**ESCUELA POLITÉCNICA NACIONAL**

**ESCUELA DE FORMACIÓN DE TECNÓLOGOS**

**DESARROLLO DE SISTEMA Y APLICACIÓN MÓVIL INFORMATIVA SOBRE MÚSICA NACIONAL ECUATORIANA**

**DESARROLLO DE UN FRONTEND**

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN DESARROLLO DE SOFTWARE**

**DENISSE NOEMI CUMBAL GANCHOZO**

**DIRECTOR: ING BYRON LOARTE**

**DMQ, julio, 2022**

CERTIFICACIONES

Yo, Denisse Noemi Cumbal Ganchozo declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

**  
DENISSE CUMBL**

**denisse.cumbal@epn.edu.ec**

**denissecumbal2015@hotmail.com**

Certifico que el presente trabajo de integración curricular fue desarrollado por Denisse Noemi Cumbal Ganchozo, bajo mi supervisión.

**  
Ing. BYRON LOARTE, MSc.**

**DIRECTOR**

**byron.loarteb@epn.edu.ec**

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

DENISSE NOEMI CUMBAL GANCHOZO

LOARTE CAJAMARCA BYRON GUSTAVO

DEDICATORIA

(Opcional)

AGRADECIMIENTO

(Opcional)

ÍNDICE DE CONTENIDO

[CERTIFICACIONES I](#_Toc105198101)

[DECLARACIÓN DE AUTORÍA II](#_Toc105198102)

[DEDICATORIA III](#_Toc105198103)

[AGRADECIMIENTO IV](#_Toc105198104)

[ÍNDICE DE CONTENIDO V](#_Toc105198105)

[RESUMEN VII](#_Toc105198106)

[ABSTRACT VIII](#_Toc105198107)

[1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO 1](#_Toc105198108)

[1.1 Objetivo general 1](#_Toc105198109)

[1.2 Objetivos específicos 1](#_Toc105198110)

[1.3 Alcance 2](#_Toc105198111)

[1.4 Marco Teórico 3](#_Toc105198112)

[2 METODOLOGÍA 6](#_Toc105198113)

[2.1 Metodología de Desarrollo 6](#_Toc105198114)

[Roles 6](#_Toc105198115)

[*Product Owner* 6](#_Toc105198116)

[Scrum Master 7](#_Toc105198117)

[Development Team 7](#_Toc105198118)

[Artefactos 7](#_Toc105198119)

[2.2 Diseño de interfaces 11](#_Toc105198120)

[Herramienta utilizada para el diseño 11](#_Toc105198121)

[2.3 Diseño de la arquitectura 11](#_Toc105198122)

[Patrón arquitectónico 12](#_Toc105198123)

[2.4 Herramientas de desarrollo 12](#_Toc105198124)

[3 RESULTADOS 14](#_Toc105198125)

[3.1 Sprint 0. Configuración del ambiente de desarrollo 14](#_Toc105198126)

[Descripción de requerimientos 14](#_Toc105198127)

[3.2 Sprint 1. Resultado de las interfaces generales para el usuario administrador y artista. 15](#_Toc105198128)

[Consumir varios métodos para Iniciar sesión, cerrar sesión y recuperar contraseña del rol administrativo y artista. 15](#_Toc105198129)

[Consumir varios métodos para modificar el perfil del rol administrador y artista 16](#_Toc105198130)

[3.3 Sprint 2. Resultado de las interfaces para el usuario administrador y artista 16](#_Toc105198131)

[Consumir varios métodos para gestionar las solicitudes rol administrador. 16](#_Toc105198132)

[Consumir varios métodos para registrar el rol artista. 17](#_Toc105198133)

[Consumir varios métodos para enviar la solicitud rol artista 17](#_Toc105198134)

[3.4 Sprint 3 Resultado de las interfaces para el usuario artista 17](#_Toc105198135)

[Consumir varios métodos para gestionar los géneros musicales. 18](#_Toc105198136)

[Consumir varios métodos para gestionar álbumes. 18](#_Toc105198137)

[Consumir varios métodos para gestionar canciones. 19](#_Toc105198138)

[3.5 Sprint 4 Resultado de las interfaces para el usuario artista 19](#_Toc105198139)

[Buscar canciones 19](#_Toc105198140)

[3.6 Sprint 5 Pruebas en el *frontend* 19](#_Toc105198141)

[Conclusión de la realización de las pruebas unitarias. 20](#_Toc105198142)

[Conclusión de la realización de las pruebas de compatibilidad. 20](#_Toc105198143)

[Conclusión de la realización de las pruebas de aceptación. 20](#_Toc105198144)

[Sprint 1. 20](#_Toc105198145)

[Sprint N. 20](#_Toc105198146)

[4 Conclusiones 21](#_Toc105198147)

[5 Recomendaciones 21](#_Toc105198148)

[6 REFERENCIAS BIBLIOGRÁFICAS 22](#_Toc105198149)

[7 ANEXOS 24](#_Toc105198150)

[ANEXO I 25](#_Toc105198151)

[ANEXO II 26](#_Toc105198152)

[ASPECTOS DE FORMATO (Eliminar esta página) 9](#_Toc105198153)

[Tipografía 9](#_Toc105198154)

[Espaciado 10](#_Toc105198155)

[Numeración 10](#_Toc105198156)

RESUMEN

(Máximo 250 palabras)

**PALABRAS CLAVE:** palabra1, palabra2, …, palabra6.

ABSTRACT

(Máximo 250 palabras)

**KEYWORDS:** word1, word2, …, word6.

# DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

En el Ecuador han surgido varios géneros musicales, que cuentan la historia de los antepasados indígenas, donde cada ritmo es parte de la identidad musical ecuatoriana, por ejemplo: el Pasillo refleja el espíritu del pueblo y es reconocido por el Patrimonio Cultural Inmaterial de la Humanidad como un distintivo musical ecuatoriano [1]. Por otra parte, el San Juanito como menciona Diario el Universo tiene un ritmo distintivo, originario de la provincia de Imbabura y el cual es otro distintivo musical ecuatoriano [2]. Sin embargo, en el país no existe una plataforma digital dedicada exclusivamente a los géneros ecuatorianos, en donde se realce la identidad musical ecuatoriana y en donde se dé la oportunidad a los artistas de colaborar de forma directa e indirecta en un sistema de reproducción de música. Logrando de esta manera, que los artistas opten por promover su contenido musical en plataformas internacionales como *Spotify*, *YouTube*, etc. [3] [4].

No obstante, en dichas plataformas internacionales los artistas ecuatorianos no tienen un lugar significativo en las mismas, provocando que las poblaciones del Ecuador no opten por escuchar sus canciones al momento de hacer uso de estas, por ejemplo, en el 2021 según el Diario Metro los géneros más escuchados en la plataforma de música *Spotify* por la población ecuatoriana correspondieron al Reguetón y otros géneros [5].

En el Ecuador existe el Derecho de propiedad Intelectual, la cual respalda el derecho de autor de cada artista ecuatoriano, el cual menciona “Los artistas, intérpretes y ejecutantes tienen el derecho de autorizar o prohibir la comunicación al público de sus interpretaciones o ejecuciones en directo, así como la fijación de sus interpretaciones y la reproducción de tales ejecuciones, por cualquier medio o procedimiento” [6].

Por lo antes mencionado, en el actual Trabajo de Integración Curricular se ha desarrollado un *frontend* llamado “Taki-Tri”, el cual pretende ser un espacio propio para la música ecuatoriana en donde los artistas den a conocer su música, géneros que resaltan la belleza de la mujer, álbumes que contribuyan a la identidad del pueblo y canciones que destaquen el talento del país. Logrando de esta manera que las actuales y nuevas generaciones tengan un espacio dedicado a la música nacional ecuatoriana gracias al uso de la tecnología.

## Objetivo general

Desarrollar un sistema y aplicación móvil informativa sobre música nacional ecuatoriana.

## Objetivos específicos

1. Levantar requerimientos para el *frontend* de música nacional ecuatoriana.
2. Diseñar el *frontend* en base a los requerimientos que se han obtenido.
3. Codificar los módulos del *frontend* en base a los requerimientos que se han obtenido
4. Probar el *frontend* y realizar el respectivo despliegue a producción para su difusión.

## Alcance

El presente proyecto se contempla de un componente para desarrollar una aplicación del lado del cliente o comúnmente llamado *frontend*, el cual permite proporcionar al usuario la facilidad de realizar acciones de forma sencilla, esto es gracias al desarrollo web que se centra en trabajar correctamente con el consumo de datos, experiencia de usuario y la correcta integración con una serie de interfaces adaptables a cualquier dispositivo [7].

Es por esta razón, que se ha desarrollado un *frontend* que se enfoca principalmente en proporcionar una experiencia de usuario significativa, correcto funcionamiento, adaptabilidad, integración y comunicación con un *backend* previamente desarrollado mediante el consumo de una serie de métodos para que toda la información sea visualizada en tiempo real y de forma organizada en cada una de las interfaces de usuario del *frontend*, así como el respectivo reproductor de música ecuatoriana. Utilizando para ello, una amplia gama de tecnologías de desarrollo actuales del lado del cliente, un patrón de arquitectura para una correcta organización a nivel de código, una metodología de desarrollo para el seguimiento y cumplimento de los objetivos del proyecto y una serie de pruebas para garantizar la calidad del producto *software* [8]. Por último, el acceso a los datos se garantiza en su totalidad al contar con una serie de perfiles y acciones que se describen a continuación:

**Perfiles dentro del *frontend***

* Administrador y Artista

**Información general**

* Consumir un método para presentar información en una página informativa.

