ATP TOUR

Aplicación de ATP TOUR, prepara para la creación de Tenistas y Torneos que pudiera haber en un partido de tenis. Siendo posible la modificación, la eliminación y la edición de imágenes para cada torneo y tenista dentro de nuestra aplicación creada en Laravel

Laura Garrido Arredondo

Índice:

Contenido

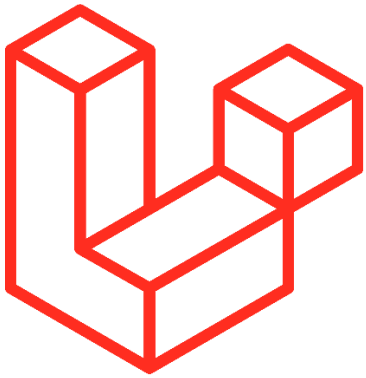
Introducción:

El propósito de este proyecto basándose en ATP es crear una aplicación web que no solo sea fácil de usar, segura y escalable, sino que también redefina y mejore significativamente la experiencia del usuario. Buscamos ofrecer una plataforma sólida y efectiva utilizando el poderoso Laravel para el desarrollo backend y PostgreSQL como sistema de gestión de bases de datos.

Nuestro objetivo principal es simplificar los procesos complejos que actualmente enfrentan nuestros usuarios en la era digital que vivimos estos días. Trabajamos para desarrollar una solución digital que no solo cumpla con sus expectativas, sino que las supere al comprender y anticipar las necesidades específicas de nuestro cliente. Queremos ofrecerles una herramienta adaptable que pueda adaptarse a sus necesidades y al entorno dinámico en el que trabajan.

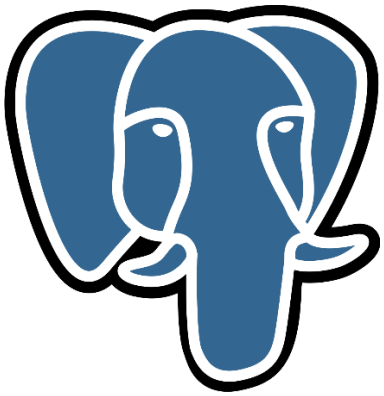
Dejando de igual manera que nuestros administradores de torneos y tenistas puedan usar estos datos de manera rápida, segura e intuitiva para nuestro cliente.

Tecnologías Utilizadas:



Descripción: Laravel, reconocido como uno de los frameworks de PHP más populares y versátiles, será la columna vertebral de nuestro proyecto. La elección de Laravel no es casual, ya que su enfoque en la simplicidad y elegancia del código nos permite desarrollar aplicaciones robustas y mantenibles. Laravel nos proporciona una amplia gama de características que facilitan el desarrollo rápido y eficiente, como el sistema de enrutamiento intuitivo, el motor de plantillas Blade y el ORM Eloquent.

Razón: Estas herramientas no solo aceleran el proceso de desarrollo, sino que también garantizan que nuestro código sea limpio y fácil de entender. Además, una de las fortalezas de Laravel es su capacidad para manejar la autenticación y autorización de usuarios de manera segura. Con funciones integradas para la gestión de usuarios, roles y permisos, podemos garantizar que nuestra aplicación proteja adecuadamente los datos sensibles y mantenga altos estándares de seguridad. Asimismo, Laravel ofrece soporte para la creación de API robustas, lo cual es esencial para la integración con otros sistemas y aplicaciones, asegurando que nuestra solución pueda interactuar sin problemas con el ecosistema tecnológico existente de nuestros clientes.



Descripción: Complementando a Laravel, hemos seleccionado PostgreSQL como nuestro sistema de gestión de bases de datos. PostgreSQL es conocido por su fiabilidad, robustez y características avanzadas que lo convierten en una elección ideal para aplicaciones de cualquier escala. Su capacidad para manejar grandes volúmenes de datos y realizar consultas complejas con alta eficiencia es crucial para el rendimiento de nuestra aplicación.

