



Urband



# Índice

- Estilo
- Dominio
- Requisitos
- Sprints
- Resultados

U

# ESTILO

## Definimos



Nova Square



Documentación Formal...



Urband

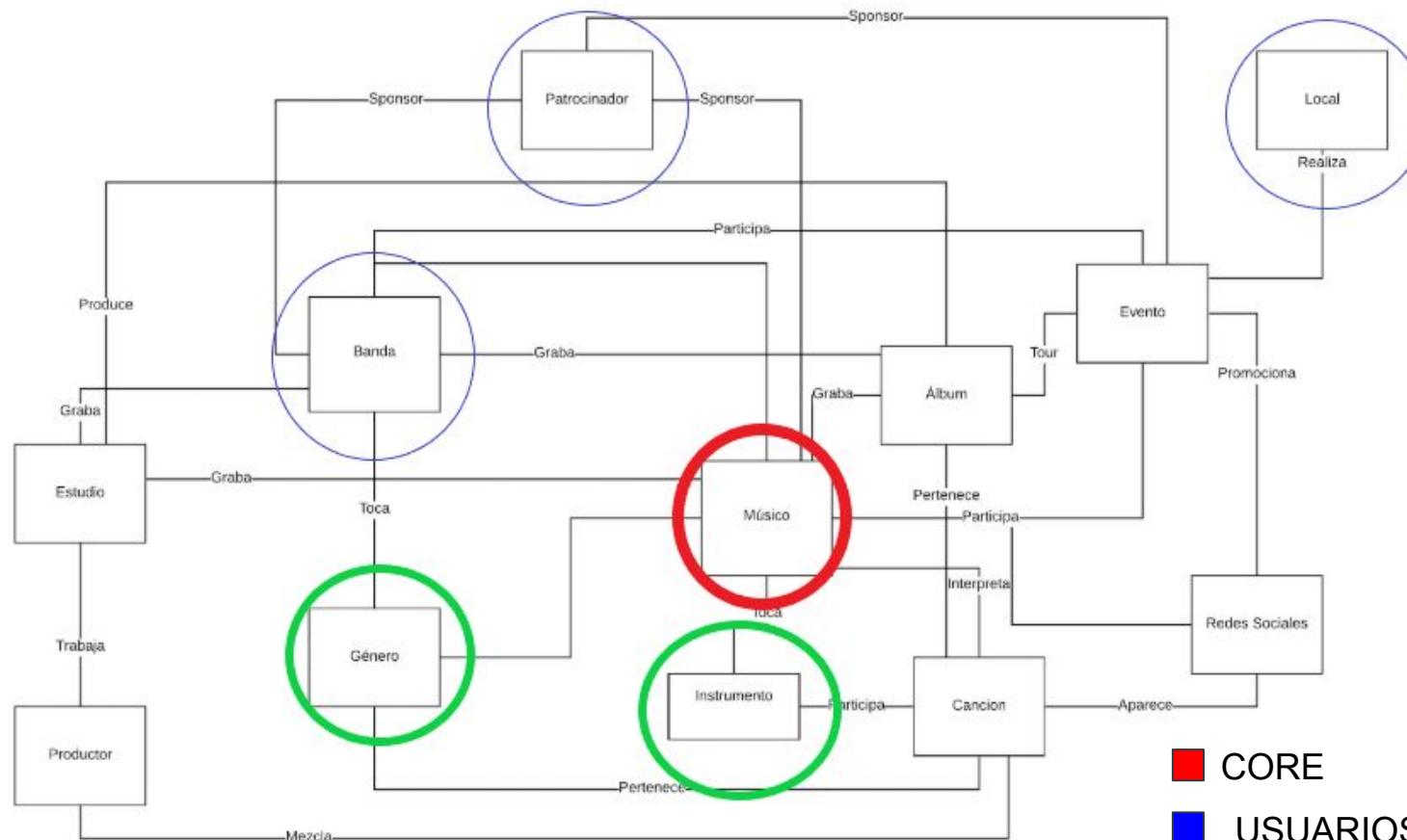
## Dividimos

- Análisis
- Testing
- Programación
- Documentación
- Estilado



# DOMINIO





**CORE**

**USUARIOS**

**PARTES APP**

# REQUISITOS



## *Requisito de usuario*

### 8. El usuario debe poder contactar con resultantes de un match



8.1 El usuario solo podrá acceder al perfil de un usuario que no sea su match, con información reducida, pero no podrá ponerte en contacto con él, tiene que ser un match

