()).log(Level.SEVERE, null,
ex);
        }
        return mensaje;
    }
    /**
     * El paciente se registra metiéndose en la cola y libera el
semaforo para
     * que lo pueda coger el auxiliar
     * @param paciente
     */
    public void registrarse(Paciente paciente) {
        cola.add(paciente);
        pacienteEnRecepcion.release();
    }

    public LinkedBlockingQueue<String> getColaIDs() {
        ArrayList<Paciente> listaCopia = new ArrayList<Paciente>();
        listaCopia.addAll(cola);

        LinkedBlockingQueue<String> listaIDs = new
LinkedBlockingQueue<String>();
        for (int i = 0; i < listaCopia.size(); i++) {
            listaIDs.add(listaCopia.get(i).getID());
        }

        return listaIDs;
    }

    public String getIdPacienteDentro() {
        return idPacienteDentro;
    }

}
```

Clase SalaDescanso

```
package recursos_compartidos;

import hilos.Sanitario;
import java.io.Serializable;
import static java.lang.Thread.sleep;
import java.util.Random;
import java.util.concurrent.CopyOnWriteArrayList;

import java.util.logging.Level;
import java.util.logging.Logger;
/**
 * En esta clase están los métodos para los descansos de los
auxiliares y los sanitarios.
 * Además de un CopyOnWriteArrayList con los IDs de los que estén
descansando en ese momento.
 * @author David
 */
public class SalaDescanso implements Serializable {

    private CopyOnWriteArrayList<String> salaDescansoId = new
CopyOnWriteArrayList<String>();

    public SalaDescanso() {

    }

    public void entrarDescansoAux1() throws InterruptedException {

        salaDescansoId.add("A1");
        sleep(new Random().nextInt(3000) + 3000);
        salaDescansoId.remove("A1");

    }

    public void entrarDescansoAux2() throws InterruptedException {
        salaDescansoId.add("A2");
        sleep((new Random()).nextInt(4000) + 1000);
        salaDescansoId.remove("A2");

    }

    public void entrarDescansoSanitario(Sanitario sanitario) throws
InterruptedException {
        salaDescansoId.add(sanitario.getID());
        sleep((new Random()).nextInt(4000) + 5000); //Entra a la
sala de descanso a cambiarse
        salaDescansoId.remove(sanitario.getID());
    }

    public void cambiarse(String idSanitario) throws
InterruptedException {
        salaDescansoId.add(idSanitario);
        sleep((new Random()).nextInt(3000) + 1000); //Entra a la
sala de descanso a cambiarse
        salaDescansoId.remove(idSanitario);
    }
```

```
    public CopyOnWriteArrayList<String> getIdsDescanso() {
        return salaDescansoId;
    }

}
```

Clase SalaObservacion

```
package recursos_compartidos;


import hilos.Sanitario;
import hilos.Paciente;
import java.io.Serializable;
import static java.lang.Thread.sleep;
import java.util.ArrayList;
import java.util.Random;
import java.util.concurrent.BrokenBarrierException;
import java.util.concurrent.CopyOnWriteArrayList;
import java.util.concurrent.CyclicBarrier;
import java.util.concurrent.LinkedBlockingQueue;
import java.util.concurrent.Semaphore;
import java.util.logging.Level;
import java.util.logging.Logger;


/**
 * En esta clase se gestiona todo lo relacionado con la sala de
observación, como
 * puede ser la propia observación o las reacciones de los
pacientes
 * @author David
 */
public class SalaObservacion implements Serializable {
    private CopyOnWriteArrayList<PuestoObservacion> puestos = new
CopyOnWriteArrayList<PuestoObservacion>();

    //En estas colas se guardan los puestos libres y los que tienen
un padiente con una reacción
    private LinkedBlockingQueue<PuestoObservacion>
colaObservacionPacientes;
    private LinkedBlockingQueue<PuestoObservacion>
colaReaccionesVacuna;

    //Semáforo que utiliza el auxiliar en la recepcion para asignar
un puesto al paciente
    private Semaphore puestosObservacionPaciente;

    public SalaObservacion(Semaphore puestosObservacion) {
        this.puestosObservacionPaciente = puestosObservacion;
        this.colaObservacionPacientes = new
LinkedBlockingQueue<PuestoObservacion>();
        this.colaReaccionesVacuna = new
LinkedBlockingQueue<PuestoObservacion>();
        for(int i = 1; i < 21; i++) {
            puestos.add(new PuestoObservacion(i));
            colaObservacionPacientes.add(puestos.get(i-1));
```

