**UNIVERSIDAD AUTÓNOMA GABRIEL RENÉ MORENO**

**FACULTAD DE INGENIERÍA EN CIENCIAS   
DE LA COMPUTACIÓN Y** **TELECOMUNICACIONES**

**SOFTWARE PARA FOTÓGRAFOS CON RECONOCIMIENTO FACIAL**

**ESTUDIANTE:** Moises Leonardo Mogiano Gutierrez

**REGISTRO:** 218034121

**CARRERA:** Ingeniería en Sistemas

**MATERIA:** Ingeniería de Software I (INF-422) – SC

**DOCENTE:** Ing. Rolando Antonio Martínez Canedo

**SEMESTRE:** 1-2023

Mayo - 2023

Santa Cruz de la Sierra – Bolivia

**INDICE**

[**1. FUNDAMENTACIÓN TEÓRICA** 4](#_Toc134693420)

[**1.1. Desarrollo de Software Basado en Componentes** 4](#_Toc134693421)

[**1.1.1. Concepto de Componente** 4](#_Toc134693422)

[**1.1.2. Características de un Componente** 4](#_Toc134693423)

[**1.1.3. Componentes e Interfaces** 5](#_Toc134693424)

[**1.1.4. Componentes en el Software PicStore** 5](#_Toc134693425)

[**1.2. Arquitectura del Software** 5](#_Toc134693426)

[**1.2.1. ¿Porque es Importante la Arquitectura del Software?** 6](#_Toc134693427)

[**1.2.2. Definición de la Estructura** 6](#_Toc134693428)

[**1.2.3. Patrón Modelo-Vista-Controlador** 6](#_Toc134693429)

[**1.2.4. Infraestructura del Software** 7](#_Toc134693430)

[**1.3. Ingeniería de Software de Sala Limpia** 9](#_Toc134693431)

[**1.3.1. Proceso de Desarrollo de Sala Limpia** 9](#_Toc134693432)

[**1.3.2. Beneficios de la Ingeniería de Software de Sala Limpia** 10](#_Toc134693433)

[**1.3.3. Buenas Prácticas Aplicadas en el Desarrollo del Software** 10](#_Toc134693434)

[**1.4. Alternativas más Efectivas en el Contenido de Fotos** 11](#_Toc134693435)

[**1.4.1. Amazon S3** 11](#_Toc134693436)

[**1.4.2. Azure Blob Storage** 11](#_Toc134693437)

[**1.5. Metodología SCRUM** 11](#_Toc134693438)

[**1.5.1. ¿Cuándo se utiliza?** 12](#_Toc134693439)

[**1.5.2. El Equipo de SCRUM** 12](#_Toc134693440)

[**1.5.3. Eventos de SCRUM** 12](#_Toc134693441)

[**1.6. C4 como Herramienta para la Arquitectura** 14](#_Toc134693442)

[**1.6.1. Descripción General** 15](#_Toc134693443)

[**1.7. Inteligencia Artificial** 16](#_Toc134693444)

[**1.7.1. Amazon Rekognition** 16](#_Toc134693445)

[**2. SPRINT 0** 19](#_Toc134693446)

[**2.1. Definiciones Iniciales** 19](#_Toc134693447)

[**2.2. Base de Datos Inicial** 21](#_Toc134693448)

[**3. SPRINT 1** 22](#_Toc134693449)

[**3.1. Sprint Backlog** 22](#_Toc134693450)

[**3.2. Historias de Usuario** 23](#_Toc134693451)

[**3.3. Sprint Review** 25](#_Toc134693452)

[**3.3. Sprint Retrospective** 26](#_Toc134693453)

[**4. SPRINT 2** 27](#_Toc134693454)

[**4.1. Sprint Backlog** 27](#_Toc134693455)

[**4.2. Historias de Usuario** 28](#_Toc134693456)

[**4.3. Sprint Review** 30](#_Toc134693457)

[**4.4. Sprint Retrospective** 31](#_Toc134693458)

[**5. MODELO C4** 32](#_Toc134693459)

[**5.1. Nivel 1: Diagrama de Contexto** 32](#_Toc134693460)

[**5.2. Nivel 2: Diagrama de Contenedor** 33](#_Toc134693461)

[**5.2. Nivel 3: Diagrama de Componentes** 34](#_Toc134693462)

[**5.4. Modelo de Datos** 35](#_Toc134693463)

[**6. GUÍA DE SOFTWARE** 41](#_Toc134693464)

[**6.1. Navegación de Página Principal** 41](#_Toc134693465)

[**6.2. Registrase** 42](#_Toc134693466)

[**6.3. Dashboard** 42](#_Toc134693467)

[**6.4. Perfil de Usuario** 43](#_Toc134693468)

[**6.5. Métodos de Pago** 43](#_Toc134693469)

[**6.6. Eventos** 44](#_Toc134693470)

[**6.7. Contrato de Fotógrafos** 46](#_Toc134693471)

[**6.8. Solicitudes de Trabajo** 47](#_Toc134693472)

[**6.9. Subida de Fotos** 47](#_Toc134693473)

[**6.10. Compra de Fotos** 48](#_Toc134693474)

[**6.11. Fotos Compradas** 49](#_Toc134693475)

# **1. FUNDAMENTACIÓN TEÓRICA**

## **1.1. Desarrollo de Software Basado en Componentes**

La ingeniería de software basada en componentes (CBSE) es un enfoque de desarrollo de software que surgió en los años 90 y se centra en la reutilización de entidades denominadas "componentes de software". Este enfoque se originó debido a las limitaciones del desarrollo orientado a objetos en cuanto a la reutilización efectiva. A diferencia de las clases de un solo objeto, los componentes de software son entidades más generales y abstractas que se pueden reutilizar en diferentes contextos y proyectos. De esta manera, CBSE permite a los desarrolladores construir software de alta calidad de manera más eficiente y económica.

