DP1 2020-2021
Documento de Requisitos y Análisis del Sistema

Mineral House Spa

https://github.com/gii-is-DP1/dp1-2020-g2-07

Miembros:

- JUAN MANUEL GARCIA CRIADO
- FERNANDO MIGUEL HIDALGO AGUILAR
- MIGUEL MOLINA RUBIO
- JAVIER RAMOS ARRONDO
- FRANCISCO JAVIER RODRIGUEZ ELENA

Tutor: Irene Bedilia Estrada

GRUPO G2-07

Versión 2.0

29/11/2020

Historial de versiones

Fecha	Versión	Descripción de los cambios	Sprint
03/11/2020	V1	● Creación del documento ● Añadidas historias H1-H10 ● Añadidas reglas de negocio 1-6 ● Descripción del proyecto ● Roles	1
29/11/2020	v2	● Añadidas historias H11-H28 ● Añadidas reglas de negocio 7-10	2
08/11/2020	v3	● Corregidas historias de usuario, eliminando las H3, H5, H12, H18, H19. Se añaden historias relativas a correos (H 23 - 25) ● R Simple 12 eliminada ● R Simple 16 y 17 añadidas	3
	v4	● R Simple 1, 6, 10, 11, 12, 13,14 cambiadas ● Modelo de datos terminado (diagrama de clases) ● R Negocio 11 ● Actualización de la tabla de planificación por Sprint ● Actualizadas historias de usuario, con las clases que intervienen en cada una, así como nuevos escenarios	4

Descripción general del proyecto

Con este software se desea no solo agilizar el proceso de administración de un Spa sino también mejorar la experiencia de los clientes a la hora de facilitar funcionalidades del mismo. Por parte de la administración, implementaremos funciones que permitan registro de usuarios y empleados y gestión de los mismos, creación de nuevas salas y gestión de la contabilidad del propio Spa.

Como hemos comentado anteriormente, no solo los administradores tendrán acceso a nuestro software, sino que los clientes también se verán beneficiados de él. A través de nuestra aplicación los clientes podrán acceder a su información para visualizar sus pagos, canjear bonos que les hayan podido llegar y pedir citas en las salas a las que deseen acudir.

Tipos de Usuario / Roles

***Administrador**: Gerente del spa que se encarga de la administración dando de alta a otros usuarios y gestionar la información registrada.*

***Empleado**: Trabajador del spa que se encarga de realizar diferentes actividades según su categoría.*

***Cliente**: Usuarios que pagan una cuota mensual para disfrutar de las actividades e instalaciones del spa.*

Historias de Usuario

Cliente:

H1ºRegistra clientes:
(Cliente.java,ClienteRepository.java,ClienteService.java,ClienteController.java,Pago.java, createOrUpdateClientsForm.jsp,User.java, Authorities.java, UserService.java)
Como administrador deseo que el sistema me permita consultar los clientes junto con sus datos (nombre, DNI, dirección, etc.) y registrarlos, para poder visualizar de manera cómoda.

Escenario Positivo:
H1+E1->Alta clientes Pedro Sánchez.
Dado que estamos registrados como administrador y que Pedro Sánchez no está dado de alta, cuando pulsemos el botón de añadir usuario se nos desplegará un formulario donde introducir los datos y una vez guardado con éxito nos aparecerá, si lo buscamos en la lista de clientes.

Escenario Negativo:
H1-E1->Alta clientes con campos sin rellenar.
Dado que estamos registrados como administradores y que Pedro Sánchez no está dado de alta, cuando pulsemos el botón de añadir nuevo cliente y dejemos cualquier campo vacío o solo ocupado con espacios (" "," ") al intentar añadirlo nos indicará que para agregar un usuario debe rellenarse todos los campos.

H1-E2->Alta usuario con campos mal rellenados
Dado que estamos registrados como administrador y que Pedro Sánchez no está dado de alta, cuando pulsemos el botón de añadir nuevos usuarios y dejemos cualquier campo de manera incorrecta (letras en un número de telf., espacios en un correo, ...) al intentar añadirlo nos indicará qué campos se han puesto de manera incorrecta y cómo deberían ser rellenados.

H2º Modificación de un cliente ya registrado

(Cliente.java,ClienteRepository.java,ClienteService.java,ClienteController.java,Pago.java, createOrUpdateClientsForm.jsp,User.java, Authorities.java, UserService.java)

Como administrador quiero poder modificar los datos de los clientes que están registrados para tener sus datos actualizados.

Escenario positivo:

H2+E1->Reemplazar uno de los datos básicos del cliente por uno nuevo.

Dado que estamos registrados como administrador y podemos ver el listado de clientes le pulsamos al botón "Editar" en el cliente deseado. Se nos desplegará un menú donde podremos reescribir algunos de los datos del cliente.

Al pulsar el botón "Guardar" y volver al menú de clientes veremos los cambios guardados.

Escenario negativo:

H2-E1->Eliminar uno de los datos y dejarlo vacío.

Dado que estamos registrados como administrador y podemos ver el listado de clientes le pulsamos al botón "Editar" en el cliente deseado. Se nos desplegará un menú donde podremos reescribir algunos de los datos del cliente.

