Iván Moreno Quirós

Este es un documento explicativo del software RentSoft para la presentación de este en un instituto oficial como trabajo final de grado.

RentSoft

Documentación

Tabla de contenido

[DOCUMENTACIÓN RENTSOFT 2](#_Toc25609810)

[1. IDENTIFICACIÓN DEL PROYECTO. 2](#_Toc25609811)

[2. ANÁLISIS DE REQUISITOS. 2](#_Toc25609812)

[2.1. OBJETIVOS. 2](#_Toc25609813)

[2.2. TECNOLOGÍAS UTILIZADAS. 2](#_Toc25609814)

[3. DISEÑO DE LA APLICACION 2](#_Toc25609815)

[3.1. DISEÑO DE LA BASE DE DATOS 2](#_Toc25609816)

[3.1.1. PISTAS 3](#_Toc25609817)

[3.1.2. USUARIOS 3](#_Toc25609818)

[3.1.3. ALQUILER 4](#_Toc25609819)

[3.1.4. ADMINISTRADOR/ESCRITORIO 5](#_Toc25609820)

[3.1.5. CLIENTE/ANDROID 13](#_Toc25609821)

[3.1.6. SERVIDOR 13](#_Toc25609822)

[3.1.7. MODELO 14](#_Toc25609823)

[4. MANUALES. 15](#_Toc25609824)

[4.1. Inicio de sesion / login 15](#_Toc25609825)

[4.2. Pantalla de gestion de usuarios. 15](#_Toc25609826)

[4.2.1. Insertar 16](#_Toc25609827)

[4.2.2. Modificar 18](#_Toc25609828)

[4.2.3. Borrar 19](#_Toc25609829)

[4.3. Reservar Pista 19](#_Toc25609830)

[5. CONCLUSIONES 20](#_Toc25609831)

[6. PROPUESTA DE AMPLIACIÓN. 21](#_Toc25609832)

[6.1. INTERACTIVIDAD 21](#_Toc25609833)

[6.2. AMBICION 21](#_Toc25609834)

[6.3. CONTROL 21](#_Toc25609835)

[6.4. SEGURIDAD 21](#_Toc25609836)

[7. BIBLIOGRAFÍA. 21](#_Toc25609837)

[8. ANEXOS. 22](#_Toc25609838)

[8.1. ANEXO 1 22](#_Toc25609839)

# DOCUMENTACIÓN RENTSOFT

## IDENTIFICACIÓN DEL PROYECTO.

Iván Moreno Quirós.

Desarrollo de aplicaciones multiplataformas (DAM).

I.E.S. Augustobriga, Navalmoral de la mata, Cáceres, Extremadura.

Proyecto de renta de pistas deportivas para un ente deportivo por la falta de información de los clientes que reservan en las pistas ya que muchos reservan y no asisten a la reserva, con este sistema se sabrá que cliente a reservado y cuando, y así se podrá tomar medidas al respecto.

## ANÁLISIS DE REQUISITOS.

### OBJETIVOS.

La aplicación tiene 3 partes diferentes:

* + 1. Por un lado, tiene un servidor con una base de datos que podrá subirse a cualquier dominio para acceder a él a través de una ip.
    2. Después estará, por otro lado, un cliente diseñado para labores administrativas dentro del servicio de atención al cliente, donde se podrá añadir, modificar, borrar nuevos clientes y reservar alguna pista en caso de que sea necesario.
    3. Y, por último, un cliente diseñado para terminales Android para poder reservar la pista una vez estés inscrito en el club deportivo que corresponda.

Para reservar la pista se tendrá en cuenta que tendrán que ser un máximo de dos días contando con el mismo día, lo que es lo mismo, solo podrás alquilar una pista el día en el que estés y el día siguiente a ese día. También se tendrá en cuenta que se alquilará desde que abra el ente deportivo hasta la hora de su cierre y se dividirá por horas.

### TECNOLOGÍAS UTILIZADAS.

* 1. Utilizare un servidor en Java hecho con socket atendiendo peticiones de cualquier cliente que entre a esa ip hecho con Eclipse y sin ninguna interfaz gráfica.
  2. Una base de datos en MySQL manejando el lenguaje SQL.
  3. Un cliente administrador hecho con Java con una interfaz gráfica hecha en NetBeans con la librería swing y una añadida llamada edisonCorX y la ayuda grafica hecha con JavaHelp.
  4. También incluirá un proyecto Android para el cliente hecho con xml para la interfaz y con Java para los controladores en el IDE de Android Studio.

## DISEÑO DE LA APLICACION

El diseño de esta aplicación está basado en tres tablas de la base de datos ya que es lo más practico a la hora de guardar la información es hacer una buena relación entre las tablas.

### DISEÑO DE LA BASE DE DATOS

Para el diseño de la base de datos he preferido que sea chiquitita y tenga una muy buena relación para que a la hora de guardar los datos sea lo más breve posible.

La base de datos está basada en una sola relación ya que no se necesita más para la realización de este proyecto.

Existe una relación que es la principal que es la de pistas (1: N ya que una pista puede tener varios usuarios, pero los usuarios solo pueden estar en una pista) con usuarios (1: N ya que un usuario solo puede estar en varias pistas, pero en distinta hora, pero a la misma hora no puede estar en dos pistas diferentes).

De esa relación sale una relación N: N que al ser de esta manera forma una nueva tabla que llamaremos alquiler que servirá para distinguir las horas y los días que los usuarios usan las pistas.