### **1.1.1. Concepto de Componente**

El concepto de componente en software se refiere a una “unidad de composición de aplicaciones que posee un conjunto de interfaces y un conjunto de requisitos, y que ha de poder ser desarrollado, adquirido, incorporado al sistema y compuesto con otros componentes de forma independiente, en tiempo y espacio” (Szyperski, 1998).

### **1.1.2. Características de un Componente**

* **Estandarizada:** La estandarización del componente se refiere a que un componente que es utilizado en un proceso de CBSE, debe ajustarse a un estándar de modelo de componentes. Este modelo puede definir interfaces de componentes, metadatos del componente, documentación, composición y despliegue.
* **Independiente:** Un componente debe ser independiente, debe ser posible componerlo, y desplegarlo sin la necesidad de utilizar otro componente en específico.
* **Accesible a ser compuesto:** Para que un componente sea componible, todas las interacciones externas deben tener lugar a través de interfaces definidas públicamente. Además, se debe acceso externo a la información de este, tales como sus métodos y atributos.
* **Desplegable:** Para ser desplegable, un componente debe ser autónomo, debe ser capaz de operar como una entidad autónoma en una plataforma de componente que provee una implementación del modelo del componente. Esto usualmente significa que el modelo es binario y que no tiene que ser compilado antes de ser desplegado.
* **Documentado:** Los componentes deben ser totalmente documentados para que todos los usuarios puedan decidir si los componentes cumplen sus necesidades. La sintaxis y la semántica de las interfaces de los componentes deben ser especificadas.

### **1.1.3. Componentes e Interfaces**

Las interfaces de un componente determinan tanto las operaciones que el componente implementa como las que precisa utilizar de otros componentes durante su ejecución. En los modelos de componentes habituales cada interfaz va a venir determinada por el conjunto de atributos y métodos públicos que el componente implementa, y por el conjunto de eventos que emite. Los eventos especifican la forma en la que el componente notifica al exterior una respuesta a un estímulo externo o bien un cambio en una condición.

### **1.1.4. Componentes en el Software PicStore**

Los componentes utilizados son:

* **Página de Inicio:** Componente que dará la bienvenida a los usuarios que hayan adquirido algún paquete, el cual le mostrará las características del paquete adquirido.
* **Sidebar:** Componente que se utiliza para mostrar las opciones que tendrá cada usuario de acuerdo al paquete escogido.
* **Navbar:** Componente que se utiliza para mostrar información rápida del usuario conectado y las notificaciones del mismo.

## **1.2. Arquitectura del Software**

La arquitectura de software es el conjunto de patrones y abstracciones que proporcionan una estructura clara y definida para interactuar con el código fuente del software. Esta arquitectura se selecciona y diseña en función de los objetivos y restricciones del sistema de información. Los objetivos incluyen los requisitos funcionales y no funcionales, como el mantenimiento, la auditoría, la flexibilidad y la interacción con otros sistemas de información. Las restricciones se derivan de las tecnologías disponibles para implementar sistemas de información. La arquitectura de software es importante porque define la estructura de la aplicación y puede afectar su capacidad para cumplir con los requisitos del sistema y para ser mantenida y escalada con el tiempo.

### **1.2.1. ¿Porque es Importante la Arquitectura del Software?**

La arquitectura de software nos traza un camino por el cual seguir para lograr cumplir con los requerimientos de la aplicación, por lo tanto, tenemos que analizar cada uno de estos requerimientos para definir qué se va a hacer y cómo se va a hacer, aquí se comienza a definir temas como servidores, tecnologías, bases de datos entre otras cosas.

### **1.2.2. Definición de la Estructura**

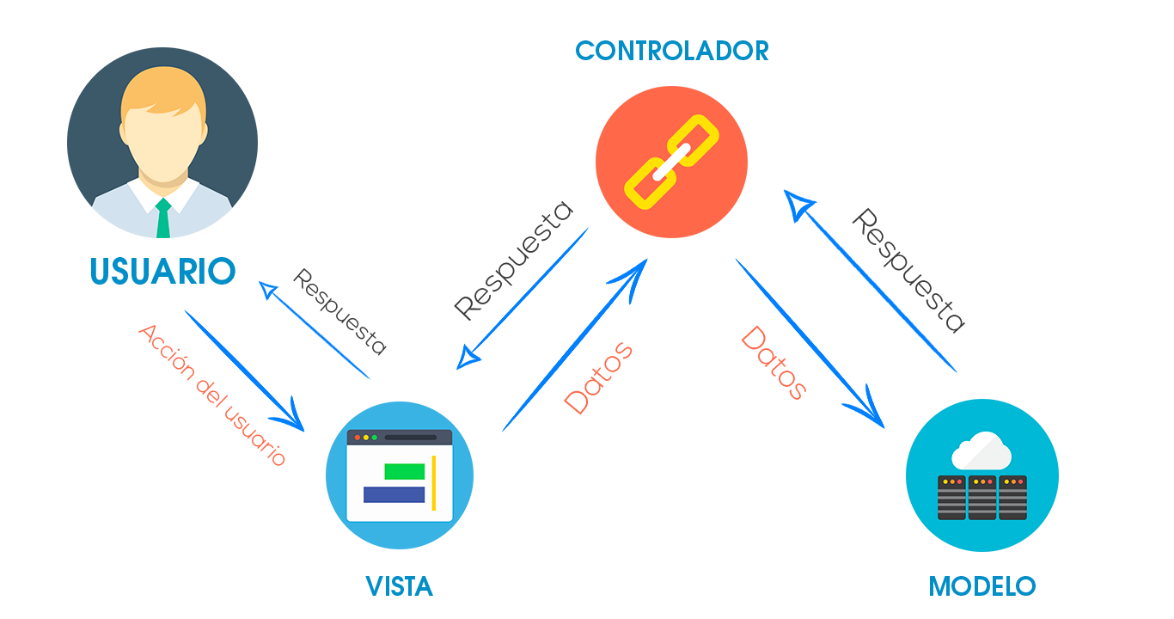
Para definir de la mejor forma la estructura de nuestra aplicación con base en la arquitectura de software tenemos 4 etapas principales:

