Diploma de Especialización de Desarrollo de Aplicaciones Android

Proyecto Final

Quick Quizz

"Publica cuestionarios de una o varias preguntas rápidas, para obtener un sondeo rápido de resultados"

Quick Quizz, de aquí adelante por abreviar QQ, es una app que surge de una idea principal que expuso el tutor del máster de llevar a cabo una aplicación que nos permitiera hacer preguntas rápidas tipo test del mismo modo que el famoso juego de mesa Trivial, partiendo de esta idea y de otras obtenidas durante la realización del máster surge la propuesta de llevar a cabo el desarrollo de QQ como proyecto final del master.

Necesidades detectadas

Todos sabemos que cuando alguien intenta transmitir algo a otro conjunto de usuarios, durante una charla o ponencia el locutor la mayoría de las veces desearía poder tener un feedback rápido de los asistentes, el cual por distintas dificultades nunca se logra, ya se por limitaciones del medio en el que se imparte la charla o por la gran cantidad de asistentes a esta.

Por este motivo QQ es una apuesta interesante ya que el locutor podrá ir lanzando preguntas a todos los asistentes que se añadan al grupo de la charla, evento o simplemente como asistentes dentro de un local con el fin de poder recibir dichas preguntas y poder dar su respuesta con un simple click.

Otra necesidad que podría cubrir QQ o más que necesidad otra uso que se le podría dar, sería para fines de ocio o lúdicos, un par de ejemplos podría ser: nos encontramos dentro de un Karaoke y el dueño quiere premiar a quien haya hecho la mejor actuación de la noche para darle un premio, pues con QQ podría usar a los asistentes al local para que fueran ellos quien juzgarán, el dueño tan solo tendría que ir generando una pregunta por cada actuación y los asistentes podrían ir puntuando.
También estando con un grupo de amigos uno podría lanzar una pregunta como por ejemplo ¿Qué fin de semana os va mejor quedar para cenar? a continuación dar una serie de fechas como opciones y el resto votar las opciones que les va mejor, para así poder elegir el día que mejor se les ajuste.

Estas son las principales ideas en que se podrá utilizar QQ, pero no son las únicas ya que a ser una aplicación móvil y basar su funcionamiento en consultas sobre un servidor web, puede dársele infinidad de usos.

Arquitectura de QQs

La arquitectura y tecnologías utilizadas para el desarrollo de QQs abarcan distintos conocimientos obtenidos durante la realización del máster, en el esquema de la Figura 1 se intenta plasmar dicha arquitectura.

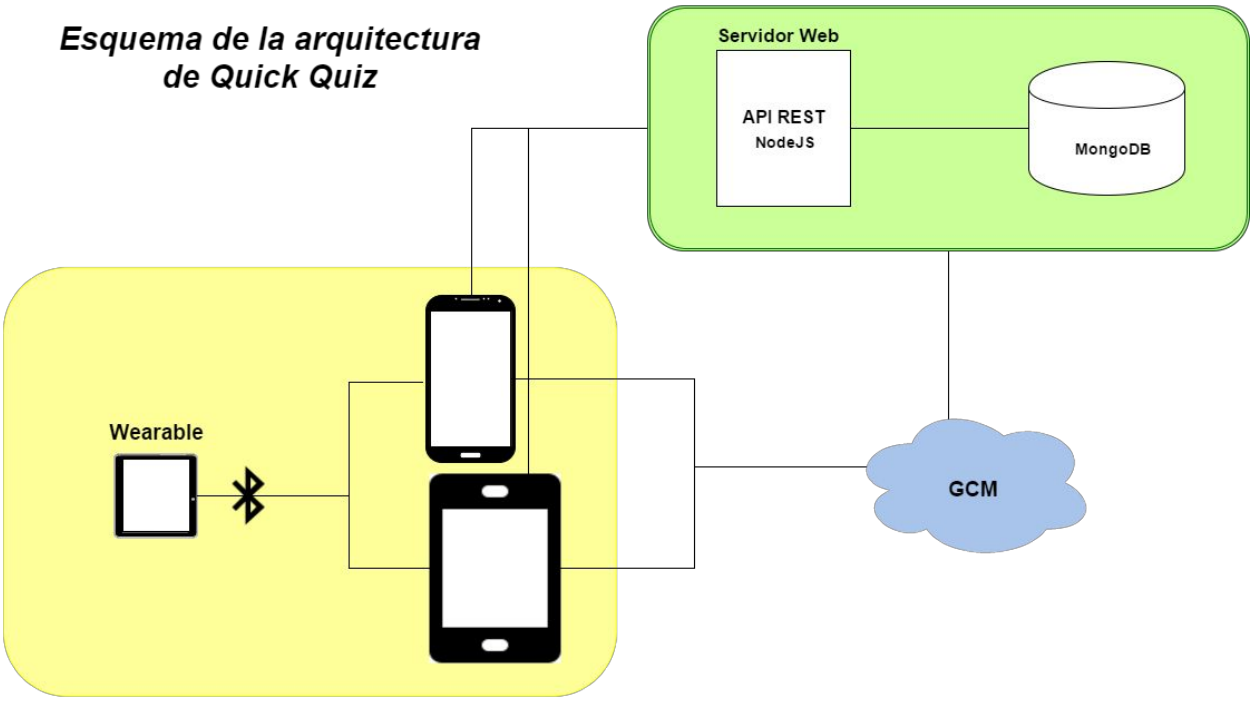


Figura 1. Esquema de la arquitectura de QQs

QQ dispone de una versión hospedada en un servidor web, por el momento está hospedada en el servidor Heroku (https://www.heroku.com/), por su sencillez y agilidad para poder desarrollar una aplicación REST con NodeJs, ExpressoJs y MongoDb.

Para la comunicación la api REST he utilizado una librería de google llamada Volley, la cual nos permite crear un singleton en nuestra aplicación con el cual podemos ir haciendo peticiones HTTP e ir encolándolas a una pila de peticiones de manera muy sencilla.

Para avisar a los participantes de los grupos cuando se publica o cierra una pregunta utilizaremos el servicio de Google para mensajería en la nube GCM (Google Cloud Messaging), más adelante detallaremos esta funcionalidad.

La aplicación nativa android tiene soporte para móvil y tablet (esta última por tiempo no dispone de UI customizada), también con el fin de utilizar más tecnologías del máster y por darle un valor añadido a la app, también tenemos soporte para smartwatch con sistema Android Wear, en los dispositivos wearables la idea es poder recibir la pregunta y poder responder directamente desde él si el usuario quisiera, además podremos visualizar la gráfica de resultados en el reloj.