#### PISTAS

Como podemos comprobar tenemos una tabla pistas que incluirá todas las pistas que tengamos en ese institución o ente deportivo, esta tabla incluye:

* “id\_pista” este dato servirá para distinguir todas las pistas que haya en la base de datos definidas.
* “num” número que se refiere al número de pista que le quiera dar la institución o ente deportivo a su pista por si ellos tienen una enumeración propia de sus pistas.
* “tipo” y por último tenemos el tipo de la pista que queremos guardar que se guardara como una cadena de letras y así poder asignarle cualquier tipo de pista que quiera poner el administrador. En este programa trabajaremos pensando en 3 pistas estáticas de pádel.

#### USUARIOS

En esta tabla tendremos todos los registros para tener la información primordial de un cliente de la empresa donde que contrate este producto. Estos datos pueden adaptarse según las necesidades de cada institución o ente deportivo.

Pero las más importantes las nombrare aquí y se podrán ver en la imagen. Estas son las siguientes:

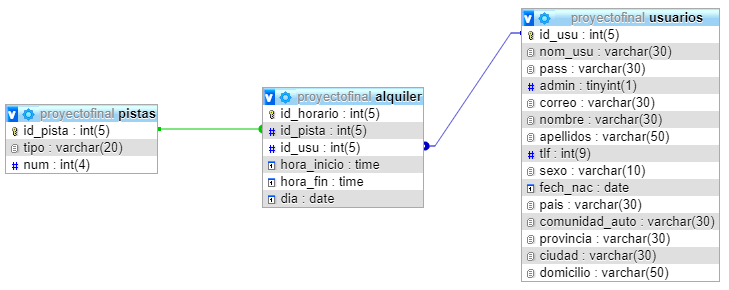
* ”id\_usu” para distinguir imparcialmente cada usuario y que no se repitan. Este dato será un número que será único y auto incrementable y no hace falta que nadie lo introduzca.
* “nom\_usu” este dato definirá el nombre de usuario para acceder a la aplicación tanto móvil como administrador en el escritorio.
* “pass” este será el dato para la contraseña del usuario anteriormente nombrado.
* “admin” será el dato que guardara un 0 o un 1 para saber si es administrador y si puede acceder a la aplicación de escritorio.
* “correo” será el dato que guarde el correo electrónico del usuario.
* “nombre” será el dato que guarde el nombre del usuario guardado.
* “apellidos” será el dato que guarde los apellidos del usuario guardado.
* “tlf” será el dato que guarde el teléfono del usuario guardado.
* “sexo” será el dato que guarde si el usuario es hombre o mujer u otra cosa.
* “fech\_nac” será el dato que guarde el día de su nacimiento.
* “pais” será el dato que guarde el país en el que vive.
* “comunidad\_auto” será el dato que guarde la comunidad autónoma del usuario guardado.
* “provincia” será el dato que guarde la provincia a la que pertenece el usuario guardado.
* “ciudad” será el dato que guarde la ciudad del usuario.
* “domicilio” será el dato que guarde la dirección de la vivienda del usuario guardad.

#### ALQUILER

Esta tabla se crea a partir de la relación de las dos anteriores, un usuario puede tener varias pistas y una pista puede tener varios usuarios el mismo día así es que nace esta tercera tabla con la intención de mezclar estas dos y puesto que ya está creada la aprovechamos para poner los horarios de las pistas la cual los usuarios pueden acceder para alquilarla. Esta tabla contiene los siguientes datos:

* “id\_horario” este dato guardara un id irrepetible para cada fila de la tabla.
* “id\_pista” este dato es una referencia directa al “id\_pista” de la tabla de la tabla pistas para que cuando se cree un alquiler se guarde una referencia de la pista donde se está haciendo ese alquiler.
* “id\_usu” este dato es una referencia directa al dato “id\_usu” de tabla usuarios para que cuando se cree un alquiler se sepa que usuario está realizando este alquiler.
* “hora\_inicio” este dato guarda un tipo llamado time que marca la hora a la cual empieza la reserva.
* “hora\_fin” este dato guarda un tipo llamado time que marca la hora a la cual termina la reserva.
* “dia” este dato guarda el día en el que se realiza la reserva.

En esta foto se representan todas las tablas y lo explicado anteriormente.



**Fig.1 Modelo E/R de la base de datos.**

DISEÑO DE LA APLICACIÓN

Para empezar, mostrare los diferentes diagramas UML para que se vea bien el funcionamiento de la aplicación.

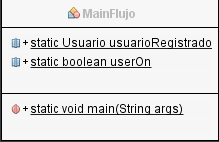
Antes de nada, señalar que el modelo de la aplicación se hace aparte ya que todos los puntos explicados más adelante tienen el mismo sistema de modelos lo que es lo mismo tienen las mismas clases para que no falle en el intercambio de archivos.

#### ADMINISTRADOR/ESCRITORIO

Este programa está hecho en Java y será multiplataforma ya que con la clase “File” de java tenemos una herramienta muy potente en uno de sus métodos llamado “separator” que hará que los slash sean admitidos para el sistema operativo donde este corriendo este programa. Esto y usando rutas relativas en todos los procesos hace que uniendo todo en un mismo proyecto el programa funcione en cualquier plataforma en donde se lance.

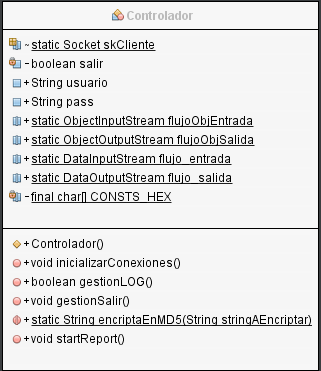
Podemos ver que este programa está pensado para estar en la oficina o incluso al lado de las pistas donde se quiere reservar para que cualquier persona que esté cerca y haya alguien del personal pendiente del programa pueda realizar su alquiler. También está pensado por si existe algún recepcionista y también así se pueda coger alquileres por teléfono si el recepcionista y el centro disponen de teléfono.

