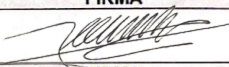




Proyecto Final de Carrera Ingeniería Informática	Informe de estado del proyecto	FICH	UNL
---	--------------------------------	------	-----

REALIZADO POR	FECHA	FIRMA
Facundo Salmerón	15/04/18	
REVISADO POR	FECHA	FIRMA
Emmanuel Rojas Fredini	15/04/18	
REVISADO POR	FECHA	FIRMA
Horacio Sagardoy	15/04/18	
APROBADO POR	FECHA	FIRMA
Lucila Romero		

Nombre del Proyecto: DESARROLLO DE APLICACIÓN MÓVIL PARA EL DISEÑO Y PROCESAMIENTO DE ENCUESTAS PÚBLICAS SOBRE SALUD.

Periodo del Informe: 16/03/2018 – 11/04/2018

Alcance: Etapa 2

Proyecto Final de Carrera Ingeniería Informática	Informe de estado del proyecto	FICH	UNL	
---	--------------------------------	------	-----	--

Estado del Proyecto					
Cronograma	Etapa 2: Diseño.	Actividad	Fecha realización		<i>Resultados obtenidos</i> Se deja un anexo con un detalle de las actividades realizadas.
			Estimada	Real	
		2.1. Diseño de la base de datos.	11/10/17	18/03/18	
		2.2. Diseño del Web Service.	17/10/17	24/03/18	
		2.3. Diseño de la Aplicación.	26/10/17	29/03/18	
		2.4. Diseño de la Interfaz Gráfica.	03/11/17	01/04/18	
		2.5. Diseño de la Integración de Módulos.	07/11/17	03/04/18	
		2.6. Desarrollo del documento de diseño.	21/11/17	11/04/18	

Proyecto Final de Carrera Ingeniería Informática	Informe de estado del proyecto	FICH	UNL	
---	---------------------------------------	-------------	------------	--

Riesgos	Riesgo	Se efectivizó		Impacto	Mitigación
	R001: Falta de disponibilidad del director.		No		
	R002: Retrasos en los entregables.	Si		Debido que el ejecutor del proyecto tomó un empleo de 8 horas diarias y debido a situaciones de cursado en la carrera, el desarrollo de la etapa de diseño se vio postergado.	
	R003: Indisponibilidad de los recursos.		No		
	R006: Cambios en los requerimientos.		No		
	Riesgos futuros			Probabilidad	Impacto
	Falta de disponibilidad del recurso humano del proyecto			Alta	Medio
Notas	Como mitigación para solventar los retrasos producidos en el inicio del proyecto, se comenzó a trabajar más horas de las previstas por día y tiempo completo los fines de semana y feriados.				



UNIVERSIDAD NACIONAL DEL LITORAL
Facultad de Ingeniería y Ciencias Hídricas

PROYECTO FINAL DE CARRERA
INGENIERÍA EN INFORMÁTICA

**Desarrollo de aplicación móvil para el
diseño y procesamiento de encuestas
públicas sobre salud.**

ETAPA 2: DISEÑO

Alumno: Salmerón Facundo

Director: Rojas Fredini Emmanuel

Co-Director: Sagardoy Horacio

Santa Fe, Abril de 2018

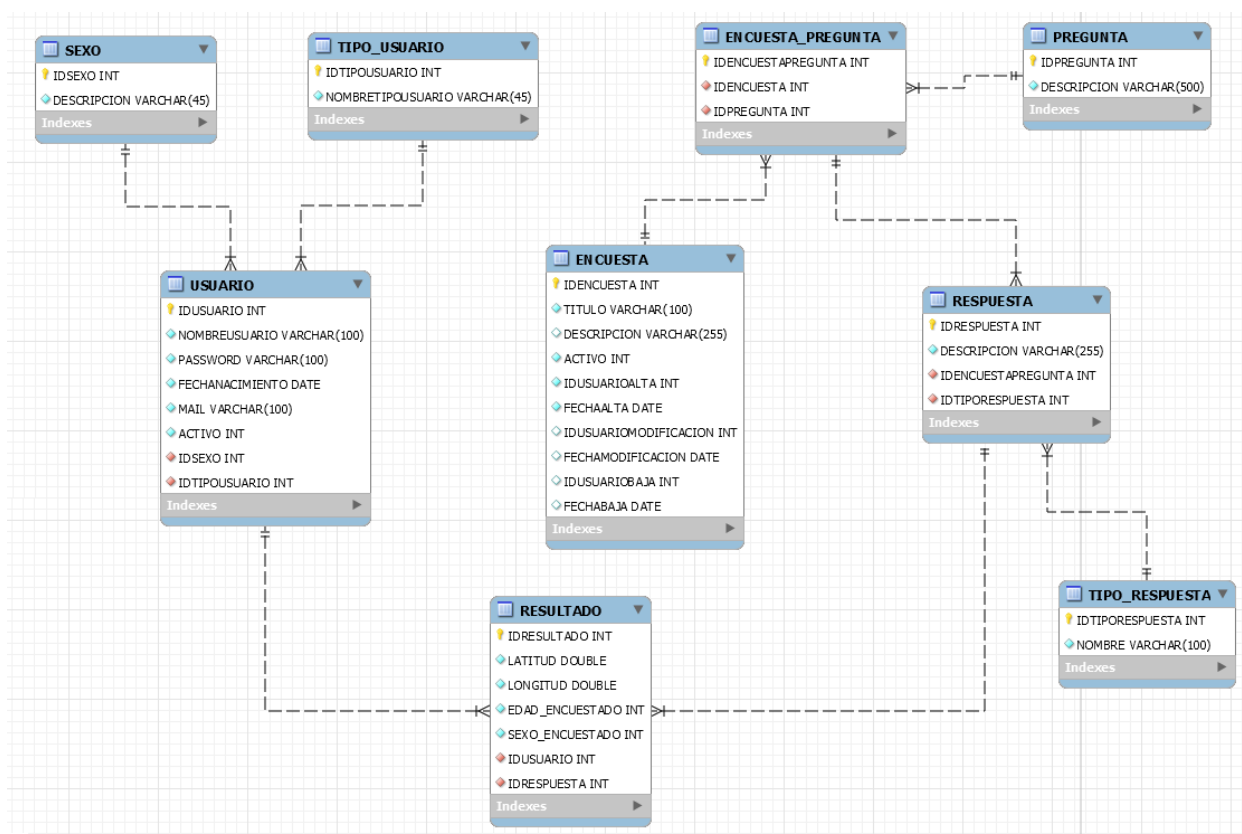
INFORME DE AVANCE N° 2:

En el siguiente anexo se detallan los resultados obtenidos de las tareas realizadas dentro de la segunda etapa que constituye el ciclo de vida del proyecto, denominada de Diseño.

DISEÑO DE LA BASE DE DATOS:

Debido a los requerimientos obtenidos en la etapa de análisis, es necesario el uso de un sistema de almacenamiento para poder manejar toda la información con la que la aplicación va a trabajar. Los datos a guardar serán relacionados a los usuarios que van a interactuar con el sistema y las encuestas en sí. Los mismos luego serán accedidos gracias al Web Service.

A continuación, se detalla el **diagrama de tablas** de la base de datos de forma relacional, implementada en el servidor:



(Fig. 1 – Diagrama de Base de Datos)

DESCRIPCIÓN DE LAS TABLAS:

Cabe aclarar que el diseño presentado es una aproximación inicial, donde el mismo podría sufrir cambios a futuro a medida que se avanza la etapa de desarrollo.

USUARIO: ésta tabla va a almacenar los datos de los usuarios que van a interactuar con la aplicación. La misma posee el nombre del usuario, la contraseña, fecha de nacimiento, mail y su estado, es decir si se encuentra activo o no. A su vez posee

claves foráneas en relación a las entidades SEXO y TIPO_USUARIO. Ésta última tabla representa los tipos de usuario que se pueden escoger, como lo serán por ejemplo “Usuario Específico” y “Usuario Común”.

ENCUESTA: en ésta tabla se van a representar las encuestas que se van a llevar a cabo. Las mismas poseen como datos un título, una descripción, el estado (si la encuesta se encuentra activa o no), y un registro sobre los ABM que se producen sobre cada encuesta, es decir mediante el idUsuario y la fecha (de Alta, Baja y Modificación), llevar a cabo un control de las mismas.

PREGUNTA: aquí se van a almacenar todas aquellas preguntas a incluir dentro de una encuesta. Las mismas poseen una descripción, que sería la pregunta en sí.

ENCUESTA_PREGUNTA: como la relación que se produce entre ENCUESTA y PREGUNTA es Many To Many (muchos a muchos), los datos de éstas entidades se van a representar mediante ésta tabla, almacenando de tal forma el id de la encuesta y el id de la pregunta.

RESPUESTA: a su vez, cada pregunta va a poseer una o más respuestas (dependiendo del tipo de respuesta). Es por ello que en ésta tabla se almacenan los datos relacionados a las posibles respuestas (descripción) y como claves foráneas el id de la encuesta y pregunta a la que pertenece, por lo tanto, se produce la relación entre la presente tabla y ENCUESTA_PREGUNTA. Además, cada respuesta va a poseer una clave foránea hacia la entidad TIPO_RESPUESTA.

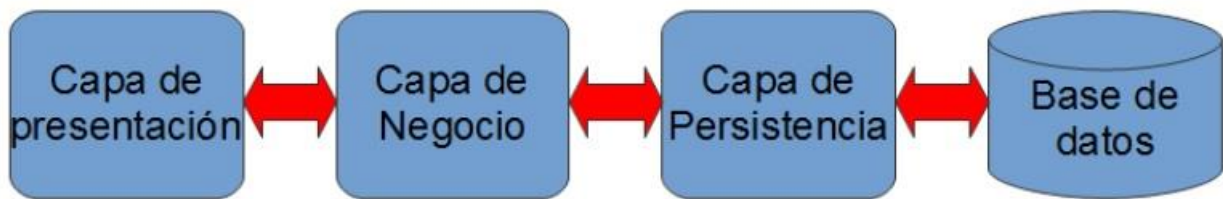
TIPO_RESPUESTA: en ésta tabla se van a representar los tipos de respuestas que van a ser posibles en una pregunta. Por ejemplo, pueden ser “Múltiple Choise”, “Libre”, “Numérica”, etc.