Razón: PostgreSQL ofrece un fuerte soporte para la integridad de los datos, con funciones como claves foráneas, lo que asegura que los datos de nuestros clientes se mantengan seguras. La integración de Laravel con PostgreSQL nos permite aprovechar al máximo las capacidades de ambos sistemas. El ORM Eloquent de Laravel facilita la interacción con la base de datos de manera fluida y natural, permitiéndonos escribir consultas complejas de manera sencilla y eficiente. Además, Laravel soporta migraciones de base de datos, lo que nos permite gestionar los cambios en la estructura de la base de datos de forma controlada y segura, reduciendo el riesgo de errores y asegurando que el desarrollo y la implementación sean procesos ordenados y rápidas.

Seguridad:

La seguridad es una prioridad fundamental en el desarrollo de nuestra aplicación, asegurando que tanto la información del usuario como la integridad del sistema estén protegidas contra accesos no autorizados y vulnerabilidades potenciales.

Estrategias para la autenticación y la autorización.

Implementamos métodos de autenticación y autorización robustos para controlar el acceso a los recursos de la aplicación, garantizando que solo los usuarios con los permisos adecuados puedan acceder a información sensible o realizar acciones críticas. Esto incluye el uso de tokens de seguridad, para mantener una sesión de usuario segura y verificar su identidad en cada solicitud.

La identificación de Usuarios.

La identificación de usuarios se realiza mediante un proceso de inicio de sesión seguro, mediante el cual las credenciales del usuario se verifican a través de los registros almacenados en nuestra base de datos PostgreSQL. Las contraseñas se almacenan de manera segura utilizando técnicas de hash avanzadas.

Protección de Caminos.

Para proteger las rutas críticas de la aplicación, utilizamos software de seguridad de seguridad que verifica la autenticación y autorización del usuario antes de permitir el acceso a ciertas áreas o funcionalidades de la aplicación. Esto garantiza que solo los usuarios con los permisos necesarios puedan llevar a cabo operaciones sensibles o acceder a datos protegidos.

Pruebas y Despliegue

Para garantizar la calidad y el correcto funcionamiento de nuestra aplicación en ATP, implementamos un riguroso proceso de pruebas, seguido de un plan de despliegue cuidadosamente estructurado.

Pruebas

Nuestro enfoque de pruebas es exhaustivo y sistemático, abarcando todos los aspectos de la aplicación. Este proceso nos permite identificar y corregir errores, asegurar la compatibilidad entre diferentes componentes y validar que la experiencia del usuario cumpla con nuestras expectativas de calidad y funcionalidad.

Pruebas Unitarias con PHPUnit

En ATP, utilizamos PHPUnit, un framework de pruebas para PHP, para realizar pruebas unitarias en nuestra aplicación. Estas pruebas se centran en pequeñas unidades de código, como funciones o métodos, asegurando que funcionen como se espera de manera aislada. Este enfoque nos ayuda a detectar errores en las etapas más tempranas del desarrollo, facilitando su corrección y mejorando la estabilidad del código.

Pruebas de Integración y Funcionales

Además de las pruebas unitarias, llevamos a cabo pruebas de integración y funcionales para evaluar cómo los diferentes componentes de la aplicación trabajan juntos. Las pruebas de integración se centran en las interacciones entre módulos o servicios, asegurando que se comuniquen y cooperen correctamente. Las pruebas funcionales, por otro lado, evalúan la aplicación desde la perspectiva del usuario final, verificando que todas las características y flujos de trabajo funcionen como se espera en un entorno similar al de producción.

Despliegue

Nuestro plan de despliegue en ATP está cuidadosamente estructurado para minimizar riesgos y asegurar una transición suave hacia el entorno de producción. Implementamos una estrategia de despliegue continuo, lo que nos permite lanzar actualizaciones frecuentes y controladas. Cada despliegue es precedido por un riguroso proceso de pruebas, asegurando que solo el código más estable y probado llegue a producción.

