Nivel de acabado de la aplicación

1. Descripción de la aplicación

El proyecto consiste en hacer una red social parecida a Spotify o SoundCloud para la subida de beats. Incluye la gestión de los mismos, un módulo de estadísticas y amistades, playlists y mucho más.

1.1. Descomposición en microservicios

Existen varios microservicios, siendo estos los siguientes:

- user-auth: Gestiona toda la lógica de autenticación vía JWT, apoyándose en Redis para la validación y seguridad de tokens. Además de la administración de usuarios y perfiles con personalización avanzada, integra mecanismos de seguridad 2FA y verificación de identidad externa. Desarrollado por Ramón Gavira y Benjamín Flores
- beats-upload: Es el microservicio encargado de gestionar la subida de beats. Desarolado por Daniel Vela y Miguel Encina
- beats-interactions: Se encarga de la lógica de las playlists, los comentarios, los ratings y la moderación de los mismos. Desarrollado por Daniel Galván y Jaime Linares.
- analytics-and-dashboards: Se encarga de la lógica de cálculo de métricas de un beat, creación de dashboards y visualización de estas métricas mediante widgets. Desarrollado por Daniel Ruiz y Rafael Pulido.
- social: Se encarga de la lógica de las amistades entre usuarios, el feed y las conversaciones y mensajes. Desarrollado por Andrés Martínez y Sergio Álvarez.
- payments-and-suscriptions: Se encarga de gestionar los pagos y los cambios en el plan de precios. Desarrollado por Miguel Encina y Ramón Gavira

Customer Agreement

El Customer Agreement se encuentra definido aquí. Este documento establece el marco contractual que regula los derechos y obligaciones entre SocialBeats y el Cliente, definiendo de forma clara aspectos clave del servicio como las condiciones económicas, la terminación del contrato, la responsabilidad de las partes, la propiedad intelectual y la protección de datos personales.

En particular, el acuerdo contempla garantías relevantes para el Cliente, entre las que destacan:

- Protección ante cambios de precio (Cláusula 2.5 y 2.6): La compañía está obligada a notificar con 30 días de antelación cualquier subida, otorgando al cliente el derecho a cancelar sin penalización antes de que se aplique.

- Política de Terminación Justa (Cláusula 5.2): Si la compañía decide terminar el contrato por conveniencia, se garantiza la devolución prorrateada del dinero por el tiempo de servicio no disfrutado.

- Responsabilidad (Cláusula 3.2): SocialBeats no se exime de responsabilidad en casos de negligencia grave, fraude o conducta intencional, cumpliendo con la normativa legal vigente.

- Propiedad Intelectual (Cláusula 6.2): Se especifica claramente que el contenido subido sigue siendo propiedad del usuario, otorgando a la plataforma únicamente la licencia necesaria para operar el

servicio.

- Cumplimiento Normativo (Cláusula 7): Adhesión explícita al GDPR y la LOPD española para la protección de datos.

Cláusulas abusivas detectadas por iCan

Tras analizar el *Customer Agreement* con iCan, la herramienta identificó ciertas cláusulas como potencialmente abusivas. No obstante, consideramos que estos resultados corresponden a **falsos positivos**, por los siguientes motivos:

- **Cláusula 1.2 (Aceptación del Acuerdo por uso del servicio):**
  iCan la clasifica como *contract by using*. Consideramos que no es abusiva, ya que el Cliente acepta explícitamente el Acuerdo, el SLA y la Política de Privacidad durante el proceso de registro antes de poder utilizar el servicio.

- **Cláusula 2.4 (Suspensión del servicio por impago):**
  Identificada como posible terminación unilateral. Se mantiene porque la suspensión solo se produce ante un incumplimiento objetivo del Cliente (impago) y después de un plazo de 30 días, lo que constituye una medida proporcionada.

- **Cláusula 2.6 (Aceptación de cambios de tarifas):**
  Marcada por aceptación tácita. No se considera abusiva, ya que cualquier cambio de precio se comunica con al menos 30 días de antelación y el Cliente puede cancelar el contrato sin penalización antes de que el cambio entre en vigor.

- **Cláusula 4 (Suspensión o eliminación de cuentas):**
  iCan la señala como eliminación unilateral de contenido. Se mantiene porque solo es aplicable en caso de incumplimiento de las normas de uso y es necesaria para garantizar la seguridad y el correcto funcionamiento del servicio.

- **Cláusula 5.3 (Aceptación tácita tras notificación de cambios):**
  iCan identifica esta cláusula como *contract by using*, al considerar que la aceptación se produce por el uso continuado del servicio. Consideramos que se trata de un falso positivo, ya que la aceptación solo tiene lugar tras una notificación previa clara y con un plazo de 30 días, durante el cual el Cliente puede cancelar el contrato sin penalización si no está de acuerdo con los cambios.