RESULTADO: aquí se van a almacenar las resoluciones de las encuestas. Para obtener todos los datos relacionados a la solución de la misma, se mapea la relación de la entidad con la tabla RESPUESTA, obteniendo las claves foráneas necesarias a almacenar como encuesta, pregunta y respuesta.

A su vez, se posee una relación con la tabla USUARIO, almacenando el id de la misma como clave foránea. Es decir, con la resolución de la encuesta se almacenará el usuario que la resolvió. Si se tratara de un usuario del tipo común, además de saber el encargado de la resolución, de aquí se obtendrán los datos de edad y sexo del encuestado, mientras que, si la resolución fuera por parte de un usuario específico, estos datos serán obtenidos de distinta manera a la hora de la resolución. También, esta entidad posee una latitud y longitud, las cuales van a almacenar la ubicación donde se resolvió la encuesta.

DISEÑO DE LA ARQUITECTURA:

La arquitectura del sistema escogida será la de **división por tres capas**. El principal objetivo de este modelo de desarrollo de software es la separación de las partes que componen una arquitectura del tipo cliente – servidor. Con esta arquitectura el software se divide en tres niveles diferentes: **capa de presentación**, **capa de negocio** y **capa de datos**. Cada nivel se desarrolla y mantiene como una capa independiente.



(Fig. 2 – Arquitectura del sistema)

- **CAPA DE PRESENTACIÓN:**

Es el nivel superior de la aplicación, básicamente con la que interactúa el usuario, por ello es llamada también “*capa de usuario*”. Esta capa proporciona la interfaz de usuario de la aplicación y además le comunica la información y captura datos del usuario. Ya que se trata de la capa que tiene relación directa con el usuario, la misma debe presentar una interfaz de uso intuitivo, sin mayores complejidades. Dicha capa en la aplicación estará compuesta por un conjunto de archivos en formato XML, que es el formato estándar para el diseño de interfaces en Android. Se comunica con la capa de negocios. Mientras que en el Web Service se implementará mediante controladores (detallado más adelante).

- **CAPA DE NEGOCIO:**

Es la capa que contiene los procesos a realizar (reglas de negocios) con la información recibida desde la capa de presentación. Se encarga de realizar toda la lógica de negocio. Trata las peticiones que el usuario ha realizado y se encarga de enviarle una respuesta hacia la capa de presentación. Esta capa intermedia se encuentra relacionada con la capa de presentación y la capa de persistencia. En la relación con la capa de presentación, como ya se mencionó se comunica para recibir solicitudes y presentar resultados, mientras que en la comunicación con la capa de persistencia se pueden recuperar datos o persistirlos en la base de datos, así como también procesar la información que proviene de la capa de datos.

- **CAPA DE PERSISTENCIA:**

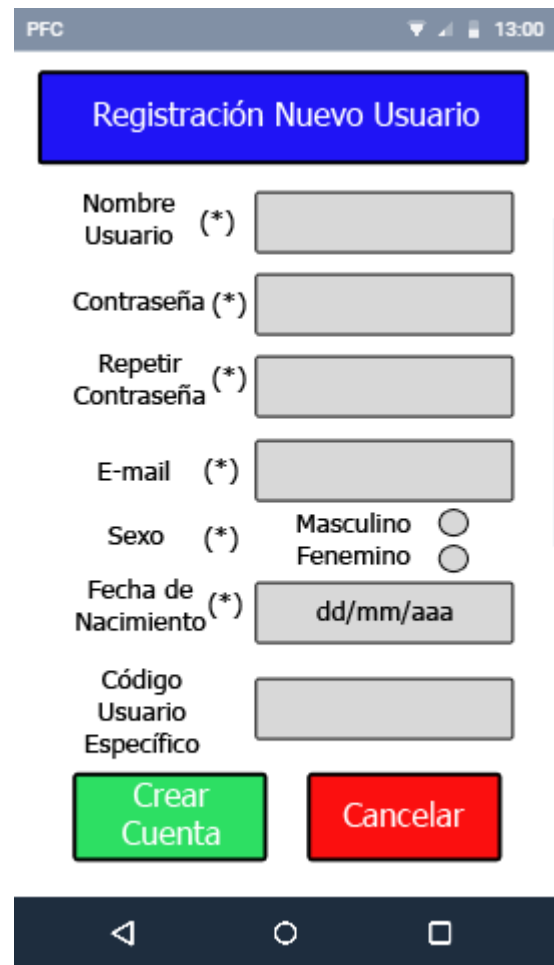
Es también denominada como “capa de datos”. Es la última capa encargada de proporcionar una biblioteca de funcionalidades para el acceso a la base de datos, lugar en el que residen los datos. Está formada por el gestor de base de datos, que permite el acceso a los datos para su consulta, actualización, almacenamiento o eliminación. Se comunica con la capa de negocios facilitando datos o recibiendo los.