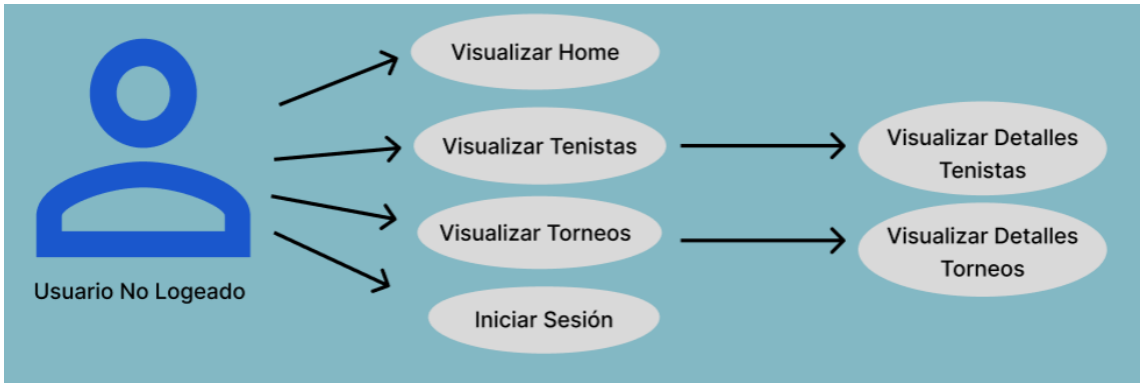
En resumen, nuestro enfoque de pruebas y despliegue en ATP está diseñado para asegurar que la aplicación no solo funcione correctamente, sino que también ofrezca una experiencia de usuario de alta calidad y sin problemas. Al combinar pruebas unitarias, de integración y funcionales con un plan de despliegue estructurado, garantizamos la estabilidad, seguridad y escalabilidad de nuestra solución digital.

Diagrama de Casos de Uso

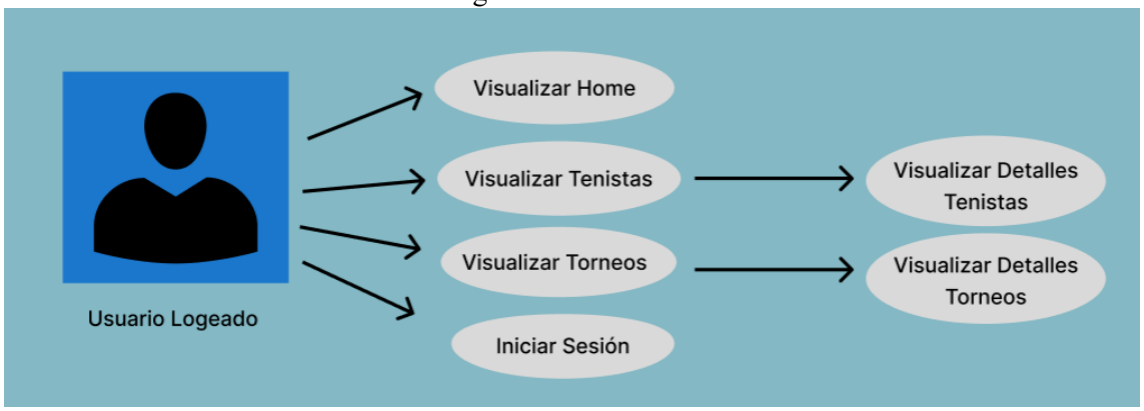
Aquí se tiene en cuenta todos nuestros casos de usos realizados en nuestra aplicación ATP en Laravel. Creada con cada una de las vistas.

Tanto como Usuarios Logeados, no logeados y administradores, en detalles se podrán imprimir los datos del tenista a ver.

- Casos de Uso de un Usuario No Logeado.



- Casos de Uso de un Usuario Logeado.



- Casos de Uso de un Administrador.