Roles y tipos de usuarios

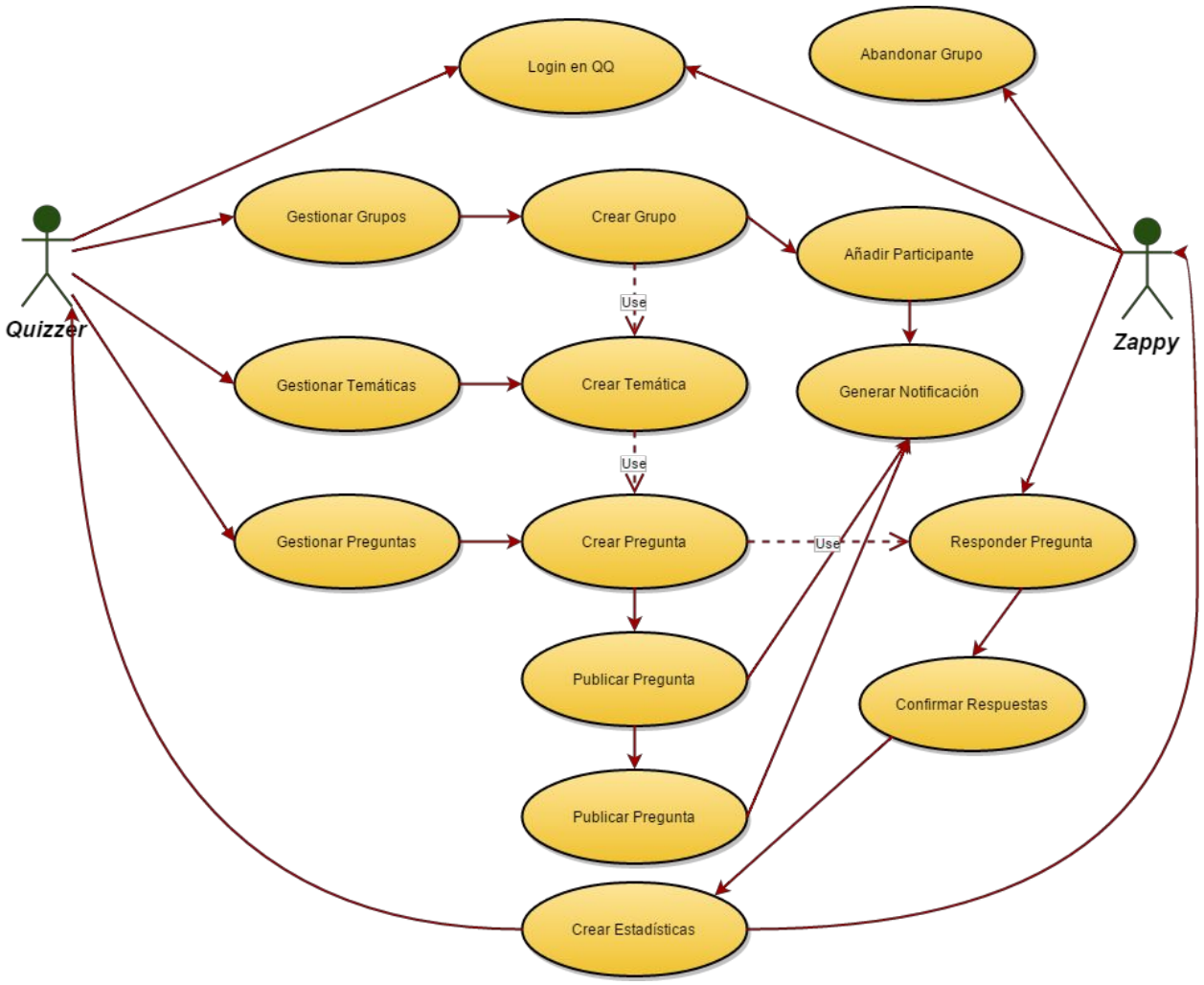
Dentro de QQs podemos identificar distintos tipos de usuarios en función del papel que desempeñen a la hora de participar en una QQ, con esto queremos decir que un usuario no es que tenga un rol propio por conectar a la aplicación. A continuación se detallan los distintos tipos de rol.

- ***Quizzer.*** Cuando un usuario dá de alta un grupo o evento al cual le irá lanzando las QQs automáticamente asume para ese grupo o evento el rol de Quizzer. Las acciones que puede realizar:
    - Administrar las QQs de su grupo.
    - Dar de alta/baja/modificar las QQs
    - Gestionar los Zappy o asistentes dentro de sus QQs
    - Publicar o parar una QQ
    - Ver resultados parciales de una QQ
- ***Zappy.*** Cualquier usuario añadido a un grupo automáticamente se convierte en un Zappy para ese grupo. Las acciones que puede realizar son las siguientes:
    - Responder las QQs de su grupo.
    - Responder las QQs desde un dispositivo wearable.
    - Ver los resultados finales de una QQ.
    - Salirse de un grupo o evento.

Estos son los roles que como participante de un grupo podemos asumir dentro de Quick Quiz.

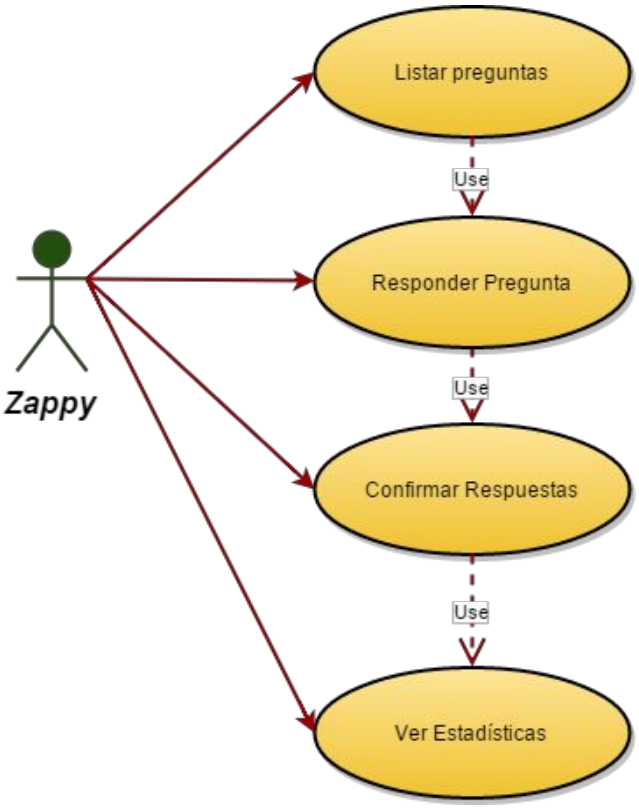
Diagramas de casos de uso

En el siguiente diagrama mostramos los casos de uso para los roles antes descritos para la versión mobile:



En el diagrama vemos representado como cualquier usuario puede ser Quizzer de sus grupos, temáticas y preguntas de estas, y como un participante con rol Zappy dentro de un grupo puede salirse de este, responder preguntas y confirmarlas para finalmente ver los resultados en forma de estadísticas.