Si al realizar alguna modificación dejamos alguno de los campos en blanco el sistema nos avisará y no dejará aplicar los cambios.

H3º Eliminación de un cliente ya registrado

(Cliente.java,ClienteRepository.java,ClienteService.java,ClienteController.java,Pago.java, clienteDetails.jsp,User.java, Authorities.java, UserService.java)

Como administrador quiero poder eliminar los clientes ya registrados.

Escenario positivo:

H3+E1->Eliminar exitosamente un cliente.

Dado que estamos registrados como administrador y podemos entrar al perfil de un cliente, pulsamos el botón de borrar y se elimine satisfactoriamente.

H4º Mostrar clientes

(Cliente.java,ClienteRepository.java,ClienteService.java,ClienteController.java,Pago.java, ClientsListing.jsp)

Como administrador quiero poder ver mis clientes listados y los detalles de manera individual, como cliente quiero poder ver mi perfil.

Escenario positivo:

H4+E1->Accedo a los empleado y se me muestran correctamente.

Escenario negativo:

H4-E1->Accedo como un cliente que no existe y me muestra un mensaje con el error y me redirige a mi perfil o al listado según con que me haya registrado.

H4-E2->Como cliente accedo a un perfil que no es el mío y me muestra un mensaje con el error y me redirige a mi perfil.

Empleado:

H5ºRegistra empleados:

(Employee.java,EmployeeRepository.java,EmployeeService.java,EmployeeRevenue.java,User.java, Authorities.java, UserService.java,createOrUpdateEmployee.jsp)

Como administrador deseo que el sistema me permita consultar a los empleados junto con sus datos (nombre, DNI, dirección, etc.) y registrarlos, para poder visualizar de manera cómoda.

Escenario Positivo:

H5+E1->Alta María Jimenez.

Dado que estamos registrados como administrador y que Maria Jimenez no está dado de alta, cuando pulsemos el botón de añadir empleado se nos desplegará un formulario donde introducir los datos y una vez guardado con éxito nos aparecerá, si lo buscamos en la lista de empleados.

Escenario Negativo:

H5-E1->Alta clientes con campos sin rellenar.

Dado que estamos registrados como administrador y que Maria Jimenez no está dado de alta, cuando pulsemos el botón de añadir nuevo empleado y dejemos cualquier campo vacío o solo ocupado con espacios (" "," ") al intentar añadirlo nos indicará que para agregar un usuario debe rellenarse todos los campos.

H5-E2->Alta empleado con campos mal rellenados.

Dado que estamos registrados como administrador y que Maria Jimenez no está dado de alta, cuando pulsemos el botón de añadir nuevo empleado y dejemos cualquier campo de manera incorrecta (letras en un número de telf., espacios en un correo, ...) al intentar añadirlo nos indicará qué campos se han puesto de manera incorrecta y cómo deberían ser rellenados.

H6º Modificación de un empleado ya registrado

(Employee.java,EmployeeRepository.java,EmployeeService.java,EmployeeRevenue.java,User.java, Authorities.java, UserService.java,createOrUpdateEmployee.jsp)

Como administrador quiero poder modificar los datos de los empleados que están registrados para tener sus datos actualizados.

Escenario positivo:

H6+E1->Reemplazar uno de los datos básicos del cliente por uno nuevo

Dado que estamos registrados como administrador y podemos ver el listado de empleados le pulsamos al botón "Editar" en el empleado deseado. Se nos desplegará un menú donde podremos reescribir algunos de los datos del empleado.

Al pulsar el botón "Guardar" y volver al menú de empleados veremos los cambios guardados.

Escenario negativo:

H6-E1->Eliminar uno de los datos y dejarlo vacío.

Dado que estamos registrados como administrador y podemos ver el listado de empleado le pulsamos al botón "Editar" en el empleado deseado. Se nos desplegará un menú donde podremos reescribir algunos de los datos del empleado.

Si al realizar alguna modificación dejamos alguno de los campos en blanco el sistema nos avisará y no dejará aplicar los cambios.

H7º Eliminación de un empleado ya registrado

(Employee.java,EmployeeRepository.java,EmployeeService.java,EmployeeRevenue.java,User.java, Authorities.java, UserService.java,employeeDetails.jsp)

Como administrador quiero poder eliminar los empleados ya registrados.

Escenario positivo:

H7+E1->Eliminar exitosamente un empleado.

Dado que estamos registrados como administradores y podemos entrar al perfil de un empleado, pulsamos el botón de borrar y se eliminará satisfactoriamente.

H9º Mostrar empleados

(Employee.java,EmployeeRepository.java,EmployeeService.java,EmployeeRevenue.java, employeesListing.jsp)

Como administrador quiero poder ver mis clientes listados y los detalles de manera individual, como empleado quiero poder ver mi perfil

Escenario positivo:

H9+E1->Accedo a los empleado y se me muestran correctamente.

Escenario negativo:

H9-E1->Accedo como un empleado que no existe y me muestra un mensaje con el error y me redirige a mi perfil o al listado según con que me haya registrado.