##### CONTROLADOR

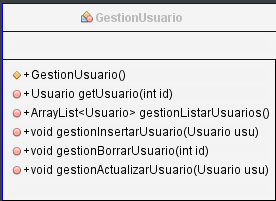


En este apartado hablaremos sobre el administrador de las pistas orientado a la aplicación de escritorio.

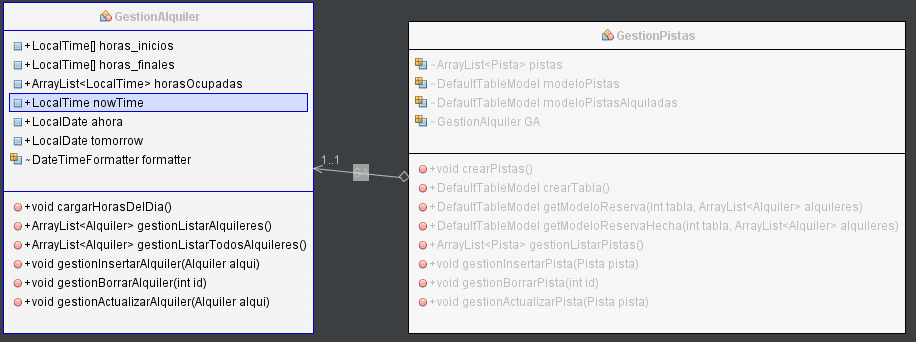
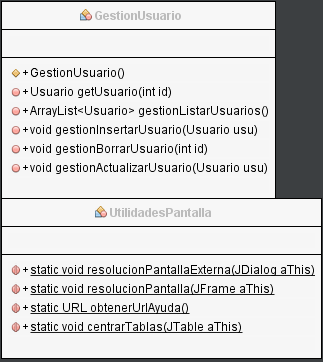
Aquí vemos la clase Main que es con la que comienza el programa, esta clase llamara al controlador



En el controlador ejecutamos el control de usuario de la aplicación con un usuario y una contraseña y que este usuario sea administrador.



En esta imagen veremos a la clase de GestionUsuario que servirá para conectar el programa con el servidor y realizar las peticiones que queramos para los usuarios.

En esta imagen podemos ver las dos clases de gestión que son GestionAlquiler y GestionPistas que tienen relación ya que se complementan una a otra a la hora de obtener las horas y las pistas. También uso métodos para rellenar las listas en la GestionPistas porque si no sería muy inútil en esta aplicación, pero lo correcto sería hacer otra clase para eso y realizar las pistas dinámicas y que el usuario pudiera añadirlas, modificarlas y borrarlas.

También tenemos una tercera clase de gestión que se llamara GestionUsuario para transmitir datos entre el servidor y el programa. Y por último tenemos UtilidadesPantalla que servirá para ejecutar métodos en todas las vistas y como son los mismo podemos usarlo varias veces escribiéndolos una sola vez lo que llamamos abstracción.

##### VISTA

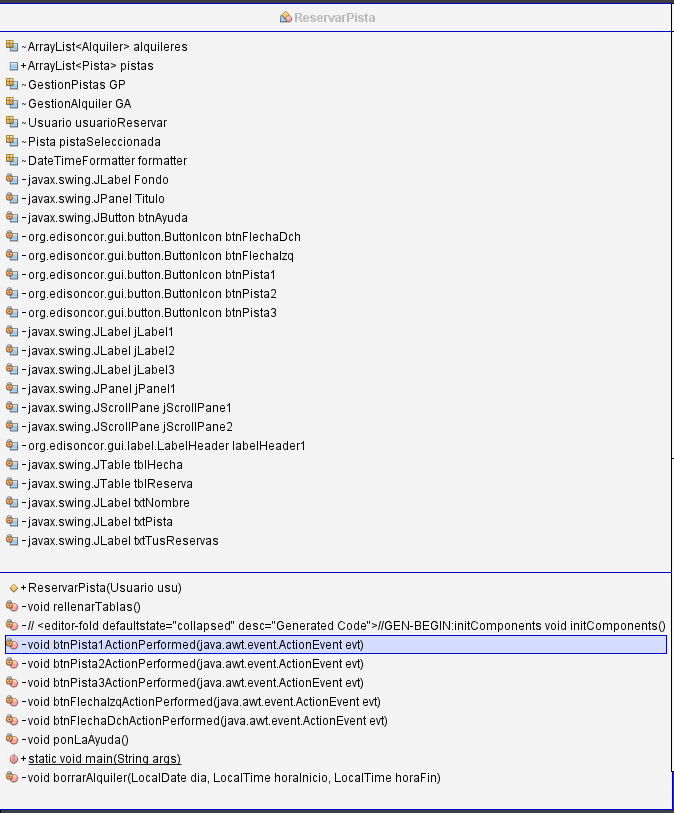
Todo este apartado estará compuesto por vistas del proyecto o lo que es lo mismo las partes visibles del proyecto y las que vera al final el cliente.

La primera vista que mostraremos será el control de usuario(login).

Aquí se puede ver como recogemos el usuario y la contraseña y lo mandamos al controlador.

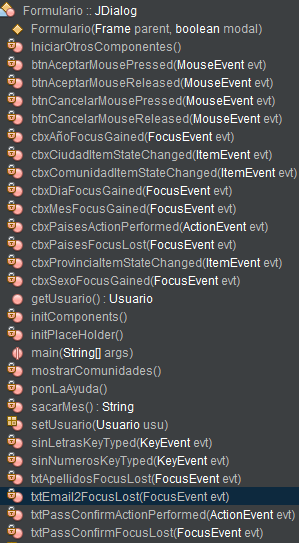
Después se abrirá la pantalla principal.

