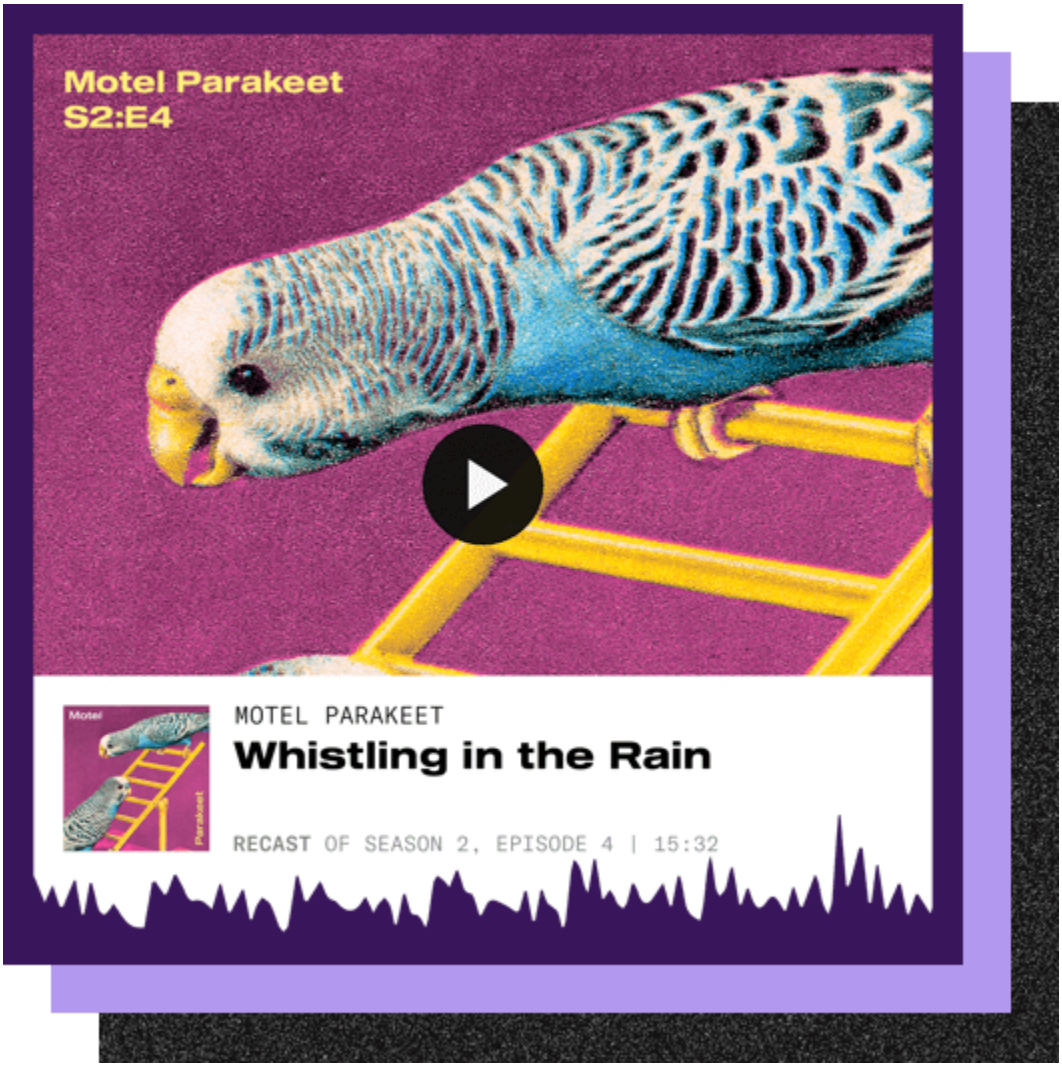
El Podcast
Manual de Programador



Alumno: Rojas Aranda José Luis
Materia: Aplicaciones Web
Profesor: Francisco Everardo Estrada Velazquez
Semestre 2020-2021 II

Introducción

En este anexo se describe la documentación técnica de programación, incluyendo la instalación del entorno de desarrollo, la estructura de la aplicación, su compilación, la configuración de los diferentes servicios de integración utilizados.

Requerimientos

El sistema puede ser desarrollado en cualquier sistema operativo que cumpla con los siguientes requerimientos

- El proyecto está desarrollado utilizando la última versión estable de laravel la cual es laravel 8.12 y PHP 7.0 o superior.
- Para la base datos se está utilizando MySql versión 8.0.23.
- Para poder desarrollar también es necesario tener instalado Node.js y NPM para el manejo de paquetes de javascript, La versión utilizada del Node.js es las v12.18 y la de NPM es 7.11.
- Para hacer uso del repositorio es necesario tener instalado el manejador de versión Git.