DISEÑO DE LA INTERFAZ GRÁFICA:

A continuación, se presenta un diseño (mockups) de las pantallas que componen la aplicación y una breve explicación de las mismas, en relación a lo establecido en los requerimientos funcionales.



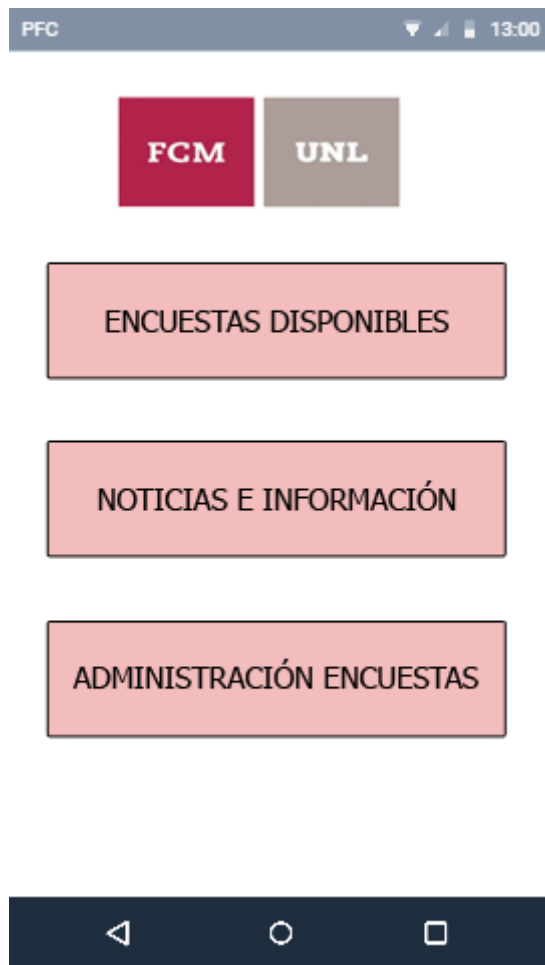
(Fig. 3 – Inicio de la aplicación)



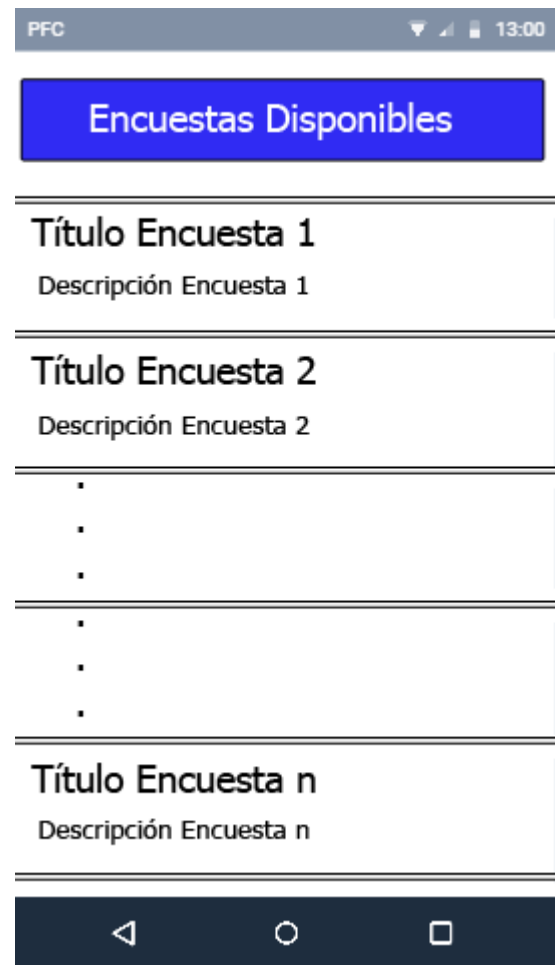
(Fig. 4 – Registración de Usuarios)

En la Fig. 3 se presenta la pantalla de inicio de la aplicación. La misma posee los campos para ingresar nombre de usuario, contraseña y poder iniciar sesión, así como también el botón de “Registrar nueva cuenta” para el alta de las mismas.

En la Fig. 4 se detalla la pantalla de registración de un nuevo usuario. Esta pantalla es iniciada cuando el usuario presiona sobre el botón de “Registrar nueva cuenta” (mencionado en la Fig. 3). Posee los distintos campos que el usuario debe completar de manera obligatoria para la creación de una nueva cuenta y el campo de “Código usuario específico”) que no es de carácter obligatorio, pero debe ser completado por aquellos usuarios que quieran adquirir los beneficios de usuarios específicos.