**Usuario administrador**

* Consumir varios métodos para iniciar sesión, cerrar sesión y recuperar contraseña.
* Consumir varios métodos para modificar el perfil de usuario.
* Consumir varios métodos para gestionar las solicitudes de artistas.

**Usuario artista**

* Consumir varios métodos para iniciar sesión, cerrar sesión y recuperar contraseña.
* Consumir varios métodos para modificar el perfil de usuario.
* Consumir varios métodos para gestionar géneros musicales.
* Consumir varios métodos para gestionar álbumes.
* Consumir varios métodos para gestionar canciones.

## Marco Teórico

En los últimos años la forma en que las personas disfrutan escuchar música ha ido cambiando radicalmente, siendo cada vez más fácil de realizar esta acción, esto es a causa de que la industria musical de ha digitalizado, haciendo que gran cantidad de canciones de distintos géneros y álbumes se encuentre en Internet [9], por ello al tener un *frontend* que proporcione al artista la oportunidad de llegar a sus fans de forma sencilla y rápida, abre sus posibilidades de seguir con su carrera musical.

Los servicios de *Streaming* de música, le dan al artista la libertar de gestionar su contenido, un control casi totalitario dentro de su carrera musical, especialmente a los artistas independientes, estos medios digitales les proporcionan la oportunidad de decidir como difundir su música, mediante el consumo de sus canciones en estos medios digitales, además de disminuir el plagio de sus canciones según indica que los *Streaming* de música están causando que las personas no opten por la piratería al momento de escuchar música, ya que existen otras formas más sencillas y rápidas de realizar esta acción de forma gratuita [9].

Es por ello, que para llevar a cabo el desarrollo de un *frontend* es necesario utilizar varias tecnologías del lado del cliente, las cuales admiten la interacción del usuario con varios elementos como: menús, imágenes, botones, entre otros, donde la usabilidad permite una experiencia del usuario satisfactoria, para ello los principales lenguajes que permiten el desarrollo de aplicaciones del lado de *frontend* son: JavaScript, HTML y CSS pero no son los únicos pues también existen los *Frameworks* que se utilizan como plantillas para obtener un código más estructurado y ordenado como lo son Angular [10] y Vue [11] por citar algunos proporcionando estabilidad y escalabilidad a la misma cuando se requiera [7].

Las metodologías agiles son técnicas utilizadas en el desarrollo de *software*, mediante el seguimiento de varios pasos se obtiene un *software* de forma alífera y configurable a las necesidades del consumidor, donde su prioridad es poder complacer al cliente con entregas continuas. Permitiendo de esta manera, responder de una forma rápida y organizada a los problemas que se presente durante alguna de las etapas de la metodología [12].

Angular es un *Framework* que permite trabajar en una sola página, de esa forma se evita que la aplicación sea lenta o demore en cargar sus componentes. Además, integra de forma organizada los módulos que son necesarios dentro de un sistema como lo son vistas, rutas, controladores logrando de eta manera tener una programación modular [13].

La metodología ágil *Scrum* permite tener un control de cada uno de los errores y resolverlos de forma rápida y segura en conjunto con el grupo de desarrolladores que reflejan el compromiso, el enfoque y el respeto para lograr de esta manera cumplir el objetivo del proyecto y seguimiento minucioso de cada fase de desarrollo [14].

*Figma* es una plataforma colaborativa para el diseño de interfaces pensada para que los desarrolladores de *software* puedan obtener código CSS, SVG y un primer acercamiento del sistema final mediante la creación de prototipos [15]. De esa forma se implementa cada aspecto solicitado por el dueño del producto de una forma rápida y sin tantas complicaciones ya que esta plataforma permite comenzar un diseño desde cero o trabajar una plantilla acoplándose a la necesidad del usuario.

Un patrón de arquitectura dentro del desarrollo del desarrollo de un *frontend*, ayuda a prevenir incompatibilidades entre las herramientas en uso ya que se analiza la comunican entre los equipos a utilizar, garantizando así el rendimiento de cada módulo en un lapso de tiempo corto. En donde su enfoque primordial es la implementación del MVC, las cuales se enfoca en 3 capas las cuales son: Lógica, Negocio e Interfaz [16].

Los métodos que proporciona una librería permiten hacer uso de técnicas en el desarrollo que colaboran entre sí para obtener un resultado favorable ante un objetivo propuesto [17], los cuales son una solución diseñada para cumplir con lo determinado dentro del proyecto en el menor tiempo posible.

Para el consumo e integración con el *backend* se hace por medio de una serie de métodos que ya están establecidos, configurados y testeados similar como si se tratara de una API para que únicamente sea consumido por cualquier aplicación del lado del cliente o aplicación móvil sin importar el lenguaje de programación o arquitectura que se esté utilizando. Además, proporciona todo lo necesario para trabajar y gestionar con imágenes, archivos mp3, documentos y toda la información necesaria que necesita el *frontend* para poder presentar en cada una de sus interfaces de usuario sin dejar de lado la capa de seguridad [18].

Dentro del desarrollo *web* existen diferentes servicios que se acomodan a las necesidades de los clientes, los cuales permiten al proyecto que sea ágil y flexible. Esto es posible porque al construir un proyecto mediante servicios permite dividir un sistema en componentes que interactúan entre sí. Es por esta razón que al dividir el *software* en componentes *frontend* y *backend* será mucho más eficiente y permitirá la integración de más aplicaciones de terceros sin causar inconvenientes a futuro [19].

La etapa de pruebas es importante dentro del desarrollo de *software*, ya que es un enfoque primordial para comprobar que la funcionalidad sea la correcta, es decir, que realice las actividades solicitadas como se ha estipulado en la etapa inicial de requerimientos. Por otra parte, es importante centrarse en esta etapa pues se puede prevenir y corregir todos los errores antes que salga a producción [20].

El despliegue dentro del desarrollo es una parte fundamental ya que el sistema software está alojado en un servidor externo y no localmente, posibilitando que esté al alcance de los usuarios que tengan acceso a Internet [21].

# METODOLOGÍA

Un estudio de casos es una indagación profunda que permite comprender de forma detallada la realidad de un tema determinado. El objetivo esencial es obtener la información real de la situación, por medio de la extracción de datos fundamentales que se adquieren en encuestas, entrevistas etc., y así lograr resultados prósperos para un buen análisis [22].

Correspondiente a esto, el presente Trabajo de Integración Curricular maneja un estudio de casos para la realización del componente específicamente el desarrollo del *frontend*, ya que se centra en el análisis de un tema determinado para su desarrollo con la ayuda de encuestas, entrevistas e indagación en otros sitios *web*, se ha llegado a la conclusión que los artistas ecuatorianos no cuenta con una aplicación que les permita gestionar la música nacional de forma gratuita, en consecuencia a esto, se ha llevado a cabo el desarrollo de un *frontend* de música nacional ecuatoriana denominado “TAKI-TRI” en donde los artistas ecuatorianos puedan gestionar desde una sola plataforma sus géneros, álbumes y canciones de forma gratuita gracias al uso de la tecnología y herramientas de desarrollo actuales.

* 1. Metodología de Desarrollo

Las metodologías de desarrollo permiten adaptarse a las necesidades del cliente dando solución y aceptación positiva a los cambios que puedan surgir durante el proceso de desarrollo de *software* y reflejando un servicio de calidad en periodos de tiempo relativamente cortos [23].

En la actualidad el uso de metodologías ágiles, permite tener un mayor control en cada *Sprint* que se planifica, realiza y entrega, es por ello que, en el desarrollo de este *frontend* se ha implementado la metodología ágil *Scrum*, para llevar un control adecuado de cada una de las iteraciones que se han planificado realizar y entregar paulatinamente. Además, esta metodología ágil tiene un conjunto de artefactos y roles los cuales permiten guiar de manera satisfactoria todo el proceso de desarrollo de *software* desde un enfoque totalmente ágil e iterativo [12].

### Roles

*Scrum* permite incorporar varios roles con diferentes actividades dentro de un equipo de trabajo, lo cual permite que el proyecto se lo realice de forma eficiente en un lapso de tiempo relativamente corto. Por otra parte, cada uno de estos roles permiten obtener un producto final de calidad [24], a continuación, se presentan a cada uno de los roles que han sido participes en el desarrollo del *frontend.*

### *Product Owner*

Este rol se encarga de establecer los elementos del componente a realizar y que los miembros del equipo comprendan lo solicitado. Además, la persona asignada con este rol toma todas decisiones antes del desarrollo del producto *software* y dependiendo de los objetivos a cumplir [24]. En ese sentido, la **TABLA I** presenta a la persona asignada a este rol la cual posee todas las habilidades necesarias para desempeñar dicho rol mencionado anteriormente.

### *Scrum Master*

Este rol al ser el líder del equipo de desarrollo se encarga de supervisar que todo el proceso *Scrum* se lo realice de la forma correcta, además de que se esté aplicando las buenas prácticas en cada implementación del producto *software*, pues el desempeño correcto de este rol se refleja en consecuencia en la última etapa de la fase de la metodología y de la aprobación del producto final por parte del *Product Owner* [24]. En ese sentido, en la **TABLA I** se presenta a la persona asignada a este rol para desempeñar las funcionalidades anteriormente mencionadas, mediante la ejecución de reuniones semanales y una buena comunicación con todo el equipo de trabajo.

### *Development Team*

Este rol generalmente está formado por un grupo de 3 a 9 desarrolladores que de forma organizada cada uno sabe cuáles son sus actividades a realizar para lograr de esta manera obtener un incremento en cada iteración del desarrollo [24]. Por consiguiente, en la **TABLA I** en donde se visualiza el rol que se ha desempeñado.