* **Requerimientos:** En esta etapa se recolecta la información y se documentan los requerimientos que influyen en la arquitectura de la aplicación.
* **Diseño:** Es la etapa más crucial, aquí se define el uso de tecnologías adecuadas para resolver el problema y no solo porque una tecnología está de moda.
* **Documentación:** Una vez se ha definido el diseño es necesario comunicarlo de manera eficiente y eficaz a todos los involucrados, es importante crear documentación que sirva como referencia a todos y sea el marco de trabajo para todos.
* **Evaluación:** Es importante luego de tener la documentación evaluar, esto se puede hacer incluso sin haber hecho una línea de código.

### **1.2.3. Patrón Modelo-Vista-Controlador**

El patrón de arquitectura de software Modelo-Vista-Controlador (MVC) se utiliza para separar la lógica de negocio y los datos de una aplicación de su representación y manejo de eventos. MVC propone la construcción de tres componentes separados para la representación de la información (modelo), la interacción del usuario (vista) y la gestión de eventos (controlador). Este enfoque busca promover la reutilización de código y la separación de conceptos, lo que puede hacer que el desarrollo y mantenimiento de aplicaciones sea más sencillo. En resumen, el patrón MVC es una forma de organizar y estructurar el software para que sea más fácil de desarrollar y mantener.

Este patrón se divide una aplicación interactiva en tres partes diferenciadas:

* **Modelo:** Contiene la funcionalidad central y los datos.
* **Vista:** Muestra la información al usuario, siempre es posible definir una o más vistas para una misma aplicación.
* **Controlador:** Maneja la entrada del usuario. Esto se hace para separar las representaciones internas de la información de las formas en que se presenta y se acepta la información del usuario. De esta manera se desacopla los componentes y permite una reutilización eficiente del código.

*Ilustración 1: Funcionamiento de Patrón MVC*

*Fuente: Carlos Alfaro (2020). “Mi primer sistema PHP y MVC”*

### **1.2.4. Infraestructura del Software**

La infraestructura que se necesita para desplegar el software es el siguiente:

1. **Hardware**

Los recursos físicos necesarios para el funcionamiento del software serán:

* 1. **Servidor**
* Procesador Intel Core i5-10300H
* Memoria RAM 16 GB DDR4
* Tarjeta Gráfica NVIDIA GeForce GTX 1650
* Disco Sólido SSD 256 GB
* Disco Duro HDD 1 TB
  1. **Cliente**
* Conexión estable a internet
* Computadora o celular

1. **Software**

Entre los recursos lógicos se manejarán los siguientes:

* 1. **Servidor**
* **Servidor Web:** AWS ( Amazon Web Service )
* **Editor de Código:** Visual Studio Code
* **Lenguaje de Programación:** PHP
* **Framework:** Laravel 10
* **Motor Base de Datos:** PostgreSQL
* **Sistema Operativo**: Windows 10
  1. **Cliente**
* **Sistema Operativo:** Windows 10
* **Navegador:** Chrome o Microsoft Edge
* **Dispositivo:** Celular Móvil

1. **Datos**

El software manejará los siguientes datos:

* Usuario (ID, nombre, correo, contraseña, phone, photo1, photo2, locación, ci)
* Eventos(ID, nombre, tipo, lugar, imagen, fecha, hora, direccion, org\_id)
* Suscripciones(ID, nombre, precio, logo, funciones)
* Tarjetas (ID, numero, titular, fecha\_v, cvc, user\_id)
* Portafolios(ID, foto, user\_id)
* Fotografias (ID, foto, foto\_c, precio, autor\_id, tipo)
* QR (url, evento\_id)
* Notificaciones

1. **Procesos**
   1. **Proceso de Registro de Usuario**

Se registrará al usuario para que tenga acceso al sistema, cada uno con su respectivo nivel de privilegio. Estos contarán con un nombre de usuario y contraseña la cual le permitirá iniciar sesión en el sistema.

* 1. **Proceso de Registro de Evento**

Se registrarán los eventos creados con su respectiva información de acuerdo con los requerimientos del organizador como ser el tipo de evento, dirección, fecha, hora y el fotógrafo que trabajara para ese evento.