8.2 El usuario contará con diferentes opciones externas para ponerte en contacto, dependiendo de los datos del perfil

- 8.2.1 **Métodos de contacto externos** : Whatsapp, Instagram, correo electrónico y facebook ofrecidos por el usuario en su perfil
- 8.2.2 Se podrá contactar por la misma aplicación, implementaremos un chat para que los usuarios no tengan que usar un método externo de contacto

## *Requisito de sistema*

### 13. Nuestro sistema debe integrar diferentes sistemas de pago online.



En la aplicación, se tendrá que poder pagar por varios métodos cuando queramos comprar, esto conlleva una integración de diversos métodos de pagos online.

13.1 **Integración API de Paypal:** Una opción de pago muy usada, usaremos la API para desarrolladores que la misma empresa ofrece de manera gratuita, en este caso la versión para Java.

13.2 **Integración API de RedSys:** Menos conocido, pero permite pagar con tarjeta, una opción que no podemos perder de cara a los clientes. Igual que en la opción anterior, la propia empresa proporciona una API Rest para facilitar la integración.

ENTREVISTAS

ESCENARIOS

ENCUESTAS

# USER-STORIES

1. El usuario debe poder guardar/editar su geolocalización
  - a. Implica añadir un campo en la tabla de usuario, añadir el campo por la API, integrarlo gráficamente en el registro. Pedir permiso de geolocalización
2. El usuario debe poder verse geolocalizado en la pantalla por defecto
  - a. Implica añadir librería de google maps y generar el mapa gráficamente
3. El usuario ha de poder ver que usuario tiene cerca de sí
  - a. Implica añadir markers por cada usuario en la pantalla de google maps. Expansión 11



10. El usuario debe poder visualizar un mapa en la aplicación que le ofrezca interacciones.

10.1 Mapa : se podrá elegir el tipo de mapa que queremos visualizar (satélite, mapa, relieve), éste se podrá ampliar o alejar, dependiendo la zona en que quiera buscar el usuario.

- 10.1.1 La aplicación tendrá la opción de poder buscar usuarios cercanos en el mapa, estos aparecerán con un marker acompañado de un icono para su clara interpretación.
  - 10.1.1.1 Marker : éste elemento permitirá que los usuarios vean situado en el mapa a los otros usuarios. Al pulsar el marker aparecerá el perfil del usuario, en él habrá un resumen de ese usuario, si queremos más información del usuario buscado.
  - 10.1.2 La aplicación tendrá la opción de poder buscar eventos, éstos aparecerán en un mapa en que los diferentes eventos programados estarán marcados con un marker que estará resaltado para que los usuarios lo puedan pulsar fácilmente.
    - 10.1.2.1 Músicos, grupos o conciertos : en la parte inferior del mapa habrá la opción de poder elegir que lo que vemos marcado en el mapa pueden ser o conciertos, músicos o grupos.

Práctica 2. Ingeniería de los requisitos

17

3 marzo

Práctica 4. Sprint 1, 2 y 3

19

5 mayo

Práctica 5. Sprint de consolidación

20

20 mayo

**LOS REQUISITOS EVOLUCIONAN**

## 2. El usuario necesita un perfil para identificarse

2.1 El usuario tiene que poder instanciar y modificar su perfil en todo momento.

### Requisitos de sistema

2.1.1 **Acceso al perfil** : Si es la primera vez que entras, se abrirá inmediatamente al hacer click en "Sign in". También se accede desde un botón del menú principal.

2.1.2 **Pantalla del perfil** : una estructura de formulario, con campos obligatorios y campos opcionales para completar su descripción de usuario. Contendrá un botón de "Aplicar Cambios", un botón de vuelta atrás.