```
        }
    }
    /**
     * Se asigna el puesto de observación al paciente quitándolo de
la cola
     * para indicar que ya no está libre
     * @return puesto de observación
     * @throws InterruptedException
     */
    public PuestoObservacion getPuestoObservacionPaciente() throws
InterruptedException {
            puestosObservacionPaciente.acquire();
        if (!colaObservacionPacientes.isEmpty()) {
            return colaObservacionPacientes.poll();
        } else {
            return null;
        }
    }

    /**
     * Si hay alguna reaccion en la cola, el sanitario atiende la
reaccion
     * @param sanitario que va a atender la reaccion
     * @throws InterruptedException
     */
    public void atenderReaccion(Sanitario sanitario) throws
InterruptedException {
        if (!colaReaccionesVacuna.isEmpty()) {
            try {
                //Vemos cual es el paciente que lleva mas tiempo
sin ser atendido
                int numeroPuesto =
colaReaccionesVacuna.poll().getId();
                //Incluimos al sanitario que va a atender la
reaccion en el Array
                puestos.get(numeroPuesto -
1).setSanitarioAtendiendoReaccion(sanitario.getID());
                //Indicamos que el sanitario empieza a atender al
paciente
                puestos.get(numeroPuesto -
1).getBarreraReaccion().await();
                //Se le atiende
                sleep((new Random()).nextInt(4000) + 2000);

                //Se abre la CyclicBarrier para que el paciente se
vaya
                puestos.get(numeroPuesto -
1).getBarreraReaccion().await();

                puestos.get(numeroPuesto -
1).setSanitarioAtendiendoReaccion("");

            } catch (BrokenBarrierException ex) {

Logger.getLogger(SalaObservacion.class.getName()).log(Level.SEVERE,
null, ex);
            }
        }
```

```
    }

    /**
     * Funcion para que un paciente entre a un puesto de
observación
     * @param numeroPuesto
     * @param paciente
     */
    public void entrarSalaObservacion(int numeroPuesto, Paciente
paciente){
        puestos.get(numeroPuesto-
1).setPacieneteEnObservacion(paciente.getID());
    }

    /**
     * Si el paciente tiene una reacción tendrá que esperar a que
un sanitario
     * le atienda.
     * @param numeroPuesto
     * @return ID del sanitario que atiende la reacción
     * @throws InterruptedException
     */
    public String esperarAtencionReaccion(int numeroPuesto) throws
InterruptedException {
        String idSanitario = "";
        //Se añade el peusto del paciente a la cola de reacciones
        colaReaccionesVacuna.add(puestos.get(numeroPuesto-1));
        try {
            //Esperamos a que venga el sanitario a atender al
paciente
            puestos.get(numeroPuesto -
1).getBarreraReaccion().await();
            //Guardamos el sanitario que atiende al paciente
            idSanitario = puestos.get(numeroPuesto-
1).getSanitarioAtendiendoReaccion();
            //Esperamos a que el sanitario atienda al paciente
            puestos.get(numeroPuesto -
1).getBarreraReaccion().await();
        } catch (BrokenBarrierException ex) {

Logger.getLogger(SalaObservacion.class.getName()).log(Level.SEVERE,
null, ex);
        }
        //Devolvemos el id del sanitario que ha atendido a este
paciente
        return idSanitario;
    }

    /**
     * Al salir se vacía el string del paciente en ese puesto,
     * se añade el puesto a la cola de puestos libres y se libera
el semaforo
     * para dejar otro hueco
     * @param numeroPuesto
     */
    public void salirSalaObservacionPaciente(int numeroPuesto) {
        puestos.get(numeroPuesto-1).setPacieneteEnObservacion("
");
```

```
        colaObservacionPacientes.add(puestos.get(numeroPuesto-1));
        puestosObservacionPaciente.release();
    }

    public ArrayList<String> getSanitariosAtendiendoReaccion(){
        ArrayList<String> idSanitarios= new ArrayList<String>();
        for(int i=0;i<puestos.size();i++){

idSanitarios.add(puestos.get(i).getSanitarioAtendiendoReaccion());
        }
        return idSanitarios;
    }
    public ArrayList<String> getPacientesEnObservacion(){
        ArrayList<String> idSanitarios= new ArrayList<String>();
        for(int i=0;i<puestos.size();i++){

idSanitarios.add(puestos.get(i).getPacieneteEnObservacion());
        }
        return idSanitarios;
    }
}
```

Clase SalaVacunacion