(Fig. 5 – Menú Principal)



(Fig. 6 – Encuestas Disponibles)

En la Fig. 5 se presenta el menú principal que va a contener la aplicación, una vez que se ha iniciado sesión. En este caso, cabe aclarar que se presenta la vista para los usuarios específicos, ya que los usuarios comunes solamente visualizarán los dos primeros ítems de “Encuestas Disponibles” y “Noticias e Información”. En cuanto a la “Administración de encuestas”, solamente será visualizado por los usuarios específicos que podrán llevar a cabo los ABMs de las encuestas.

En la Fig. 6 se presenta la pantalla de “Encuestas disponibles”. La misma es accedida por el usuario al presionar sobre el ítem de encuestas disponibles mencionado en la Fig. 5. En dicha pantalla se presentan todas las encuestas disponibles de resolución a modo de lista, traídas por el Web Service, con un título y una breve descripción. Luego para ser respondidas se deberá presionar sobre la encuesta deseada.

(Fig. 7 – Resolución de Encuesta)

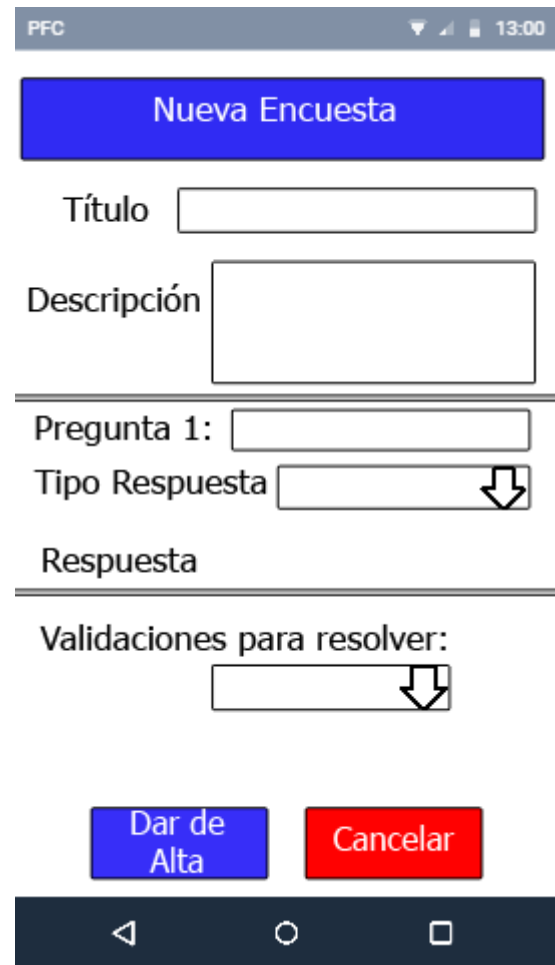
(Fig. 8 – Noticias Disponibles)

En la Fig. 7 se visualiza un ejemplo de resolución de encuesta. En la parte superior se muestra el título de la encuesta en cuestión y en los campos inferiores se van presentando todas las preguntas con sus respectivas respuestas. A modo de ejemplo se colocaron distintos tipos de respuestas que puede contener una pregunta. La última opción de “residencia habitual” está relacionada con el requerimiento de obtener la ubicación de resolución por geoposicionamiento o de forma manual (en caso de responder “No” se habilitarían los campos para ésta lectura).

En la Fig. 8 se detalla la sección de noticias, la cual es accedida presionando sobre el ítem “Noticias e información” mencionado en la Fig. 5. De forma similar a como se presentan las encuestas, a modo de lista se encuentran con un título y una breve descripción, y presionando sobre una de ellas se abrirá la noticia en cuestión.



(Fig. 9 – Administración de encuestas)



(Fig. 10 – Alta de Encuesta)

En la Fig. 9 se presenta la pantalla para llevar a cabo los ABMs de las encuestas. La misma es accedida presionando sobre el ítem “Administración Encuestas” mencionado en la Fig. 5. Dentro de la pantalla se presentan los distintos ítems para llevar a cabo la administración, como “Crear nueva encuesta”, “Modificar encuesta” y “Eliminar encuesta”. Para los últimos dos casos, al presionar sobre uno de ellos se abrirá una pantalla similar a la mencionada en la Fig. 6. Si se tratara de una modificación al presionar sobre una encuesta determinada, se abrirá una pantalla al estilo de la Fig. 10, con los campos ya cargados para poder modificar lo que se requiera, mientras que, si se tratase de una eliminación, al presionar sobre una encuesta determinada se abrirá un popup para confirmar la baja de la misma.

En el caso de presionar sobre “Crear nueva encuesta”, se presentará la pantalla diseñada en la Fig. 10. En la misma se encuentran los distintos campos a completar, como el título, descripción y las distintas preguntas adheridas. Por cada pregunta que se vaya agregando se debe colocar el tipo de respuesta (lista desplegable) y en base a la que se haya escogido, se cargarán los campos correspondientes para las posibles respuestas. Finalmente se encuentra un campo de validaciones que se deben cumplir para la resolución de la encuesta, de acuerdo a lo establecido en la etapa de análisis.