H9-E2->Como empleado accedo a un perfil que no es el mio y me muestra un mensaje con el error y me redirige a mi perfil.

Sala:

H10º Crear sala

(SalaService.Java,SalaFormatter.java,SalaRepository.java,Sala.java,SalaController.java,creatOrUpdateSalasForm.jsp)

Como administrador quiero poder crear salas para que los clientes puedan proceder a su reserva siempre dependiendo de su capacidad máxima..

Escenario Positivo:

H10+E1->Crear salas con datos correctos.

Dado que estamos registrados como administrador podemos crear varias salas con características semejantes a las del negocio.

Escenario Negativo:

H10-E1->Crear sala con datos incorrectos o nombres duplicados

El sistema no permite que el administrador pueda crear salas introduciendo datos erróneos en las campos del formulario

H11º Eliminar una sala

(SalaService.Java,SalaRepository.java,Sala.java,SalaController.java)

Como administrador quiero poder eliminar salas para la nueva introducción de otras.

Escenario Positivo:

H11+E1-> Eliminar sala que existe.

El administrador puede eliminar las salas existentes del negocio.

Escenario Negativo:

H11-E1-> Eliminar sala que no existe.

El sistema no permite que el administrador elimine alguna sala no disponible en la base de datos de la aplicación web.

H12º Editar una sala

(SalaService.Java,SalaFormatter.java,
SalaRepository.java,Sala.java,SalaController.java,createOrUpdateSalasForm.jsp)

Como administrador quiero poder editar salas por la posible modificación de algunas de estas..

Escenario positivo:

H12+E1->Con todos los campos rellenados correctamente.

El administrador puede editar salas siempre que se produzca alguna modificación de estas. La edición de estas se llevará a cabo en cada uno de los campos requeridos. Por ej: si se modifica el aforo por una ampliación en la reforma del edificio, el administrador modifica solo el aforo.

Escenario Negativo:

H12+E1-> Editar una sala con un campo vacío o nombres duplicados.

El sistema no permite que el administrador inserte nombres de salas duplicados o que modifique algún campo, siendo la modificación un campo vación en alguna parte del formulario o datos erróneos tales como una cadena de texto en el aforo.

H13º Mostrar salas

(SalaService.Java,SalaFormatter.java
SalaRepository.java,Sala.java,SalaController.java,salasListing.jsp,
salaDetails.jsp,Citas.java,CitasService.java,CitasRepository.java)

Como administrador/empleado quiero poder ver las salas disponibles y los detalles de manera individual, como cliente con sesión iniciada quiero poder ver las salas disponibles y los detalles que contengan las próximas sesiones sobre las que pedir cita.

Escenario positivo:

H13+E1->Accedo como administrador a las salas y se me muestran correctamente.

H13+E2->Accedo como cliente con sesión iniciada y puedo ver las sesiones disponibles para pedir cita, si las hubiese.

Escenario negativo:

H13-E1->Accedo sin iniciar sesión y se me mostrará la lista de salas pero no podré entrar a ver los detalles hasta que no se inicie la sesión.

H14º Pedir cita en una sala

(SalaService.Java,SalaRepository.java,Sala.java,SalaController.java,salasListing.jsp, salaDetails.jsp,Cita.java,CitaService.java,CitaRepository.java, ClienteService.java,ClienteRepository.java)

Como cliente quiero poder pedir cita en una sesión creada por el empleado para acudir a disfrutar del spa.

Escenario Positivo:

H14+E1->Sesión disponible y aforo no completo.

Dado que estamos registrados como cliente, existe una sesión disponible y que el aforo de la sala no está completo, cuando accedamos a los detalles de la sala se nos mostrará un listado con las sesiones que cumplen la condición anterior.

Escenario negativo:

H14-E1-> Aforo de la sala completo.

Dado que estamos registrados como clientes y que la sala tiene todas las plazas reservadas, esa sesión no aparecerá como disponible dentro de los detalles de la sala.

H14-E2-> Sesión fuera del horario de la categoría.

Dado que los clientes tienen un tipo de pase, si el horario de la sesión no se corresponde con el pase del cliente, esta no se mostrará en el listado de sesiones disponibles.

H14-E3->Cita ya pedida a cierta hora.

Aquellas citas que se pisen con las que el cliente ya tiene reservadas no se mostrarán en el listado de sesiones disponibles.

Circuito:

H15º Crear circuito

(Circuito.java,CircuitoService.java, SalaService.Java,SalaRepository.java,Sala.java,CircuitoRepository.java,CircuitoController.java, createOrUpdateCircuitosForm.jsp)

Como administrador quiero poder crear un circuito enlazando diferentes salas, para así tener experiencias más completas para los clientes.

Escenario Positivo:

H15+E1-> Crear circuito.

El administrador puede crear circuitos, añadiendo a cada circuito una lista de salas,nunca ningún circuito puede tener menos de 2 salas. El aforo se rellena automáticamente, siendo este el mínimo de los aforos de las salas incluidas en el circuito.

Escenario Negativo:

H15-E1-> Crear circuito con datos erróneos.

El sistema no permite que el administrador cree circuitos con menos de 2 salas, que se cree un circuito con un nombre duplicado o que alguno de los campos se deje vacío.