```
package recursos_compartidos;

import hilos.Sanitario;
import hilos.Paciente;
import java.io.Serializable;
import static java.lang.Thread.sleep;
import java.util.ArrayList;
import java.util.Random;
import java.util.concurrent.BrokenBarrierException;
import java.util.concurrent.CopyOnWriteArrayList;
import java.util.concurrent.CyclicBarrier;
import java.util.concurrent.Exchanger;
import java.util.concurrent.LinkedBlockingQueue;
import java.util.concurrent.Semaphore;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 * En esta clase se gestiona todo lo relacionado con la vacunación
 * @author David
 */
public class SalaVacunacion implements Serializable {

    private CopyOnWriteArrayList<PuestoVacunacion> puestos;
    private LinkedBlockingQueue<PuestoVacunacion>
puestosLibresPaciente;
    private LinkedBlockingQueue<PuestoVacunacion>
puestosLibresSanitario;

    private Semaphore vacunasDisponibles = new Semaphore(0);
    private Semaphore puestosVacunacionPaciente;

    public SalaVacunacion(Semaphore puestosVacunacionPaciente) {
```

```
        this.puestosVacunacionPaciente = puestosVacunacionPaciente;
        this.puestos = new
CopyOnWriteArrayList<PuestoVacunacion>();
        this.puestosLibresPaciente = new
LinkedBlockingQueue<PuestoVacunacion>();
        this.puestosLibresSanitario = new
LinkedBlockingQueue<PuestoVacunacion>();
        for (int i = 1; i < 11; i++) {
            puestos.add(new PuestoVacunacion(i));
            puestosLibresSanitario.add(puestos.get(i - 1));
        }
    }

    /**
     * función usada por el auxiliar 1 para asignarle el puesto de
vacunacion al
     * paciente.
     * @return
     * @throws InterruptedException
     */
    public PuestoVacunacion getPuestoVacunacionPaciente() throws
InterruptedException {
        puestosVacunacionPaciente.acquire();

        if (!puestosLibresPaciente.isEmpty()) {
            //Se elimina de la cola el puesto libre y se retorna
ese puesto
            return puestosLibresPaciente.poll();
        } else {
            return null;
        }
    }

    /**
     * función para que el sanitario entre a un puesto de
vacunación
     * @param sanitario
     * @return el puesto al que entra en sanitario
     */
    public PuestoVacunacion
entrarPuestoVacunacionSanitario(Sanitario sanitario) {
        if (!puestosLibresSanitario.isEmpty()) {

            PuestoVacunacion puesto =
puestosLibresSanitario.poll();
            int numeroPuesto = puesto.getId();
            puestos.get(numeroPuesto -
1).setIdSanitario(sanitario.getID()); //Se añade el id del
sanitario en su puesto
            puestosLibresPaciente.add(puesto); //Cuando entra un
sanitario se añade como disponible el puesto para un paciente
            puestos.get(numeroPuesto - 1).setPuestoAbierto(true);
//El puesto ahora no está cerrado
            puestosVacunacionPaciente.release();//Libera un permit
del semaforo para que pueda entrar un paciente
            return puesto;
        } else {
            return null;
```

```
        }
    }

    /**
     * En esta función el sanitario espera a que el paciente llegue
o a que se tenga que cerrar
     * el puesto de vacunacion. Esto se gestiona con un semaforo
para esperar lo que le llegue primero
     * (si el paciente o la notificación de cerrar) y un exchanger
para saber qué es lo que le ha llegado
     * y actuar en consecuencia
     * @param numeroPuesto
     * @return boolean que indica si ha llegado a vacunar a un
paciente, o no
     * ha vacunado porque se ha cerrado el puesto
     * @throws InterruptedException
     */
    public boolean vacunar(int numeroPuesto) throws
InterruptedException {
        try {
            puestos.get(numeroPuesto -
1).getSemaforoGestionBoton().release();
            int accion = puestos.get(numeroPuesto -
1).getAccion().exchange(null);

            if (accion == 1) {

                //Coger una vacuna, si hay disponibles. Si no hay,
espera.
                vacunasDisponibles.acquire();
                //en caso de haber un paciente esperando vacunamos,
en otro caso esperamos
                puestos.get(numeroPuesto - 1).getBarrier().await();
                //El sanitario procede a vacunar
                sleep((new Random()).nextInt(3000) + 3000);
                //le decimos al paciente que le hemos vacunado
                puestos.get(numeroPuesto - 1).getBarrier().await();
                return true;
            }

