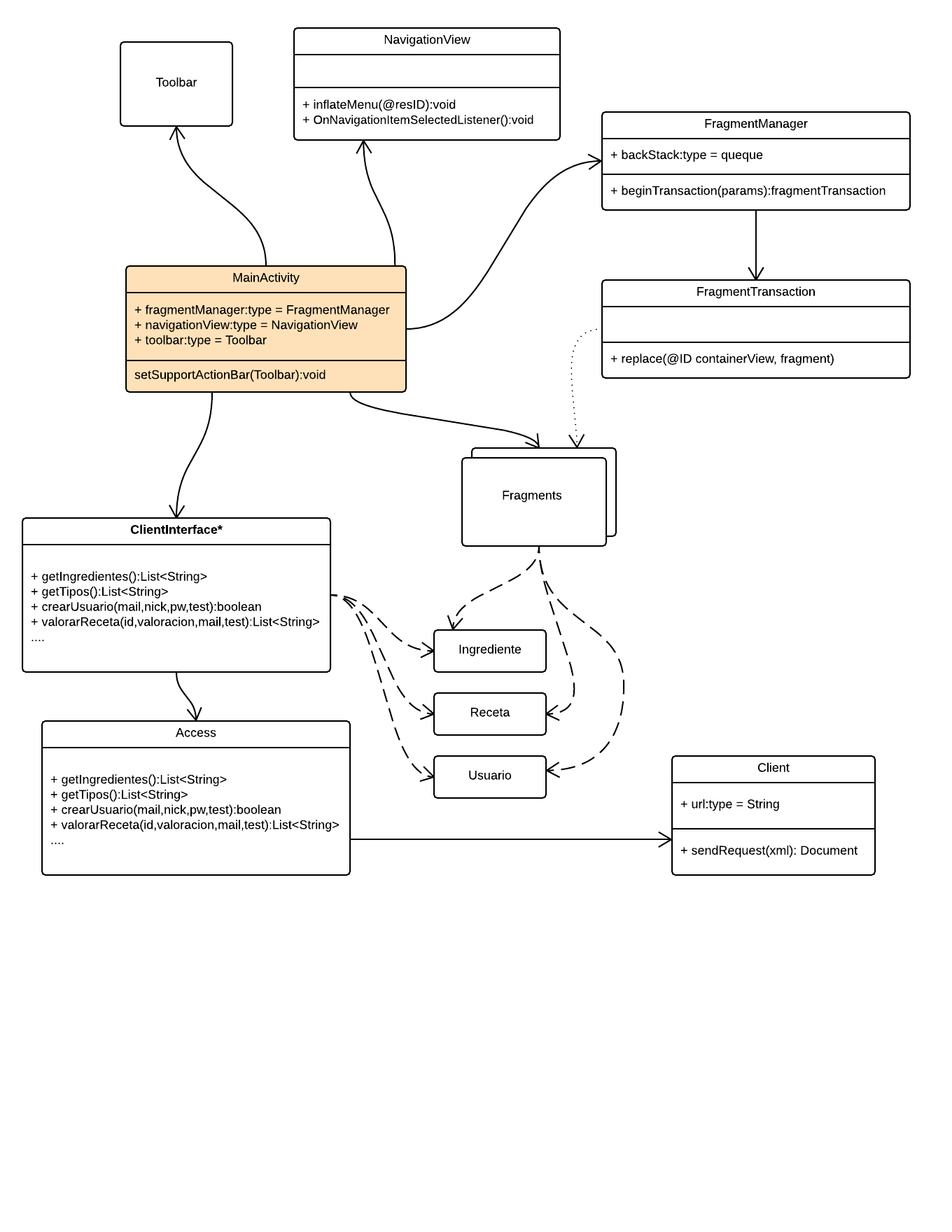
APLICACION



Tal y como muestra el diagrama, la actividad principal de la aplicación corresponde a la clase MainActivity. Esta clase gestionará toda la arquitectura de la aplicación, formada por 4 partes:

CONTENIDO PRINCIPAL

La aplicación tendrá contenidos dinámicos, y deberá permitir una navegación fluida entre las pantallas, es por eso que se ha decido usar una estructura de Fragments (<http://developer.android.com/intl/es/training/appbar/index.html>) Esta tecnología permite modificar únicamente una parte de la Actividad principal en la que se encuentre el usuario, de esta forma evitamos volver a cargar elementos estáticos de la aplicación, como es el Menú Lateral y la Barra de Aplicación que se describirán a continuación. A través de la clase FragmentManager se mantendrá una pila (backStack) almacenando en orden todos los fragments por los que ha navegado el usuario, de esta manera al pulsar el botón atrás, la aplicación volverá al Fragment anterior, y no saldrá de la aplicación, por lo que la experiencia de usuario se verá mejorada considerablemente. Por otro lado, para hacer la transacción entre Fragments se hace uso de la clase FragmentTransaction junto al método replace.



MENU SUPERIOR

La zona superior de la aplicación estará ocupada por una AppBar (<http://developer.android.com/intl/es/training/appbar/index.html>) Esta barra de acciones estará implementada a través de un objeto Toolbar, que muestre las opciones necesarias. Se ha decidido usar esta implementación ya que así, es muy sencillo modificar las opciones disponibles en esta barra a través del Fragment en el que se encuentre el usuario.



MENÚ LATERAL

Para la implementación del menú lateral se ha usado la clase NavigationView (<http://developer.android.com/intl/es/reference/android/support/design/widget/NavigationView.html>) Esta clase permite cargar las opciones del menú lateral a través de un .xml con el método (inflateMenu()), el cual puede ser modificado para mostrar distintas opciones según sea necesario, por ejemplo, modificar las opciones según el usuario esté registrado o no. Además, permite instanciar un ActionListener para cada opción del menú.

CONEXIÓN CON LA BD

Para permitir el trabajo concurrente entre el equipo backend junto el frontend, se han separado las clases necesarias para realizar la conexión, asi como las clases que contienen los métodos que realizan las conexión en un paquete aparte con nombre “connection”, de esta forma se divide el código evitando mezclas entre el código perteneciente a la GUI y la BD.LA aplicación usará únicamente los métodos de la clase ClientInterface. Dicha clase ejecutará los métodos de la clase Access, que hará uso de la clase Client para enviar y recibir peticiones al servidor.