Aquí podemos ver los diferentes métodos que controlan la clase principal que llamaremos tabla, el objeto principal será el GestionUsuario que vimos en el controlador, este método permitirá tanto añadir, borrar o actualizar los usuarios. Tambien llamara a la clase de ReservaPadel al pulsar el botón de “Reservar Pista”. La clase ReservaPadel será la siguiente



Aquí accedemos a las dos clases de gestión tanto de las pistas como del alquiler. Según la pista pulsada accederemos a la clase de gestión de pistas para añadirle un nuevo modelo a la tabla y así cambiar los datos sin tener que cambiar de tabla.

Por último, pondré las ultimas vistas que son diálogos que son para meter o sacar información de los usuarios. Empezare con el más grande, el Formulario, que el propio programa no me dejo sacar el UML así que dejare una captura de todos los métodos y dejare las variables por motivos obvios.



Como podemos observar en esta clase hacemos un control de errores muy estricto y siempre que se puede en tiempo real ya que capturamos los eventos para que el usuario sepa que se está equivocando al momento que lo está escribiendo.

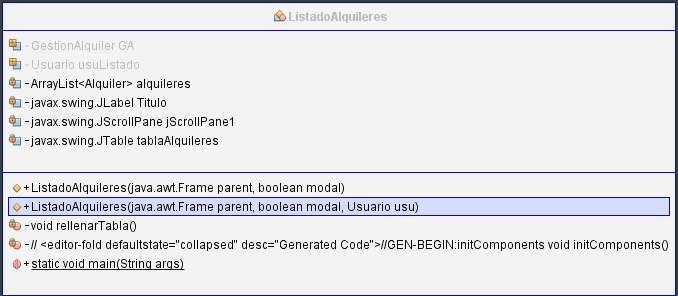
Esto es primordial ya que un fallo en la inserción de los datos puede ser muy perjudicial tanto para el programa como para el cliente ya que puede que quiera acceder al usuario y los datos estén incompletos.

También podemos ver que el único método que es público es el “getUsuario()”, ya que este método lo tendremos que llamar desde donde lo llamemos para obtener el objeto usuario que creemos en esta clase.

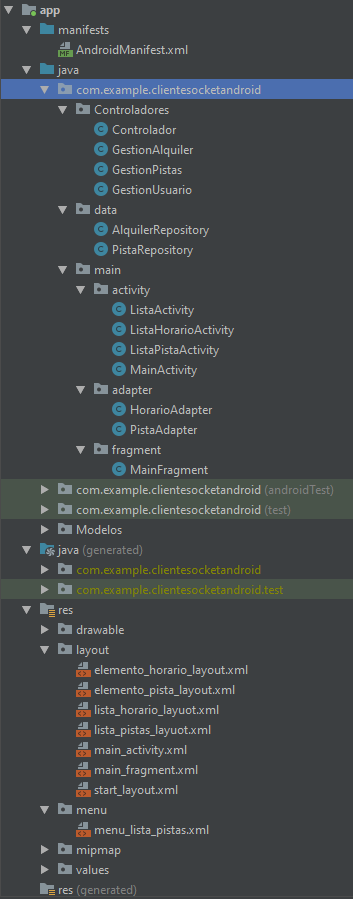
Una clase prácticamente calcada de esta última sería la de modificar.

Aquí podemos ver que es prácticamente igual, pero sin tanto controles de errores ya que rellenamos todos los campos con los datos que ya tenemos del usuario que queremos modificar y resulta más fácil para el usuario ya que no tiene que rellenar todos los campos en blanco.

Y por último tenemos un listado de todas las reservas de un usuario seleccionado, para seleccionarlo solo tenemos que hacer doble clic en la tabla de los usuarios aquí tenemos el modelo.



#### CLIENTE/ANDROID



Para empezar, explicare un poco el proceso del programa

Se trata de una app que empieza con una cabecera del programa, que al pulsarla se conecta con el servidor, si este no pasa de esa ventana significa que no tiene acceso al servidor. Después una vez conectado accederá al login de la aplicación y rellenando los datos correctos podrá acceder a las pistas, y al seleccionar una pista podrá ver las listas de horas disponibles. Al seleccionar una hora a la que alquilar te preguntara para confirmar y una vez aceptado alquilara.

Me ha sido muy difícil sacar los UML del proyecto en Android Studio por eso he subido una captura del directorio de carpetas.

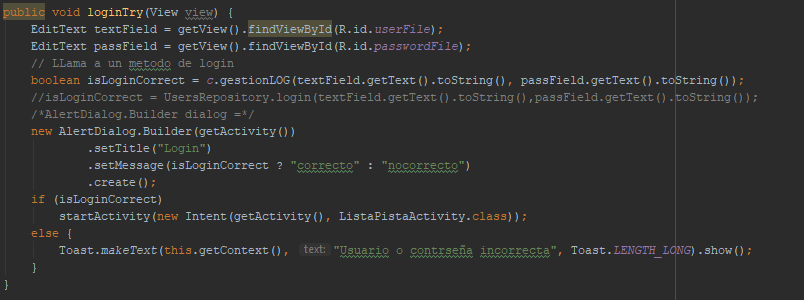
Para empezar, tendremos el AndroidManifest.xml que es donde se pedirán los permisos de accesos y se declaran las actividades. Una actividad en Android es todo aquello que el usuario puede ver a través de una pantalla y puede interactuar con ella.