- 2.1.1.1 *Campos obligatorios* : ímagen de perfil, nombre completo, género musical, al menos uno, que instrumentos tocas, al menos uno (pudiendo seleccionar más de uno incluyendo el canto), experiencia como músico ,una breve descripción de los usuarios que buscas en la aplicación, e información detallada de tu ubicación y disponibilidad.
- 2.1.1.2 *Campos opcionales* : una breve descripción personal, fotografías complementarias, referencias a tu contenido.
- 2.1.1.3 El botón de aplicar cambios, guardará los cambios en el perfil, para ser vistos para otros usuarios

2.1.3 El usuario debe aceptar unos permisos para poder subir imágenes desde su teléfono.

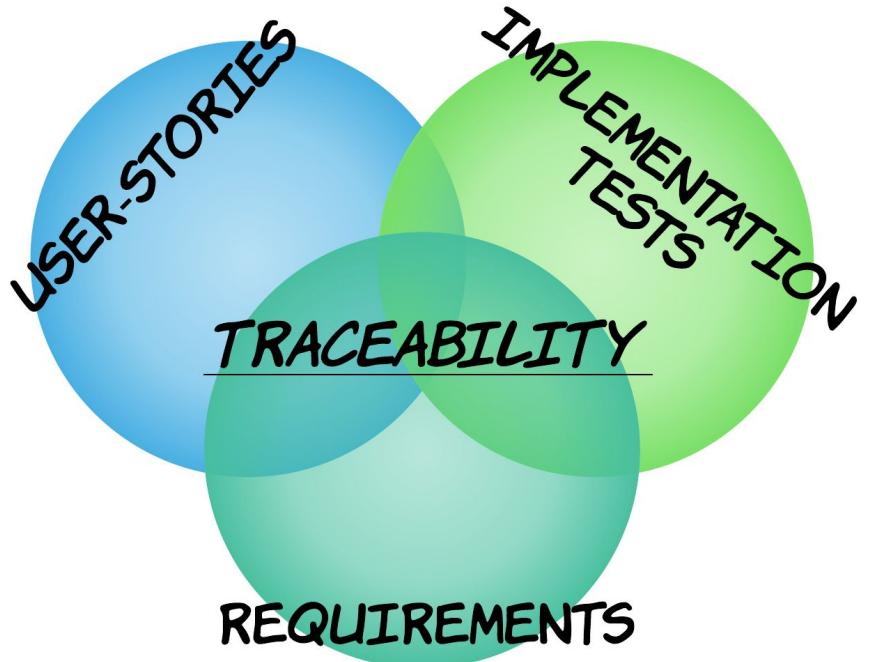
- 2.1.3.1 se le aparecerá una "Toast" o un "popup", donde el usuario deberá aceptar el permiso de que el sistema pueda recoger imágenes del usuario, si no se acepta, el usuario no podrá subir multimedia.

## 2. El usuario necesita un perfil para identificarse

2.1. **Instanciación y modificación del perfil:** dividimos la configuración del perfil en 3 etapas

- 2.1.1 Diálogo con los datos más personales del usuario
    - Nombre completo, fecha de nacimiento, género, experiencia general en la música, foto de perfil y una pequeña descripción.
  - 2.1.2 Diálogo con las habilidades musicales, instrumentos, etc.
    - El sistema debe mostrar una tabla dinámica que se actualice a medida que el usuario añade o borra instrumentos. Incorporando elementos visuales para añadir o eliminar elementos
  - 2.1.3 Diálogo con géneros musicales favoritos o identificativos.
    - El sistema debe mostrar una tabla dinámica que se actualice a medida que el usuario añade o borra géneros musicales. Incorporando elementos visuales para añadir o eliminar géneros
- 2.1.2 El usuario debe aceptar unos permisos para poder subir imágenes desde su teléfono.
- 2.1.3.1 se le aparecerá una "Toast" o un "popup", donde el usuario deberá aceptar el permiso de que el sistema pueda recoger imágenes del usuario, si no se acepta, el usuario no podrá subir multimedia.

2.2 **Acceso al perfil** : Si es la primera vez que entras, se abrirá inmediatamente al hacer click en "Sign in". También se accede desde un botón del menú principal.