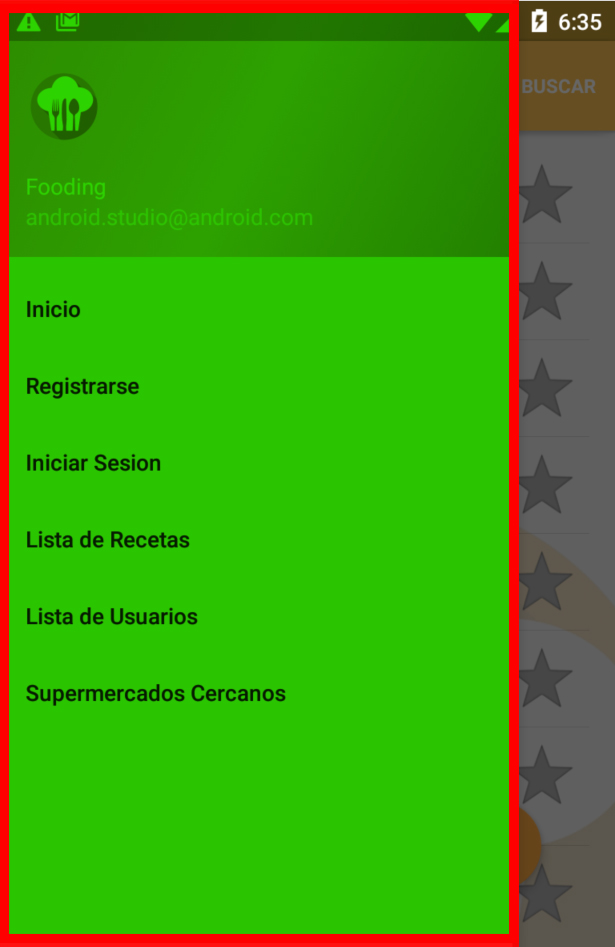
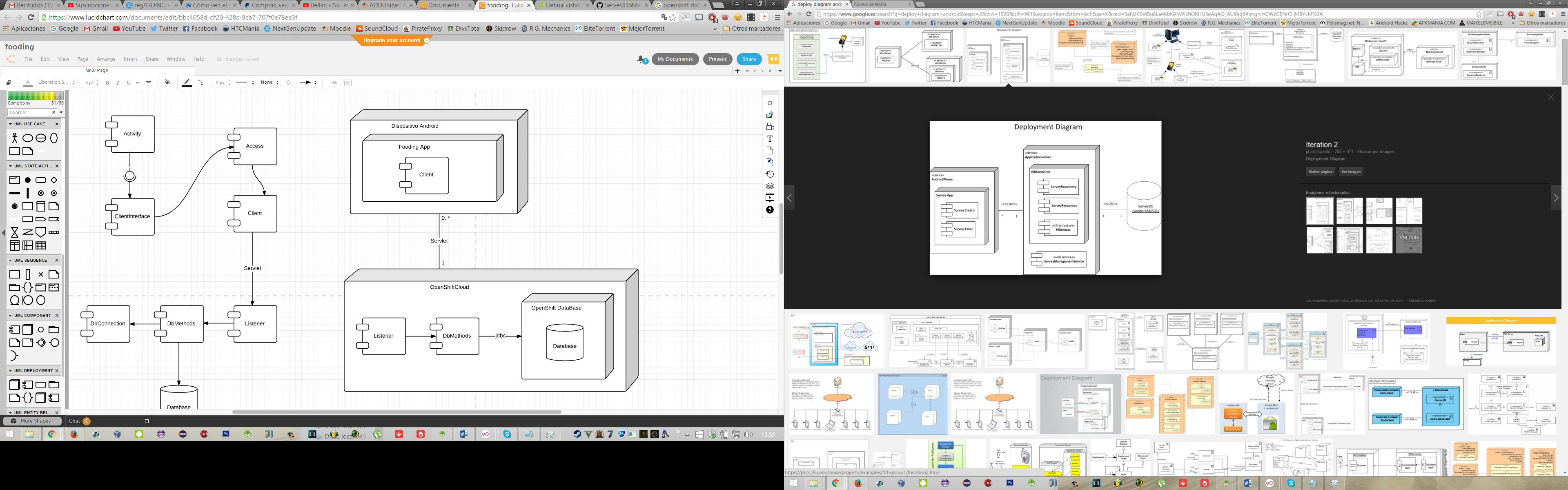