- **Cláusula 10 (Ley aplicable y resolución de disputas):**
  Identificada como potencialmente problemática. Sin embargo, no impone arbitraje obligatorio ni limita el acceso a la vía judicial, permitiendo al Cliente recurrir a los mecanismos legales disponibles en su jurisdicción.

2. Nivel de acabado

MICROSERVICIO BÁSICO (REQUISITOS GRUPALES)

- Debe estar desplegado y ser accesible desde la nube (ya sea de forma individual o como parte de la aplicación): **REALIZADO**.

    - El frontend está accesible en https://socialbeats.es/socialbeats/ y la API en https://api.socialbeats.es/socialbeats-api/health. Se ha usado *Digital Ocean* como proveedor Cloud, donde tenemos un cluster de Kubernetes, y se ha usado IONOS para el DNS y la gestión del dominio. La documentación de la API esta disponible en https://api.socialbeats.es/socialbeats-api/api/v1/docs/

- Integración continua: El código debe compilarse, probarse y generar la imagen de Docker automáticamente usando GitHub Actions u otro sistema de integración continua en cada commit: **REALIZADO**.

    - Para la parte de CI/CD revisar el documento de nivel de acabado de cada microservicio (pero se cumple en todos). Las releases se generan a partir de un push a la rama main de cada repositorio, y se suben a Dockerhub. El repositorio con todas las imagenes del proyecto es https://hub.docker.com/repositories/socialbeats.

MICROSERVICIO AVANZADO (REQUISITOS GRUPALES)

- Para las características avanzadas relativas al frontend, si el frontend implementado en el microservicio es parte del frontend común (ver más adelante), las características avanzadas relativas al frontend se valorarán únicamente si la parte del frontend común que ha implementado la pareja incluye estos aspectos específicamente: **REALIZADO**.
    - El frontend es común para todos los microservicios, y se encuentra en https://github.com/SocialBeats/frontend.

APLICACIÓN BASADA EN MICROSERVICIOS BÁSICA

- Interacción completa entre todos los microservicios de la aplicación integrando información. La integración debe realizarse a través del backend: **REALIZADO**.
    - Se puede observar por la existencia de commands, la gestión de eventos de kafka y el frontend en común.
- Tener un frontend común que integre los frontends de cada uno de los microservicios. Cada pareja debe ocuparse, al menos, de la parte específica de su microservicio en el frontend común: **REALIZADO**.
    - Está especificado en los detalles del nivel de acabado de cada microservicio, pero el frontend común puede encontrarse en https://github.com/SocialBeats/frontend. La plantilla básica del frontend fue realizada por Rafael Pulido y Daniel Ruiz.
- Permitir la suscripción del usuario a un plan de precios y adaptar automáticamente la funcionalidad de la aplicación según el plan de precios seleccionado: **REALIZADO**.
    - Se ha usado Space para la gestión de planes de precio en nuestros microservicios. Además, existe un microservicio dedicado a los planes de precios y suscripciones, que está disponible en https://github.com/SocialBeats/payments-and-suscriptions.
    - La gestión y división de tareas en este aspecto ha sido la siguiente:
        - En el diseño del YAML y subida a SPHERE del pricing ha sido realizado por Rafael Pulido disponible aquí.
        - La integración de SPACE con la API Gateway y gestión del Pricing Token para poder evaluar tanto en back como en front ha sido realizado por Ramón Gavira. **NOTA:** La implementación concreta del pricing en cada microservicio tanto en backend como en frontend ha sido realizada por los miembros del microservicio en cuestión (explicado más adelante).

        - Miguel Encina se ha encargado de crear las suscripciones directamente en SPACE de los planes y controlar upgrades/downgrades y contrataciones de add-ons.