TABLA :Designación de Roles

|  |  |
| --- | --- |
| **ROLES** | **NOMBRES** |
| *Product Owner* | Denisse Cumbal. |
| *Scrum Master* | Ing. Byron Loarte, MSc. |
| *Development Team* | Denisse Cumbal |

### Artefactos

Los artefactos dentro de *Scrum* destacan los efectos de una excelente implementación de la metodología para poder garantizar de esa manera el rendimiento y eficacia del proyecto en cada *Sprint* entregado al usuario [24]. A continuación, se presenta a cada uno de los artefactos que se han seleccionado para el desarrollo del *frontend* y que de esta manera garanticen el correcto registro de datos e información.

**Recopilación de Requerimientos**

Dentro del desarrollo del *frontend*, la Recopilación de requerimientos ayuda a tener un punto de referencia para las actividades que se realizan y también estimar tiempos, recursos, cronograma de entregables que se llevan a cabo durante todo el desarrollo del proyecto. Por otra parte, permite evaluar el avance y cumplimento del mismo [24].

La Recopilación de requerimientos para el desarrollo del *frontend* se llevan a cabo mediante una encuesta con el dueño del producto, en donde la información que se ha obtenido se refleja en la implementación y diseño de los diferentes módulos. Prueba de ello, la **TABLA II** se visualiza el formato que se ha utilizado para la elaboración de la recopilación de requerimientos. No obstante, la tabla completa con toda la información más profunda y detallada se encuentra en el **ANEXO II** del actual documento.

TABLA II:Requerimientos recopilados

|  |  |  |
| --- | --- | --- |
| **RECOPILACIÓN DE REQUERIMIENTOS** | | |
| **TIPO DEL SISTEMA** | **ID - RR** | **ENUNCIADO DEL ÍTEM** |
| ***FRONTEND*** | **RR001** | El usuario con perfil invitado, artista y administrador tienen la posibilidad de visualizar una página informativa. |

**Historias de Usuario**

Se utilizan para plasmar los requerimientos de los clientes de una forma sintetizada, corta y fácil de entender para el equipo de desarrollo, en donde se especifica que funcionalidad va a tener el producto o servicio a desarrollar. Además, estas Historias deben ser sencillas de recodar, entender y de implementarse en un periodo de tiempo determinado [24]. En consecuencia, a lo citado anteriormente cada una de las Historias de Usuario representa una de las diferentes funcionalidades del *frontend* es por ello que en la **TABLA III** se presenta un ejemplar de las Historias de Usuario que se ha realizado bajo los parámetros ya mencionados, sin embargo, las demás Historias se encuentran en el **ANEXO II** del presente documento.

TABLA : Historia de usuario 01 - Visualizar una página informativa.

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU001 | **Usuario:** Invitado |
| **Nombre historia:**  Visualizar una página informativa | |
| **Prioridad en Negocio:** Media | **Riesgo en Negocio:** Media |
| **Iteración asignada:** 2 | |
| **Responsable:** Denisse Cumbal | |
| **Descripción:** El usuario invitado en el *frontend* pueden visualizar la página informativa, además de interactuar en el menú principal con la siguiente información:   * ¿Quiénes somos? * Misión y Visión * Géneros Musicales | |
| **Observación:**  Todos los perfiles pueden visualizar la página informativa, pero necesitan iniciar sesión en la misma para tener acceso a módulos distintos dependiendo el rol correspondiente. | |

***Product Backlog***

Es un conjunto de todos los requerimientos extraídos en donde su finalidad es poder maximizar la aplicación dentro de la etapa de desarrollo. Además, dentro del contenido de este *Product Backlog* se encuentran las tareas principales y secundarias dependiendo su prioridad en el desarrollo y el estado de cada una de estas tareas [14]. En ese aspecto, la  **TABLA IV** presenta el formato que se ha realizado para plasmar los requerimientos que se han solicitado en base a la prioridad del negocio, por otra parte, la tabla completa del *Product Backlog* se encuentra en el **ANEXO II** del actual documento.

TABLA IV:*Product Backlog.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ELABORACIÓN DEL *PRODUCT BACKLOG*** | | | | |
| **ID – HU** | **HISTORIA DE USUARIO** | **ITERACIÓN** | **ESTADO** | **PRIORIDAD** |
| HU001 | Visualizar una página informativa | 1 | Finalizada | Media |
| HU002 | Iniciar sesión, cerrar sesión y recuperar contraseña | 1 | Finalizada | Alta |

**Sprint Backlog**

Es una tabla donde se encuentran todas las iteraciones del producto a desarrollar dependiendo de la prioridad del cliente y permitiendo de esta manera ver paso a paso el detalle de cada una de las tareas pendientes [14]. Por lo tanto, la **TABLA V** muestra el formato que se ha utilizado para elaborar el *Sprint Backlog* mediante en el cual se lista 5 *Sprints* que son: Configuración del ambiente de desarrollo, información principal del *frontend*, recuperación de contraseña del usuario, modificar el perfil administrador; solicitudes de artistas, registro de perfil artista, géneros musicales, álbumes, canciones, pruebas y despliegue del *frontend*. Por último, la tabla completa con el detalle de los demás *Sprints* se encuentra en el **ANEXO II** del actual documento.

TABLA : *Sprint Backlog*.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ELABORACIÓN DEL *SPRINT BACKLOG*** | | | | | | |
| **ID-SB** | **NOMBRE** | **MÓDULO** | **ID-HU** | **HISTORIA DE USUARIO** | **TAREAS** | **TIEMPO ESTIMADO** |
| SB000 | Configuración del ambiente de desarrollo | ------------ | ---------- | ------------ | * Definir los requerimientos funcionales y no funcionales del *frontend* * Configuración de las herramientas del *frontend* * Configuración del ambiente de desarrollo del *frontend* | 20 horas |

## Diseño de interfaces

Es un boceto que se crea en base a los requerimientos del dueño del producto, es decir es una parte visual en donde el usuario puede interactuar de mejor forma con los colores, estilos y demás elementos que va a tener producto una vez que empiece la etapa de codificación, logrando de esta manera tener una primera aceptación por parte del cliente y corregir posibles errores o fallas a nivel de interactividad o experiencia de usuario [25].

### Herramienta utilizada para el diseño

A través de una herramienta de diseño enfocada principalmente en el desarrollo de interfaces como es *Figma* se procede a crear los diferentes prototipos de cada uno de los módulos que corresponden al *frontend,* los cuales toman realce en la fase de desarrollo. Prueba de ello la **Fig. 1,** muestra un primer prototipo con la información general, la cual está compuesta por una serie de elementos contenedores como: *header, footer, navbar, cards* entre otros. Por último, los demás prototipos que se han diseñado por cada módulo se encuentran en el **ANEXO II** del actual documento.

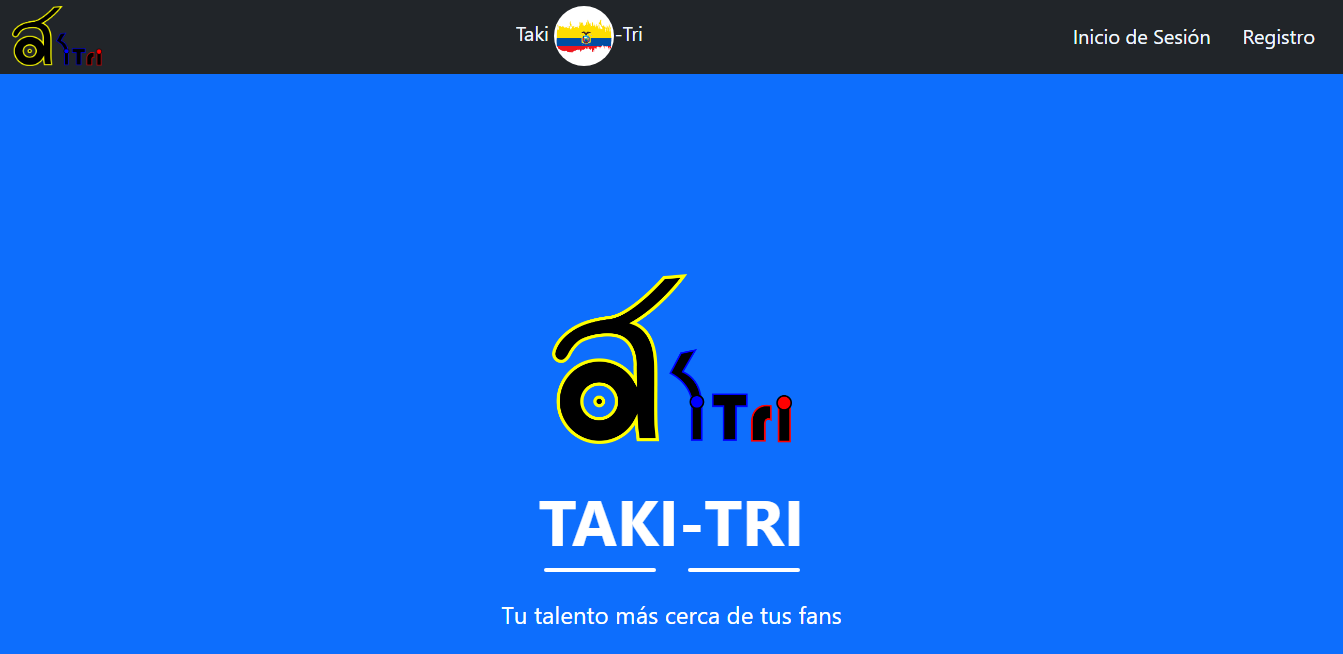


Fig. : Prototipo para mostrar la información general.