Instalación

1. Clonar el repositorio usado el comando: git clone https://git.example.com
2. Usando una terminal navegar al directorio para instalar las dependencias de node usando el comando: npm install
3. Una vez instalado los paquetes de Node, asegurarse de que el puerto y usuario de MySql sean correctos en el archivo .env.
4. Una vez que la configuración de la base de datos es necesario crear la base de datos directamente en la bash de mysql usando el comando: create database elpodcast;

5. Para hacer las migraciones y seeder de la base de datos correr el siguiente comando: php artisan migrate:fresh && php artisan db:seed

Compilación de proyecto para desarrollo

Para correr el proyecto en un entorno de desarrollo primero es necesario compilar los componentes de Vue, esto lo podemos hacer usando el comando npm run dev, pero se recomienda mejor utilizar el comando npm run watch, ya que este volverá a compilar cada vez que se realice un cambio al código. Finalmente para correr el servidor usar el comando php artisan serve, el cual asignará un puerto para el servidor de php.

Estructura de directorios

El repositorio del proyecto se distribuye de la siguiente manera:

- `/` : contiene archivos de dependencias, configuración del proyecto, README y licencia del proyecto
- `/app/`: módulo correspondiente a la aplicación.
- `/app/Http/`: contiene los controladores y middleware de la aplicación.
- `/app/Models/`: las clases de las entidades del proyecto.
- `/database/`: módulo correspondiente a la base de datos.
- `/database/migrations/`: las migraciones necesarias para la base de datos.
- `/database/seeders/`: los seeders con datos de pruebas
- `/public/`: carpeta public de la aplicación web.
- `/resources/`: módulo correspondiente a los recursos de la aplicación.
- `/resources/js/`: contiene la configuración y componentes de Vue
- `/resources/sass/`: estilos del app usando Sass.
- `/view/`: las vistas blade de la aplicación
- `/routes/`: las diferentes rutas de la aplicación web y API
- `/storage/`: carpeta con los recursos y archivos de la aplicación.

Laravel

El proyecto está desarrollado usando el framework de Laravel como elemento de manejo de backend, autenticación y arquitectura del proyecto. Tomando ventaja de los modelos Elocuentes nos permite modelar el problema a resolver de una manera muy eficiente.

Bootstrap

Bootstrap se está utilizando como framework de diseño combinado con estilos personalizados para una mejor experiencia para el usuario.

Vue.js

Se está utilizando Vue como framework de front-end que nos permite interfaces de usuario y single page applications. El uso de este framework nos permite reciclar de una manera más fácil diferentes componentes de interfaz y crear una interacción más completa con el usuario.

Estructura de la Aplicación

Debido a que se está utilizando Vue para el desarrollo de la interfaz la estructura de la aplicación es un poco diferente a otros proyectos de Laravel. La principal diferencia es que los archivos .blade se encarga únicamente del renderizado de los componentes de

Vue, y la información de la vista es obtenida con el API de la aplicación usando Restful Controllers de Laravel. Un ejemplo de un archivo blade es el siguiente:

```
@extends('layouts.app')

@section('content')
    <podcast-details></podcast-details>
@endsection
```

Y el componente “podcast-details” obtendrá los datos de la pantalla durante su creación usando el API.

```
<template>
    <div> … </div>
</template>

<script>
export default {
    data() {
        return {
            podcast: null,
        }
    },
    created() {
       this.podcast = axios.get(‘/api/.../.../’)
    },
}
</script>
```

Esto nos permite sacar un mayor provecho de las capacidad de Vue y también nos permite separar más el front-end del back-end lo cual en un futuro puede ser útil en caso de que se hagan cambios a la arquitectura de la aplicación.

Modelos

El nombre de las propiedades van de acuerdo a los estándares de laravel y estan en ingles, esto nos permite sacar más provecho de las funcionalidades elocuentes de laravel.

User Model

Este modelo guarda toda la información del usuario, este modelo esta basado en el modelo que Laravel proporciona con la autenticación básica. A este modelo se agrego una propiedad boolean que nos permite identificar si un usuario es creador de podcasts.

Nombre	Tipo	Descripción
name	String	Nombre del usuario
email	String	Correo del usuario
creator	Boolean	Si el usuario es un creador de podcasts, por defecto es falso.

Category Model

Este modelo guarda los diferentes tipos de categorías de podcasts que existen en la aplicación.

Nombre	Tipo	Descripción
id	String	Id de la categoría
name	String	Nombre de la categoría
image	String	Ruta con la imagen de

		previsualización de la imagen.