* 1. **Proceso de Registro de Portafolio**

Se podrá permitirá a los fotógrafos cargar diferentes imágenes para el portafolio, de tal manera permitiéndoles a los organizadores una mejor elección.

* 1. **Proceso de Registro de Tarjetas**

Se podrá registrar diferentes tarjetas por cada usuario, para poder comprar fotografías como recibir ganancias de las ventas realizadas.

* 1. **Proceso de Registro y Control de Fotografía**

Se utilizará una inteligencia artificial para poder controlar en que fotos del evento donde aparece el cliente y así poder notificarle que aparece en una foto.

## **1.3. Ingeniería de Software de Sala Limpia**

La ingeniería de software de Sala Limpia es un enfoque de diseño de software que se enfoca en la arquitectura de software. La arquitectura de software es un conjunto de patrones y abstracciones que proporcionan una estructura clara y definida para interactuar con el código fuente del software. La elección y diseño de la arquitectura de software se basa en objetivos y restricciones. Los objetivos incluyen los requisitos del sistema de información, así como otros objetivos como el mantenimiento, la auditoría, la flexibilidad y la interacción con otros sistemas de información. Las restricciones se derivan de las tecnologías disponibles para implementar sistemas de información.

### **1.3.1. Proceso de Desarrollo de Sala Limpia**

Los enfoques de desarrollo de software de sala limpia constan de cuatro procesos clave, es decir:

* **Gestión:** Es persistente durante toda la vida útil del proyecto, que consiste en la misión del proyecto, el cronograma, los recursos, el análisis de riesgos, la formación, la gestión de la configuración, etc.
* **Especificación:** Se considera como el primer proceso de cada incremento que consiste en análisis de requisitos, especificación de funciones, especificación de uso, planificación de incrementos, etc.
* **Desarrollo:** Se considera como el segundo proceso de cada incremento que consiste en la reingeniería del software, la verificación de la corrección, el diseño incremental, etc.
* **Certificación:** Se considera como el proceso final de cada incremento que consiste en el modelado de uso y la planificación de pruebas, la capacitación estadística y el proceso de certificación, etc.

### **1.3.2. Beneficios de la Ingeniería de Software de Sala Limpia**

* Brindar un producto de alta calidad.
* Incrementa la productividad en desarrollo.
* Reduce el costo de desarrollo.
* Los errores se encuentran temprano.
* Reduce el tiempo total del proyecto.
* Ahorra recursos.

### **1.3.3. Buenas Prácticas Aplicadas en el Desarrollo del Software**

* **Definición de Requisitos:** Es imprescindible definir a dónde se quiere llegar y cómo se hará. Para ello, se debe redactar un documento de alcance y asegurarse de que todos los participantes del proyecto (programadores, analistas, jefes de proyecto, clientes que solicitan el desarrollo y usuarios finales que harán uso de la aplicación) estén completamente de acuerdo.
* **Divide y Vencerás:** En lugar de buscar una solución que abarque todas las posibilidades o funcionalidades, es recomendable dividir los desarrollos complejos en varios más sencillos. De esta manera, se pueden programar las funcionalidades atendiendo a su función principal y a la integración con el resto.
* **Comenta tu Código:** Para facilitar las modificaciones y mantenimiento, se debe comentar el código. Sin embargo, hay que tener en cuenta que un buen código es aquel que no necesita comentarios. Por ello, es recomendable redactar sentencias simples e intentar elaborar una solución sencilla y corta, para minimizar los errores y facilitar su localización y solución.
* **Documentación:** Durante los desarrollos, es importante documentar todo lo que se hace para facilitar el entendimiento del desarrollo y funcionamiento del software al personal del proyecto y a futuras personas que trabajen en él.

## **1.4. Alternativas más Efectivas en el Contenido de Fotos**

### **1.4.1. Amazon S3**

Amazon S3 es un servicio de almacenamiento de datos en la nube diseñado para permitir a las empresas realizar copias de seguridad de sus archivos en cualquier momento y desde cualquier lugar. Destaca por su alta seguridad en el resguardo, integridad y confidencialidad de los datos, proporcionando privacidad y seguridad de extremo a extremo. Es ideal tanto para archivos personales como empresariales. Para los desarrolladores, S3 ofrece la facilidad de conectar los datos almacenados con aplicaciones de terceros, tanto móviles como web, con un excelente rendimiento.

### **1.4.2. Azure Blob Storage**

Azure Blob Storage permite crear lagos de datos para análisis y proporciona almacenamiento para aplicaciones móviles y nativas de la nube. Ofrece almacenamiento en niveles para datos a largo plazo y escalabilidad flexible para cargas de trabajo de aprendizaje automático y computación de alto rendimiento. Diseñado específicamente para satisfacer las necesidades de escala, seguridad y disponibilidad de los desarrolladores de aplicaciones móviles, web y nativas de la nube. Es el único servicio de almacenamiento en la nube con un nivel de almacenamiento de objetos Premium basado en SSD para escenarios interactivos y de baja latencia.

## **1.5. Metodología SCRUM**

Scrum es un proceso ágil que se utiliza comúnmente en el desarrollo de productos, en particular en el desarrollo de software. Es un marco de gestión de proyectos que se aplica a cualquier proyecto con plazos agresivos, requisitos complejos y un alto grado de singularidad. En Scrum, el equipo de desarrollo trabaja en ciclos cortos, llamados sprints, para entregar un producto funcionando de alta calidad. El proceso de Scrum se basa en la transparencia, la inspección y la adaptación, y se centra en la colaboración entre los miembros del equipo.