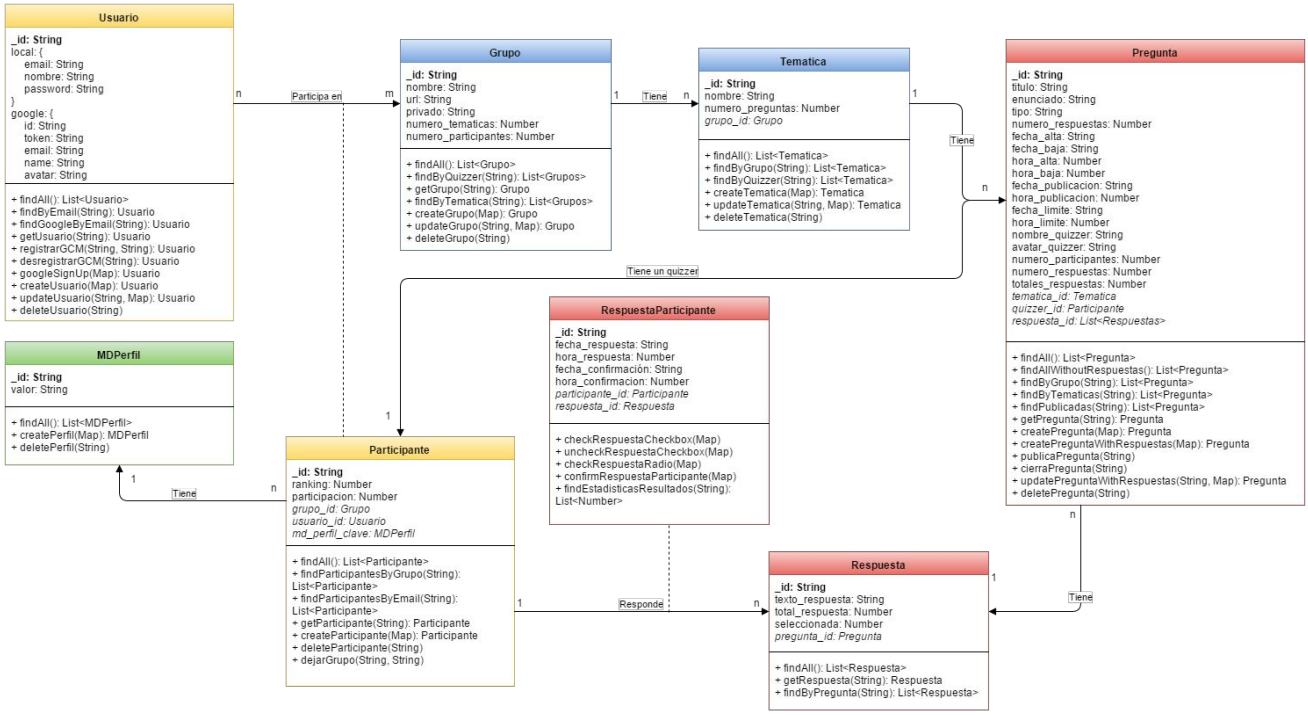
Otro diagrama de casos de uso es el que corresponde a la aplicación wear, el cual por la manera de interactuar con este tipos de dispositivos, da como resultado menores casos de usos:



En este diagrama lo que podemos observar es que la aplicación wear solo está diseña para que podamos adoptar el rol de Zappy y por lo tanto podamos desde la aplicación acceder a la lista de preguntas de los grupos en los que estamos participando y desde el mismo dispositivo podamos responderlas, confirmarlas y posterior visualizar la gráfica de resultados de las estadísticas.

Diagramas de clases (Api REST)

El modo de persistir los datos en QQ esto mediante servidor web ya que cualquier acción que realicemos tiene que estar sincronizada con el resto de participantes del grupo, por este motivo se ha tenido que implementar una api de tipo REST, en el siguiente diagrama de clases se muestra los objetos que persistimos en la base de datos MongoDB, y cada uno de los Endpoint que dispone cada clase de la api:




```
//routes Usuario
app.get('/api/usuarios', usuario.findAll);
//app.get('/api/usuario/:id', usuario.findById);
app.post('/api/usuario/local', usuario.createUsuario);
app.get('/api/usuario/email/:email', usuario.findByEmail);
app.get('/api/usuario/google/:email', usuario.findGoogleByEmail);
app.get('/api/usuario/:id', usuario.getUsuario);
app.put('/api/usuario/gcm/registrar/:id', usuario.registrarGCM);
app.put('/api/usuario/gcm/desregistrar/:id', usuario.desregistrarGCM);
app.post('/sendGCM', usuario.sendGCM);
app.post('/api/usuario/google/signup', usuario.googleSignUp);
app.put('/api/usuario/:email', usuario.updateUsuario);
app.delete('/api/usuario/:email', usuario.deleteUsuario);

//routes Calificacion
app.get('/api/calificaciones', calificacion.findAll);
app.get('/api/calificacion/:participante_id', calificacion.findByParticipante);
app.get('/api/calificacion/:tematica_id', calificacion.findByTematica);
app.get('/api/calificacion/:participante_id/tematica_id', calificacion.getCalificacion);
app.post('/api/calificacion', calificacion.createCalificacion);
app.put('/api/calificacion/:participante_id/tematica_id', calificacion.updateCalificacion);
app.delete('/api/calificacion/:participante_id/tematica_id', calificacion.deleteCalificacion);

//routes Grupo
```

```
        app.get('/api/grupos', grupo.findAll);
        app.get('/api/grupos/usuario/:usuarioId', grupo.findByQuizzer);
        app.get('/api/grupo/:id', grupo.getGrupo);
        app.get('/api/grupo/tematica/:tematicaId', grupo.findByTematica);
        app.get('/api/grupo/nombre/:nombre', grupo.findByNombre);
        app.get('/api/grupos/publicos', grupo.findAllPublicos);
        app.post('/api/grupo', grupo.createGrupo);
        app.put('/api/grupo/:id', grupo.updateGrupo);
        app.delete('/api/grupo/id', grupo.deleteGrupo);

        //routes MD_Perfil
        app.get('/api/perfiles', perfil.findAll);
        app.post('/api/perfil', perfil.createPerfil);
        app.delete('/api/perfil/:clave', perfil.deletePerfil);

        //routes Participante
        app.get('/api/participantes', participante.findAll);
        app.get('/api/participantes/grupo/:grupo', participante.findParticipantesByGrupo);
        app.get('/api/participante/email/:email', participante.findParticipantesByEmail);
        app.get('/api/participante/:id', participante.getParticipante);
        app.post('/api/participante', participante.createParticipante);
        app.delete('/api/participante/:clave', participante.deleteParticipante);
        app.delete('/api/participante/grupo/:grupo_id/:usuario_id', participante.dejarGrupo);

        //routes Tematica
        app.get('/api/tematicas', tematica.findAll);
        app.get('/api/tematicas/grupo/:grupoId', tematica.findByGrupo);
        app.get('/api/tematicas/quizzer/:usuarioId', tematica.findByQuizzer);
        app.post('/api/tematica', tematica.createTematica);
        app.put('/api/tematica/:id', tematica.updateTematica);
        app.delete('/api/tematica/:id', tematica.deleteTematica);

        //routes Preguntas
        app.get('/api/preguntas', pregunta.findAll);
        app.get('/api/preguntas/sinrespuestas', pregunta.findAllWithoutRespuestas);
        app.get('/api/preguntas/grupo/:grupo', pregunta.findByGrupo);
        app.get('/api/preguntas/tematicas/:usuarioId', pregunta.findByTematicas);
        app.get('/api/preguntas/vigentes/:dataActual', pregunta.findVigentes);
        app.get('/api/preguntas/usuario/:usuarioId', pregunta.findPublicadas);
        app.get('/api/preguntas/tematica/:tematica/:fechaInici/:fechaFin',
pregunta.findPublicadasByTematica);
        app.get('/api/pregunta/:id', pregunta.getPregunta);
        app.get('/api/pregunta/:id/:usuarioId', pregunta.getPreguntaByUsuario);
        app.post('/api/pregunta', pregunta.createPregunta);
        app.post('/api/pregunta_w_respuestas', pregunta.createPreguntaWithRespuestas);
        app.put('/api/pregunta/publicar/:preguntaId', pregunta.publicaPregunta);
        app.put('/api/pregunta/cerrar/:preguntaId', pregunta.cierraPregunta);
        app.put('/api/pregunta/:id', pregunta.updatePreguntaWithRespuestas);
        app.delete('/api/pregunta/:id', pregunta.deletePregunta);

        //routes Respuesta
        app.get('/api/respuestas', respuesta.findAll);
        app.get('/api/respuestas/:id', respuesta.getRespuesta);
        app.get('/api/respuestas/pregunta/:pregunta', respuesta.findByPregunta);
        app.post('/api/respuesta', respuesta.createRespuesta);
        app.put('/api/respuesta/:id', respuesta.updateRespuesta);
        app.delete('/api/respuesta/:id/:preguntaid', respuesta.deleteRespuesta);

        //routes Respuesta Participante
        app.get('/api/respuestas_participantes', respuestaParticipante.findAll);
        app.get('/api/respuesta_participante/:id', respuestaParticipante.getRespuestaParticipante);
        app.get('/api/respuestas_participantes/:participante',
respuestaParticipante.findRespuestasByParticipante);
        app.post('/api/respuesta_participante', respuestaParticipante.createRespuestaParticipante);
        app.post('/api/respuesta_participante/checkbox/check',
respuestaParticipante.checkRespuestaCheckbox);
```