DISEÑO DEL WEB SERVICE:

Siguiendo la arquitectura establecida anteriormente, para el diseño del Web Service se realizó un modelo de tres capas. Como ya se especificó anteriormente, el diseño del mismo está sujeto a cambios, conforme avance el proyecto en función de las necesidades que vayan surgiendo.

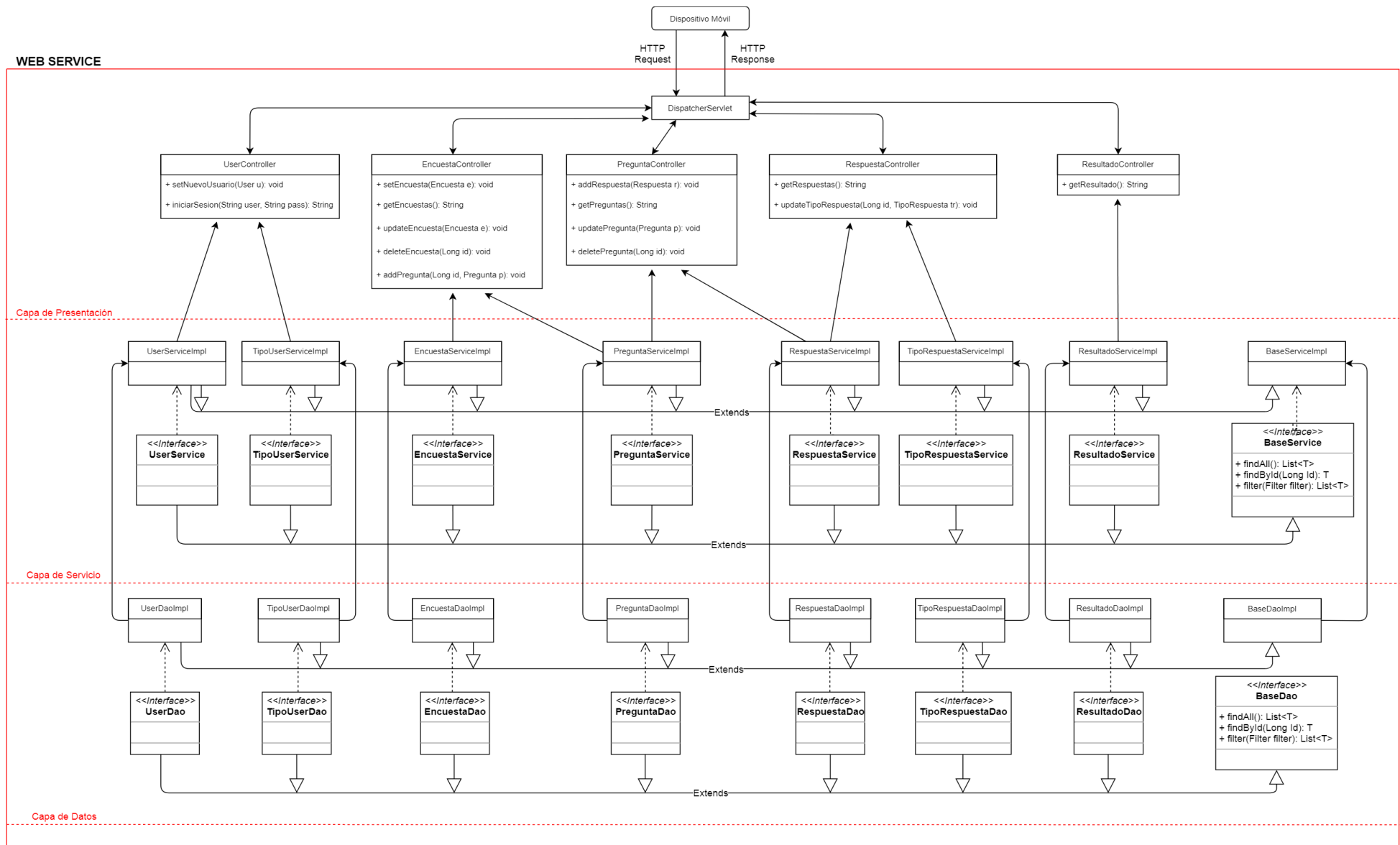
En este caso la capa de presentación será aquella que atienda todas las peticiones (recibiendo request y generando response) que lleguen al Web Service. Dicha capa cuenta con un **DispatcherServlet** cuya funcionalidad principal es la de tomar un **request** y encontrar la combinación correcta dentro de los **controladores** para generar una **response** correcta hacia el usuario. A su vez, el Dispatcher es el encargado de encapsular y abstraer muchas funcionalidades estandarizadas en Spring (framework utilizado de Java), mediante la declaración de beans específicos o el uso de anotaciones. Un ejemplo es el de la anotación *@RequestMapping* la cual notifica la petición concreta del cliente.