U

# SPRINTS

# Tareas destacadas de los sprints

**Tarea 1 – Registro de Usuario:** implementar el sistema de registro.

**Tarea 2 – Inicio de sesión:** implementar el sistema de inicio de sesión.

**Tarea 2 – Implementación géneros musicales del usuario**

**Tarea 3- Inicio lógica suscripciones entre usuarios.**



# SPRINT I: Trazabilidad

IDENTIFICACION		ESTADO			CARACTERISTICAS			DETALLE Y VAUDR		
	Id	Tipo	Versión	Estado	Fecha de estado	Prioridad	Complejidad	Objetivo	Test usado	Resultados test
1.2.1	1.2.1.1	Sistema	0.1	Developement	20/03/2020	1	3	Comprobar usuario	Test 4	Cool- Comprobado
	1.2.1.2	Sistema	0.1	Developement	22/03/2020	2	4	Mensaje de error	Instrumented test 1	Cool- Comprobado
	1.2.1.3	Sistema	0.1	Developement	27/03/2020	1	7	Crear Token en DB	Postman 1, Test 4	Cool- Comprobado
	1.2.1.4	Sistema	0.1	Developement	27/03/2020	1	6	Guardar Token en disp	Shared pref Test	Cool- Comprobado
	1.2.1.5	Sistema	0.1	Developement	02/04/2020	3	4	Borrar Token en DB y disp	Postman 2, Test4	Cool- Comprobado
	1.2.1	Usuario	0.1	DONE	07/03/2020	2	2	Pantalla Logeo	Instrumented test	Cool- Comprobado
	1.2	Usuario	1	DONE	12/03/2020	1	3	Logeo de Usuario	All sons tests	Achieved!

# SPRINT I: Pruebas

## Pruebas unitarias

```
@Test  
public void isValidPassword() throws IOException {  
    String password = "Mmmmmmmmm88@";  
    assertTrue( message: "La contrasenya es vàlida" ,LoginUtils.isValidPassword(password));  
  
@Test  
public void isValidEmailAddress() throws IOException {  
    String email = "usuari1@gmail.com";  
    assertTrue( message: "L'email és vàlid", LoginUtils.isValidEmailAddress(email));  
  
@Test  
public void isValidPhone() throws IOException {  
    String phone = "607689879";  
    assertTrue( message: "El telèfon es vàlid", LoginUtils.isValidPhone(phone));  
  
@Test  
public void Check_EnviemApi_Valid() throws IOException {  
    UserViewModel viewModel;
```