## Diseño de la arquitectura

Son las diferentes interacciones que existen entre los componentes internos de un *software* que permita un buen funcionamiento y escalabilidad a futuro e integración con otras herramientas y librerías [25]. Por consiguiente, en el desarrollo del *frontend* se ha implementado un diseño arquitectónico el cual se detalla a continuación.

### Patrón arquitectónico

El patrón arquitectónico MVC comúnmente llamado Modelo, Vista y Controlador se encarga de constituir y organizar los diferentes componentes internos de un *software*, separando la lógica, interfaz e información del mismo, que dentro del desarrollo de *software* es un modelo a seguir y que muchos sistemas o aplicaciones de gran escala lo integran como parte de núcleo, posibilitando de esta manera tener un control mucho más ordenado y estructurado en cada una de las capas del sistema *software* [26].

* **Modelo**: es la capa que gestiona todos los datos que va a interactuar con el sistema, especialmente la parte de la lógica.
* **Vista**: es la capa donde se presenta el contenido al usuario y toda la información de forma visual.
* **Controlador**: es la capa que se encarga de gestionar la información entrante por parte del usuario siendo el mediador entre el modelo y la vista.

en ese sentido, se procede a mostrar el patrón arquitectico que se ha implementado para el desarrollo del *frontend* como se visualiza en la **Fig. 2**, lo cual posibilita una alta compatibilidad con todas las librerías y herramientas de desarrollo, pruebas y despliegue.

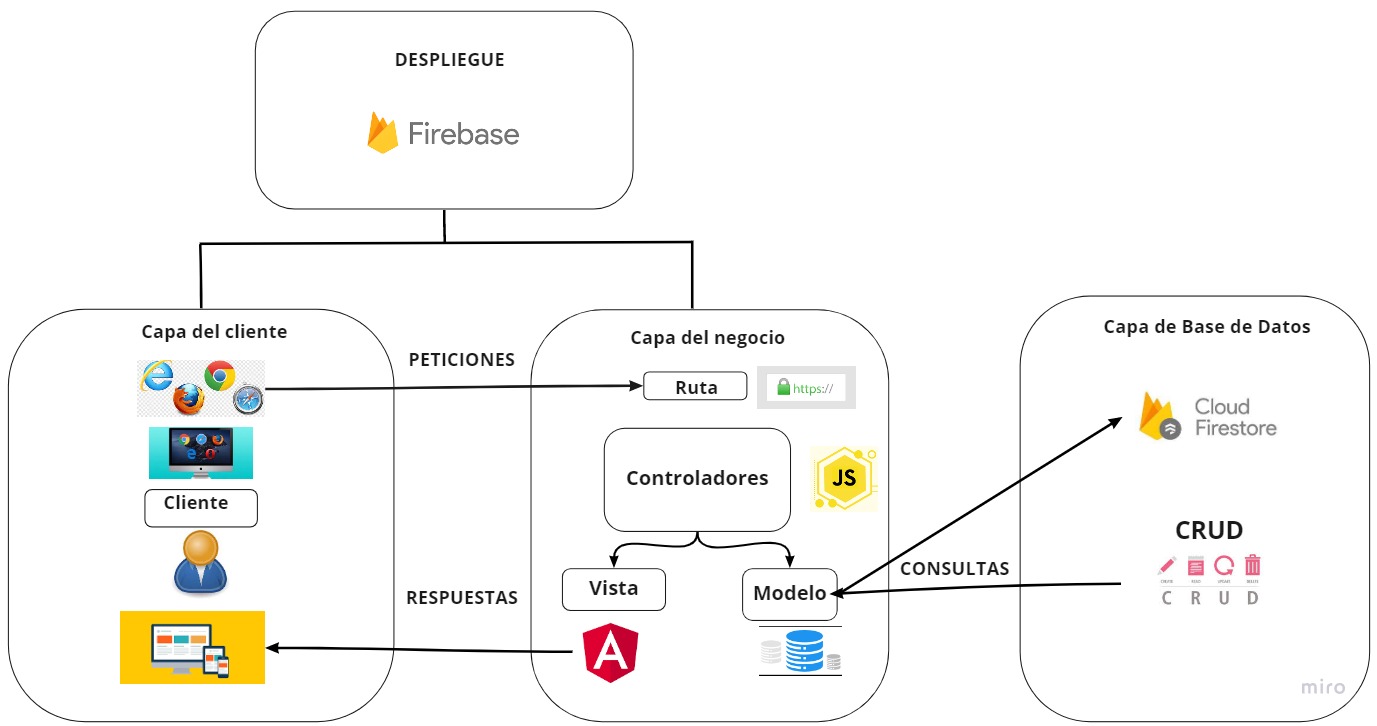


Fig. : Patrón arquitectónico – *frontend.*

## Herramientas de desarrollo

Las herramientas de desarrollo para el presente proyecto son seleccionadas en base a cada uno de los requerimientos, haciendo óptimas para el desarrollo del mismo y que sean compatibles entre ellas las cuales se describen a continuación por medio de la **TABLA VI**, la cual además muestra una breve justificación de su implementación.

TABLA : Herramientas para el desarrollo del *frontend*.

|  |  |
| --- | --- |
| **HERRAMIENTA** | **JUSTIFICACIÓN** |
| ***Angular*** | Angular es un *Framework* que permite crear aplicaciones web de una sola página (SPA) mediante el uso de *TypeScript* y una estructura bien definida [10]. |
| ***Angular CLI*** | Es un CLI o intérprete de línea de comando la cual colabora con Angular en la creación de proyectos y demás componentes por medio de una serie de comandos preestablecidos [27] |
| ***Angular Material*** | Es un conjunto de módulos y componentes visuales creados primordialmente para la parte visual de Angular, los mismos que permiten que sean adaptables a cualquier tamaño del dispositivo [28]. |

**Librerías**

Dentro del desarrollo del *frontend*, existen diferentes librerías las cuales aportan una serie de ventajas para que el proyecto sea más eficiente, evitando errores al momento de la compilación del mismo y ahorrando tiempo en cada *Sprint* que se ha planificado [31]. La **TABLA VII**, muestra una breve justificación de todas las librerías que se han utilizado.

TABLA : Librerías para el desarrollo del *frontend.*

|  |  |
| --- | --- |
| **LIBRERÍA** | **DESCRIPCIÓN** |
| ***material-app*** | Librería para el diseño de las interfaces del *frontend* [28]. |
| ***mat-icon*** | Librería para mostrar íconos en Angular [29]. |
| ***sweetalert2*** | Librería para mostrar las alertar o mensaje al usuario de una forma sencilla [30]. |

# RESULTADOS

A continuación, se presentan a detalle los resultados que se han alcanzado en cada módulo y elementos visibles que se han realizado para el competente del *frontend*, además de las diferentes pruebas que se han realizado y el respectivo despliegue del mismo. Todos los resultados que se han alcanzado se los muestran por medio de *Sprints*, los cuales se han planificado con anterioridad como se detalla en el **ANEXO II** del actual documento.

* 1. *Sprint* 0. Configuración del ambiente de desarrollo

Las tareas para el *Sprint* 0 tienen que ver con:

* Descripción de requerimientos.
* Organización del proyecto *frontend.*
* Roles de usuarios.

### Descripción de requerimientos.

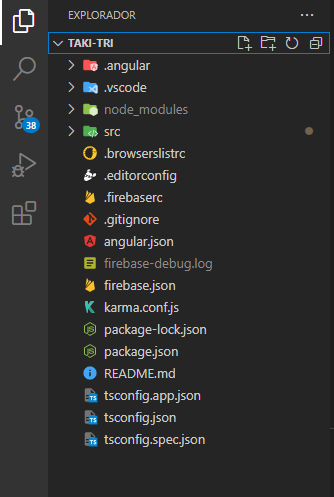
**Visualizar página informativa**.

En el *frontend* los usuarios con perfil invitado, artista y administrador pueden visualizar cada una de las páginas informativas del componente que son: ¿Quiénes Somos?, Misión, Visión y Home, mediante diversos métodos que son consumidos. Es importante mencionar que para visualizar esta página el usuario independiente de su rol no debe iniciar sesión en el *frontend.*

**Organización del proyecto *frontend***

La herramienta *Visual Studio Code* se usa como entorno de desarrollo del componente *frontend*, la cual colabora con la estructura de los diferentes módulos, directorios y archivos en correlación con la arquitectura que se ha mostrado de antemano, es por ello que en la **Fig. 3** se muestra la organización para el desarrollo*.*

Fig. : Organización del *frontend.*

**

**Roles de usuarios**

A continuidad la **Fig. 4**muestra los diferentes roles de cada usuario dentro del frontend los cuales son: administrador, artista e invitado. Al momento de iniciar sesión cada rol tiene acceso a módulos distintos, menos el rol invitado.