Podcast Model

Este modelo guarda la información de un podcast publicado en la aplicación|

Nombre	Tipo	Descripción
id	String	Id del podcast.
user_id	String	Id del usuario que creó el podcast.
category_id	String	Id de la categoría que corresponde al podcast.
title	String	Titulo del podcast.
hosts	String	Anfitriones del podcast.
short_description	String	Descripción corta del podcast.
long_description	String	La descripción corta del podcast.
image	String	Ruta de la imagen del podcast.

Episode Model

Este modelo guarda la información de los episodios del podcast.

Nombre	Tipo	Descripción
id	String	Id del episodio.

podcast_id	String	Id del podcast al que corresponde el episodio.
title	String	Título del episodio.
description	String	Descripción del podcast.
audio	String	Ruta al archivo de audio del episodio.
image	String	Ruta de la imagen del episodio.
publish_date	Date	Fecha en la que se publicó el episodio.

Subscription Model

Guarda la información de una suscripción que hizo el usuario en algún podcast.

Nombre	Tipo	Descripción
id	String	Id de la subscripción.
podcast_id	String	Id del podcast de la subscripción.
user_id	String	Id del usuario que creó la subscripción.
date	Date	Fecha en la cual se realizó la suscripción.

Restful API

Creator Routes

The creator routes allow data operations for podcast creators

`POST: /api/creator/{id}`
Convierte al usuario “id” en un creador de podcasts.

`POST: /api/creator/{id}/podcast`
Publica un nuevo podcast del creador. La información del podcast es enviada utilizando un Form.

`GET: /api/creator/{id}/podcast`
Obtiene la información todos los podcasts de ese creador.

`POST: /api/creator/{id}/podcast/{id_podcast}`
Actualiza los datos del podcast del creador.

Podcasts Routes

Las rutas de podcast permiten hacer operaciones relacionadas a la información de los podcasts

`GET: /api/podcast/categories`
Obtiene todas las categorías disponibles en la aplicación.

`GET: /api/podcast/categories/{id}`
Obtiene la información de una categoría en específico.

`GET: /api/podcast/categories/{id}/pods`
Obtiene todos los podcasts registrados en esa categoría.

`POST: /api/podcast/search`

Realiza una búsqueda de podcasts, la consulta de la búsqueda se envía a través de un Form.

`GET: /api/podcast/recommended`
Obtiene los podcasts recomendados para el usuario, esta información se utiliza en el homepage de la aplicación.

`GET: /api/podcast/{id}`
Obtiene toda la información de un podcast en específico.

`GET: /api/podcast/{id}/episodes`
Regresa todos los episodios del podcast. Esta ruta también regresa el podcast, ya que esta información se utiliza en conjunto con la de los episodios.

`POST: /api/podcast/{id}/episodes`
Publica un nuevo episodio del podcast. La información del episodio será enviada a través de un Form.

`GET: /api/podcast/{id}/episodes/{episode_id}`
Obtiene la información de un episodio en específico del podcast.

`POST: /api/podcast/{id}/episodes/{episode_id}`
Actualiza la información del episodio.

`DELETE: /api/podcast/{id}/episodes/{episode_id}`
Borra de la base de datos el episodio.

Subscription Routes

Las rutas de podcast permiten hacer operaciones relacionadas a la información de los podcasts

`GET: /api/subscriptions/{user_id}`
Obtiene todos los podcasts a los cuales está suscrito el usuario.

`GET: /api/subscriptions/{user_id}/{podcast_id}`
Regresa true si el usuario está suscrito al podcast en específico.

`POST: /api/subscriptions/{user_id}/{podcast_id}`
Subscribe al usuario al podcast en específico.

`DELETE: /api/subscriptions/{user_id}/{podcast_id}`
Borra la suscripción del usuario al podcast en específico.

Web Routes

`elpodcast.com/`
En caso de que no exista ningún usuario iniciado sesión lleva al landing page de la página. Si el usuario está autenticado, entonces lo redirecciona a /home.

`elpodcast.com/home`
Página home en donde se muestra podcasts recomendados y lo popular del momento. Esta página no se puede acceder si no está autenticado el usuario.

`elpodcast.com/subscriptions`
Página en donde se muestran los podcasts a los cuales está suscrito el usuario. Esta página no se puede acceder si no está autenticado el usuario.

`elpodcast.com/discover`
Página en donde se muestran las categorías de podcasts que existen y la opción de realizar una búsqueda de podcasts. Esta página si se puede acceder aunque no esté autenticado el usuario.

`elpodcast.com/search`
Página en donde se muestran los resultados de una búsqueda realizada por el usuario, la consulta es concatenada a la ruta como un parámetro de ruta. Esta página si se puede acceder aunque no esté autenticado el usuario.