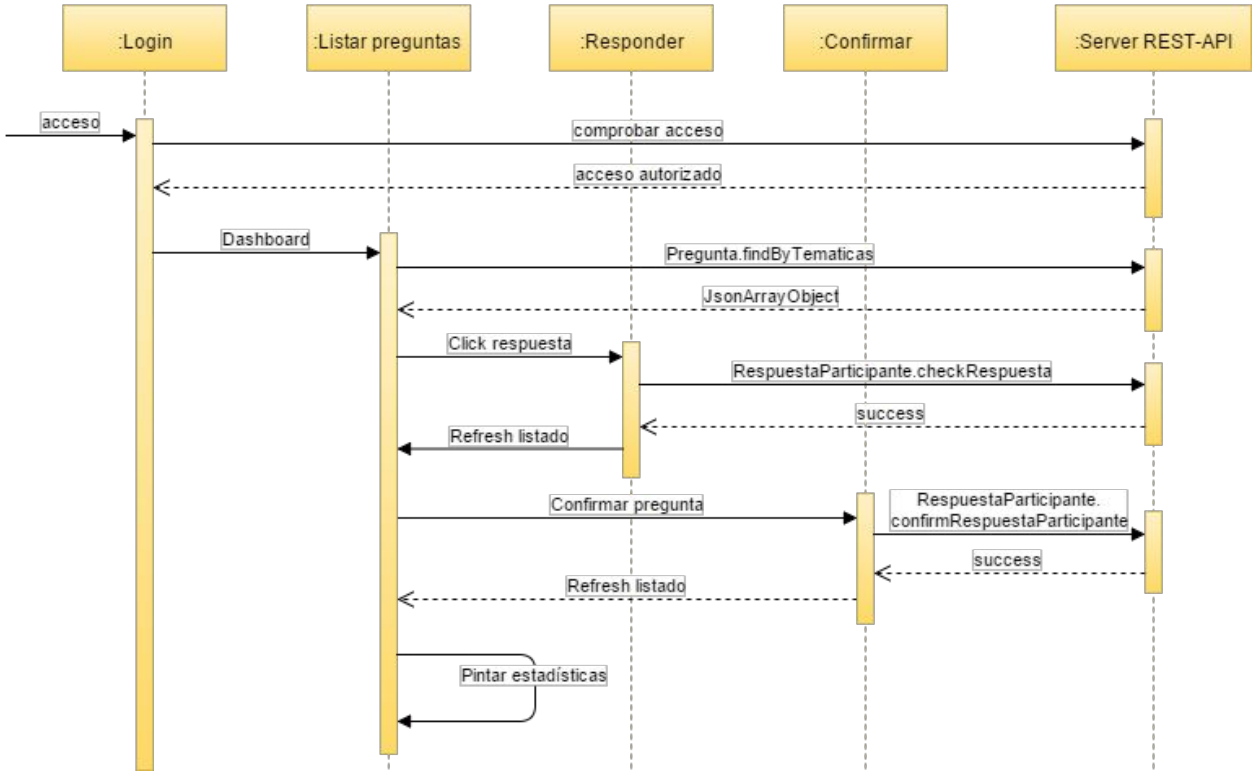
```
        app.post('/api/respuesta_participante/checkbox/uncheck',
respuestaParticipante.uncheckRespuestaCheckbox);
        app.post('/api/respuesta_participante/radio/check', respuestaParticipante.checkRespuestaRadio);
        app.put('/api/respuesta_participante/confirm',
respuestaParticipante.confirmRespuestaParticipante);
        app.delete('/api/respuesta_participante/:id',
respuestaParticipante.deleteRespuestaParticipante);
        app.get('/api/respuesta_participante/resultados/:preguntaId',
respuestaParticipante.findEstadisticasResultados);
```

No todos estos endpoints se utilizan en la actual versión de la aplicación, ya que algunos de ellos se han implementado para la versión web.

Diagramas de secuencia

Para aclarar con más detalles el flujo de datos entre los distintos dispositivos y el server REST, a continuación detallaremos dos de los diagramas de secuencia principales, el resto de CRUDs de la aplicación mantienen una secuencia básica.

El siguiente diagrama nos muestra el flujo de un usuario utilizando la versión mobile de la aplicación para responder una pregunta desde su terminal.