        } catch (BrokenBarrierException ex) {

Logger.getLogger(SalaVacunacion.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return false;
    }

    /**
     * el paciente, una vez en el puesto de vacunacion debe esperar
a que el sanitario
     * le vacune
     * @param numeroPuesto
     * @param paciente
     * @return el id del sanitario que nos ha vacunado
     * @throws InterruptedException
     */
```

```
    public String esperarVacuna(int numeroPuesto, Paciente
paciente) throws InterruptedException {
        try {
            //Le indicamos al sanitario que le ha llegado algo
            puestos.get(numeroPuesto -
1).getSemaforoGestionBoton().acquire();
            //Le mandamos un mensaje indicandole que lo que le ha
llegado es un paciente para vacunar
            puestos.get(numeroPuesto - 1).getAccion().exchange(1);
            //Metemos al paciente en el puesto
            puestos.get(numeroPuesto -
1).setIdPaciente(paciente.getID());
            //Indicmos al sanitario que el paciente ya está en el
puesto
            puestos.get(numeroPuesto - 1).getBarrier().await();


            String idSanitario = puestos.get(numeroPuesto -
1).getIdSanitario();
            //el paciente espera a que el sanitario le vacune
            puestos.get(numeroPuesto - 1).getBarrier().await();
            return idSanitario;
        } catch (BrokenBarrierException ex) {

Logger.getLogger(SalaVacunacion.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return null;

    }
    /**
     * Al salir se vacía el string del paciente en ese puesto,
     * se añade el puesto a la cola de puestos libres y se libera
el semaforo
     * para dejar otro hueco
     * @param numeroPuesto
     */
    public void salirVacunacionPaciente(int numeroPuesto) {
        puestos.get(numeroPuesto - 1).setIdPaciente("
");
        puestosLibresPaciente.add(puestos.get(numeroPuesto - 1));
        puestosVacunacionPaciente.release();

    }

    /**
     * Al salir se vacía el string del sanitario en ese puesto,
     * se añade el puesto a la cola de puestos libres y se coge un
permit
     * del semaforo para indicar que hay un puesto menos disponible
     * @param numeroPuesto
     */
    public void salirVacunacionSanitario(int numeroPuesto) throws
InterruptedException {

        puestosVacunacionPaciente.acquire();
        puestos.get(numeroPuesto - 1).setIdSanitario("");
        puestosLibresSanitario.add(puestos.get(numeroPuesto - 1));
```

```
    }

    /**
     * El auxiliar 2 prepara una vacuna
     * @throws InterruptedException
     */
    public void auxiliar2PrepararVacuna() throws
InterruptedException {
        sleep((new Random().nextInt(500) + 500));
        vacunasDisponibles.release();
    }

    /**
     * Se cierra el/los puesto/s pasado/s como parametro
     * @param numeroPuestos
     * @throws InterruptedException
     */
    public void cerrarPuesto(CopyOnWriteArrayList<Integer>
numeroPuestos) throws InterruptedException {
        if (!numeroPuestos.isEmpty()) {
            for (int i = 0; i < numeroPuestos.size(); i++) {

                //Le indicamos al sanitario que le ha llegado algo
                puestos.get(numeroPuestos.get(i) -
1).getSemaforoGestionBoton().acquire();
                //Se cierra el puesto
                puestos.get(numeroPuestos.get(i) -
1).setPuestoAbierto(false);
                //Le mandamos un mensaje indicandole que lo que le
ha llegado es un peusto a cerrar
                puestos.get(numeroPuestos.get(i) -
1).getAccion().exchange(2);
            }
        }
    }

    public ArrayList<String> getIDSanitarios() {
        ArrayList<String> idSanitarios = new ArrayList<String>();
        for (int i = 0; i < puestos.size(); i++) {
            idSanitarios.add(puestos.get(i).getIdSanitario());
        }
        return idSanitarios;
    }

    public ArrayList<String> getIDPacientes() {
        ArrayList<String> idPacientes = new ArrayList<String>();
        for (int i = 0; i < puestos.size(); i++) {
            idPacientes.add(puestos.get(i).getIdPaciente());
        }
        return idPacientes;
    }

    public String getVacunasDisponibles() {
        return
String.valueOf(vacunasDisponibles.availablePermits());
    }

    public ArrayList<Boolean> getPuestosDisponibles() {
```

```
        ArrayList<Boolean> puestosDisponibles = new
ArrayList<Boolean>();
        for (int i = 0; i < puestos.size(); i++) {

puestosDisponibles.add(puestos.get(i).isPuestoAbierto());
        }
        return puestosDisponibles;
    }
}
```

Clase servidor