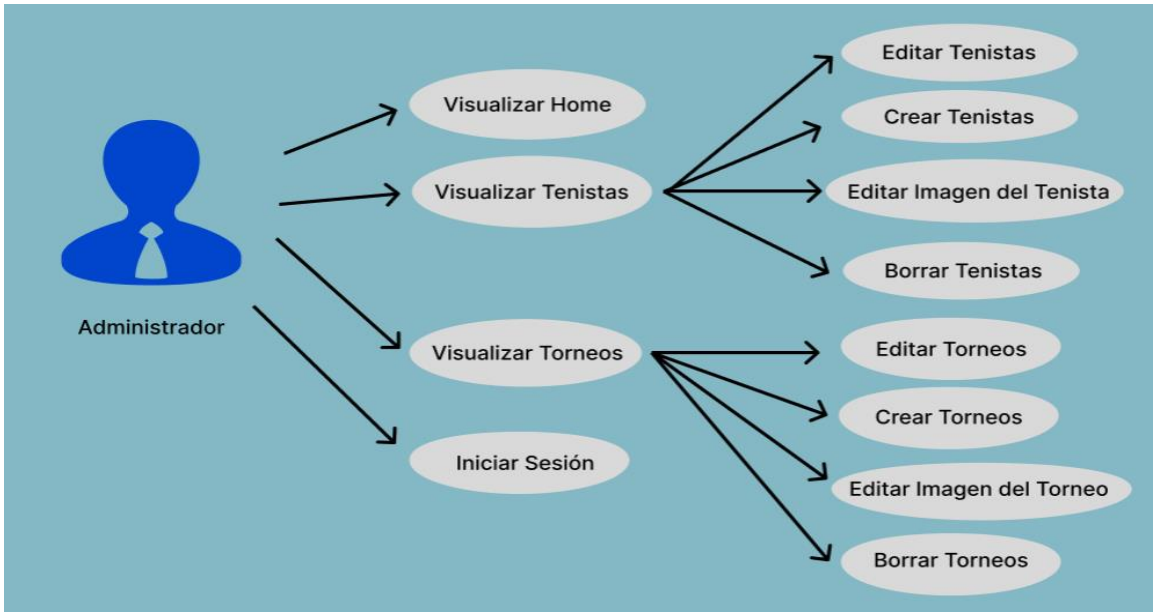


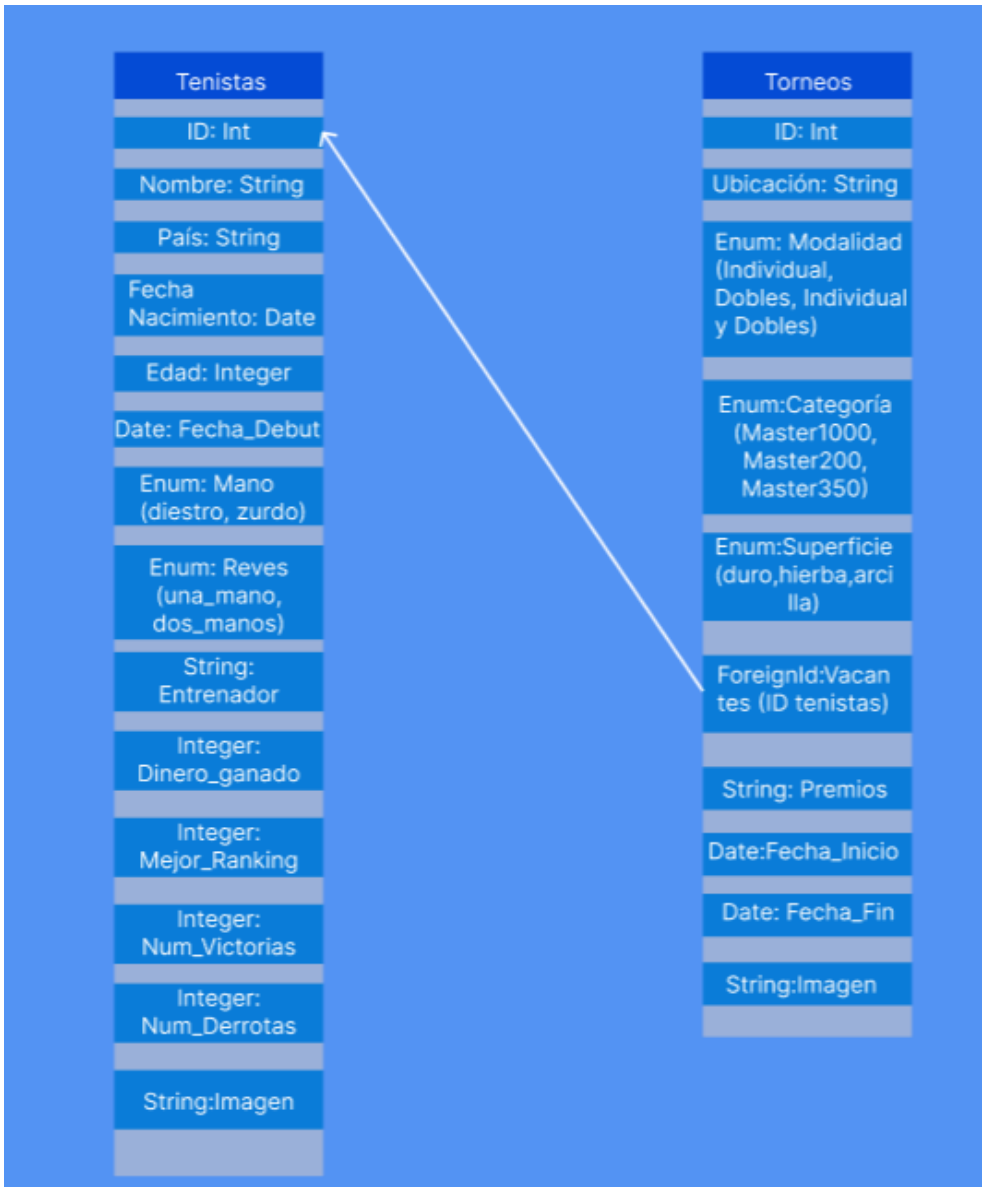
Diagrama de Clases

Aquí se tiene de referencia estos datos dentro de nuestra aplicación, siendo la base de la mayoría de las operaciones y datos que se tienen y debe tener en cuenta en base a nuestra aplicación Laravel.

Diagrama de Entidad y Relación

Aquí se tiene en cuenta Tomando en referencia el Diagrama de clases, la relación que hay entre Tenistas y Torneos.

Dando de que, si Torneos tiene algún tenista dentro de su relación, le sea imposible borrar un tenista por la relación encontrada.



Requisitos Funcionales

- Ver Detalles de Tenistas

La vista de detalles de Tenistas se mostrará tanto con Usuarios no logeados, Usuarios logeados y Administradores dentro del sistema. Dando de que Puedan visualizar todos los datos de este tenista e imprimirlas en un pdf si gustan.

- Ver Detalles de Torneos

La vista de detalles de Torneos se mostrará tanto para Usuarios y no Usuarios registrados o no en nuestra App. Dando que al igual que tenista, se visualice los datos de un torneo. A excepción de no ser impresas en un pdf.

- Añadir Tenistas

La vista de añadir tenistas se mostrará solo a administradores, donde podrá crear en la vista correspondiente, un tenista. Con los datos anteriormente mencionados en el diagrama de entidad y relación.

- Añadir Torneos

La vista de añadir torneos se mostrará solo a administradores, donde al igual que tenista, en su vista correspondiente se tendrá en cuenta los datos que se guarden en ella. Guardándose automáticamente en la Base de Datos.

- Editar Tenistas