### **1.5.1. ¿Cuándo se utiliza?**

La metodología Scrum es especialmente útil en proyectos de desarrollo de software donde se requiere una alta flexibilidad y capacidad de adaptación a los cambios. También es útil en proyectos con un alto nivel de incertidumbre y complejidad, donde el cliente no tiene una idea clara de lo que quiere o donde los requisitos del proyecto son difíciles de definir de antemano.

Además, Scrum es una buena opción cuando se trata de equipos de desarrollo distribuidos geográficamente, ya que permite una colaboración efectiva y una comunicación continua y transparente entre los miembros del equipo.

### **1.5.2. El Equipo de SCRUM**

El equipo Scrum es una parte fundamental de la metodología y está compuesto por tres roles clave: el Dueño de Producto, el Equipo de Desarrollo y el Scrum Master. El Dueño de Producto es responsable de maximizar el valor del producto y del trabajo del equipo. El Equipo de Desarrollo es responsable de crear el incremento de producto "Terminado" en cada iteración del Sprint y el Scrum Master es responsable de asegurarse de que se siga el proceso Scrum y se eliminen los obstáculos que impiden el progreso del equipo. Todos los miembros del equipo son multifuncionales y auto organizados, lo que les permite colaborar y trabajar juntos para lograr los objetivos del proyecto. Además, las entregas incrementales de producto "Terminado" permiten obtener retroalimentación y aseguran que siempre esté disponible una versión potencialmente útil y funcional del producto.

### **1.5.3. Eventos de SCRUM**

Scrum es un marco de trabajo que establece una serie de eventos predefinidos para fomentar la regularidad en el desarrollo del producto. Estos eventos están diseñados para maximizar la transparencia y la inspección y permitir la adaptación en el proceso. Además del Sprint, que es el contenedor de los demás eventos, Scrum cuenta con otros eventos clave, como la Planificación del Sprint, la Reunión Diaria de Scrum, la Revisión del Sprint y la Retrospectiva del Sprint. Cada uno de estos eventos tiene una duración máxima y está diseñado para cumplir un objetivo específico, asegurando que se emplee una cantidad adecuada de tiempo y que se evite el desperdicio en el proceso. La falta de alguno de estos eventos puede llevar a una reducción de la transparencia y constituir una oportunidad perdida para inspeccionar y adaptarse.

1. **Sprint**

El Sprint es el corazón de Scrum, un bloque de tiempo de un mes o menos en el que se crea un incremento de producto "Terminado", utilizable y potencialmente desplegable. Es más conveniente que la duración de los Sprints sea consistente a lo largo del esfuerzo de desarrollo. Cada nuevo Sprint comienza inmediatamente después de la finalización del Sprint previo. Los Sprints contienen y consisten en la Reunión de Planificación del Sprint, los Scrum Diarios, el trabajo de desarrollo, la Revisión del Sprint y la Retrospectiva del Sprint.

Durante el Sprint:

* No se realizan cambios que puedan afectar al Objetivo del Sprint.
* Los objetivos de calidad no disminuyen.
* El alcance puede ser clarificado y renegociado entre el Dueño de Producto y el Equipo de Desarrollo a medida que se va aprendiendo más.

1. **Sprint Planning Meeting**

El trabajo a realizar durante el Sprint se planifica en la Reunión de Planificación del Sprint. Este plan se crea mediante el trabajo colaborativo del Equipo Scrum completo. La Reunión de Planificación del Sprint tiene una duración máxima de ocho horas para un Sprint de un mes. Para Sprints más cortos, el evento es usualmente más corto. El Scrum Master se asegura de que el evento se lleve a cabo y de que los asistentes entiendan su propósito. El Scrum Master enseña al Equipo Scrum a mantenerse dentro del bloque de tiempo.

1. **Objetivo del Sprint**

El Objetivo del Sprint es una meta establecida para el Sprint, que es una iteración de trabajo en Scrum. Proporciona una guía al Equipo de Desarrollo sobre por qué están construyendo el incremento y se crea durante la reunión de Planificación del Sprint. El objetivo del Sprint ofrece al equipo de desarrollo cierta flexibilidad con respecto a la funcionalidad implementada en el Sprint, ya que los elementos de la Lista del Producto seleccionados ofrecen una función coherente, que puede ser el objetivo del Sprint.

1. **Daily Scrum**

El Daily Scrum es una reunión diaria de 15 minutos para que el Equipo de Desarrollo sincronice sus actividades y cree un plan para las siguientes 24 horas. Durante la reunión, cada miembro del equipo responde a tres preguntas:

* ¿Qué hice ayer que ayudó al Equipo de Desarrollo a lograr el Objetivo del Sprint?
* ¿Qué haré hoy para ayudar al Equipo de Desarrollo a lograr el Objetivo del Sprint?
* ¿Veo algún impedimento que evite que el Equipo de Desarrollo o yo logremos el Objetivo del Sprint?

El objetivo del Daily Scrum es mantener al equipo alineado y enfocado en el Objetivo del Sprint, identificar rápidamente los problemas y ajustar el plan según sea necesario para lograr el objetivo.