H16º Eliminar circuito

(Circuito.java,
CircuitoService.java,SalaService.Java,
SalaRepository.java,Sala.java,CircuitoRepository.java,CircuitoController.java)

Como administrador quiero poder eliminar un circuito para la nueva introducción de otros.

Escenario Positivo

H16+E1-> Eliminar circuito.

El administrador puede eliminar circuitos por algún motivo.

Escenario Negativo:

H16-E1-> Eliminar circuito que no existe.

El sistema no permite que el administrador elimine un circuito no existente en la base de datos de la aplicación web.

H17º Editar circuito

(Circuito.java,CircuitoService.java,
SalaService.Java,SalaRepository.java,Sala.java,CircuitoRepository.java,CircuitoController.java, createOrUpdateCircuitosForm.jsp)

Como administrador quiero poder editar un circuito por la posible modificación de alguno de estos.

Escenario Positivo

H17+E1-> Edición correcta de circuitos..

El administrador puede editar circuitos siempre que se produzca alguna modificación de estos. La edición de estos se llevará a cabo en cada uno de los campos requeridos. Por ej: si alguna de las salas deja de formar parte del circuito por la modificación de la estructura o por la eliminación de esta, la modificación se hará en el campo correspondiente a salas.

Escenario Negativo:

H17-E1-> Editar circuito a menos salas del mínimo

El sistema no permite que el administrador incluya menos de dos salas, o que se deje algún campo vacio o incluso llamar al circuito con un nombre que ya existe en la base de datos.

H18º Mostrar Circuitos

(Circuito.java,CircuitoService.java,CircuitoRepository.java,CircuitoController.java,
CircuitosListing.jsp)

Como administrador/empleado/cliente o usuario no dado de alta quiero poder consultar todos los circuitos disponibles en el SPA.

Escenario positivo

H18+E1-> Siendo admin, consultas los circuitos que quieras.

H18+E2 -> Siendo un empleado puedes ver los circuitos disponibles.

H18+E3-> Siendo cliente , consultas los circuitos que quieras.

H18+E4 -> Siendo un usuario no dado de alta, puedo consultar los circuitos disponibles en el SPA.

Horarios:

H19º Añadir horarios

(Horario.java,Employee.java,HorarioService.java,EmployeeService.java,HorarioRepository.java, EmployeeService.java, HorarioControllert.java,horarioForm.jsp)

Como administrador o empleado quiero poder añadir nuevos horarios para organizar la jornada laboral.

Escenario positivo:

H19+E1->Añadir nuevo horario

Dado que estamos registrados como administradores o empleados podemos añadir un nuevo horario pulsando en el botón "Añadir horario". Esto nos desplegará un formulario con la fecha que deseemos añadir. Una vez hayamos elegido la fecha, el empleado se asigna automáticamente desde el perfil que hayamos entrado. El nuevo día del calendario se mostrará en el listado de horarios.

Escenario negativo:

H19-E1->Añadir horario en una fecha pasada.

Si intentamos añadir un nuevo horario con una fecha anterior al día de hoy, saltará un mensaje de error.

H19-E2->Añadir horario en una fecha ya existente.

Si el empleado ya tiene un horario creado para una fecha concreta, no se podrá añadir esa fecha como nuevo horario.

H20º Añadir sesiones a un empleado en un determinado horario

(Horario.java,Employee.java,Sesion.java,SesionFormatter.java HorarioService.java,EmployeeService.java,HorarioRepository.java,EmployeeService.java, Sala.java,SalaService.java,HorarioController.java,horarioForm.jsp)

Como administrador/empleado, quiero poder añadir sesiones dentro de mi jornada laboral para poder especificar en qué sala trabajaré cada hora, y que así los clientes puedan pedir cita sobre ella.

Escenario positivo:

H20+E1-> Se añade una sesión nueva al empleado.

Estando registrado como administrador/empleado se puede crear sesiones de trabajo para un horario, seleccionando Add session to this day(que hace referencia a la fecha del horario creado) para terminar de completar el formulario y seleccionar horar de inicio y fin y la sala en la que el empleado va a estar trabajando.

Escenario negativo:

H20-E1-> Dado que no estamos registrados como empleados no podemos generar sesiones laborales.

H20-E2-> No se podrá tener una fecha de fin anterior a la de inicio.

H20-E3-> El trabajador ya tiene una sesión creada a esa hora en otra sala, por lo que saltará un mensaje de error que le permitirá elegir un horario diferente.

H20-E4-> Existe otro trabajador que ya tiene creada una sesión en esa sala a esa hora, por lo que saltará un mensaje de error que le permitirá elegir un horario diferente.

H20-E5-> Los trabajadores tienen una categoría o empleo como profesionales (masajistas, socorristas, etc), y las salas tienen un tipo según su uso, por lo que es probable que existan salas en las que el empleado no esté capacitado para trabajar. Por ejemplo, un socorrista no va a proporcionar masajes, por lo que saltará un mensaje de error que le permitirá elegir una sala diferente.