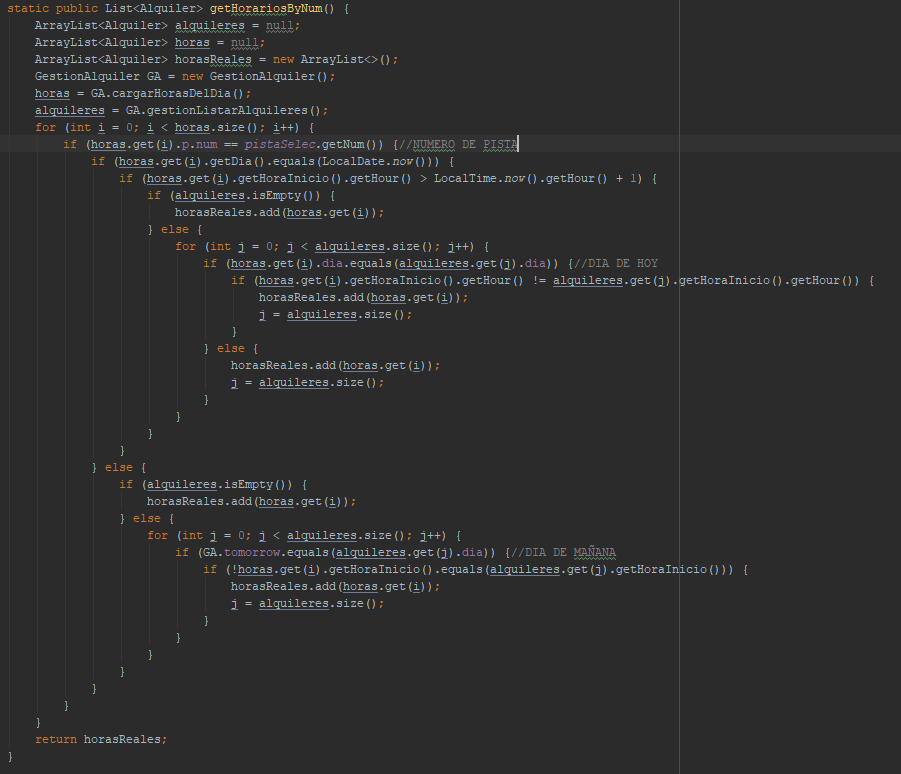
Después tenemos los paquetes de los controladores que serán los mismos que la aplicación de escritorio.

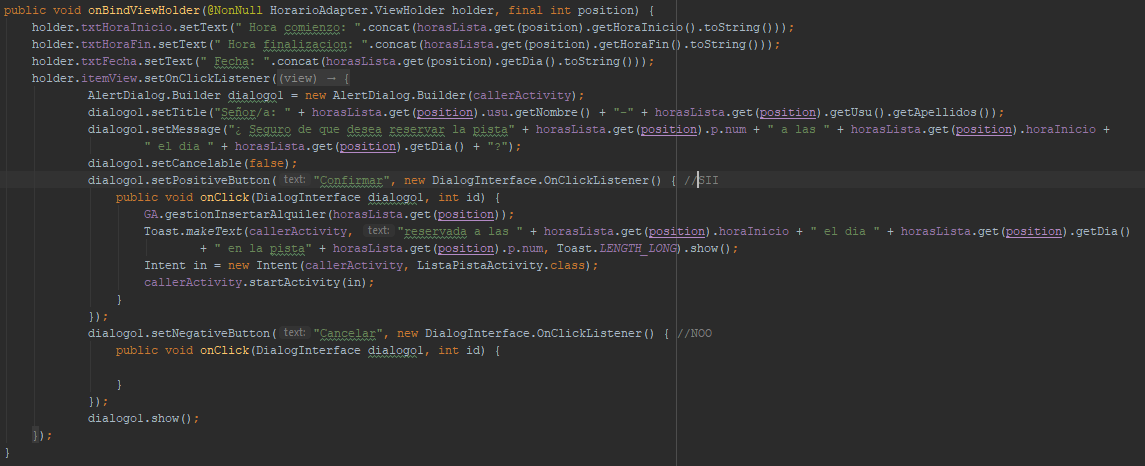
Tenemos el paquete data donde contiene los repositorios de los datos. Esto es básicamente un almacenamiento de los datos en memoria RAM que nos envía el servidor.

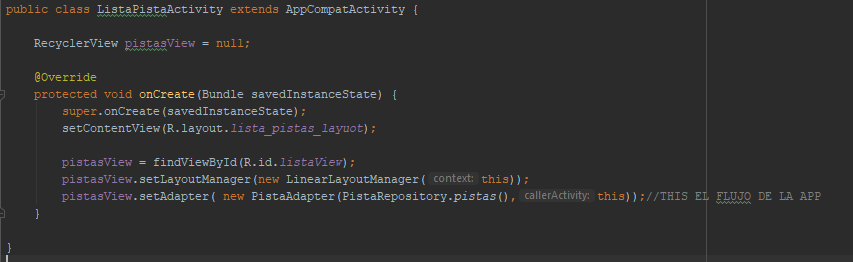
Después entramos en el main de la app donde tenemos las actividades, que son las vistas de la aplicación donde se le asignan los layout que podemos ver más abajo, tenemos los adaptadores que son los que conectan las listas que tenemos en los repositorios en la vista y por ultimo tenemos el MainFragment que es el estilo, esquema, modelo que seguirá esa lista para verse.