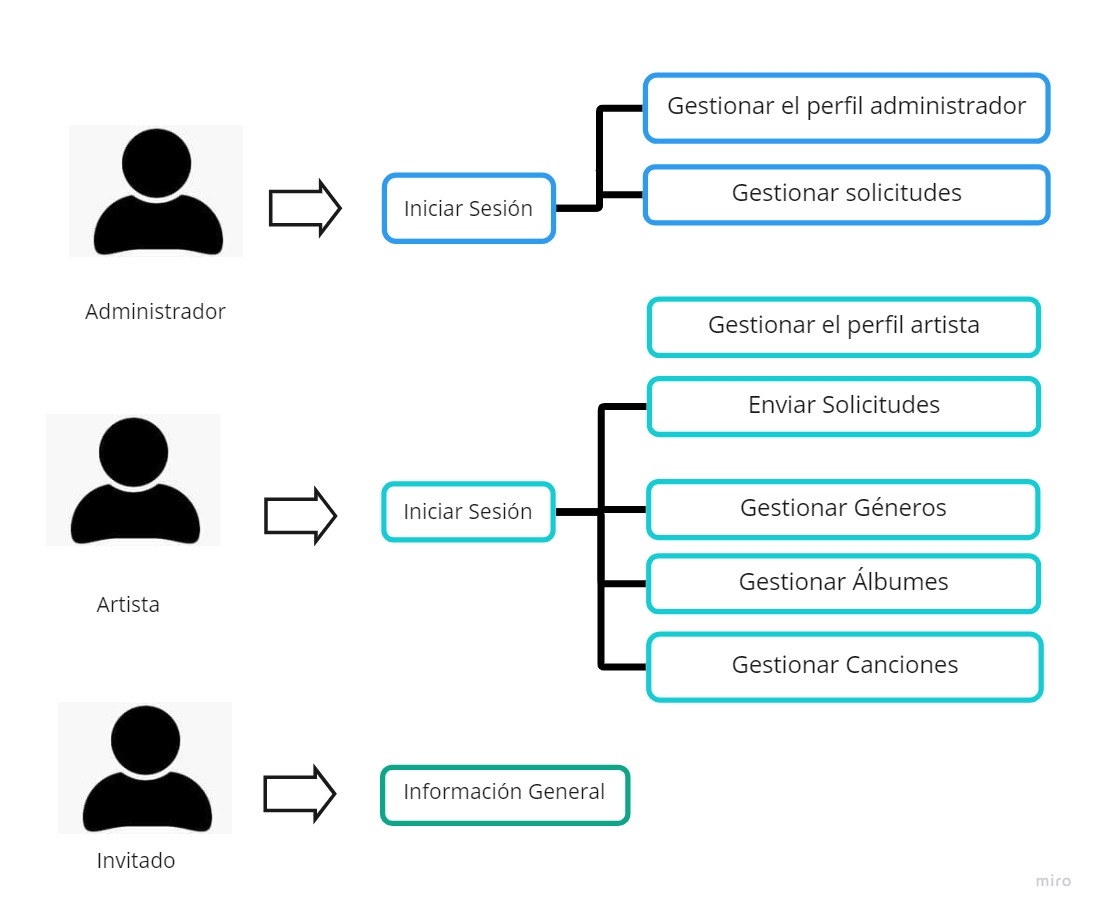


Fig. : Roles de usuario para el *frontend.*

* 1. Sprint 1. Resultado de las interfaces para el usuario administrador.

Las tareas para el *Sprint* 1 están conformado por:

* Consumir varios métodos para iniciar sesión, cerrar sesión y recuperar contraseña del rol administrativo.
* Consumir varios métodos para modificar el perfil del administrador.
* Consumir varios métodos para gestionar las solicitudes de artistas.

### Consumir varios métodos para iniciar sesión, cerrar sesión y recuperar contraseña del rol administrativo y artista.

En el *frontend* el usuario con perfil administrativo y artista tiene la posibilidad de iniciar sesión mediante un correo y contraseña generada y otorgada por el *backend* en el caso del perfil administrador, pero para el perfil artista ingresa con sus credenciales generadas por su autoría, además una vez dentro del *frontend* puede cerrar sesión, en el caso que el usuario administrador y artista se olviden sus credenciales tienen la opción de recuperar su cuenta mediante un formulario en donde se solicita el ingreso del correo electrónico. Todo lo mencionado anteriormente es posible gracias al consumo de métodos por parte del *backend* y que es presentado en los componentes visuales del *frontend* como se puede ver en las **Fig. 5 Fig. 6 y Fig. 7.** Por otra parte, el detalla del consumo y el funcionamiento completo de las funcionalidades se pueden ver de mejor manera en el **ANEXO III** del actual proyecto.

Fig. : Ingreso de usuarios administrador y artista.

Fig. : Cierre de sesión de usuarios administrador y artista.

Fig. : Recuperar contraseña de usuarios administrador y artista.

### Consumir varios métodos para modificar el perfil del administrador.

En el *frontend* el usuario con perfil administrador y artista tienen la posibilidad de visualizar su perfil que es: nombre, apellido, correo, fecha, *nickname* y una imagen por defecto de su nombre y apellido, además pueden editar su perfil en los campos seleccionados como lo es: nombre, apellido, *nickname*, entre otros. Todo esto es posible gracias al consumo de varios métodos por parte del backend y que es presentado en los componentes visuales del frontend como se puede ver en la **Fig. 8.** Por otra parte, el detalla del consumo y el funcionamiento completo de las funcionalidades se pueden ver de mejor manera en el **ANEXO III** del actual documento.

Fig. : Modificar el perfil del usuario administrador.

### Consumir varios métodos para gestionar las solicitudes de artistas.

En el *frontend* el usuario con perfil administrador tiene la posibilidad de visualizar las solicitudes de los usuarios artistas que desean ser parte de “Taki-Tri”, para poder analizar si son aceptadas o rechazadas, es importante mencionar que las solicitudes son aprobadas si cuentan con un documento para su validación y aprobación, caso contrario el administrador puede rechazar dicha solicitud, explicando porque fue rechazada y enviado al usuario artista un mensaje de la razón. Todo esto es posible gracias al consumo de varios métodos por parte del *backend* y que es presentado en los componentes visuales del *frontend* como se puede ver en las **Fig. 9, Fig.10, Fig.11.** Por otra parte, el detalla del consumo y el funcionamiento completo de las funcionalidades se pueden ver de mejor manera en el **ANEXO III** del actual documento.

Fig. : Nuevas solicitudes de artistas.

Fig.: Solicitudes aceptadas.

Fig.:Solicitudes rechazadas.

* 1. *Sprint* 3. Resultado de las interfaces para el usuario artista.

Las tareas para el Sprint 3 está conformado por:

* Consumir varios métodos para registrar el rol artista.
* Consumir varios métodos para enviar solicitud de afiliación.
* Consumir varios métodos para gestionar los géneros musicales.
* Consumir varios métodos para gestionar álbumes.
* Consumir varios métodos para gestionar canciones.
* Buscar canciones

### Consumir varios métodos para registrar el rol artista.

En el *frontend* el usuario con perfil artista tiene la posibilidad de registrarse mediante un formulario en el cual debe ingresar la siguiente información: nombre, apellido, correo, contraseña, fecha de nacimiento y *nickname*. Cabe recalcar que esta vista del registro está disponible también para el usuario con perfil invitado. Todo esto es posible gracias al consumo de varios métodos por parte del *backend* y que es presentado en los componentes visuales del *frontend* como se puede ver en la **Fig.12.** Por otra parte, el detalla del consumo y el funcionamiento completo de las funcionalidades se pueden ver de mejor manera en el **ANEXO III** del actual documento.

Fig.: Registro del rol artista.

### Consumir varios métodos para enviar solicitud de afiliación.

En el *frontend* el usuario con perfil artista tiene la posibilidad de enviar una solicitud mediante un formulario llenando la siguiente información: nombre artístico, número de cédula y un documento en formato pdf. Además, esta actividad la puede realizar varias veces hasta que sea aprobada su solicitud por medio del administrador y que de esta manera el artista pueda tener acceso a más módulos dentro del *frontend*, caso contrario solo puede navegar bajo las páginas, como home, perfil y solicitud. Todo esto es posible gracias al consumo de varios métodos por parte del *backend* y que es presentado en los componentes visuales del *frontend* como se puede ver en la **Fig. 13**. Por otra parte, el detalla del consumo y el funcionamiento completo de las funcionalidades se pueden ver de mejor manera en el **ANEXO III** del actual documento.

Fig.: Enviar solitud de afiliación.

### Consumir varios métodos para gestionar los géneros musicales.

En el *frontend* el usuario con perfil artista tiene la posibilidad de crear géneros musicales, mediante un formulario llenando la siguiente información: nombre del género, nombre del artista e imagen, además de poder editar el mismo en los campos seleccionados que son: nombre del género, nombre del artista e imagen, también tiene la opción de eliminar géneros, esta actividad solo se la realiza cuando no contenga álbumes ni canciones en él mismo. Todo esto es posible gracias al consumo de varios métodos por parte del *backend* y que es presentado en los componentes visuales del *frontend* como se puede ver en las **Fig. 14, Fig. 15** y **Fig. 16.** Por otra parte, el detalla del consumo y el funcionamiento completo de las funcionalidades se pueden ver de mejor manera en el **ANEXO III** del actual documento.

Fig.: Crear género musical.

Fig.: Editar género musical.

Fig.: Eliminar género musical.

### Consumir varios métodos para gestionar álbumes.

En el *frontend* el usuario con perfil artista tiene la posibilidad de crear álbumes, mediante un formulario llenando la siguiente información: nombre del álbum, nombre del autor, año e imagen, además de poder editar el mismo en los campos seleccionados que son: nombre del álbum, nombre del autor, año e imagen, también tiene la opción de eliminar álbumes, esta actividad solo se la realiza cuando no contenga canciones en él. Es importante mencionar que para que el usuario artista se le permita crear un álbum debe tener un género creado con anterioridad. Todo esto es posible gracias al consumo de varios métodos por parte del *backend* y que es presentado en los componentes visuales del *frontend* como se puede ver en las **Fig. 17, Fig. 18** y **Fig. 19**. Por otra parte, el detalla del consumo y el funcionamiento completo de las funcionalidades se pueden ver de mejor manera en el **ANEXO III** del actual documento.

Fig.: Crear álbum.

Fig.: Editar álbum.

Fig.: Eliminar álbum.