H21º Ver Horarios trabajador

(HorarioController.java,employeeSchedule.jsp,employeePastSessions.jsp, horarioService.java, horarioRepo.java)

Como trabajador quiero poder ver los horarios, para así saber que días tengo que ir a trabajar.

Escenario Positivo:

H21+E1->Dado que estamos registrados como empleado y que los horarios de las fechas que buscamos ya están fijados cuando accedemos a los mismos, aparecerán aquellos de los que nos encargamos nosotros en nuestro perfil.

Escenario Negativo:

H21-E1->Dado que estamos registrados como empleado y que los horarios de las fechas que buscamos no están fijados, no se mostrarán en nuestro perfil.

Bono:

H22º Crear Bono

(Bono.java, Sesion.java, SalaController.java, BonoService.java, createToken.jsp)

H22+E1-> Como administrador quiero poder crear bonos descuentos. Estos bonos serán utilizados por los clientes para poder acceder a citas fuera de su horario de suscripción.

H23º Mostrar Bonos

(Bono.java, BonoController.java, BonoService.java, BonosListing.jsp)

H23+E1-> Como administrador quiero poder ver un listado con todos los bonos disponibles.

H24º Consumir Bono

(Bono.java,Cliente.java, BonoController.java, BonoService.java, BonoRepository.java, ClienteService.java, ReedemToken.jsp)

Como cliente quiero poder canjear bonos en mi perfil, para poder acceder a citas fuera de mi horario de suscripción.

Escenario positivo:

H24+E1-> Si no hay citas para esa hora, el bono no se ha usado y no está caducado, se creará la cita y se marca el bono cómo usado.

Escenario negativo:

H24-E1-> En el caso de que exista una cita, no se creará una cita.

H24-E2-> En el caso de que el bono esté caducado, no se creará una cita.

H24-E3-> En el caso de que el bono ya haya sido usado, no se creará una cita.

H25º Eliminar Bono

(Bono.java, BonoController.java, BonoService.java, BonosListing.jsp)

Como administrador quiero poder eliminar los bonos existentes.

Escenario positivo:

H25+E1 -> El bono existe y se elimina satisfactoriamente.

Escenario negativo:

H25-E1 -> El bono no existe, así que no se puede eliminar.

H26º Recibir Bono

(Bono.java,Email.java, BonoController.java, BonoService.java, BonosListing.jsp)

Como cliente quiero poder adquirir bonos y que los códigos de estos se me envían automáticamente al correo

Escenario positivo:

H26+E1 -> El bono existe y se envía satisfactoriamente.

Escenario negativo:

H26-E1 -> El bono no existe, así que no se puede enviar.

Administrador:

H27º Añadir Pago por Suscripción

(Admin.java, Cliente.java, Pago.java, ClienteController.java, ClienteService.java, ClienteService.java, PayForm.jsp, ClienteDetails.jsp)

Escenario positivo:

H27+E1-> Como administrador quiero poder añadir los pagos de las mensualidades de los pagos de los clientes del spa.

Escenario negativo:

H27-E1->Al estar registrado como administrador y añadir un pago a una fecha anterior al último pago, nos dirá que no es posible dirigiéndonos al formulario.

H28º Añadir salario

(Admin.java, Employee.java, EmployeeRevenue.java, EmployeeController.java, EmployeeService.java, SalaryForm.jsp, EmployeeRepository.java,)

Escenario positivo:

H28+E1-> Como administrador quiero poder añadir los salarios de los empleados del spa.

Escenario negativo:

H28-E1-> Como administrador intento añadir un salario en el mismo mes que otro ya existente y me dará error.

H29º Envío de correo individual

(Email.java, AdminController.java, newEmail.jsp)

Como administrador quiero enviar correos a los usuarios registrados en la página.

Escenario positivo:

H29+E1->Enviar el correo exitosamente.

Estando registrado como admin selecciono la opción de enviar un email individual rellenando los campos correspondientes y se enviará correctamente.

Escenario negativo:

H29-E1->Fallo al intentar enviar un correo vacío.

Estando registrado como admin selecciono la opción de enviar un email individual sin rellenar los campos correspondientes y me solicitara rellenar los campos necesarios.

H30º Envío de correo circular
(Email.java, AdminController.java, newAnnouncement.jsp)

Como administrador quiero enviar correos circulares a los usuarios registrados en la página

Escenario positivo:
H30+E1->Enviar circular exitosamente.
Estando registrado como admin selecciono la opción de enviar una circular rellenando los campos correspondientes y se enviará correctamente.
Escenario negativo:
H30-E1->Fallo al intentar enviar circular.
Estando registrado como admin selecciono la opción de enviar una circular sin rellenar los campos correspondientes y me solicitara rellenar los campos necesarios.

H31º Dar de alta a usuarios
(AdminController.java, UserService.java, CienteService.java, ClienteService.java, Cliente.java, Categoria.java, checkUsers.jsp)
Como administrador quiero poder dar de alta los usuarios registrados en la página

Escenario positivo:
H31-E1->Dar de alta usuario.
Estando registrado como administrador, me aparecerá un aviso con aquellos clientes que se han registrado recientemente y todavía no han sido dados de alta, se podrá acceder a dichos usuarios y mediante un botón darlos de alta.