La vista de editar Tenistas se mostrará al administrador, donde con los datos correspondientes del tenista elegido, se podrá editar todos los datos elegidos. Sin embargo, si quiere editar la imagen, tendrá que irse a la vista de Editar Imagen para ello.

- Editar Torneos

La vista de editar Torneos se mostrará solo al administrador, donde con los datos correspondientes del torneo elegido, se procederá a editar estos datos. Con excepción de su imagen, siendo editada en la vista Editar Imagen de Torneo correspondiente.

- Borrar Tenistas

La opción de borrar Tenistas tiene en cuenta de si esta registrado dentro de un Torneo, no se podrá borrar. Siendo de que salga una alerta comentando este suceso al administrador para que tome en cuenta esta eliminación.

- Borrar Torneos

La opción de borrar Torneos al no tener en cuenta si tiene algún tenista, se podrá borrar de la base de datos junto con su id para la generación de una nueva en su lugar.

Requisitos No Funcionales

- Integridad de Datos:

Garantizar la integridad de las claves primarias y referenciales en la base de datos.

Mantener consistencia en la información almacenada.

- Uso de Bycrypt para encriptación de datos:

Garantizar la integridad de seguridad de los usuarios por el uso de Bycrypt en sus contraseñas. Manteniendo estos datos solo al entrar al Login.

- Uso de DOMPDF para la generación de pdf:

Garantizar la integridad de los datos de los tenistas dentro de la base de datos con su generación de pdf en concreto.

- Uso de PHPUnit para las pruebas dentro de nuestro Laravel:

Garantizar el uso del testing gracias a los comandos de PHPUnit para comprobar que todos los datos y pruebas se realicen correctamente.