### Consumir varios métodos para gestionar canciones.

En el *frontend* el usuario con perfil artista tiene la posibilidad de crear canciones, mediante un formulario llenando la siguiente información: nombre de la canción y un archivo mp3, además de poder editar el mismo en los campos seleccionados que son: nombre de la canción y archivo mp3, también tiene la opción de eliminar canciones. Es importante mencionar que para que el usuario artista se le permita crear canciones debe tener creado un género y álbum con anterioridad. Todo esto es posible gracias al consumo de varios métodos por parte del *backend* y que es presentado en los componentes visuales del *frontend* como se puede ver en las **Fig. 20, Fig. 21 y Fig. 22**. Por otra parte, el detalla del consumo y el funcionamiento completo de las funcionalidades se pueden ver de mejor manera en el **ANEXO III** del actual documento.

Fig. : Subir canciones.

Fig. : Editar canciones.

Fig. : Eliminar canciones.

* 1. *Sprint 4. Resultado de las interfaces para el usuario invitado*

Las tareas para el Sprint 4 están conformado por:

* Buscar canciones

### Buscar canciones

En el *frontend* el usuario con perfil artista tiene la posibilidad de buscar canciones, ubicando el nombre de la canción, lo cual permite que se presenten los resultados obtenidos si la canción existe con la posibilidad de poder reproducirla, es importante mencionar que solo puede buscar sus canciones generadas por el mismo artista. Todo esto es posible gracias al consumo de varios métodos por parte del *backend* y que es presentado en los componentes visuales del *frontend* como se puede ver en la **Fig. 23**. Por otra parte, el detalla del consumo y el funcionamiento completo de las funcionalidades se pueden ver de mejor manera en el **ANEXO III** del actual documento.

Fig. : Buscar canciones.

* 1. *Sprint* 5 Pruebas en el *frontend*

Se termina el periodo de codificación en el *frontend*, con todos sus módulos percibidos en el componente, las tareas para el Sprint 5 están conformados por:

* Conclusión de la aplicación de las pruebas unitarias.
* Conclusión de la aplicación de las pruebas de compatibilidad.
* Conclusión de la aplicación de las pruebas de aceptación.

### Conclusión de la realización de las pruebas unitarias.

### Conclusión de la realización de las pruebas de compatibilidad.

### Conclusión de la realización de las pruebas de aceptación.

**COMENZAR CON LOS DEMÁS SPRINTS PARA TERMINAR EL CAPÍTULO 3**

# Conclusiones

Presenta lo novedoso del trabajo de integración curricular, así como evaluación del cumplimiento o no de lo propuesto en los objetivos. En el caso en que no se cumpla uno o varios objetivos, y no se logren los resultados esperados, se propone una posible respuesta que explique por qué sucedió esto o las falencias de la planteado.

1. Recomendaciones

Indicar las recomendaciones formuladas a partir del desarrollo de este trabajo de integración curricular.

# REFERENCIAS BIBLIOGRÁFICAS

[1] Instituto Nacional de Patrimonio Cultural, “El Pasillo ecuatoriano – Instituto Nacional de Patrimonio Cultural.” https://www.patrimoniocultural.gob.ec/el-pasillo-ecuatoriano/ (accessed May 31, 2022).

[2] ElUniverso, “La danza y el sanjuanito también tienen festejos | Intercultural | Noticias | El Universo.” https://www.eluniverso.com/vida-estilo/2015/06/23/nota/4979551/danza-sanjuanito-tambien-tienen-festejos/ (accessed May 31, 2022).

[3] “Escuchar es todo - Spotify.” https://www.spotify.com/ec/ (accessed May 31, 2022).

[4] “YouTube.” https://www.youtube.com/ (accessed May 31, 2022).

[5] Carlos Bolaños, “Spotify: el reggaetón fue lo más escuchado en Ecuador en 2020; aquí los artistas y canciones más reproducidas,” Dec. 02, 2020. https://www.metroecuador.com.ec/ec/noticias/2020/12/02/spotify-reggaeton-fue-lo-mas-escuchado-ecuador-2020-aqui-los-artistas-canciones-mas-reproducidas.html (accessed May 31, 2022).

[6] “Derechos Conexos – Servicios.” https://www.derechosintelectuales.gob.ec/derechos-conexos/ (accessed May 31, 2022).

[7] S. Graciela, P. Ibarra, R. Quispe, F. F. Mullicundo, D. A. Lamas, and L. Presente, “HERRAMIENTAS Y TECNOLOGÍAS PARA EL DESARROLLO WEB DESDE EL FRONTEND AL BACKEND”, Accessed: May 31, 2022. [Online]. Available: https://www.campusmvp.es/recursos/post/Desar

[8] E. Serna, R. Martínez, and P. Tamayo, “Una revisión a la realidad de la automatización de las pruebas del software A Review of Reality of Software Test Automation,” vol. 23, no. 1, pp. 169–183, 2019, doi: 10.13053/CyS-23-1-2782.

[9] Jhonny Caicedo, “La Industria Musical en las Plataformas Digitales en el Año 2021: ¿Cómo Emprender con la Música en la Era Digital?”.

[10] “Angular.” https://angular.io/ (accessed May 31, 2022).

[11] “Vue.js - The Progressive JavaScript Framework | Vue.js.” https://vuejs.org/ (accessed May 31, 2022).

[12] J. Garcia, B. J. Plat, and P. Salazar, “Métodologías Ágiles en el Desarrollo de Soware”, Accessed: May 31, 2022. [Online]. Available: www.agileuniverse.com.

[13] Zuñiga Vazquez and Luis Ramon, “Desarrollo de aplicaciones web utilizando Angular como framework””.

[14] M. T. Nevado and M. R. Palero, “Estudio de metodologías ágiles. Aplicación SCRUM”.

[15] “▷¿Qué es figma?| La mejor herramienta de prototipado web.” https://cei.es/que-es-figma/ (accessed Jun. 01, 2022).

[16] J. Camilo, G. Mejía, F. A. Vargas Agudelo, and K. G. Gil, “Marco de Trabajo para Seleccionar un Patrón Arquitectónico en el Desarrollo de Software”.

[17] “Métodos de Desarrollo de Software - Free Download PDF.” https://silo.tips/download/metodos-de-desarrollo-de-software (accessed Jun. 02, 2022).

[18] “Cloud Storage para Firebase  |  Firebase Documentation.” https://firebase.google.com/docs/storage?hl=es-419 (accessed Jun. 01, 2022).

[19] Víctor Alberto Iranzo Jiménez, “Desarrollo de software basado en microservicios: un caso de estudio para evaluar sus ventajas e inconvenientes.” https://riunet.upv.es/bitstream/handle/10251/111173/Iranzo%20-%20Desarrollo%20de%20software%20basado%20en%20microservicios%3a%20un%20caso%20de%20estudio%20para%20evaluar%20sus%20ven....pdf?sequence=1&isAllowed=y (accessed Jun. 01, 2022).

[20] C. Blanco Bueno, “Ingeniería del So8ware II.”

[21] M. D. Panizzi, G. Hodes, F. Ortiz, P. Vazquez, and R. A. Bertone, *Despliegue de sistemas de software: estado de la práctica actual en PyMES de Argentina*. [Online]. Available: https://www.cessi.org.ar/opssi.

[22] U. Autónoma De Madrid *et al.*, “Métodos de investigación educativa [EL ESTUDIO DE CASOS].”

[23] Jesús Santiago Rial Huerta, “Aplicación de Metodologías Ágiles a Desarrollo de Proyectos”.

[24] E. Bahit, “Scrum y eXtreme Programming para Programadores.”

[25] J. L. Comesaña, “DISEÑO DE INTERFACES WEB Desarrollo de Aplicaciones Web.”

[26] C. Reynoso, “Arquitectura de Soware-Manual completo.”

[27] Edison Patricio Farinango Caiza, “ESTUDIO DEL FRAMEWORK ANGULARPARA DESARROLLAR APLICACIONES SINGLE-PAGE. DESARROLLO DEL SISTEMA DE INFORMACIÓN, SEGUIMIENTO Y CONTROL PARA LA HACIENDA GANADERA”.

[28] José Carlos Fatjó, “Introducción a Angular Material.” https://tech.tribalyte.eu/blog-introduccion-angular-material (accessed Jun. 02, 2022).

[29] Arunkumar Gudelli, “Mat-Icon List : 900+ Angular Material Icons.” https://www.angularjswiki.com/angular/angular-material-icons-list-mat-icon-list/ (accessed Jun. 02, 2022).

[30] “Sweet Alert 2 - Tutorial con ejemplos - Parzibyte’s blog.” https://parzibyte.me/blog/2019/12/16/sweet-alert-2-tutorial-ejemplos/ (accessed Jun. 02, 2022).

# ANEXOS

[En caso necesario, el documento escrito deberá incluir los anexos y secciones que incorporan información que sea relevante, pero que, por su extensión, no pueden ser incorporadas directamente en ninguna de las secciones anteriores. Normalmente, en la sección de Anexos se incluyen conjuntos de datos extensos, formatos de encuestas, entrevistas, enlaces hacia videos o programas que sean producto o formen parte del Trabajo de Integración Curricular, entre otros.

Ejemplo de Anexos se muestran a continuación:]

ANEXO I. Turnitin porcentaje máximo 12%.

ANEXO I. Manual técnico

ANEXO II. Manual de usuario

ANEXO III. Manual de instalación (video)

[La numeración de los Anexos debe realizarse con números en formato romano.]

## ANEXO I

Incluir el contenido del Anexo I.