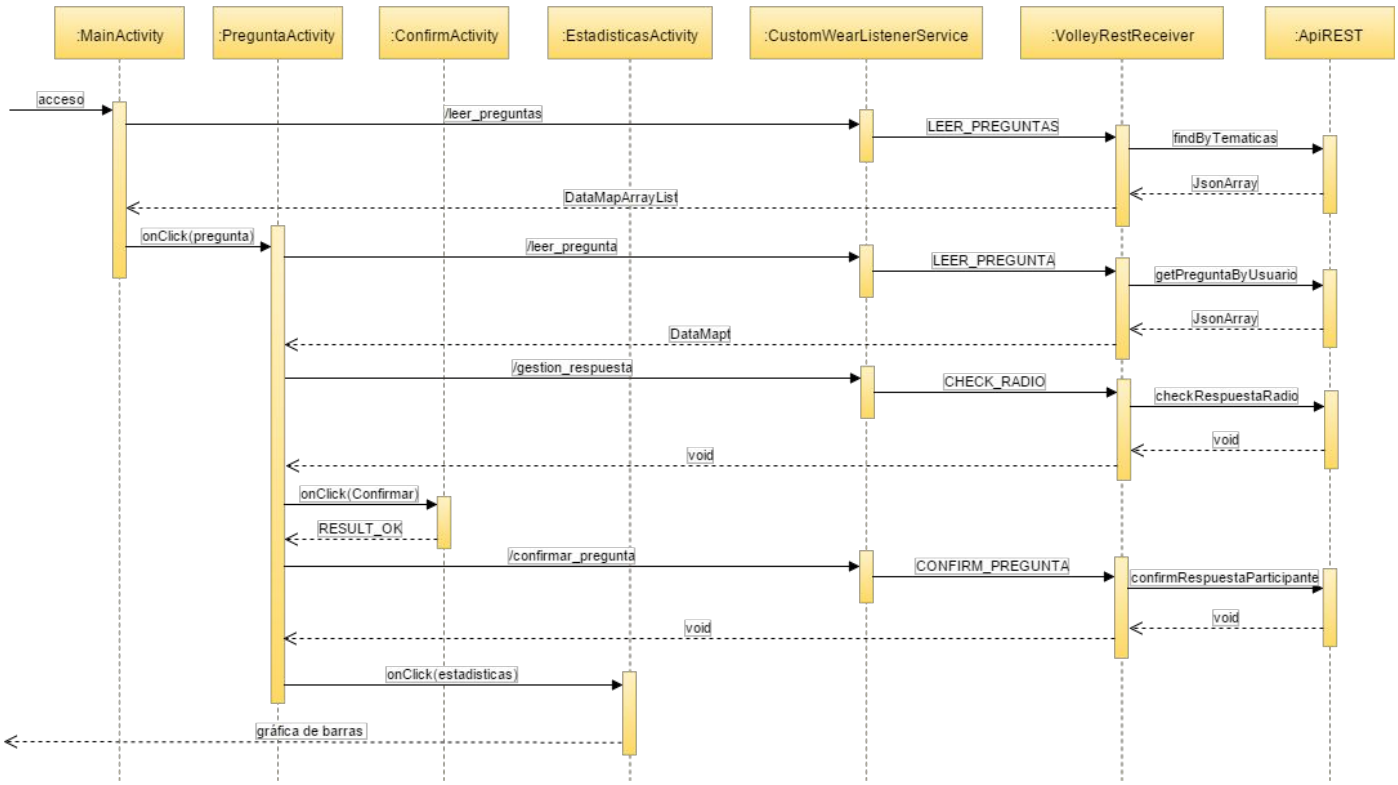


Como se puede apreciar en el diagrama el usuario al abrir la aplicación en el terminal se debe logear contra el servidor el cual si nos devuelve el usuario de base de datos nos indica que tenemos acceso autorizado.
Una vez logeados iremos a la pantalla principal que en su esencia es un dashboard donde podemos ir visualizando las preguntas de los grupos en los que participamos.
Desde esta misma pantalla podremos interactuar con cada una de estas pregunta pudiendo responderlas, después confirmar nuestra respuesta y una vez confirmamos nuestra respuesta podremos ver una gráfica de barras con los porcentajes de las respuestas del resto de participantes.

Dentro de la aplicación mobile como ya veremos más adelante en la sección de pantallas, QQ dispone de un menú de tipo NavigationDrawer desde el cual podremos acceder a tres

secciones que nos permite gestionar los grupos que nosotros creemos, las temáticas de estos y las preguntas que publicaremos en cada una de las temáticas.

Por otro lado como ya hemos mencionado también hay una versión wear de la cual el siguiente diagrama de secuencia nos muestra el flujo principal de dicha aplicación.



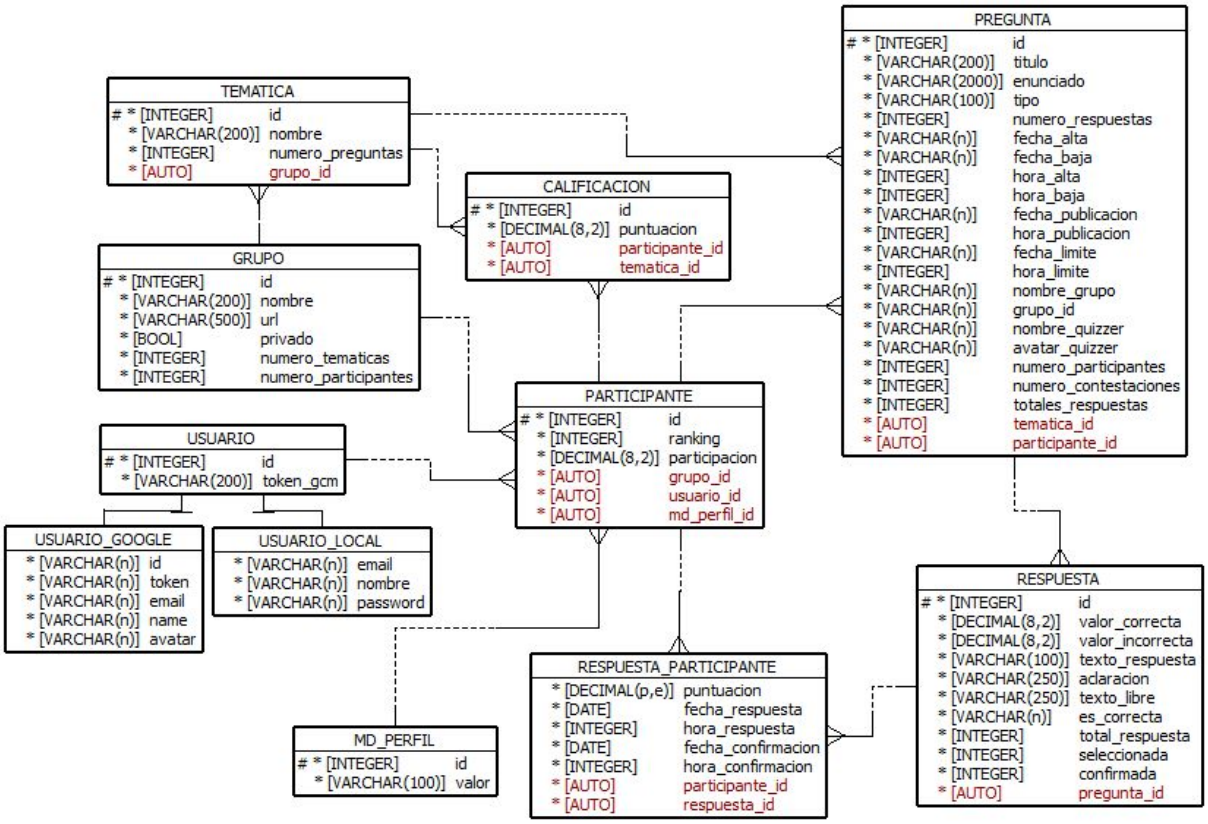
El usuario accede a la aplicación wear y lo primero que visualiza es listado de las preguntas de los grupos en los que está participando como zappy, la aplicación wear para obtener esta información se la solicita al móvil el cual tiene un escuchador para atender las peticiones del wear,
una vez procesa el móvil qué tipo de petición nos está solicitando llama a un broadcastReceiver el cual se encarga mediante Volley de obtener la información del server y enviarla al listener de la actividad del wear.
Una vez el usuario visualiza las preguntas escoge una y esto nos lleva a otra actividad que vuelve a enviar un mensaje al móvil solicitando que le de los datos de la pregunta que ha escogido, el proceso por parte del móvil es el mismo el listener discrimina y el receiver se encarga de llamar al endpoint necesario para realizar la operación.
Una vez el usuario ya visualiza la pregunta podrá escoger la respuesta o respuestas que cree conveniente y después deberá confirmar la pregunta para poder acceder a las estadísticas de resultados, representadas en una gráfica de barras.

Diagrama Entidad-Relación

A continuación se detalla los requisitos funcionales que debe recoger el modelo de base de datos, para garantizar cubrir las necesidades de la lógica de QQ.