1. **Revisión del Sprint**

Se lleva a cabo al final del Sprint para inspeccionar el Incremento y adaptar la Lista de Producto si es necesario. Durante la reunión, el Equipo Scrum y los interesados colaboran para determinar las siguientes cosas que podrían hacerse para optimizar el valor. La presentación del Incremento tiene como objetivo facilitar la retroalimentación de información y fomentar la colaboración. La reunión es informal y no es una reunión de seguimiento.

1. **Sprint Retrospective**

Tiene lugar después de la Revisión de Sprint y antes de la siguiente Reunión de Planificación de Sprint. Es una oportunidad para el Equipo Scrum de inspeccionarse a sí mismo y crear un plan de mejoras que sean abordadas durante el siguiente Sprint. La duración de la reunión está restringida a un bloque de tiempo de tres horas para Sprints de un mes, y para Sprints más cortos se reserva un tiempo proporcionalmente menor. El Scrum Master es responsable de asegurarse de que el evento se lleve a cabo y que los asistentes entiendan su propósito, y participa en la reunión como un miembro del equipo ya que la responsabilidad del proceso Scrum recae sobre él.

## **1.6. C4 como Herramienta para la Arquitectura**

El modelo C4 es una técnica gráfica utilizada para modelar la arquitectura de sistemas de software. Su enfoque se basa en la descomposición estructural del sistema en contenedores y componentes, lo que permite una visión clara y detallada de los bloques de construcción arquitectónicos. Está inspirado en técnicas de modelado existentes, como UML y ERD.

### **1.6.1. Descripción General**

El modelo C4 documenta la arquitectura del sistema de software mediante múltiples puntos de vista que explican la descomposición en contenedores y componentes, la relación entre ellos y, cuando sea necesario, la relación con los usuarios. En resumen, proporciona una visión global y detallada del sistema, permitiendo una mejor comprensión y toma de decisiones en el desarrollo y mantenimiento del software.

Los miradores están organizados según su nivel jerárquico:

* **Diagramas de contexto (nivel 1):** muestran el alcance del sistema y su relación con los usuarios y otros sistemas.
* **Diagramas de contenedores (nivel 2):** Descomponen un sistema en contenedores interrelacionados. Un contenedor representa una aplicación o un almacén de datos
* **Diagramas de componentes (nivel 3**): Descomponen los contenedores en componentes interrelacionados y relaciona los componentes con otros contenedores u otros sistemas
* **Diagramas de código (nivel 4):** Proporcionan detalles adicionales sobre el diseño de los elementos arquitectónicos que se pueden asignar a código.

El modelo C4 se basa en este nivel en notaciones existentes, como el lenguaje de modelado unificado (UML), los diagramas de entidad-relación (ERD) o los diagramas generados por entornos de desarrollo integrados (IDE). Para los niveles 1 a 3, el modelo C4 utiliza 5 elementos básicos de diagramación: personas, sistemas de software, contenedores, componentes y relaciones. La técnica no es prescriptiva para el diseño, la forma, el color y el estilo de estos elementos. En cambio, el modelo C4 recomienda el uso de diagramas simples basados en cajas anidadas para facilitar el dibujo colaborativo interactivo. La técnica también promueve buenas prácticas de modelado, como proporcionar un título y una leyenda en cada diagrama, y un etiquetado claro e inequívoco para facilitar la comprensión por parte de la audiencia destinataria.

## **1.7. Inteligencia Artificial**

### **1.7.1. Amazon Rekognition**

Amazon Rekognition es un servicio de análisis de imágenes y videos que utiliza el aprendizaje profundo y la inteligencia artificial para identificar, etiquetar y analizar objetos, personas y acciones en imágenes y videos. El servicio se integra con Amazon S3, Amazon Lambda y otros servicios de AWS, lo que lo hace fácil de implementar en aplicaciones web y móviles.

*Ilustración 2: Amazon Rekognition*

*Fuente: https://aws.amazon.com/es/rekognition/*

Rekognition tiene múltiples capacidades, incluyendo la detección de caras, reconocimiento de celebridades, análisis de emociones, detección de texto y etiquetado de contenido inapropiado. Su característica más destacada es el reconocimiento facial, que utiliza tecnología de aprendizaje profundo para detectar y comparar caras en imágenes y videos con una precisión y velocidad impresionantes. Esta característica permite la identificación de personas, el control de acceso y la búsqueda de rostros en grandes colecciones de imágenes y videos.

**Reconocimiento Facial**

El reconocimiento facial es una de las funcionalidades principales de Amazon Rekognition. Esta herramienta utiliza técnicas de inteligencia artificial para identificar y analizar rostros en imágenes y vídeos. Al procesar una imagen, Rekognition puede detectar los rostros y extraer información como la posición de los ojos, la nariz, la boca, la forma de la cara, entre otros detalles. Luego, la herramienta compara la información extraída con una base de datos previamente configurada para identificar a personas específicas.

*Ilustración 3: Reconocimiento Facial AWS Rekognition*

*Fuente: https://aws.amazon.com/es/rekognition/*

El proceso comienza con la detección de caras en la imagen o video. Luego, se crea un modelo de características para cada rostro detectado en la imagen o video, incluyendo la ubicación y forma de los ojos, la nariz y la boca. Posteriormente, se compara este modelo con una base de datos de rostros previamente registrados para determinar la coincidencia.