Y por último un poco más abajo y un poquito escondido tenemos los Modelos. Que serán los mismo que las demás aplicaciones y se explicarán en el Anexo x









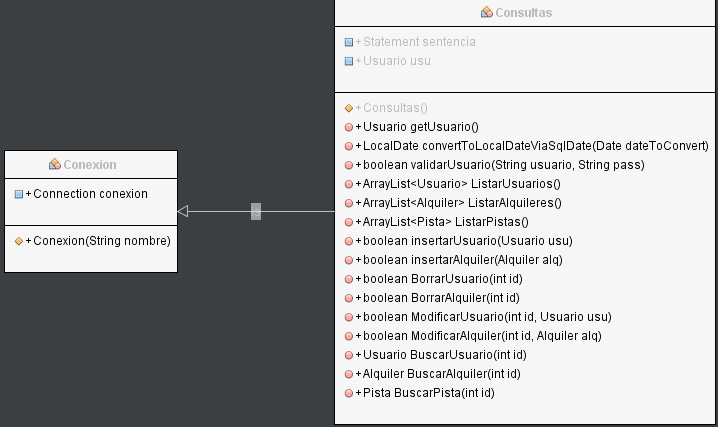
#### SERVIDOR



En este apartado veremos que solo hay una clase general movidas por hilos (*Threads*) que llamaremos hilo a cada programa ejecutado que quiera acceder a la base de datos, con lo cual hará una llamada al servidor.

Dentro del método “run()” se ve como están todos los métodos de gestión que se podrán acceder a ellos según el protocolo que envié el cliente. Los protocolos son variables globales dentro del modelo.

Para obtener algún mensaje de este servidor hay que ejecutarlo desde un terminal. Muestra los clientes que se conectan y desde la ip que viene.

También tendremos clases aparte que son utilizadas para el acceso del servidor a la base de datos.

Aquí podemos ver que siempre que se llama a la clase consulta esta crea una conexión a la base de datos y una vez obtenido los datos necesarios cierra la conexión para aprovechar al máximo el rendimiento del sistema gestor de base de datos.

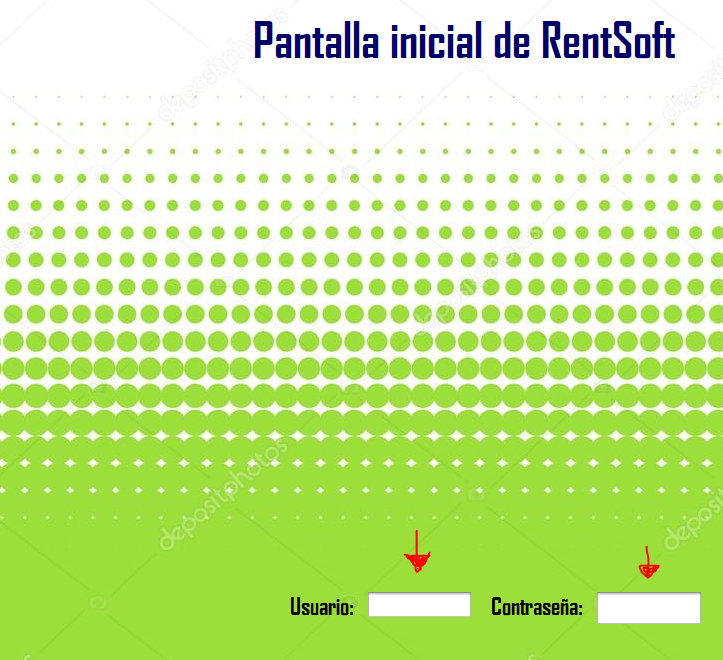
#### MODELO

En este cuadro veremos lo que contiene un objeto de cada relación de la base de datos. (Véase Anexo 1 Pág. X).

## MANUALES.

### Inicio de sesion / login

Para empezar, veremos dos cuadros de texto:



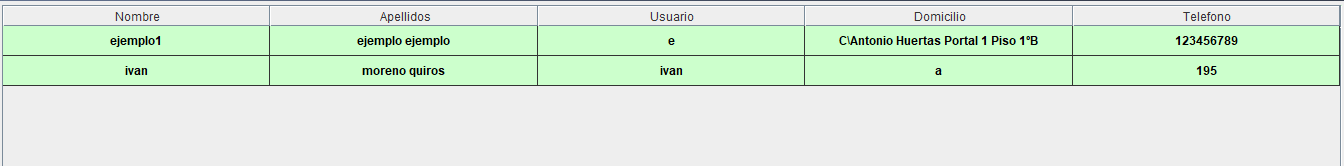
Uno para poner el nombre de usuario del administrador y otro para poner la contraseña de ese usuario, una vez insertado pulsaríamos el botón aceptar para acceder a la página principal del programa.

También dispones de un botón arriba a la derecha del programa para cerrarlo en caso de arrepentimiento.

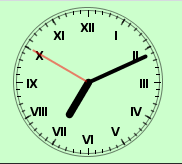


### Pantalla de gestion de usuarios.

En esta pantalla como podemos comprobar tenemos lo principal del programa que son los usuarios.



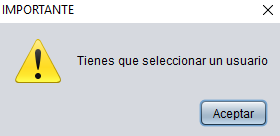
Cuando pulsamos dos veces en un usuario nos muestra todas las veces que ha reservado una pista para que sepamos la experiencia de este usuario con la empresa. Te muestra la pista en la que estaba la hora a la que la cogió y el día. Podemos pulsar en el título de la tabla para ordenarlos.