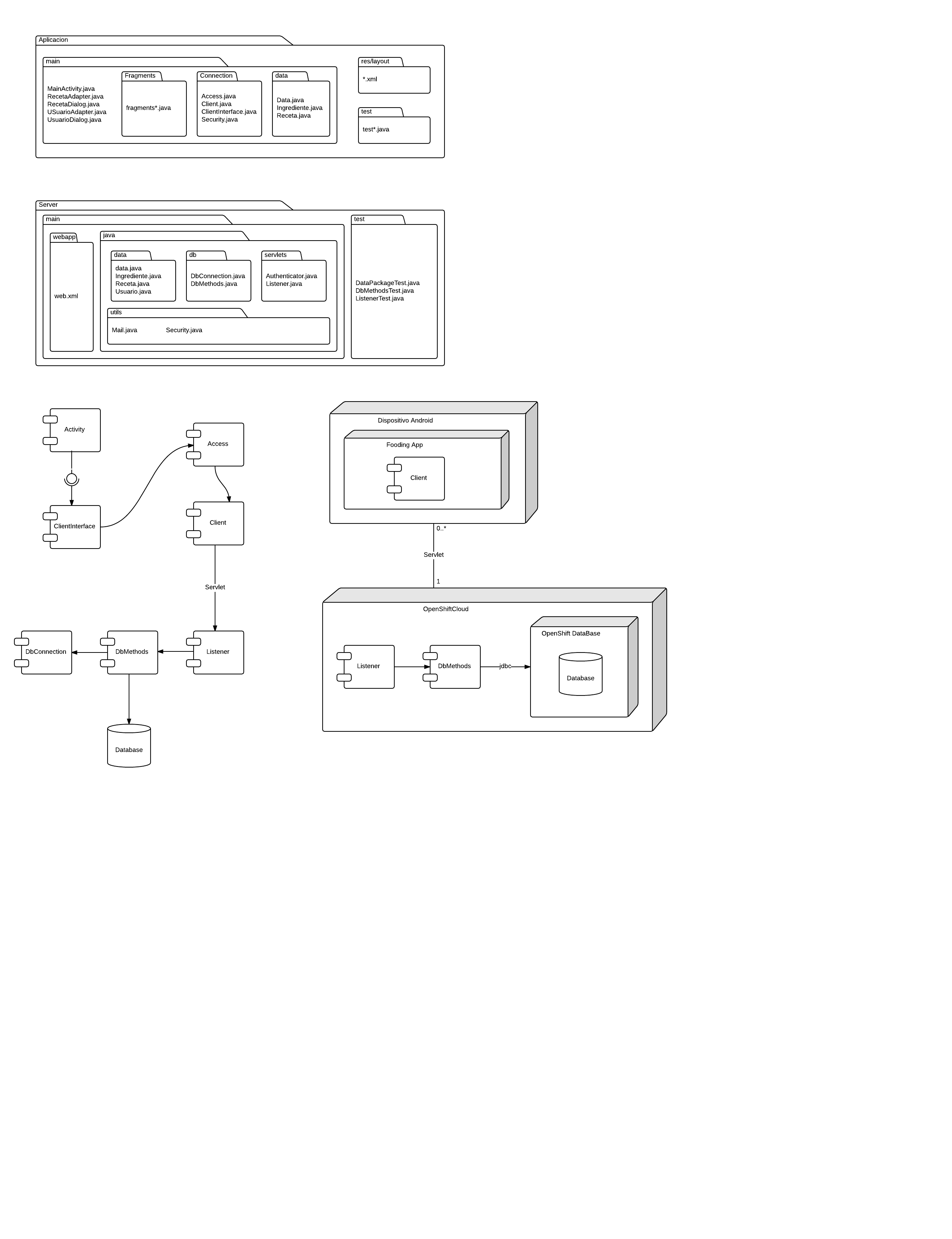
DIAGRAMA DESPLIEGUE



El sistema se compone de dos nodos. El cliente que ejecutará la aplicación en su disposivo Android, y por otro lado el servidor en OpenShift. Este servidor en la nube tendrá un componente Listener, que esté escuchando peticiones de todos los clientes a través de servlets. Se ha decido usar esta comunicación ya que la aplicación no requiere muchas peticiones al servidor, y no se prevee una gran cantidad de usuarios concurrentes. En todo caso, si llegara a dar problemas se podrían replicar los listener y por medio de una cola y redirigiendo a las peticiones según vayan siendo gestionadas.

Por último la base de datos, que ha sido alojada junto al servidor en el mismo “Gear” de Openshift. El listener hará uso de los métodos de DbMethods, el cual accederá a la base de datos a través de jdbc.