En caso de que exista una coincidencia, el sistema proporciona información sobre la identidad del individuo detectado. Esto puede ser utilizado en una variedad de aplicaciones, desde la identificación de personas para fines de seguridad hasta el reconocimiento de celebridades en imágenes de redes sociales.

**Ventajas**

Las ventajas de utilizar Amazon Rekognition para el reconocimiento facial son diversas. En primer lugar, esta herramienta permite automatizar y acelerar el proceso de identificación de personas en grandes volúmenes de imágenes y vídeos. De esta manera, se puede ahorrar tiempo y esfuerzo en la tarea de análisis de imágenes y vídeos. Además, el uso de inteligencia artificial en el reconocimiento facial permite una mayor precisión y eficacia en la identificación de personas.

Otra ventaja importante de Amazon Rekognition es su capacidad para analizar y comparar rostros en tiempo real. Esto puede ser especialmente útil en situaciones en las que se requiere una identificación rápida, como en el caso de la seguridad pública, la vigilancia y la prevención de fraudes. Por ejemplo, Rekognition puede utilizarse para identificar a personas sospechosas en grandes eventos, estaciones de transporte público, entre otros lugares.

*Ilustración 3: Reconocimiento Facial AWS Rekognition*

*Fuente: https://aws.amazon.com/es/rekognition/*

# **2. SPRINT 0**

## **2.1. Definiciones Iniciales**

1. **Requisitos Iniciales**

* **Usuarios:** Se define los usuarios que existirán en el sistema: fotógrafo, organizador y cliente.
* **Fotógrafo:** Cuando este usuario se registra en nuestro software como fotógrafo, tendrá la capacidad de ofrecer sus servicios al crear diferentes paquetes que los organizadores de eventos puedan contratar. Una vez que un organizador crea un evento, el fotógrafo puede subir fotografías públicas y privadas relacionadas con dicho evento.
* **Organizador:** Este usuario al registrarse como un organizador el cual podrá crear eventos y escoger que fotógrafo contratar para dicho evento.
* **Cliente:** Este usuario mediante el QR podrá registrarse a la plataforma como al evento podrá recibir notificaciones de las fotografías en las que aparece.
* **Modelo Saas:** Permite a nuestros usuarios conectarse a nuestra aplicación en la nube a través de Internet y usarla. Con esto ofrecemos un modelo de pago por uso a nuestros usuarios principales que serían los fotógrafos.

1. **Equipo de Desarrollo**

|  |  |  |
| --- | --- | --- |
| **Persona** | **Rol** | **Característica** |
| Leonardo Mogiano Gutierrez | Scrum Master | Hacerse cargo de la realización de todo el Sprint 0 |
| Leonardo Mogiano Gutierrez | Product Owner | Especificar sobre todos los aspectos que requiere el software |
| Leonardo Mogiano Gutierrez | Developers | Realizar e implementar las funciones básicas de lado del servidor y prototipo |

1. **Herramienta de Implementación**

|  |  |
| --- | --- |
| **Nombre** | **Característica** |
| Sistema Operativo | Windows 11 |
| Lenguaje de programación web | PHP |
| Motor de base de datos | PostgreSQL |
| Herramienta de Diseño | Enterprise Architect, Draw.io |
| Framework | Laravel |
| Servicio en la Nube | Amazon Web Service |
| Testeo | Postman |

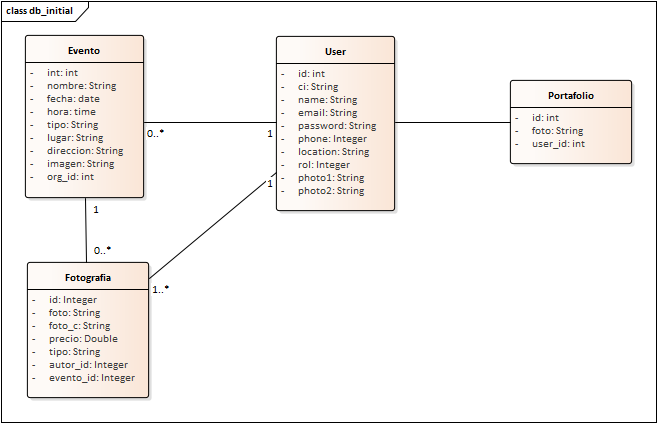
1. **Duración de cada Sprint**

Este proyecto tendrá 2 sprint con una duración de 2 semanas aproximadamente.