- Los usuario podrán acceder a la aplicación de dos maneras distintas, por acceso local en nuestro servidor propio (heroku) y por acceso con su cuenta de google.
- Un usuario puede ser participante de varios grupos, y dentro de estos grupos puede ser un participante de tipo Quizzer o Zappy, pero no ambos roles a la vez.
- Un grupo tiene de 1 a n participantes, pero solo uno de estos participantes puede tener el perfil de tipo Quizzer.
- Un grupo puede catalogar las preguntas en distintas temáticas.
- Una temática puede tener varias preguntas pero una pregunta no puede pertenecer a más de una temática.
- Las preguntas se pueden catalogar por tipos (por el momento pueden ser MONORESPUESTA o MULTIRESPUESTA)
- Toda pregunta que se desee publicar para que los participantes puedan responderla tiene que tener la fecha y hora de publicación.
- Toda pregunta tendrá que poder registrar la fecha y hora de cuando se cierra la pregunta a los participantes.
- Una pregunta puede tener varias respuestas asociadas, mientras que las respuestas solo pueden pertenecer a una pregunta.
- Un participante para una pregunta puede seleccionar de 0 a n respuestas.
- La respuesta o respuestas que un participante escoge para una pregunta, tienen que poder registrar la fecha y hora de cuando el participante confirma la pregunta.

Con el siguiente esquema se intenta conseguir cubrir todas las necesidades generales de la aplicación y por lo tanto será la base de datos que hospedará la aplicación servidora.

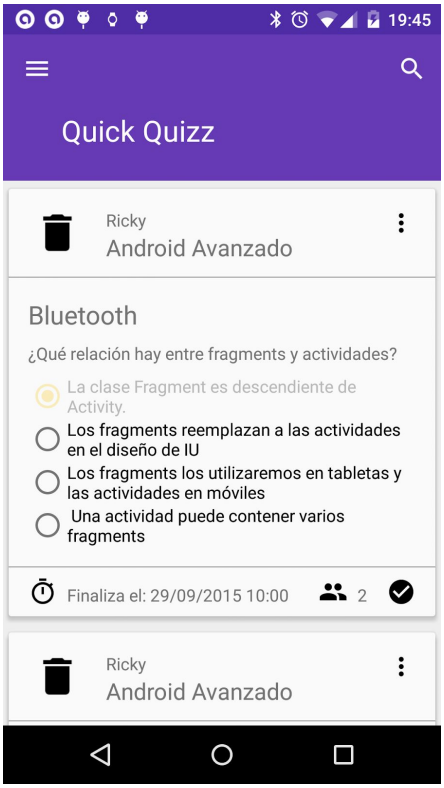

PREGUNTA
* [INTEGER] id
* [VARCHAR(200)] titulo
* [VARCHAR(2000)] enunciado
* [VARCHAR(100)] tipo
* [INTEGER] numero_respuestas
* [VARCHAR(n)] fecha_alta
* [VARCHAR(n)] fecha_baja
* [INTEGER] hora_alta
* [INTEGER] hora_baja
* [VARCHAR(n)] fecha_publicacion
* [INTEGER] hora_publicacion
* [VARCHAR(n)] fecha_limite
* [INTEGER] hora_limite
* [VARCHAR(n)] nombre_grupo
* [VARCHAR(n)] grupo_id
* [VARCHAR(n)] nombre_quizzer
* [VARCHAR(n)] avatar_quizzer
* [INTEGER] numero_participantes
* [INTEGER] numero_contestaciones
* [INTEGER] totales_respuestas
* [AUTO] tematica_id
* [AUTO] participante_id
TEMATICA
* [INTEGER] id
* [VARCHAR(200)] nombre
* [INTEGER] numero_preguntas
* [AUTO] grupo_id
CALIFICACION
* [INTEGER] id
* [DECIMAL(8,2)] puntuacion
* [AUTO] participante_id
* [AUTO] tematica_id
GRUPO
* [INTEGER] id
* [VARCHAR(200)] nombre
* [VARCHAR(500)] url
* [BOOL] privado
* [INTEGER] numero_tematicas
* [INTEGER] numero_participantes
PARTICIPANTE
* [INTEGER] id
* [INTEGER] ranking
* [DECIMAL(8,2)] participacion
* [AUTO] grupo_id
* [AUTO] usuario_id
* [AUTO] md_perfil_id
USUARIO
* [INTEGER] id
* [VARCHAR(200)] token_gcm
USUARIO_GOOGLE
* [VARCHAR(n)] id
* [VARCHAR(n)] token
* [VARCHAR(n)] email
* [VARCHAR(n)] name
* [VARCHAR(n)] avatar
USUARIO_LOCAL
* [VARCHAR(n)] email
* [VARCHAR(n)] nombre
* [VARCHAR(n)] password
RESPUESTA
* [INTEGER] id
* [DECIMAL(8,2)] valor_correcta
* [DECIMAL(8,2)] valor_incorrecta
* [VARCHAR(100)] texto_respuesta
* [VARCHAR(250)] aclaracion
* [VARCHAR(250)] texto_libre
* [VARCHAR(n)] es_correcta
* [INTEGER] total_respuesta
* [INTEGER] seleccionada
* [INTEGER] confirmada
* [AUTO] pregunta_id
RESPUESTA_PARTICIPANTE
* [DECIMAL(p,e)] puntuacion
* [DATE] fecha_respuesta
* [INTEGER] hora_respuesta
* [DATE] fecha_confirmacion
* [INTEGER] hora_confirmacion
* [AUTO] participante_id
* [AUTO] respuesta_id
MD_PERFIL
* [INTEGER] id
* [VARCHAR(100)] valor

Pantallas de QQ

QQ como ya se ha comentado anteriormente tiene una versión mobile y otra wear de la aplicación, y además por las características de cada dispositivo estas versiones tienen distintas funcionalidades.
A continuación se muestran la mayoría de las vistas o pantallas que componen cada una de las versiones de QQ.

Versión mobile

Para la versión mobile se ha optado por un diseño siguiendo los patrones que google aconseja desde su última versión de Lollipop, llamado Material Desing, con esto se intenta que la interacción del usuario con la interfaz de la aplicación sea de una manera fluida y lo más usable posible.



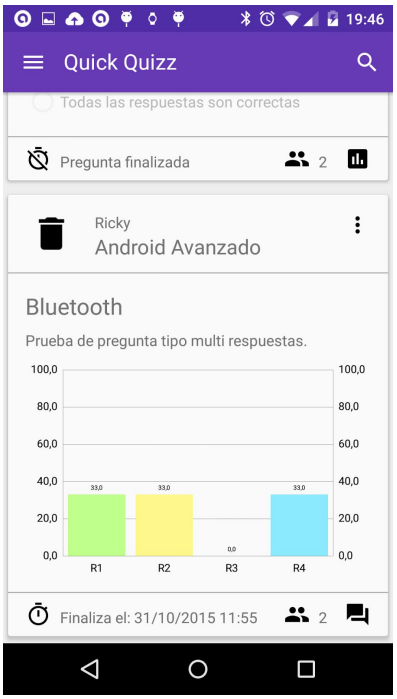
Por este motivo el usuario nada más acceder a la aplicación y después de haberse logeado, podrá visualizar la view del Dashboard donde verá un listado de todas las preguntas de los grupos donde el participa como Zappy.

Cada una de estas preguntas, en forma de tarjetas, al usuario de manera sencilla le muestra la información del grupo al que pertenece la pregunta así como la temática de esta y el Quizzer que la publicó.