Luego, la capa de presentación se comunicará con la capa de negocio que se encargará de ejecutar todas las reglas de negocios necesarias para atender las peticiones mediante los servicios. Ésta finalmente se comunica con la capa de datos, la cual se encargará de llevar a cabo todas las transacciones referidas en la base de datos.

Cabe aclarar que dentro de la capa de servicios se encuentra una interfaz **BaseService** con funcionalidades en común para todos los servicios (los cuales la extienden). Y a su vez se encuentra la clase **BaseServiceImpl** que se encarga de llevar a cabo la implementación de las funcionalidades establecidas dentro de la interfaz.

De manera similar, en la capa de datos se encuentra la interfaz **BaseDao** y la implementación de dicha interfaz **BaseDaoImpl** de la cual extienden los demás “daos” como se puede ver en la figura.

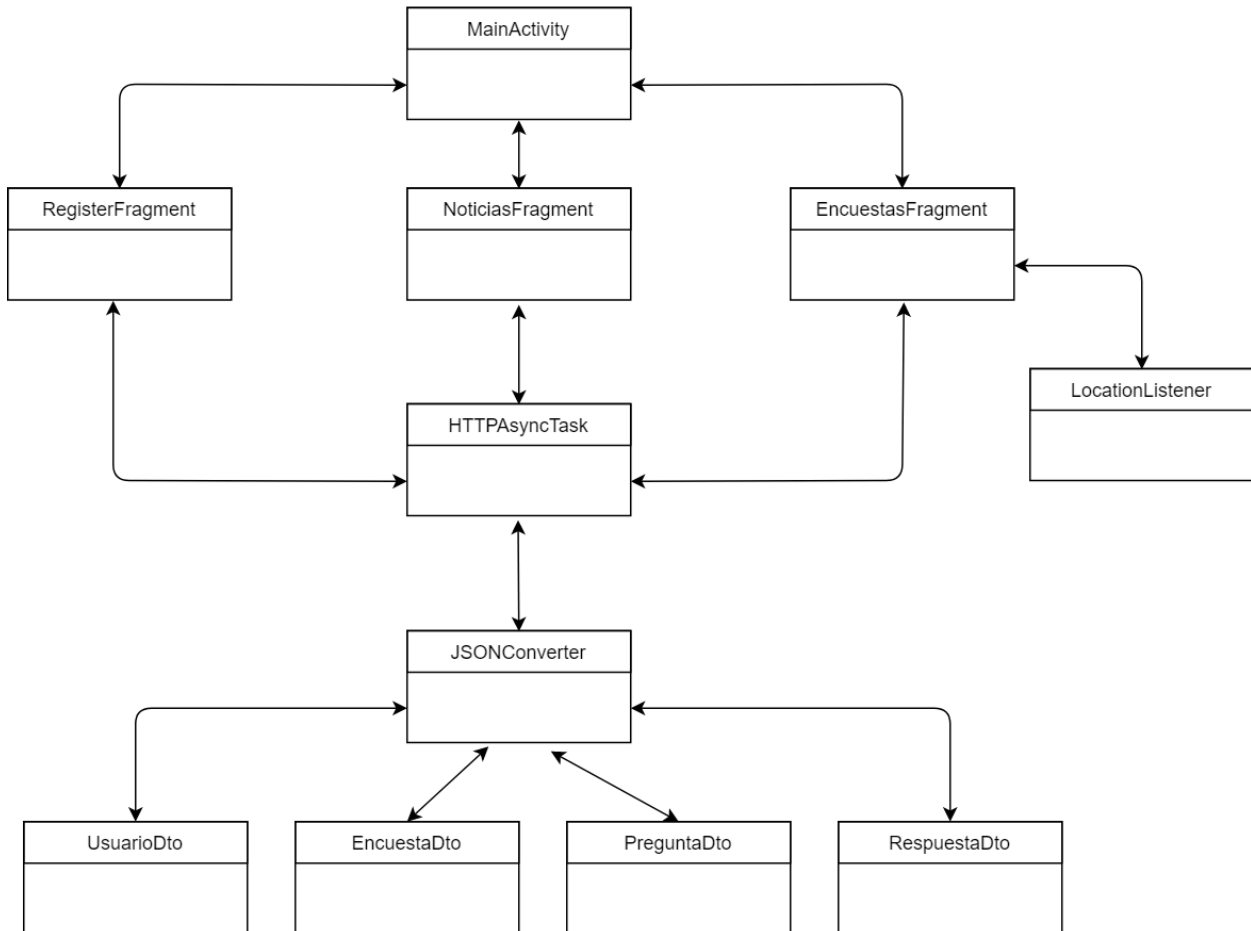
El DispatcherServlet podría recibir distintas peticiones por parte del usuario, ya sean del tipo **get**, **put**, **post** y **delete** para la realización de distintas tareas. A modo de ejemplo, una petición “get” para visualizar todas las encuestas disponibles hacia el controlador de encuestas (*EncuestasController*). Un método de ésta clase, como podría ser “*getEncuestas()*” se encargará de comunicarle a la implementación del servicio de encuestas que debe recuperar todas aquellas que se encuentren vigentes. Dicha regla de negocio es procesada y pasada a los “daos” correspondientes de la capa de datos, para luego retornar una respuesta al controlador.