## ANEXO II

**Requerimientos recopilados**

En la **Tabla VIII** se visualizan los requerimientos alcanzados a la iniciación del proyecto en donde concuerda con lo solicitado por el usuario.

Tabla VIII:Requerimientos recopilados

|  |  |  |
| --- | --- | --- |
| **RECOPILACIÓN DE REQUERIMIENTOS** | | |
| **TIPO DEL SISTEMA** | **ID - RR** | **ENUNCIADO DEL ÍTEM** |
| ***FRONTEND*** | **RR002** | El usuario con perfil administrador y artista necesitan consumir varios métodos para:   * Iniciar sesión * Cerrar sesión. * Recuperar contraseña. |
| **RR003** | El usuario con perfil administrador necesita consumir varios métodos para   * Modificar perfil de usuario. |
| **RR004** | El usuario con perfil administrador necesita consumir varios métodos para:   * Gestionar solicitudes de artistas. |
| **RR005** | El usuario con perfil invitado necesita consumir un método para:   * Registrarse a través de un formulario. |
| **RR006** | El usuario con perfil artista necesita consumir varios métodos para   * Modificar perfil de usuario. * Enviar solicitud |
| **RR007** | El usuario con perfil artista necesita consumir varios métodos para:   * Gestionar géneros musicales. |
| **RR008** | El usuario con perfil artista necesita consumir varios métodos para:   * Gestionar álbumes. |
| **RR009** | El usuario con perfil artista necesita consumir varios métodos para:   * Gestionar canciones. |
| **RR010** | El usuario con perfil artista necesita realizar lo siguiente:   * Buscador de canciones. |

**Historias de Usuario**

Terminada la etapa de recolección de requerimientos, se da paso a crear las historias de usuarios para el *frontend*. Por ello se presenta las 9 historias de usuario, guidas en los requerimientos que comienzan desde **TABLA IX** hasta **TABLA XVII.**

TABLA IX:Ingreso y salida de TAKI-TRI por parte de los usuarios administrador y artista

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU002 | **Usuario:** Administrador & Artista |
| **Nombre historia:**  Iniciar sesión, cerrar sesión y recuperar contraseña. | |
| **Prioridad en Negocio:** Alta | **Riesgo en Negocio:** Alta |
| **Iteración asignada:** 1 | |
| **Responsable:** Denisse Cumbal | |
| **Descripción:** El usuario administrador y artista en el *frontend* puede consumir varios métodospara:   * Iniciar sesión * Cerrar sesión. * Recuperar contraseña. | |
| **Observación:**  El usuario artista necesita estar registrado para acceder a los métodosanteriormente mencionados. Además, para que el usuario administrador y artista puedan recuperar su contraseña se envía un correo a su e-mail respectivamente. | |

TABLA :Modificar perfil rol administrador

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU003 | **Usuario:** Administrador |
| **Nombre historia:**  Modificar perfil de usuario. | |
| **Prioridad en Negocio:** Media | **Riesgo en Negocio:** Media |
| **Iteración asignada:** 1 | |
| **Responsable:** Denisse Cumbal | |
| **Descripción:** El usuario administrador en el *frontend* puede consumir varios métodospara visualizar y editar su perfil por medio de los siguientes campos:   * Nombre. * Imagen. * Entre otros. | |
| **Observación:**  El usuario solo puede modificar su perfil como administrador siempre y cuando inicie sesión. | |

TABLA XI:Gestionar solicitudes de artistas

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU004 | **Usuario:** Administrador |
| **Nombre historia:**  Gestionar solicitudes de artistas | |
| **Prioridad en Negocio:** Alta | **Riesgo en Negocio:** Alta |
| **Iteración asignada:** 2 | |
| **Responsable:** Denisse Cumbal | |
| **Descripción:** El usuario administrador en el *frontend* puede consumir varios métodospara:   * Visualizar solicitudes de artistas. * Aprobar solicitudes de artistas. * Rechazar solicitudes de artistas. | |
| **Observación:**  EL usuario administrador puede visualizar las solicitudes de los artistas para aprobarlas o rechazarlas.  Se rechazan las solicitudes de los artistas cuando no esté el documento correctamente enviado por el artista al momento de registrarse.  Se aprueban las solicitudes del artista cuando conste el documente enviado correctamente por el artista al momento de registrarse. | |

TABLA XII:Registro del formulario

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU005 | **Usuario:** Invitado |
| **Nombre historia:**  Registrarse a través de un formulario. | |
| **Prioridad en Negocio:** Media | **Riesgo en Negocio:** Media |
| **Iteración asignada:** 2 | |
| **Responsable:** Denisse Cumbal | |
| **Descripción:**  El usuario artista en el *frontend* puede consumir varios métodospara registrarse por medio llenando los siguientes campos:   * Nombre. * Nickname. * Fecha de nacimiento. * Correo electrónico. | |
| **Observación:**  El usuario Invitado, debe llenar correctamente todos los campos solicitados en el registro, para que el administrador lo apruebe y con ello convertirse en usuario con perfil artista. | |

TABLA XIII:Modificar solicitud artista y enviar solicitud

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU006 | **Usuario:** Artista |
| **Nombre historia:** Modificar perfil de usuario y enviar solicitud. | |
| **Prioridad en Negocio:** Media | **Riesgo en Negocio:** Media |
| **Iteración asignada:** 2 | |
| **Responsable: Denisse Cumbal** | |
| **Descripción:** El usuario artista en el *frontend* puede consumir varios métodospara modificar, visualizar su perfil además de enviar solicitudes por medio de los siguientes campos:  Perfil   * Nombre. * Imagen de perfil. * Entre otros.   Solicitud   * Nombre artístico. * Número de cédula * Documento PDF. | |
| **Observación:**  El usuario artista es el único que puede editar su perfil, siempre y cuando inicie sesión previamente, además de llenar un formulario para que pueda acceder. a los diferentes módulos del artista. | |

TABLA XIV:Gestionar géneros musicales

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU007 | **Usuario:** Artista |
| **Nombre historia:** Gestionar géneros musicales. | |
| **Prioridad en Negocio:** Media | **Riesgo en Negocio:** Media |
| **Iteración asignada:** 3 | |
| **Responsable:** Denisse Cumbal | |
| **Descripción:** El usuario artista en el *frontend* puede consumir varios métodospara:   * Crear géneros musicales. * Visualizar géneros musicales. * Eliminar géneros musicales. * Actualizar géneros musicales. | |
| **Observación:**  El usuario artista solo puede ver sus géneros registrados por el mismo, siempre y cuando inicie sesión previamente. | |

TABLA XV:Gestionar álbumes

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU008 | **Usuario:** Artista |
| **Nombre historia:**  Gestionar álbumes. | |
| **Prioridad en Negocio:** Media | **Riesgo en Negocio:** Media |
| **Iteración asignada:** 3 | |
| **Responsable:** Denisse Cumbal | |
| **Descripción:** El usuario artista en el *frontend* puede consumir varios métodospara:   * Crear álbum. * Visualizar álbum. * Actualizar álbum. * Eliminar álbum. | |
| **Observación:**  Para crear un álbum el artista debe seleccionar un género musical previamente. | |

TABLA XVI:Subir canciones

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU009 | **Usuario:** Artista |
| **Nombre historia:**  Gestionar canciones. | |
| **Prioridad en Negocio:** Alta | **Riesgo en Negocio:** Alta |
| **Iteración asignada:** 3 | |
| **Responsable:** Denisse Cumbal | |
| **Descripción:** El usuario artista en el *frontend* puede consumir varios métodospara:   * Subir canción a un álbum. * Visualizar canciones. * Actualizar canción. * Eliminar canción. | |
| **Observación:**  Solo pueden subir una canción por solicitud, dentro de un álbum creado por el artista. | |

TABLA XVII:Buscador de canciones

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU010 | **Usuario:** Artista |
| **Nombre historia:** Buscador de canciones. | |
| **Prioridad en Negocio:** Baja | **Riesgo en Negocio:** Baja |
| **Iteración asignada:** 4 | |
| **Responsable:** Denisse Cumbal | |
| **Descripción:** El usuario artista en el *frontend* tiene la posibilidad de buscar canciones. | |
| **Observación:**  El usuario artista solo puede buscar las canciones subidas por el mismo. | |

***Product Backlog***

La **TABLA XVIII** se muestra el orden de prioridad de cada módulo implementado en el componente, los cuales son asignados tomando en cuenta la prioridad del usuario y su dificultad.

TABLA XVIII:Product Backlog

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ELABORACIÓN DEL *PRODUCT BACKLOG*** | | | | |
| **ID – HU** | **HISTORIA DE USUARIO** | **ITERACIÓN** | **ESTADO** | **PRIORIDAD** |
| HU003 | Modificar perfil de usuario. | 1 | Finalizada | Media |
| HU004 | Gestionar solicitudes de artistas. | 2 | Finalizada | Alta |
| HU005 | Registrarse a través de un formulario. | 2 | Finalizada | Media |
| HU006 | Modificar perfil de usuario y enviar solicitud. | 2 | Finalizada | Media |
| HU007 | Gestionar géneros musicales. | 3 | Finalizada | Media |
| HU008 | Gestionar álbumes. | 3 | Finalizada | Media |
| HU009 | Gestionar canciones. | 3 | Finalizada | Alta |
| HU010 | Buscador de canciones. | 4 | Finalizada | Baja |

**Sprint Backlog**

La **TABLA XIX** se muestra todos los Sprint planificados en el proyecto los cuales suman un total de 5, en donde cada uno cuenta con el tiempo estimando de cada actividad para culminar con los entregables ya definidos con el usuario.