APLICACIÓN BASADA EN MICROSERVICIOS AVANZADA

- Incluir add-ons al plan de precios y adaptar automáticamente la funcionalidad de la aplicación según los add-ons utilizados: **REALIZADO**.
    - En el microservicio de user-auth se ha incluido un add-on que permite personalizar la foto de perfil del usuario con decorativos. Esto ha sido limitado tanto en frontend usando el SDK `react-space-client` como en backend usando el SDK `space-node-client`. El código de esta evaluación se aprecia en el archivo `src/components/profile/ProfileHero.jsx` en las líneas 111 a 132. También se bloquea en el backend en el archivo `src/controllers/profileController.js` evaluando dicha feature a la hora de editar un perfil.
    - En beats-upload se ha añadido un add-on que consiste en promocionar un beat para que aparezca primero en los criterios de búsqueda de la vista de explorar. Se puede ver la adaptación del add-on usando space en el archivo `src/utils/spaceConnection.js` y en el métodos *togglePromotion* del archivo `src/services/beatService.js`. En el frontend, se limita este add-on en el archivo `src/pages/app/beats/BeatDetailPage.jsx`, en las lineas de la 284 a la 322.
- Incluir en el plan de precios límites de uso y aplicarlos automáticamente según la suscripción del usuario: **REALIZADO**.
    - Se ha integrado Space en los microservicios que tienen limitaciones del pricing asociadas. El plan de precios en formato YAML se puede encontrar en https://sphere.score.us.es/pricings/collections/69527907641bc8e6c0f7397d/FIS-2526-Socialbeats
        - En beats-interactions se puede ver la adaptación a los límites del pricing usando space en el archivo `src/utils/spaceConnection.js` y en los métodos *createPlaylist* y *deletePlaylist* del archivo `src/services/playlistService.js`. En el frontend, se limita esta característica en el archivo `src/pages/app/beats-interaction/playlist/CreatePlaylist.jsx` en las lineas de la 237 a la 251.
        - En user-auth se ha impuesto un límite sobre el número de certificados que un usuario puede subir a su perfil. Esto se realiza usando el SDK de node `space-node-client` y se puede comprobar en el archivo `src/controllers/uploadControllers.js`.
        - En beats-upload se puede ver la adaptación a los límites del pricing usando space en el archivo `src/utils/spaceConnection.js` y en los métodos *generatePresignedUploadUrl*, *incrementDownloads* y *deleteBeatPermanently* del archivo `src/services/beatService.js`. En el frontend, se limitan estas características en los archivos `src/pages/app/beats/BeatDetailPage.jsx`, en las lineas de la 234 a la 263; `src/components/forms/BeatForm.jsx`, en las lineas de la 229 a la 276; `src/pages/beats/MyBeatsListPage.jsx`, en las lineas de la 59 a la 80; y en `src/components/features/player/BeatDetailPlayer.jsx`, en las lineas de la 291 a la 298.
        - En analytics-and-dashboards se puede ver la adaptación a los límites del pricing usando space en el archivo `app/utils/space_connection.py` y en los métodos *create* y *delete* del archivo `app/services/dashboard_service.py`. En el frontend, se limita esta característica en el archivo `src/pages/app/dashboards/DashboardsPage.jsx` en las lineas de la 181 a la 194. Además, se ha diseñado la clase `SpaceClient` (facade) para abstraer la comunicación con la API, con el propósito de servir como biblioteca estándar

de Python para la autoadaptación de aplicaciones. El coordinador técnico de Space ha validado esta implementación y ha solicitado un Pull Request para integrarla en su repositorio oficial.

- Realizar pruebas de integración automatizadas con los otros microservicios utilizando el sistema de integración continua: **NO REALIZADO**.
- Hacer uso de un API Gateway con funcionalidad avanzada como un mecanismo de throttling o de autenticación: **REALIZADO**.
    - El api-gateway está disponible en https://github.com/SocialBeats/api-gateway. La autenticación se puede encontrar en `src/services/aggregationService.js` y `src/services/tokenValidationService.js`. El throttling en `src/middleware/rateLimiter.js`. Realizado por Daniel Vela
- Hacer uso de un sistema de comunicación asíncrono mediante un sistema de cola de mensajes para todos los microservicios. Si no es para todos, debe justificarse de forma razonada: **REALIZADO**.
    - Se ha usado Kafka para la gestión de eventos en los microservicios. Todos los microservicios consumen o crean eventos. Recomendamos consultar individualmente el uso de kafka en cada microservicio. Un ejemplo se puede ver el archivo `src/services/kafkaConsumer.js` del microservicio de beats-interaction.
- Implementación de un mecanismo para poder deshacer transacciones distribuidas: **NO REALIZADO**.
- Cualquier otra extensión a la aplicación basada en microservicios básica acordada previamente con el profesor: **REALIZADO**.
    - Se ha hecho un repositorio .github de la organización (accesible en https://github.com/SocialBeats/.github) desde donde se heredan configuraciones comunes, como workflows, el CONTRIBUTING.md, el SECURITY.md o el CODE_OF_CONDUCT.md y donde se le da un aspecto más amigable a la organización del Github. Además, se ha hecho una plantilla de microservicio (disponible en https://github.com/SocialBeats/microservice-template), de la cual todas las parejas han partido para un desarrollo más cómodo y con un flujo de trabajo ya integrado (entorno de pruebas, metodología de commits, conexión a base de datos, etc). Por último, el despliegue se ha hecho en Kubernetes, teniendo un repositorio de infraestructura para el mismo, disponible en https://github.com/SocialBeats/infrastructure. El repositorio de la plantilla ha sido realizado por Daniel Galván y Jaime Linares, y el .github así como la infraestructura por Daniel Galván. El despliegue lo realizaron Daniel Galván y Ramón Gavira, gestionando el despliegue en Digital Ocean y realizando los cambios pertinentes en la infraestructura.