Balance:

H32º Mostrar Balance con estadísticas
(Balance.java,BalanceService.java,BalanceRepository.java,BalanceController.java, BalancesListing.jsp)
Como administrador quiero poder consultar todos mis balances y que también lo hagan los trabajadores designados.

Escenario positivo
H32+E1-> Siendo admin, consultas el balance que quieras.
H32+E2 -> Siendo un empleado designado, puedo consultar los balances que estén en mi lista.

Escenario negativo
H32-E1 -> Siendo empleado pero no estando designado para ver ese balance, el balance se muestra vacío.

H33º Crear Balance con estadísticas

(Balance.java, Balance.java, Balance.java, Balance.java)

Como administrador quiero que se generen automáticamente balances no editables de mi negocio, contando estadísticas como los sueldos, mantenimiento, bonos y suscripciones para llevar la cuenta de mi dinero.

Escenario positivo

H33+E1-> No hay balance ya creado y es día de balance, por lo que el nuevo balance se crea.

Escenario negativo

H33-E1 -> No es día de balance, por lo que el balance no se crea.

H34º Empleados designados para ver balances

(Balance.java, Balance.java, Balance.java, EmployeeService.java, BalancesEmplEdit.jsp)

Como administrador, quiero designar a ciertos trabajadores de mi confianza, a ver ciertos balances que yo elija.

Escenario positivo:

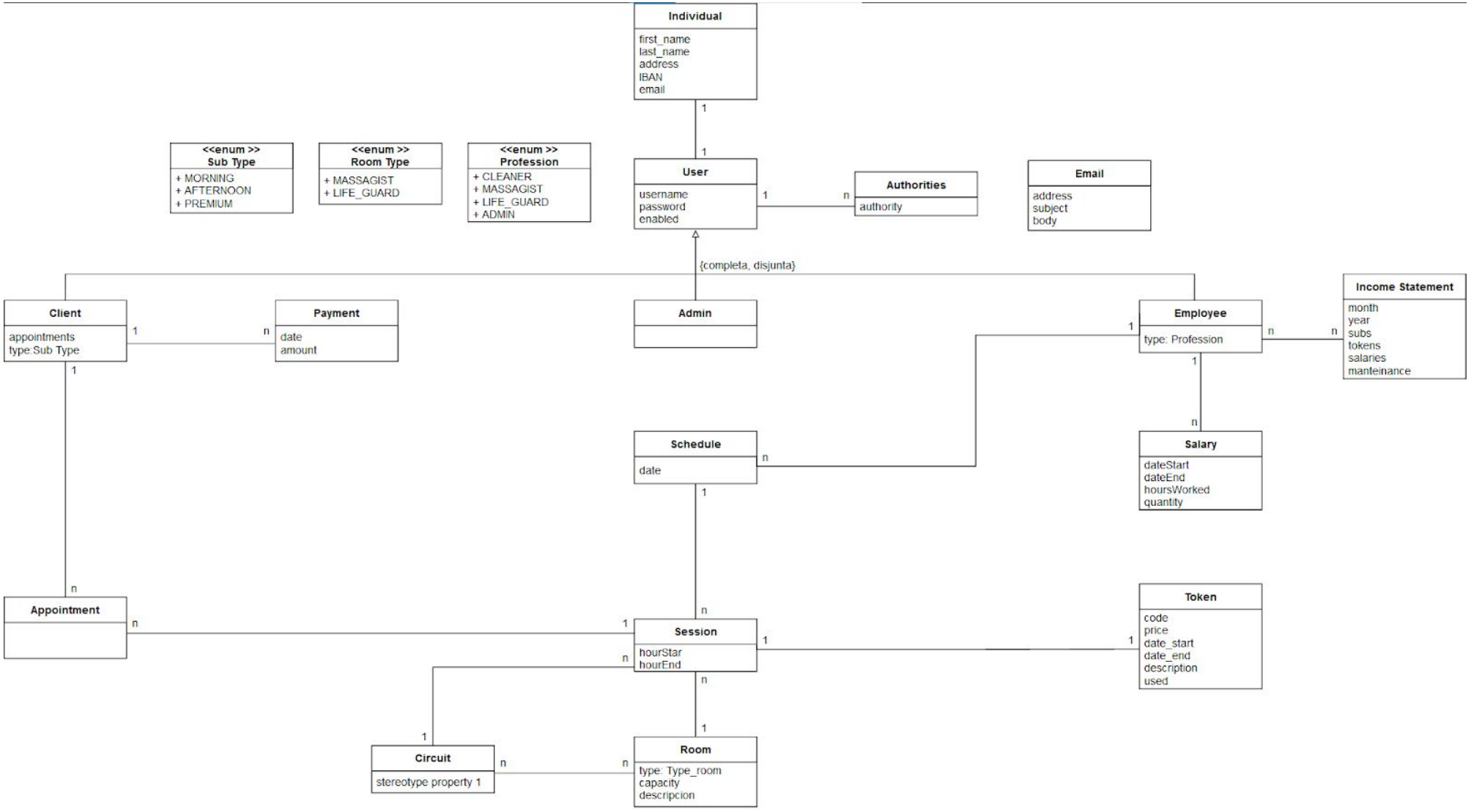
H34+E1-> El empleado se añade en el balance.