DIAGRAMA PAQUETES



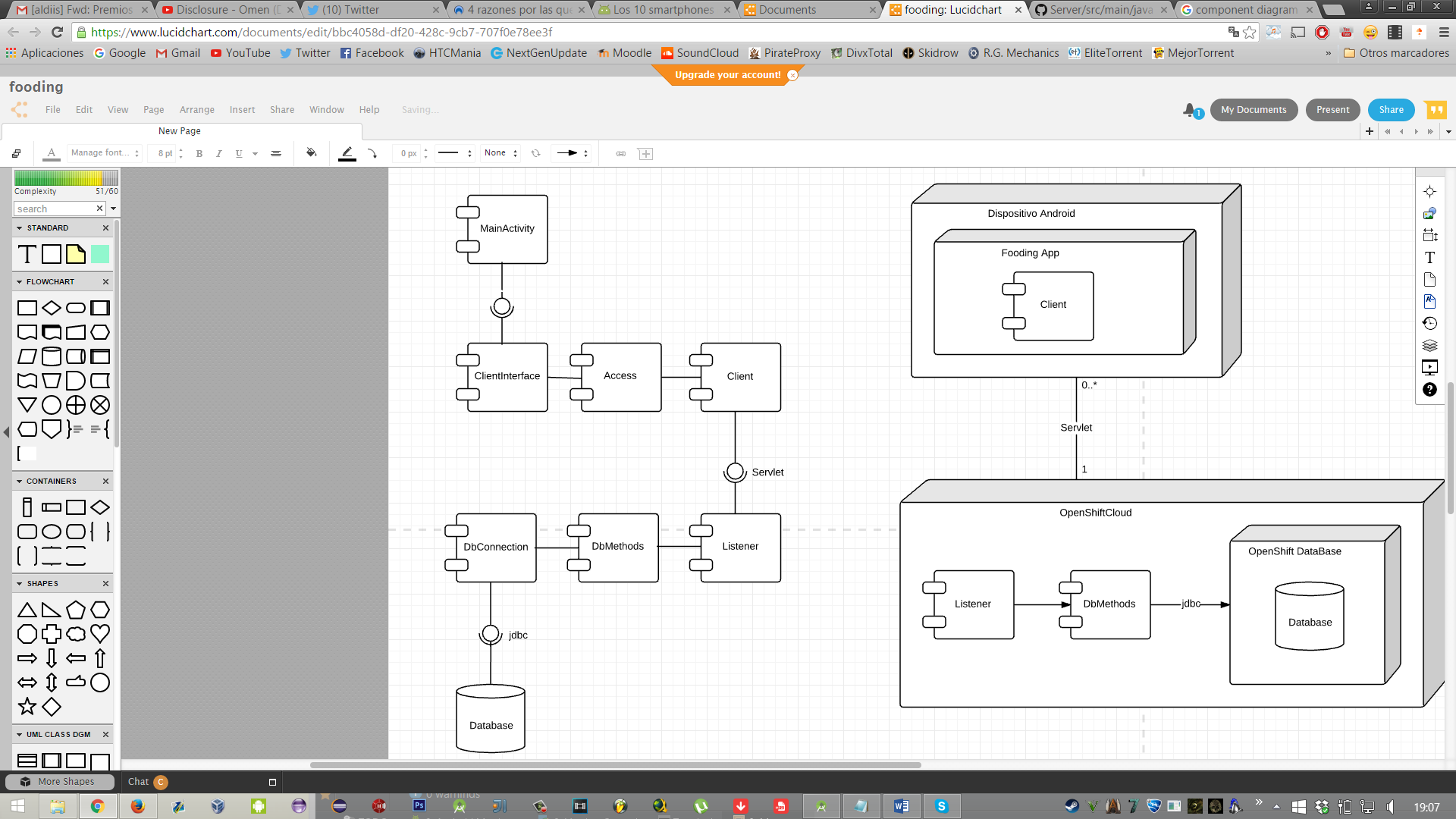
Se ha modificado el esquema de paquetes en la parte del cliente respecto al primer Sprint, ya que la complejidad de la aplicación hacía que el número de clases creciera muy rápidamente y esto dificultaba el desarrollo.

Ahora está organizado en 4 directorios:

* Data: lugar donde almacenar los objetos con los que se va a trabajar
* Connection: clases encargadas de realizar la conexión con la base de datos y enviar/recibir peticiones
* Fragments: clases que extiendan de la Clase Fragment de Android, es decir, cada pantalla
* Raíz: resto de clases, adapters o diálogos, además de la Actividad principal “MainActivity”.

Respecto a la parte del servidor, se mantiene la misma estructura, sin cambios.

DIAGRAMA COMPONENTES



A través del diagrama de componentes se ha representado el flujo que recorre la aplicación y grafo que forman sus componentes. El usuario interacciona directamente con el MainActivity, ya sea desde los menús o desde el fragment en cuestión. Este trabajo con la interfaz que ofrece la clase ClientInterface:

\*\*\*\*\*\*\*\*\*\*\*\*\*METER METODOS Y DESCRIPCION

Esta clase, usa los métodos implementados en Access, que junto al componente “Cliente” realiza una conexión con el Listener del servidor. Esta conexión se realiza a través de servlets. El listener, de manera similar al cliente, hace uso de otros dos componentes, para realizar las consultas a la base de datos a través de jdbc.