(Fig. 11 – Modelo del WebService)

DISEÑO DE LA APLICACIÓN:

A continuación, se presenta un modelo aproximado de la aplicación a realizar. Al igual que en casos anteriores, el diseño del mismo está sujeto a cambios conforme avance el proyecto y obteniendo una mejor aproximación del modelo a realizar en etapas posteriores con el Web Service ya desarrollado y en funcionamiento.



(Fig. 12 – Modelo de la Aplicación)

Inicialmente cabe aclarar que los componentes gráficos de la aplicación consisten en distintos archivos XML como ya se mencionó anteriormente. Por lo tanto, las clases presentadas en el modelo son de tipo Java.

Como se puede ver en la Figura 12, se tiene una clase **MainActivity** la cual Android escoge para manejar la actividad principal de la aplicación. Dicha clase se encarga de instanciar los métodos específicos correspondientes a las distintas funcionalidades que presenta el proyecto y de gestionar la identificación del usuario. Es por ello que en principio se tendría una relación con tres nuevas clases de tipo **Fragment**. Todos los “fragment” se encuentran embebidos dentro de éste Activity principal, lo cual permite la manipulación independiente de cada uno de ellos, teniendo su propio **layout** y ciclo de vida. Específicamente se va a desarrollar un fragment relacionado a la registración de usuarios, otro en relación a la sección de noticias, y por último lo relativo a las encuestas y sus ABMs.

La clase **LocationListener**, como lo dice su nombre, es utilizada para la obtención de la georeferencia del usuario a la hora de resolver una determinada encuesta.

Luego, los distintos fragment están relacionados con la clase **HTTPAsyncTask** que se encarga de la conexión a la red. Ésta clase se encarga de lanzar las distintas peticiones que se realizan sobre el Web Service.

Como ya se mencionó en etapas previas, se trabajará mediante JSON para las distintas peticiones, por lo cual, se precisará de una clase **JSONConverter** que realice la conversión de objetos de dicho tipo (manipulados por la clase HTTPAsyncTask) en DTOS (Data transfer object), los cuales son objetos simples sin lógica de negocio, utilizados para representar los datos a nivel aplicación.

DISEÑO DE LA INTEGRACIÓN DE MÓDULOS:

Con la creación de las tablas de base de datos, se realizarán los mapeos correspondientes a las mismas en el desarrollo del Web Service (WS). Una vez desarrolladas estas partes, se procederá a integrar las mismas en la aplicación (app) Android.

Por lo tanto, se proponen algunos flujos de actividades que brinden solución a los alcances del proyecto. Inicialmente, el usuario al abrir la app se va a encontrar con la pantalla de *login*. Al escoger crear nueva cuenta, se cargará el formulario xml de los datos a ingresar y una vez completados los mismo, la app ejecutará una petición GET al Web Service, con todos los datos de registro. El WS atenderá ésta petición, y a partir de las correspondientes clases de servicio (comunicadas con la capa de datos), validará que todos los datos ingresados sean correctos y si fuera así realizando el INSERT en la base de datos, enviando luego una respuesta al usuario mediante un JSON sobre el éxito o fracaso de la operación, el cual luego será decodificado mediante la clase JSONConverter mencionada anteriormente.

En caso de que el usuario al abrir la app proceda a iniciar sesión, los campos cargados en usuario y password se enviarán por parte de la app al WS. Los servicios encargados de validar que los datos sean correctos responderán mediante un JSON a la app para poder cargar la pantalla de inicio (teniendo en cuenta el tipo de usuario que haya iniciado sesión), o informar al usuario los errores que se hayan producido.

Una vez dentro de la app (ya iniciado sesión), si el usuario escoge por ejemplo la opción de “Encuestas disponibles”, se ejecutará una petición GET hacia el WS, que mediante los servicios correspondientes obtendrá una lista de todas aquellas que se encuentren vigentes para responder, convirtiendo posteriormente dichos datos en un JSON y así poder enviarlos a la app mediante un *response*.

En el caso de los ABMs a realizar por parte de los usuarios específicos, en el *Alta* de una encuesta, se ejecutará una petición PUT hacia el WS, con todos los datos insertados en el formulario. Al igual que en casos anteriores el WS responderá mediante un JSON el estado de la solicitud. Si se tratara de una *Modificación*, para cargar el formulario de la encuesta en cuestión, previamente se ejecutaría una petición GET para traer los datos correspondientes a la misma, en tanto que una vez modificados los datos la petición a ejecutar sería del tipo POST para actualizar los datos en el servidor. Finalmente, para la *Baja* de una encuesta, la petición a ejecutar por la app sería del tipo DELETE.