Escenario negativo:

H34-E1-> El balance no existe, por lo que el empleado no se añade.

H34-E2-> El empleado no existe, por lo que el empleado no se añade.

Modelo de Datos



Reglas de Negocio

R-<01> <Pago de la suscripción>
Los clientes tienen que estar al corriente del pago de las mensualidades de sus suscripciones mediantes tablas en su perfil.

R-<02> <Privilegios de ser Clientes Premium>
Los clientes premium pueden acceder al spa a las horas que les convengan dentro del horario del spa.

R-<03> <Horarios de reserva>
Los clientes no pueden adquirir una reserva de alguna de las salas del spa si el aforo está cubierto al máximo.

R-<04> <Realización de circuitos>
Los circuitos están compuestos mínimo por dos salas.

R-<05> <Bonos fuera de suscripción>
Los clientes pueden optar a acceder a sesiones fuera de las posibilidades de su suscripción, mediante el uso de bonos

R-<06> <Aforo de los circuitos>
En el caso de que dos salas de un circuito tengan distinto aforo máximo, el aforo del circuito será igual al aforo menor de las salas.

R-<07> <Bonos con códigos automáticos>
Los bonos tienen un código único generado automáticamente de forma aleatoria, por motivos de unicidad. En caso de haberlo introducido a mano, se comprueba que el código no está repetido

R-<08> <Balances para empleados de confianza>
Los balances también puedan ser consultados por los trabajadores, pero solo aquellos en los que yo decida

R-<09> <Número de pagos al mes>
La suscripción al spa y los salarios solo puede pagarse 1 vez al mes

R-<10> <Número de reservas simultáneas>
Un cliente no puede reservar dos salas a la misma hora.

R-<11> <Horario de empleado>
Un empleado X no puede estar en un sala Y si su horario marca que debe estar en la sala X.

R-<12> <Asistencia para los clientes>
Un cliente normal solo puede asistir a las horas posibles para su tipo de suscripción.

R-<13> <Empleados por sala>
Deberá haber un empleado por cada sala del spa

R-<14> <Cualificación de los empleados>
Los empleados solo pueden trabajar en salas en las que estén cualificados

R-<15> <Fechas de los bonos>
La fecha de inicio se establece automáticamente al momento de creación del bono. La fecha de fin se establece automáticamente para el día previo a la sesión

Restricciones simples

R-Simple<01>
La edad de los empleados debe ser mayor o igual a 18 años

R-Simple<02>
El aforo de un circuito es el menor aforo de las salas que lo componen

R-Simple<03>
Las salas no puede superar el aforo permitido

R-Simple<04>
Los anuncios enviados a los empleados o cliente siempre deben tener una categoría

R-Simple<05>
El correo debe ajustarse a la expresión regular propia de estos, esto es: x@y.z

R-Simple<06>
El IBAN debe ajustarse a la expresión regular propia de estos, por ejemplo: ES3912341234250123456789

R-Simple<07>
La longitud del código de los bonos debe ser de 12 caracteres

R-Simple<08>
La contraseña para acceder a la aplicación debe tener mínimo 8 caracteres, conteniendo por lo menos un número.

R-Simple<09>
No se pueden crear salas con el mismo nombre de otras salas ya creadas.

R-Simple<10>
No se pueden crear circuitos con el mismo nombre de otros circuitos ya creados.

R-Simple<11>
El precio de los bonos no se puede dejar en blanco

R-Simple<12>
El mes en los balances se almacenará y mostrará por su nombre en inglés y mayúsculas, por ejemplo: JANUARY

R-Simple<13>
El formato de todas las fechas será: yyyy-mm-dd

R-Simple<14>
El formato de todas las horas será: HH:MM

R-Simple<15>
Los clientes solo pueden pagar cantidades entre 30 y 50 unidades monetarias

R-Simple<16>
El aforo mínimo de una sala es de 1 persona

R-Simple<17>
El nombre de las salas no puede coincidir con ningún nombre de alguna sala que ya esté registrada en el sistema.

R-Simple<18>
El nombre de los circuitos no puede coincidir con ningún nombre de algun circuito que ya esté registrado en el sistema

Métricas del proyecto

Métrica	**Valor**
Nº de entidades (excluyendo actores)	*18*
Relaciones	*12*
Relaciones N:N	*2*
Restricciones Simples	*18*
Reglas de Negocio	*15*
Historias de Usuario totales	*34*
Historias de usuario involucrando 2 o más entidades	*21*
Historias de usuario involucrando 3 o más entidades	*13*
Actores	*3*