En la parte inferior de la tarjeta también podrá visualizar información de cuando se cierra la pregunta, o si ya está cerrada, el número de participantes de la pregunta y las acciones de confirmar, o la de visualizar las estadísticas una vez la pregunta está validada. *(Por cuestiones de infraestructura del servidor web gratuito, el proceso de cierre automático de las preguntas no se ha podido realizar)*.

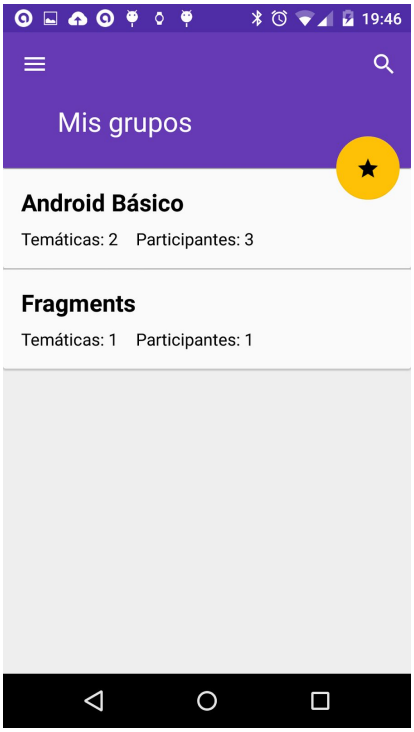
En esta misma tarjeta el usuario podrá seleccionar su respuesta o respuestas.

Si seleccionamos visualizar las estadísticas dentro del propio cuerpo de la tarjeta nos aparecerá la gráfica de resultados.



Por último cada tarjeta tiene un menú con opciones adicionales donde de momento solo hay disponible la opción de abandonar el grupo, por si no quisiéramos continuar en él.

Con este planteamiento de que nada más entrar nos vayan saliendo las preguntas que tenemos o podemos ir contestando, lo que se quiere conseguir es que la gente no le cueste participar en los cuestionarios que se vayan publicando.



Las siguientes pantallas nos muestran las secciones dedicadas al rol Quizzer.

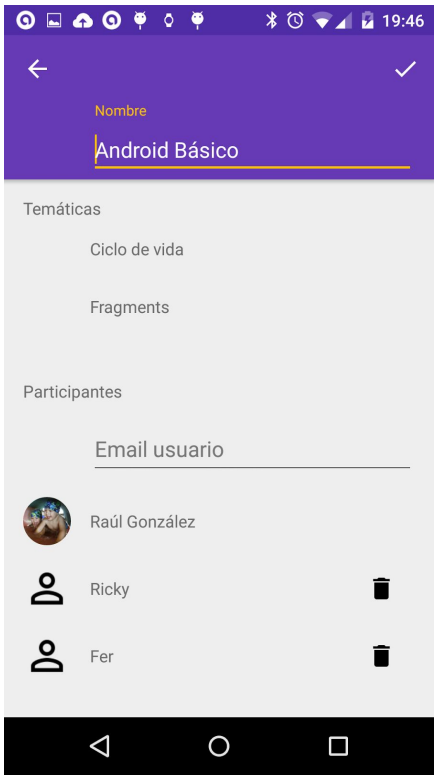
Como ya se comentó todo usuario de QQ en cualquier momento puede ser Quizzer ya que tan solo es necesario que dé de alta un grupo.

En la sección mis grupos se encuentra el listado de todos los grupos que se han registrado, de cada uno de estos grupos también se visualiza el número de temáticas asociadas y el número de participantes del grupo.

Si le damos añadir un nuevo grupo en el botón flotante o si escogemos cualquiera de los grupos pasaremos a la vista de edición de grupos, donde se puede modificar el nombre del grupo y añadir o quitar participantes.

La manera en que se ha diseñado funcionalmente este formulario está pensada para que el usuario vaya dando de alta los grupos que crea necesario y posteriormente acceda a dichos grupos para ir añadiendo o quitando los usuarios.

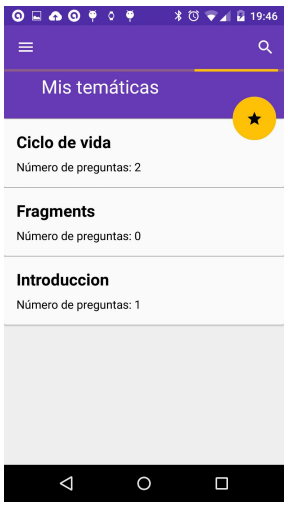
Con el fin de facilitar el añadir usuarios y por usabilidad dentro del formulario de grupos hay un campo autocompletable en el cual podemos escribir el correo del usuario que queremos añadir como participante, es importante que el usuario debe de estar registrado en QQ.

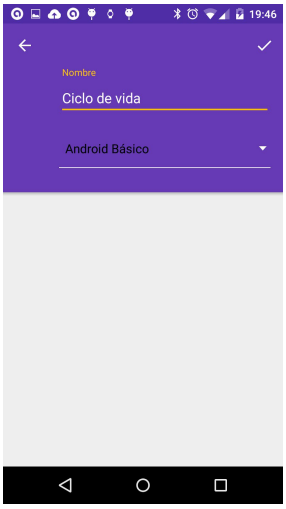


También es importante mencionar aunque se recordará en la sección correspondiente, es que recibiremos una notificación cuando nos añadan a un grupo.

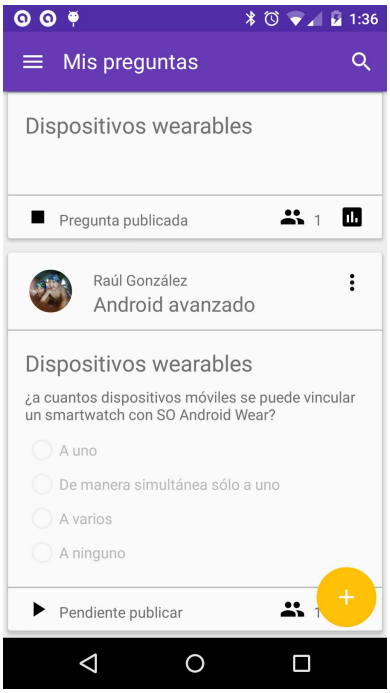
Otra sección que un Quizzer puede gestionar es Mis temáticas, donde de igual modo que en Mis grupos el usuario visualizará un listado de las temáticas que ha dado de alta y el número de preguntas que tiene registradas en cada temática.

La vista de edición de la temática es todavía más sencilla porque tan solo de una temática registramos el nombre de esta y al grupo que pertenece.





Por último la pantalla sección más importante para los Quizzer es Mis preguntas donde podrá gestionar las preguntas de cada temática, publicarlas, cerrarlas y ver los resultados de cada una de ellas.

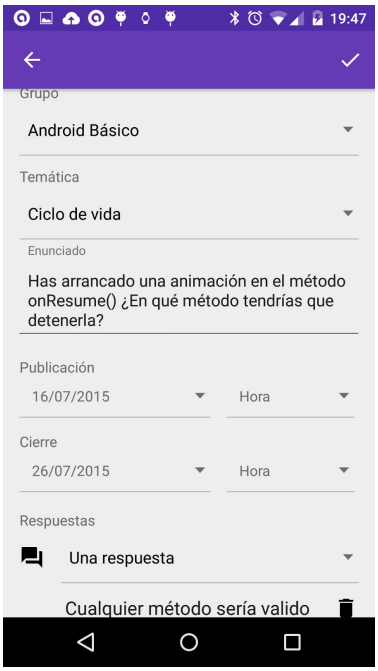


Para seguir manteniendo la usabilidad y coherencia la pantalla del listado de mis preguntas mantiene el mismo formato que la vista del Dashboard, únicamente lo que cambia son alguna de las acciones.