```
package servidor;

import mvc_cliente.Controlador;
import recursos_compartidos.Recepcion;
import recursos_compartidos.SalaDescanso;
import recursos_compartidos.SalaObservacion;
import recursos_compartidos.SalaVacunacion;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;

import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.CopyOnWriteArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;

public class Servidor extends Thread implements Serializable {
    private Controlador controlador;
    private final int PUERTO=40080;
    private ServerSocket sk;
    private Socket socket;
    private ObjectInputStream ois;
    private ObjectOutputStream oos;

    private SalaObservacion salaObservacion;
    private SalaVacunacion salaVacunacion;
    private Recepcion recepcion;
    private SalaDescanso salaDescanso;

    public Servidor(SalaObservacion salaObservacion, SalaVacunacion
salaVacunacion,Recepcion recepcion,SalaDescanso salaDescanso) {
        this.salaObservacion=salaObservacion;
        this.recepcion=recepcion;
        this.salaDescanso=salaDescanso;
        this.salaVacunacion=salaVacunacion;
```

```
    }

    public void setControlador(Controlador controlador) {
        this.controlador=controlador;
    }


    public void abrirPuerto() {
        try {
            sk=new ServerSocket(PUERTO);
        } catch (IOException ex) {


Logger.getLogger(Servidor.class.getName()).log(Level.SEVERE, null,
ex);
        }
    }
    public void esperarCliente(){
        try {
            socket=sk.accept();
        } catch (IOException ex) {

Logger.getLogger(Servidor.class.getName()).log(Level.SEVERE, null,
ex);
        }
    }

    public void crearFlujos() {
        try {
            oos= new ObjectOutputStream(socket.getOutputStream());
            ois= new ObjectInputStream(socket.getInputStream());
        } catch (IOException ex) {

Logger.getLogger(Servidor.class.getName()).log(Level.SEVERE, null,
ex);
        }
    }


    public void run(){
        System.out.println("Iniciando servidor...");
        abrirPuerto();
        System.out.println("Esperando al cliente...");
        esperarCliente();
        crearFlujos();
        System.out.println("Servidor iniciado");
        while(true){
            //Se crea la estructura de datos donde guardamos toda
la informacion
            //necesaria para actualizar la interfaz del cliente
            ArrayList<ArrayList<Object>> mensaje = new
ArrayList<>();
            for (int i = 0; i < 9; i++) {
                mensaje.add(new ArrayList<>());
            }
            try {
```

```
                //Recibimos los puestos a cerrar y llamamos a la
función de la sala
                //de vacunacion para cerrarlos
                CopyOnWriteArrayList<Integer> puestosCerrar =
(CopyOnWriteArrayList<Integer>) ois.readObject();
                salaVacunacion.cerrarPuesto(puestosCerrar);
                puestosCerrar.clear();

                //Vamos llenando los ArrayList con toda la
informacion del hospital

                //Paciente siendo atendido en recepcion
                mensaje.get(0).clear();

mensaje.get(0).add(recepcion.getIdPacienteDentro());
                //Cola recepcion
                mensaje.get(1).clear();
                mensaje.get(1).addAll(recepcion.getColaIDs());
                //Vacunas disponibles
                mensaje.get(2).clear();

mensaje.get(2).add(salaVacunacion.getVacunasDisponibles());
                //Pacientes en puestos de vacunacion
                mensaje.get(3).clear();

mensaje.get(3).addAll(salaVacunacion.getIDPacientes());
                //Sanitarios en puestos de vacunacion
                mensaje.get(4).clear();

mensaje.get(4).addAll(salaVacunacion.getIDSanitarios());
                //Pacientes en puestos de observacion
                mensaje.get(5).clear();

mensaje.get(5).addAll(salaObservacion.getPacientesEnObservacion());
                //Sanitarios en puestos de observacion
                mensaje.get(6).clear();

mensaje.get(6).addAll(salaObservacion.getSanitariosAtendiendoReacci
on());
                //Personas en sala de descanso
                mensaje.get(7).clear();

mensaje.get(7).addAll(salaDescanso.getIdsDescanso());

                //Para la gestion de los botones de Cerrar puestos

mensaje.get(8).addAll(salaVacunacion.getPuestosDisponibles());

                //Enviamos toda la informacion
                oos.writeObject(mensaje);
                oos.reset();
            } catch (Exception e){

            }

        }

    }
```

}