- Uso de AOS - Animate on scroll library para animaciones dentro de las vistas:

Garantizar el uso de animaciones dentro de nuestra aplicación para obtener unas vistas modernas y preparadas para ser utilizadas tanto para usuario y administrador.

Requisitos de Información

- Entidades de la Base de Datos:

  -Torneos: Almacenar información sobre los torneos disponibles dentro de nuestra aplicación, incluyendo ubicación, modalidad (Individual, Dobles, Individual y Dobles), categoría (Master1000, Master200, Master350), superficie (Duro, hierba, Arcilla), vacantes (conexión de tenistas), premios, fecha de inicio y fecha de fin de los torneos para tener en cuenta. Junto con el detalle de añadir imágenes.

  -Tenistas: Almacenar información sobre los tenistas disponibles y registrados dentro de nuestra aplicación para ser registrados en cualquier torneo disponible, junto con su nombre, país, fecha de nacimiento, edad, altura, peso, fecha de profesión, mano con el que juega (Diestro, zurdo), como hace su revés (una_mano, dos_manos), entrenador, dinero ganado, mejor ranking que ha tenido el tenista, numero de victorias y numero de derrotas, junto con el extra de añadir una imagen a este tenista.

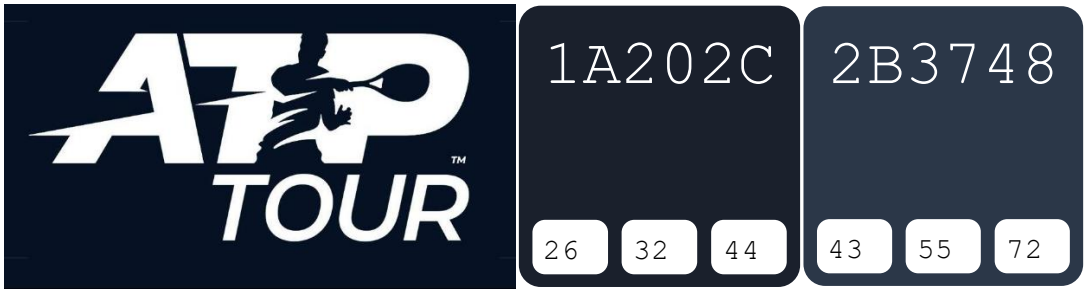
- Formato de Archivos:

  PHP y HTML: Utilizada para mostrar la información de los tenistas y torneos.
  BOOTSTRAP: Utilizada para ser mostrada las vistas de forma consistente.

Estructura de Colores

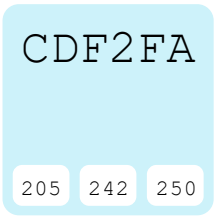
Nuestra estructura de Colores para nuestra aplicación se basa en los colores de nuestro logo de ATP. Siendo utilizados en los encabezados y nuestro footer de nuestra página web e Copyright del creador del código.

Logo de Atp junto con el código de color usado en referencia al logo para el encabezado, footer y Copyright:



También se tiene en cuenta las vistas de tenistas y torneos siendo colores simples para la simplicidad de nuestra aplicación ser para todo tipo de usuarios. Siendo utilizada en administradores y usuarios a la vez.

Código utilizado en la mayoría de las vistas fuera de home:



Y el porque el azul es una razón por el uso de nuestra marca sobre nuestros torneos y sobre el paso de la historia representándose con un azul oscuro. Sabiendo que nuestros clientes y administradores saben quienes somos con el color de nuestra marca y nuestra silueta de un tenista. Siendo registrada nuestra idea de logo a lo largo de los años.

Presupuesto

1. **Salario del desarrollador**:
    - 2,400$ al mes por desarrollador
    - Duración: 1 mes
    - Horas Mensuales: 128 $
    - Costo por Hora: 15 $
    - Total:2 ,200 $

2. **Pruebas y QA:**
    - Testeado al 85% o 90% del proyecto: $1,000

3. **Diseño UI/UX:**
    - Diseño de interfaz y experiencia de usuario: $1,000

4. **Coste de Tecnología:**
    - PHP: $1,000

5. Gestión de Usuarios:
    - Coste de Gestión de Contraseñas encriptadas: $1,200