# SPRINT II: Calidad del Software y SCRUMSU

## Calidad del Software

- Planned Software Test

## Revisión e inspección

- Review de documentación

- Plantillas

## Actas de las reuniones

Data: 3 de abril de 2020 a las 17:00h

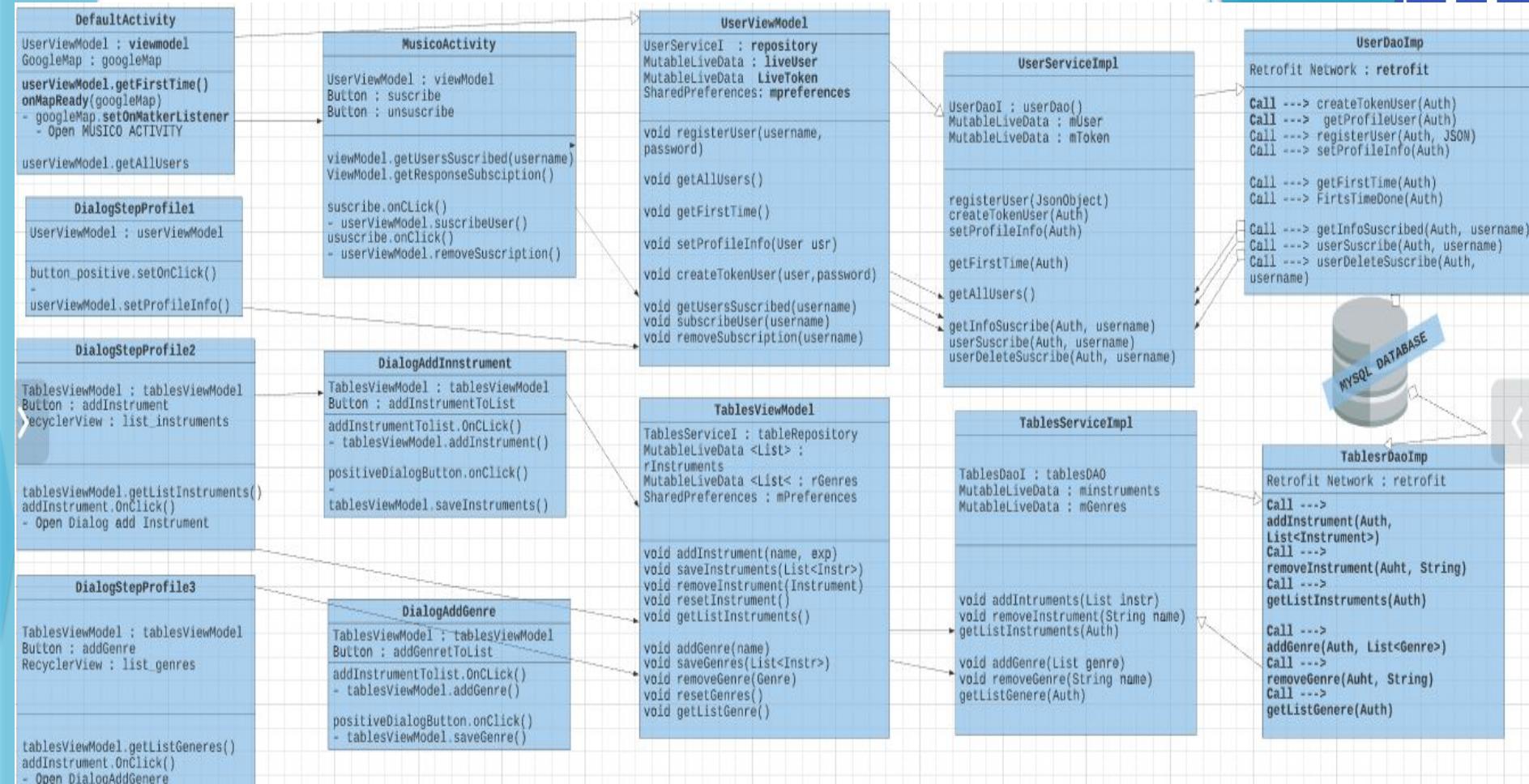
Asistentes: Toni Carol, Iván Méndez, Èlia Isart, Oriol Vivó

Orden del dia:

- General: Seguimiento de las user stories, exponer dudas a otros integrantes del grupo para resolver, discusión prototipo. División tareas sprint II Innovació.
- Toni: Documentación sobre los permisos de cámara, galería. Arreglar, cambiando nombres, las activities. Proponer nuevas maneras de hacer el código.
- Ivan: Arreglar problemas de programación android. Comentar que la ruta de fichero es diferente para cada uno y no tener problemas al subirlo.
- Oriol: Desplegar Docker y mirar API.
- Èlia: Acabar prototipo. Pasar en un PDF y subir al campus el Lean Canvas.

# SPRINT III: Diagrama de clases

11



# SPRINT IV: Principales resultados

- **Estado de las Tareas:** Revisión de los resultados Requisito por requisito 
- **Evolución de los requisitos:** Se crean nuevos y se modifican  
- **Tabla de gestión y trazabilidad de los requisitos:** Revisión de **TODOS** los requisitos
- **Proceso seguido:** Pasos para lograr los objetivos

# RESULTADOS

# Organización

- Trabajo en equipo
- Herramientas



 **Lucidchart**

# Dificultades

- ¿Qué nos ha costado más ?
- ¿Qué nos ha costado menos ?



# Resultado final

- Requisitos iniciales: 17
  - Requisitos finales: 20
- 
- Requisitos completados: 10
  - Requisitos a completar: 7