NIVEL HASTA 5 PUNTOS

- Microservicio básico completamente implementado. **REALIZADO**.
    - Revisar el nivel de acabado de cada microservicio.
- Diseño de un customer agreement para la aplicación en su conjunto con, al menos, tres planes de precios que consideren características funcionales y extrafuncionales. **REALIZADO**.
    - Explicado anteriormente.
- Ficha técnica normalizada del modelo de consumo de las APIs externas utilizadas en la aplicación y que debe incluir al menos algún servicio externo de envío de correos electrónicos con un plan de precios múltiple como SendGrid. **REALIZADO**.

    - Sí. Esta disponible en https://github.com/SocialBeats/docs/blob/main/external_apis/datasheet.md. Cada pareja ha aportado la información de las APIs externas consumidas. La API de SendGrid no está incluida en ese documento, pero porque usamos la API de Resend, que es equivalente, bajo aprobación del profesor.

- Documento incluido en el repositorio del microservicio (o en el wiki del repositorio en Github) por cada pareja **REALIZADO**.

    - El documento de nivel de acabado de cada microserivico así como el de la aplicación se encuentran disponibles en https://github.com/SocialBeats/docs/tree/main/level_of_finish.

- Vídeo de demostración del microservicio o aplicación funcionando. **REALIZADO**.

    1. Video demo de la aplicación: https://youtu.be/gb9y5TR-ftw
    2. Video demo de user-auth: https://youtu.be/RqF08LsD1wM
    3. Video demo de beats-upload: https://youtu.be/QSa-v-2zmxM
    4. Video demo de beats-interaction: https://youtu.be/-L0-ZMILihI
    5. Video demo de analytics-and-dashboards: https://youtu.be/tHK0gTlxv10
    6. Video demo de social: https://www.youtube.com/watch?v=nhaZ3wjRNCM

- Presentación preparada para ser presentada en 30 minutos por cada equipo de 8/10 personas. **REALIZADO**.

    - La presentación está disponible en https://docs.google.com/presentation/d/1ks0M5fWkYYrDGw-t2npcgh1q5nsMuCrU/edit?usp=sharing&ouid=104769228190022807550&rtpof=true&sd=true así como en https://github.com/SocialBeats/docs/tree/main/presentation en pptx y en pdf. **NOTA:** La versión con la que se expuso la presentación es la que está nombrada como `VERSIÓN DEFINITIVA`, del 9 de enero de 2026 a las 13:01.

- Análisis de los esfuerzos (en horas) dedicadas por cada uno. Para esto se recomienda utilizar una herramienta de time tracking como Clockify o Toggl: **REALIZADO**

    - Están disponibles en https://github.com/SocialBeats/docs/tree/main/time_efforts. La herramienta utilizada ha sido Clockify.

NIVEL HASTA 7 PUNTOS

- Debe incluir todos los requisitos del nivel hasta 5 puntos: **REALIZADO**.

    - Revisar el nivel de acabado de cada microservicio.

- Aplicación basada en microservicios básica implementada: **REALIZADO**.

    - Explicado anteriormente.

- Análisis justificativo de la suscripción óptima de las APIs del proyecto: **REALIZADO**.

    - El documento está disponible en https://github.com/SocialBeats/docs/blob/main/external_apis/optimal_subscriptions.md. Cada pareja ha añadido el análisis de la suscripción óptima de su(s) API(s) externa(s) a ese documento.

- Al menos 3 de las características del microservicio avanzado implementados: **REALIZADO**.
    - Revisar el nivel de acabado de cada microservicio.

NIVEL HASTA 9 PUNTOS

- Un mínimo de 20 pruebas de componente implementadas incluyendo escenarios positivos y negativos: **REALIZADO**.
    - Revisar el nivel de acabado de cada microservicio.
- Tener el API REST documentado con swagger (OpenAPI): **REALIZADO**.
    - Revisar el nivel de acabado de cada microservicio.
- Al menos 5 de las características del microservicio avanzado implementados: **REALIZADO**.
    - Revisar el nivel de acabado de cada microservicio.
- Al menos 3 de las características de la aplicación basada en microservicios avanzada implementados: **REALIZADO**.
    - Explicado anteriormente.

NIVEL HASTA 10 PUNTOS

- Al menos 6 características del microservicio avanzado implementados: **REALIZADO**.
    - Revisar el nivel de acabado de cada microservicio.
- Al menos 4 características de la aplicación basada en microservicios avanzada implementados: **REALIZADO**.
    - Explicado anteriormente.
- Documento de uso de IA: **REALIZADO**.
    - El documento está disponible en https://github.com/SocialBeats/docs/blob/main/ai_use/ai_use.md.