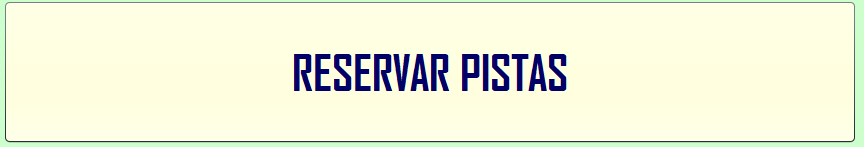
Tenemos un reloj que marca la hora para cuando se deje el programa abierto y no perder la noción del tiempo. En principio quería implementar que el programa te avisase cuando llegue la hora de cualquier sesión. Pero necesito hacer un minutero constante en el programa y no dispongo de esos conocimientos ni tiempo para aprenderlo.

Tenemos también el botón de ayuda en la parte inferior derecha de la ventana. Esta ayuda esta implementada en todo el programa al pulsar F1 en cualquier ventana para guiarte.

Tenemos tres botones abajo a la izquierda para la gestión de usuarios. Siempre que pulsemos uno de estos botones tendremos que seleccionar antes un usuario de la tabla menos en el caso de añadir que no hará falta tener seleccionado nada (Todo esto está controlado).

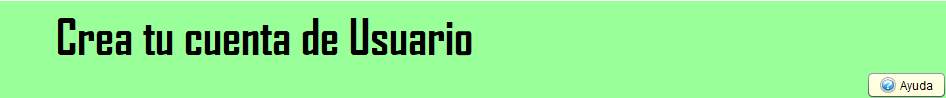


Y por último tenemos el botón de “RESERVAR PISTAS” en la parte inferior de la pantalla, el cual tendremos que seleccionar un usuario de la tabla y pulsar el botón para que se abra la ventana de las pistas.



#### Insertar

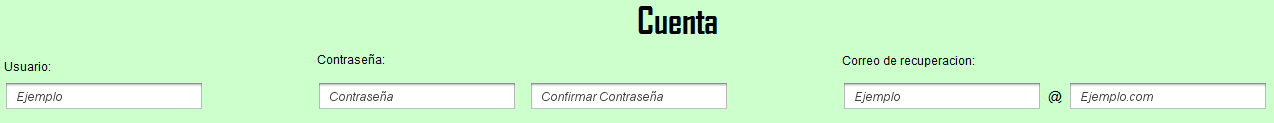
Cuando se pulsa el botón “Insertar” se abre un formulario de ingreso con muchos datos con su correspondiente control de errores. También tiene metida una clase que se llama “TextPrompt” que hace que se muestre un ejemplo de lo que hay que ingresar para que no de fallos nada. Disponemos también de un botón de ayuda en la parte superior izquierda de la pantalla por si tenemos alguna duda.



En este apartado tenemos tres fases:

##### Cuenta

Aquí ahí que ingresar todos los datos de la cuenta para acceder luego desde esta o otra aplicación como la de Android.



Cabe destacar que al poner mal la segunda contraseña saltara automáticamente el título en rojo indicando que las contraseñas no coinciden. Controles de errores como este tiene en todo el formulario.

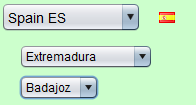
##### Datos personales

Aquí ahí que ingresar todos los datos personales de usuario que queremos insertar por si queremos ponernos en contacto con él o hay algún problema.





Cabe señalar que si se modifica el país se cambiara la banderita. Y si seleccionamos España nos mostrara automáticamente otro combo para seleccionar la comunidad autónoma.



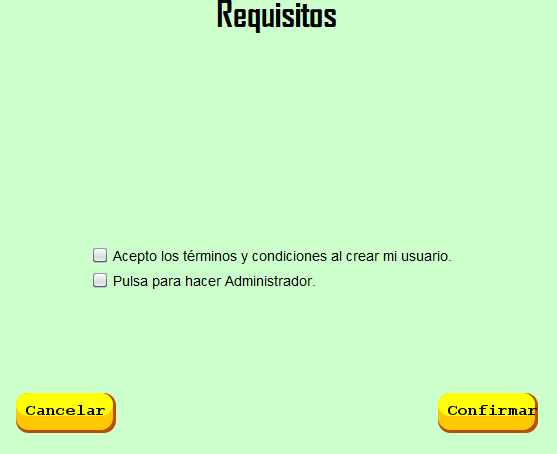
Y una vez seleccionada la comunidad autónoma nos mostrar otro combo con todas las provincias de esa comunidad autónoma seleccionada.



Y una vez seleccionada la provincia nos mostrara todas las ciudades de esa provincia.

##### Requisitos

En esta sección tenemos que aceptar que se puedan usar esos datos para el beneficio de la empresa y también si queremos si ese usuario pueda acceder al administrador o solo sea un cliente.



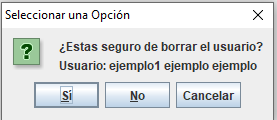
Cabe destacar que los botones están creados por el programador y están insertados en el proyecto a través de un “.jar” llamado “btn.jar” que una vez añadido a la paleta de NetBeans tiene una serie de métodos para modificar su comportamiento lo que hace que al seleccionarlos de una sensación de profundidad.

Al pulsar “Confirmar” se validan todos los campos. Y en el caso de pulsar “Cancelar” se cancela el formulario y te vuelve a la página principal.

#### Modificar

Cuando se pulsa el botón “Modificar” se abre un pequeño formulario con los datos rellenados, aquí no hay muchos controles de errores ya que en principio solo se querría cambiar algún dato erróneo o equivocado.

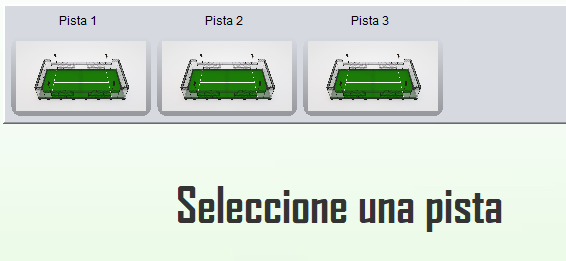
#### Borrar



Cuando se pulsa el botón de “Borrar” se abre un dialogo que te pregunta si de verdad quieres borrar el usuario y si se acepta lo borrara de todos lados.