`elpodcast.com/category/{id}`
Página en donde se muestran todos los podcasts de esa categoría en específico. Esta página no se puede acceder si no está autenticado el usuario.

`elpodcast.com/podcast/{id}`
Página en donde se muestran la información y episodios del podcast. Esta página no se puede acceder si no está autenticado el usuario.

`elpodcast.com/creator`
Página en donde se muestra los podcasts del creador, en caso de que el usuario no sea un creador, le da la opción de convertirse en uno. Esta página no se puede acceder si no está autenticado el usuario.

`elpodcast.com/newPodcast`
Página el creador puede publicar un podcast nuevo. Esta página no se puede acceder si no está autenticado el usuario.

`elpodcast.com/podcast/{id}`
Página dashboard en donde el creador puede ver los episodios publicados del podcast, publicar un nuevo podcast o editar la información del podcast. Esta página no se puede acceder si no está autenticado el usuario.

`elpodcast.com/podcast/{id}/edit`
Página en la cual el creador puede editar la información del podcast publicado. Esta página no se puede acceder si no está autenticado el usuario.

`elpodcast.com/podcast/{id}/episode`
Página en la cual el creador puede publicar un nuevo episodio del podcast. Esta página no se puede acceder si no está autenticado el usuario.

`elpodcast.com/podcast/{id}/episode/{episode_id}`
Página en la cual el creador editar la información de un episodio del podcast. Esta página no se puede acceder si no está autenticado el usuario.

Componentes de Vue

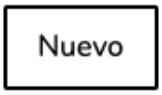
A continuación se mostraran los componentes básicos utilizados dentro del proyecto de Vue.

<fill-button>

Botón primario que se utiliza dentro de la página

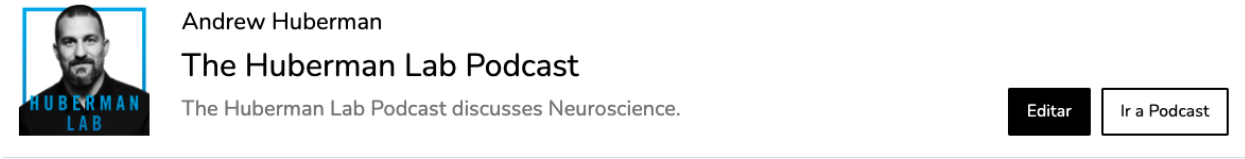
<outline-button>

Botón secundario que se utiliza dentro de la página



<podcast-row>

Renglón con la información básica del podcast, este componente se utiliza en diferentes partes dentro de la aplicación ya que las botones de operaciones pueden varias dependiendo del tipo de pantalla en la que el usuario se encuentre.

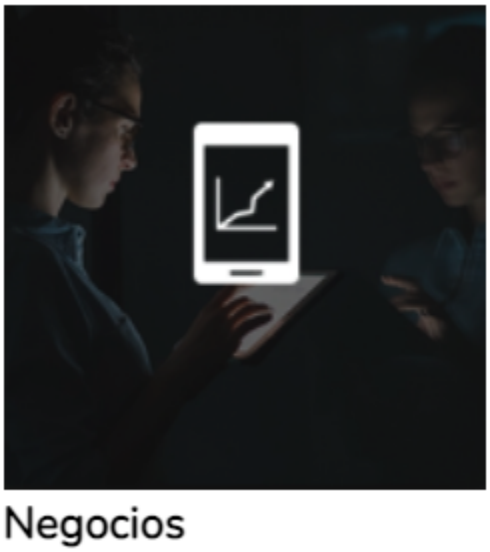


<episode-card>

Card con información básica de un episodio del podcast, también se puede tener operaciones dentro del card como lo es editar y borrar episodio

<category-card>

Card que muestra imagen y card de una categoría de pocasts

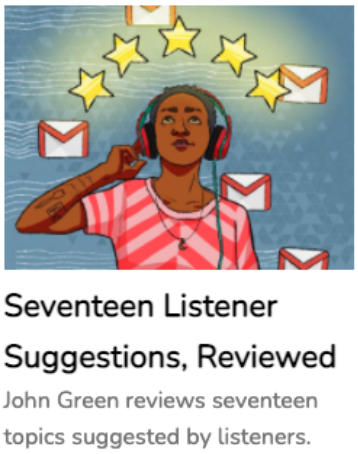


<search-bar>

Barra de búsqueda con funcionalidad integrada que se utiliza en diferentes secciones de la página.

<popular-card>

Card que muestra información básica sobre un episodio/podcast para la sección home del usuario.



<episode-row>

Renglón con información del episodio y la capacidad de reproducir el audio del episodio en específico.