TABLA XIX:Sprint Backlog

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ELABORACIÓN DEL *SPRINT BACKLOG*** | | | | | | |
| **ID-SB** | **NOMBRE** | **MÓDULO** | **ID-HU** | **HISTORIA DE USUARIO** | **TAREAS** | **TIEMPO ESTIMADO** |
| SB001 | Información principal del *Frontend* | Módulo – Información general | HU001 | Visualizar una página informativa | * Diseño de prototipos. * Definir los requerimientos funcionales y no funcionales del *Frontend* * Definir el menú principal. * Definir la información que se va visualizar en los diferentes módulos informativos. * Pruebas respectivas. | 25h |
| Módulo - ¿Quiénes Somos? | * Diseño de prototipos. * Definir los requerimientos funcionales y no funcionales del *frontend* * Definir la información que se va visualizar en los diferentes módulos informativos. * Pruebas respectivas. |
| Modulo -Visión y Misión | * Diseño de prototipos. * Definir los requerimientos funcionales y no funcionales del *frontend* * Definir la información que se va visualizar en los diferentes módulos informativos. * Pruebas respectivas. |
| SB001 | Inicio de sesión y recuperación de contraseña del usuario | Módulo – Inicio de Sesión rol administrador | HU002 | Iniciar sesión, cerrar sesión y recuperar contraseña | * Diseño de prototipos. * Definir los campos obligatorios para iniciar sesión y recuperar contraseña * Diseño de interfaz para el inicio de sesión y recuperar contraseña. * Crear las funciones para el inicio de sesión y recuperar contraseña del usuario administrador en el *frontend*. * Validación de los campos en iniciar sesión y recuperar contraseña. * Consumir varios métodos para iniciar sesión como administrador. * Pruebas con el usuario administrador en el *frontend*. | 25h |
| Módulo – Inicio de Sesión rol Artista | Iniciar sesión, cerrar sesión y recuperar contraseña | * Diseño de prototipos. * Definir los campos obligatorios para iniciar sesión y recuperar contraseña * Diseño de interfaz para el inicio de sesión y recuperar contraseña. * Crear las funciones para el inicio de sesión y recuperar contraseña del usuario administrador en el *frontend*. * Validación de los campos en iniciar sesión y recuperar contraseña. * Consumir varios métodos para iniciar sesión como artista. * Pruebas con el usuario artista en el *frontend.* |
| Modificar -Perfil administrador | Módulo – Modificar Perfil rol administrador | HU003 |  | * Diseño de prototipos. * Definir los campos que se mostraran en el módulo * Definir los campos que serán editables * Diseño de interfaz para el perfil del usuario administrador. * Consumir varios métodos para modificar el perfil * Pruebas con el usuario administrador en el *frontend.* |
| SB002 | *Solicitudes de artistas* | *Modulo –*  *Solicitud de artistas* | HU004 | Gestionar solicitudes de artistas | * Diseño de prototipos. * Definir los campos de visualización de las solicitudes de los artistas. * Definir los campos que serán editables. * Diseño de interfaz para las solicitudes de los artistas del usuario administrador. * Consumir varios métodos para visualizar las solicitudes. * Pruebas con el usuario administrador en el *frontend.* | 40h |
| *Modulo –*  *Solicitud de artistas aprobadas* | * Diseño de prototipos. * Definir los campos de visualización de las solicitudes de los artistas. * Definir los campos que serán editables. * Diseño de interfaz para las solicitudes de los artistas del usuario administrador. * Consumir varios métodos para visualizar y aprobar las solicitudes. * Pruebas con el usuario administrador en el *frontend.* |
| *Modulo –*  *Solicitud de artistas rechazadas* | * Diseño de prototipos. * Definir los campos de visualización de las solicitudes de los artistas. * Definir los campos que serán editables. * Diseño de interfaz para las solicitudes de los artistas del usuario administrador. * Consumir varios métodos para visualizar y rechazar las solicitudes. * Pruebas con el usuario administrador en el *frontend.* |
| Registrarse | *Modulo – Registrarse rol artista* | HU005 | Registrarse a través de un formulario | * Diseño de prototipos. * Definir los campos para el registro del usuario artista. * Diseño de interfaz para las solicitudes de los artistas del usuario administrador. * Consumir varios métodos para registrarse como artista. * Pruebas con el usuario artista en el *frontend*. |
|  | Modificar -Perfil artista y enviar solicitud | Módulo – Modificar perfil rol artista y enviar solicitud | HU006 | Modificar perfil de usuario. | * Diseño de prototipos. * Definir los campos que se mostraran en el módulo. * Definir los campos que serán editables. * Diseño de interfaz para el perfil del usuario artista * Consumir varios métodos para editar el perfil artista. * Pruebas con el usuario artista en el *frontend.* |
| SB003 | Géneros musicales | Módulo – Crear Géneros musicales | HU007 | Gestionar géneros musicales | * Diseño de prototipos. * Definir los campos para crear los géneros musicales * Diseño de interfaz para crear géneros musicales. * Consumir varios métodos para crear géneros musicales. * Pruebas con el usuario artista en el *frontend* | 65h |
| Módulo – Visualizar Géneros musicales | * Diseño de prototipos. * Definir los campos para visualizar los géneros musicales * Diseño de interfaz para visualizar géneros musicales. * Consumir varios métodos para visualizar géneros musicales. * Pruebas con el usuario artista en el *frontend* |
| Módulo – Editar y eliminar géneros musicales | * Diseño de prototipos. * Definir los campos que serán editables los géneros musicales * Diseño de interfaz para editar y eliminar géneros musicales. * Consumir varios métodos para editar y eliminar géneros musicales. * Pruebas con el usuario artista en el *frontend* |
| Álbumes | Módulo –  Crear álbumes | HU008 | Gestionar álbumes. | * Diseño de prototipos. * Definir los campos para crear los álbumes. * Diseño de interfaz para crear los álbumes. * Consumir varios métodos para crear álbumes musicales. * Pruebas con el usuario artista en el *frontend* |
| Módulo –  Visualizar álbumes | * Definir los campos para visualizar los álbumes. * Diseño de interfaz para visualizar álbumes. * Consumir varios métodos para visualizar álbumes musicales. * Pruebas con el usuario artista en el *frontend* |
| Módulo –  Editar y eliminar álbumes | * Diseño de prototipos. * Definir los campos para editar los álbumes. * Diseño de interfaz para editar y eliminar álbumes. * Consumir varios métodos para editar y eliminar álbumes musicales. * Pruebas con el usuario artista en el *frontend* |
| Canciones | Módulo –  Subir canción | HU009 | Gestionar canciones. | * Diseño de prototipos. * Definir los campos para subir las canciones. * Diseño de interfaz para subir las canciones. * Consumir varios métodos para subir canciones * Pruebas con el usuario artista en el *frontend* |
| Módulo –  visualizar canción | * Diseño de prototipos. * Definir los campos para visualizar las canciones. * Diseño de interfaz para visualizar las canciones. * Consumir varios métodos para visualizar canciones * Pruebas con el usuario artista en el *frontend.* |
| Módulo –  Editar y eliminar canción | * Diseño de prototipos. * Definir los campos para editar los álbumes. * Diseño de interfaz para editar y eliminar álbumes. * Consumir varios métodos para editar y eliminar canciones * Pruebas con el usuario artista en el *frontend* |
| SB004 | Buscador | Modulo –  Buscador de canciones | HU009 | Buscador de canciones | * Diseño de prototipos. * Diseño de interfaz para editar y eliminar álbumes. * Pruebas con el usuario artista en el *frontend* | 5h |
| SB005 | Pruebas y despliegue del *frontend* | Pruebas en el *frontend* | * Pruebas en el *frontend.* * Pruebas de carga. * Pruebas de compatibilidad. * Pruebas de aceptación. | | | 40h |
| Publicación del *frontend* | * Despliegue del *frontend* en *Firebase* | | |
| Documentación | | | * Trabajo de Integración Curricular. * Anexos. | | | 40h |
| TOTAL 480 HORAS | | | | | |  |

.

### ASPECTOS DE FORMATO (Eliminar esta página)

### Tipografía

Un resumen de la tipografía se presenta a continuación:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Texto** | **Tamaño** | **Fuente** | **Estilo** | **Ejemplo** |
| Normal | 11 | Arial | Normal; Alineado justificado. | Texto de párrafo |
| Nivel 1 | 14 | Arial | Negrita; Enumerada; Mayúscula, Con sangría; Alineado izquierda. | **1 INTRODUCCIÓN** |
| Nivel 2 | 14 | Arial | Negrita; Enumerada; Con sangría; Alineado izquierda. | **1.2 Objetivo general** |
| Nivel 3 | 12 | Arial | Negrita; Alineado izquierda. | **Subsección** |

### Espaciado

Todo el documento debe tener espaciado de 1,5. Las tablas pueden usar espaciado simple.

### Numeración

Un resumen de la numeración se presenta a continuación:

|  |  |  |
| --- | --- | --- |
| **Elemento** | **Estilo** | **Número** |
| Tabla | Superior al elemento, Alineado centro. | Continuo, Número arábigo. |
| Figura | Inferior al elemento, Alineado centro | Continuo, Número arábigo. |
| Ecuación | Inferior al elemento, Alineado centro. | Continuo, Número arábigo. |
| Páginas | Inferior  Centro | Continuo, Número romano hasta Abstract y en arábigo hasta el final. |
| Anexos |  | Continuo, Número romano. |

Extensión del documento

Se sugiere que el documento de Trabajo de Integración Curricular tenga una extensión mínima de 30 hojas, sin considerar los anexos.

Parte práctica

Para la carrera DS, el docente encargado de la revisión verificará el funcionamiento del trabajo final en conjunto con el estudiante