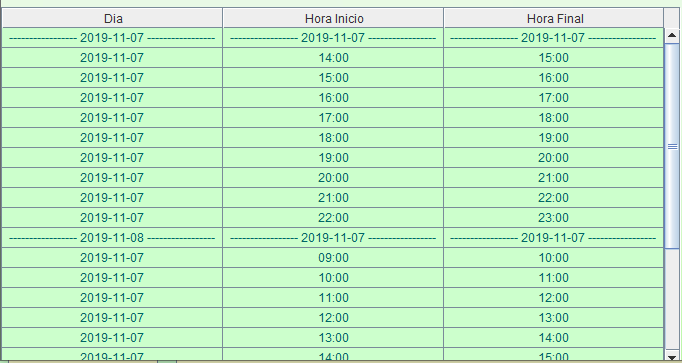
### Reservar Pista

En este apartado podemos ver las diferentes pistas. Dejamos un espacio en la parte derecha para que se puedan añadir más en caso de necesitarlo. Como se puede ver en la siguiente imagen son pistas de Pádel y por eso están representadas con ese fondo.



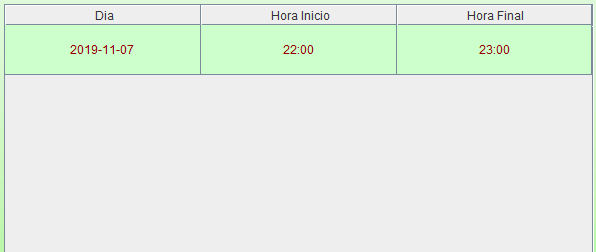
Cuando pulsamos una de las pistas tenemos un título justo debajo que nos indicara en que pista estamos actualmente y sepamos donde vamos a alquilar.

Según la pista que pulsemos se mostrara los horarios disponibles de esa pista seleccionada. También tendrá en cuenta la hora del día en la que se encuentra para que si se abre el programa a una hora no te muestre horarios anteriores del mismo día ya que sería totalmente imposible alquilarla.



Pulsando una hora de la tabla y pulsando el botón con una flecha a la derecha se pasará esa hora a la base de datos dando por hecho que el usuario seleccionado anteriormente hizo la reserva.

En la otra tabla más a la derecha se mostrará las horas que el usuario a reservado. Y pulsando el botón con la flecha hacia la izquierda se borrará la hora que tiene reservada y la devolverá a la tabla original y la borrará de la base de datos.



También se mostrará el usuario que al cual se le está haciendo la reserva.



También contiene el botón de ayuda en la parte derecha de la pantalla.

## CONCLUSIONES

Según las pruebas de campo y las investigaciones realizadas con el presente trabajo se concluyen las siguientes conclusiones:

* 1. Este sistema es el más indicado para sacarle rendimiento a un club o ente deportivo ya que permite optimizar las horas del día y obtener los datos de los que alquilan las pistas por si fallan tomar las medidas oportunas.
  2. Comparándolo con los sistemas actuales en el mercado, este programa podría llegar a representar un proyecto comercial rentable y viable.
  3. Este programa es un adelanto dentro de la gestión clásica de recopilación de documentos impresos: es más fiable, es más fácil de usar, de fácil aprendizaje, a largo plazo es más barato y es sostenible.
  4. Permite el alquiler de las pistas desde un terminal Android y así ahorrar tiempo en tele operaciones.

## PROPUESTA DE AMPLIACIÓN.

### INTERACTIVIDAD

Para ampliar este proyecto propongo darle el poder para añadir, modificar o borrar las pistas y diferenciarlas por tipos en el cliente de la administración, así se podría hacer totalmente interactivo y no dependería tanto del programador para cambiar las cosas.

También pondría algo para elegir las horas a las que se quiere abrir y cerrar y algún método para que el administrador pudiera poner las horas a la que se quieren reservar las pistas.

### AMBICION

También sería recomendable acceder a terminales IOS.

Sería muy optimo llegar a subir todo esto a un programa web y reservar desde ahí, pero sería muy poco productivo ya que perdería esa confidencialidad de un club. Habría que estudiarlo detenidamente con la empresa a la que se venda.

### CONTROL

En este programa añadiría un sistema de control del personal para que el sistema llevara las horas del personal que está trabajando en esa empresa.

### SEGURIDAD

Hice una pequeña prueba para intentar cotejar las contraseñas a un algoritmo llamado “mb5” pero e estado observando en la empresa en la que estoy que cualquiera puede tener acceso a una contraseña hecha con ese algoritmo así que mi proposición es hacer un nuevo algoritmo muy similar a ese u otro y así proteger el servidor y el programa en general.

También vería factible hacer más validaciones a la hora de las gestiones de los programas. Y mandar un mensaje en cada interacción para fortalecer la seguridad.

## BIBLIOGRAFÍA.

Casi todo lo insertado en este proyecto fue aprendido de un profesor llamado Pedro que trabajaba en Don Benito en el instituto I.E.S Donoso Cortes.

También hice un curso en Udemy de Android y manejo de Android Studio: <https://www.udemy.com/course/programacion-de-android-desde-cero/>

Para la descarga de JavaHelp: <https://jar-download.com/artifacts/javax.help/javahelp>

Para la descarga de EdisonCorX: <https://sourceforge.net/projects/edisoncorsx/>

## ANEXOS.

### ANEXO 1