En el menú de opciones tendremos la posibilidad de editar la pregunta o eliminar, es importante resaltar que tan solo nos deja eliminar las preguntas que no han sido publicadas.

Por otro lado en la parte inferior de la tarjeta también se puede observar que hay un nuevo botón de acción que nos permite publicar (botón del triángulo) y cerrar la pregunta
(botón del cuadrado).

En cuanto a la vista del formulario de edición de preguntas es algo más complejo los otros formularios de la aplicación.



Por un lado nos encontramos una anidación de listas de selección con las cuales podemos elegir el grupos donde queremos publicar la pregunta y concretar para qué categoría en concreto, ambos campos son obligatorios informarlos para poder dar de alta la pregunta.

El siguiente campo que deberemos informar será el enunciado de la pregunta, este campo no tiene límite de tamaño, pero partiendo de QQ está pensado para hacer preguntas rápidas, no debería excederse mucho el tamaño del enunciado.

Seguidamente encontramos los campos destinado a las fechas de publicación y cierre o límite del periodo en que se encuentra vigente la pregunta, estos campos están pensado para el proceso automático de publicación de preguntas, el cual ya se ha mencionado que por temas de infraestructura del servidor web, no se ha podido implementar, por este motivo estos campos en el momento de alta de la pregunta se pueden no informar, ya que a posterior podremos manualmente publicar la pregunta y cerrarla cuando lo deseemos.

Y por último debajo nos encontramos con la sección para gestionar las respuestas, primero de todos escogeremos si es una pregunta de tipo monorespuesta o multirespuesta, la diferencia entre ambas es que la monorespuestas se representa gráficamente con objetos radios y solo podremos marcar una respuestas, mientras que la multirespuesta se representa con checkbox y nos deja marcar tantos como creamos necesarios.

Una vez escogida el tipo de respuestas debajo podremos ir añadiendo o quitando tantas respuestas como queramos.

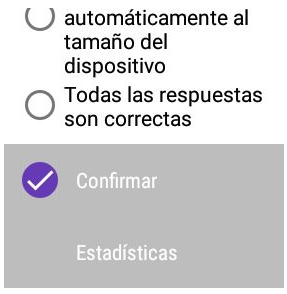
Versión wear

El sistema de views que se han diseñado para la versión de la aplicación para smartwatchs con Android Wear se basa en un diseño muy plano y minimalista, la idea es no sobrecargar dichas views ya que lo se quiere de la versión wear es una rápida interacción del usuario.

La view principal muestra al usuario un listado con el resumen de preguntas de los grupos en que participa, mostrando un información bastante ligera pero que ya le aporta indicios de la temática a la que pertenece dicha pregunta y de parte del enunciado de esta.
Otra información que nos muestra de un vistazo rápido este listado son las preguntas por colores, donde los colores tienen el siguiente significado:

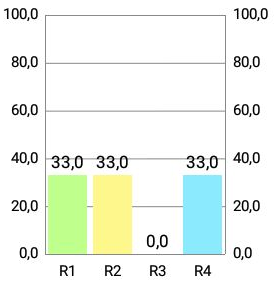
- Fondo blanco, la pregunta está publicada y no tiene fecha de fin.
- Fondo azul, la pregunta está publicada y tiene fecha de cierre o fin.
- Fondo rojo, la pregunta ya está cerrada y no se puede responder.

Una vez seleccionamos una pregunta accedemos a la ficha donde vemos la información al completo del enunciado y las respuestas a las que podemos optar.



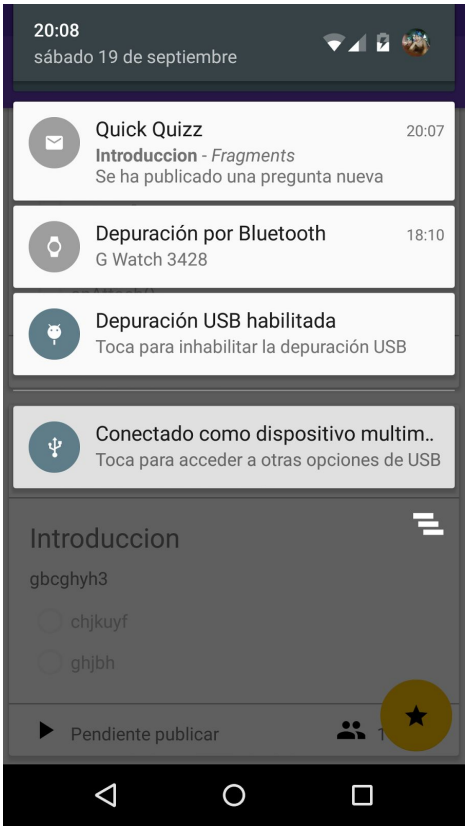
Y después de que escoja la respuesta o respuestas que cree conveniente, puede confirmar la pregunta con el botón que encontrará al final del todo de la pregunta.

Cuando ya ha confirmado la pregunta la view le mostrará el botón de acción de Estadísticas para poder visualizar los resultados de las respuestas globales mediante una gráfica de barras.

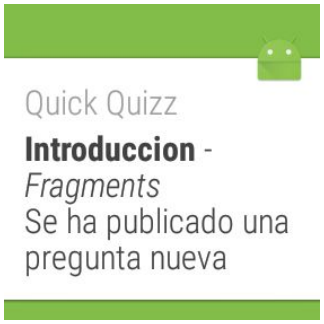


Notificaciones

Además de la interacción del usuario mediante las views, QQ genera una serie de notificaciones para comunicar al usuario de cambios de la aplicación, estas notificaciones están diseñadas para ser compatibles tanto en mobile como en wear y en concreto con acciones propias para cada tipo de dispositivo.



Tanto la notificación mobile como la wear nos ofrece la posibilidad de acceder QQ mobile, a la pantalla del dashboard.



Por otro lado la notificación wear también nos ofrece la posibilidad de abrir en el reloj, lo cual nos llevaría a la pregunta a la que hace referencia la notificación.

Notificaciones

Para poder mantener informado a los usuario de QQ, se ha implementado una integración con el servicio de GCM de google, de tal manera que desde nuestro servidor web cuando realizamos alguna de las siguientes acciones:

- *Añadir un participante a un grupo,* el Quizzer del grupo cuando escoge a un usuario para añadirlo como participante, la aplicación mobile hace una petición al servidor web para insertar al participante y seguidamente envía un mensaje al servicio GCM con el id de registro de dicho usuario.
- *Publicación de una pregunta,* esta notificación se genera en el momento que el Quizzer pública una pregunta de forma manual desde la pantalla de mis preguntas, el servidor web nada más publicar la pregunta, envía una notificación a cada uno de los participantes de ese grupo.
- *Cierre de una pregunta,* esta notificación de igual modo que la notificación de publicación de una pregunta, el servidor web envía una notificación a cada uno de los participantes del grupo donde se ha cerrado la pregunta.

Estas son las notificaciones que se han implementado hasta este momento, aunque el sistema está preparado para poder ir añadiendo nuevas notificaciones conforme se vaya detectando la necesidad.