Planificación

Tipo	***Elemento***	***Asignación***	***Sprint***
Modelado Conceptual	• *Modelado Conceptual (Versión Preliminar)*	*Javier Ramos* *Fernando Hidalgo*	*1*
Historia de Usuario	• *H1 a H10*	*Juan Manuel*	*1*
Regla de Negocio	• *RN1 a RN6*	*Francisco Javier* *Miguel Molina*	*1*
Restricciones Simples	• *RS1 a R15*	*Francisco Javier* *Miguel Molina*	*1*
Historia de Usuario	• *H11 a H28*	*Diego Márquez* *Francisco Javier*	*2*
Regla de Negocio	• *RN7 a RN1*	*Diego Márquez* *Francisco Javier*	*2*
Entidad	• *Balance*	*Fernando Hidalgo* *Juan Manuel García*	*2*

	• *Cliente, Empleado, Pagos y Sueldos, Profesión y Suscripción*		
Service	• *Balance Service (Integración con GSon y CanvasJS)* • *Empleado Service, Empleado Service, Pagos Service y Sueldos Service*	*Fernando Hidalgo* *Juan Manuel García*	*2*
Controller	• *Balance Controller* • *Cliente Controller, Empleado Controller, Pagos Controller y Sueldos Controller*	*Fernando Hidalgo* *Juan Manuel García*	*2*
Relación	• *Cliente-Pago (1:n)* • *Empleado-Salario (1:n)*	*Juan Manuel García*	*2*
Entidad	• *Salas* • *Circuito*	*Javier Ramos* Francisco Javier	*2*
Service	• *Salas Service* • *Circuito Service*	*Javier Ramos* *Francisco Javier*	*2*
Controller	• *Salas Controller,SalaFormatter* • *Circuito Controller*	*Javier Ramos* *Francisco Javier*	*2*
Entidad	• *Bono (Versión no funcional)*	*Diego Márquez* *Miguel Molina*	*2*
Entidad	• *Individual, User, Admin, Email* • *Authorities*	*Juan Manuel García* *Miguel Molina*	*3*
Historia de Usuario	• *H23 a H25 y H30 a H33*	*Juan Manuel García* *Miguel Molina*	*3*
Service	• *User Service , Admin Service , Email Service* · *Authorities Service*	*Juan Manuel García* *Miguel Molina*	*3*
Controller	• *Admin Controller*	*Juan Manuel García*	*3*
Relación	• *Individual-User (1:1)* • *User-Authorities (1:n)*	*Juan Manuel García* *Miguel Molina*	*3*
Pruebas	• *Pruebas de User Service , Authorities Service, Admin Service , Email Service ClienteService, EmployeeService*	*Juan Manuel García* *Miguel Molina*	*3*
Entidad	• *Horario, Sesión,Cita*	*Javier Ramos* *Francisco Javier*	*3*
Service	• *Horario Service,CitaService*	*Javier Ramos* *Francisco Javier*	*3*
Controller	• *Horario Controller,SesionFormatter*	*Javier Ramos* *Francisco Javier*	*3*
Relación	• *Empleado-Horario (1:n)* • *Sesión-Horario (n:1)*	*Javier Ramos* *Francisco Javier*	*3*

	• *Sesión-Sala (n:1)* • *Cliente-Cita(1:n)* • *Sesion-Cita(1:n)* • *Sala-Circuito(n:n)*		
Restricciones Simples	• *RS1 a R15*	*Javier Ramos Francisco Javier*	*3*
Pruebas	• *Pruebas de Horario Service, Sala Service, Circuito Service, Cita Service*	*Javier Ramos Francisco Javier*	*3*
Entidad	• *Bono (Versión funcional), Tipo Sala*	*Fernando Hidalgo*	*3*
Service	• *Bono Service*	*Fernando Hidalgo*	*3*
Controller	• *Bono Controller*	*Fernando Hidalgo*	*3*
Relación	• *Bono-Sesión(1:1)*	*Fernando Hidalgo*	*3*
Pruebas	• *Pruebas de Balance Service y Bono Service*	*Fernando Hidalgo*	*3*
CSS	• *CSS al completo*	*Diego Márquez*	*4*
Pruebas	• *Pruebas de Bono Controller y Balance Controller*	*Fernando Hidalgo*	*4*
Relación	• *Balance-Empleado(n:n)*	*Fernando Hidalgo*	*4*
Restricciones Simples	• *Corregir e implementar* RS 1, 6, 10, 11, 12, 13,14	*Fernando Hidalgo*	*4*
Regla de Negocio	• *RN11*	*Fernando Hidalgo*	*4*
Modelado Conceptual	• *Modelado Conceptual (Versión Final)*	*Fernando Hidalgo*	*4*
Propuesta A+: Balance	• *Implementación y documentación de la propuesta A+ relacionada con la creación automática de balances*	*Fernando Hidalgo*	*4*
Pruebas	• *Pruebas de Cliente Controller, Empleado Controller, AdminController*	*Juan Manuel García Miguel Molina*	*4*
Pruebas	• *Pruebas de Horario Controller, Sala Controller, Circuito Controller, SalaFormatter*	*Javier Ramos Francisco Javier*	*4*
Historias de usuario	• *Cambio en la numeración de historias de usuario.* • *Sobreescritura de historias de usuario de H10 a H21*	*Javier Ramos Francisco Javier*	*4*
Propuesta A+: Pruebas de Mutación	• *Implementación y documentación de la propuesta A+ relacionada con las pruebas de mutación*	*Miguel Molina Rubio*	*4*