1. **Product Backlog Inicial**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Product Backlog** | | | | | |
| **Proyecto** | | PicStore | | | |
| **Product Owner** | | Moises Leonardo Mogiano Gutierrez | | | |
| **Version** | | 1.0 | **Fecha** | 11/10/2022 | |
| **ID** | **Rol** | **Funcionalidad** | **Razón / Resultado** | | **P** |
| 1 | Usuario | Perfil de Usuario | Controlar la información de sus propios perfiles | | Alta |
| 2 | Usuario | Facturación de Usuario | Controlar las tarjetas para usar en la Facturación. | | Alta |
| 3 | Organizador | Administrar Eventos | Registra todos los eventos para un organizador y un fotógrafo | | Alta |
| 4 | Organizador | Gestionar Fotógrafos/Evento | Contratar a los fotógrafos para los Eventos Creados | | Alta |
| 5 | Fotógrafo | Administrar Portafolio | Subir fotos a su Portafolio como referencia para contrataciones | | Media |
| 6 | Fotógrafo | Subir Fotografias / Evento | Subir fotos a los eventos adjudicados para realizar la Venta | | Alta |
| 7 | Cliente | Ver Eventos/Foto Vinculados | Visualizar los eventos adjudicados y fotografías para su compra | | Media |
| 8 | Organizador | Generar QR de sus Eventos | Permite descarga de QR de sus eventos para los invitados | | Baja |
| 9 | Administrador | Implementación IA | Implementación de una IA de  reconocimiento facial | | Alta |
| 10 | Administrador | Generar Notificaciones | Implementar notificaciones para los usuarios | | Media |
| 11 | Cliente | Login App Móvil Invitado | Implementar inicio de sesión con Api y consumir en Flutter | | Media |
| 12 | Cliente | Gestión Eventos y Foto / Móvil | Visualizar los eventos adjudicados y fotografías para su compra | | Alta |
| 13 | Cliente | Perfil de Usuario Invitado / Móvil | Visualizar información básica del usuario | | Media |
| 14 | Cliente | Gestión FotoCompradas / Móvil | Visualizar fotos adquiridas por el usuario | | Media |
| 15 | Cliente | Comprar Foto / Móvil | Permite comprar las fotos desde la aplicación | | Alta |

## **2.2. Base de Datos Inicial**



# **3. SPRINT 1**

## **3.1. Sprint Backlog**

|  |  |
| --- | --- |
| **Sprint Backlog** | |
| **1** | Perfil de Usuario |
| **2** | Facturación de Usuario |
| **3** | Administrar Eventos |
| **5** | Administrar Portafolio |
| **8** | Generar QR de sus Eventos |
| **11** | Login App Móvil |
| **13** | Perfil de Usuario / Móvil |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint Backlog - Tareas** | | | | | |
| **Numero de Sprint:** 1 | | **Tiempo Programado:** 12 días | | | |
| **Fecha Inicio:** 13/04/2023 | | **Fecha de Finalización:** 25/04/2023 | | | |
| **Objetivo:** Lograr las implementaciones básicas del software PicStore | | | | | |
| **ID** | **Tarea** | **Tipo** | **Estimación** | **Responsable** | **Estado** |
| 1 | Instalación y configuración de proyecto con Laravel | Infraestructura | 1 Día | Leo Mogiano | Completado |
| 2 | Diseñar e implementar Landing page | Diseño  Implementación | 1Día | Leo Mogiano | Completado |
| 3 | Diseñar y crear Base de Datos | Diseño  Implementación | 1 Día | Leo Mogiano | Completado |
| 4 | Configurar Témplate por componentes para la Web | Diseño | 1 Día | Leo Mogiano | Completado |
| 5 | Implementar Perfil Usuario | Implementación | 1 Día | Leo Mogiano | Completado |
| 6 | Implementar Facturación | Implementación | 1 Día | Leo Mogiano | Completado |
| 7 | Implementar Administración de Eventos | Implementación | 1 Día | Leo Mogiano | Completado |
| 8 | Implementar Gestión de Portafolio | Implementación | 1 Día | Leo Mogiano | Completado |
| 9 | Implementar Facturación | Implementación | 1 Día | Leo Mogiano | Completado |
| 10 | Implementar QR invitados asociados al evento | Implementación | 0.5 Día | Leo Mogiano | Completado |
| 11 | Implementación de Api para uso en App Móvil | Implementación | 0.5 Día | Leo Mogiano | Completado |
| 11 | Instalación y configuración ambiente móvil en Flutter | Infraestructura | 0.5 Día | Leo Mogiano | Completado |
| 12 | Implementación de Login para App Móvil Invitados | Implementación  Diseño | 0.5 Día | Leo Mogiano | Completado |
| 13 | Implementación de perfil de Usuario solo Invitado | Implementación  Diseño | 1 Día | Leo Mogiano | Completado |

## **3.2. Historias de Usuario**

|  |  |
| --- | --- |
| **Perfil de Usuario** | |
| **Id** 1 | **Descripción:** Como usuario registrado en PicStore, quiero tener un perfil en el cual pueda ver y actualizar mi información personal. |
| **Prioridad** Alta | **Estimación:** 2 días |
| **Criterios de Aceptación**   * El usuario puede acceder a su perfil desde la página principal. * El usuario puede ver y editar su información personal (nombre, correo electrónico, contraseña, etc.). * La información del usuario se almacena en la base de datos de forma segura. * La interfaz de usuario es intuitiva y fácil de usar. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Facturación de Usuario** | |
| **Id** 2 | **Descripción:** Como usuario registrado en PicStore, quiero poder realizar pagos y recibir facturas de mis compras. |
| **Prioridad** Media | **Estimación:** 1 días |
| **Criterios de Aceptación**   * El usuario puede agregar y guardar su información de pago. * El usuario puede realizar pagos de forma segura. * La información del usuario se almacena en la base de datos de forma segura. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Administrar de Evento** | |
| **Id** 3 | **Descripción:** Como usuario organizador registrado en PicStore, quiero poder crear y administrar eventos. |
| **Prioridad** Alta | **Estimación:** 1 días |
| **Criterios de Aceptación**   * El usuario puede crear un evento y agregar información detallada (nombre, fecha, hora, ubicación, etc.). * El usuario puede actualizar la información del evento en cualquier momento. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Administrar de Portafolio** | |
| **Id** 5 | **Descripción:** Como usuario fotógrafo registrado en PicStore, quiero poder crear y administrar mi portafolio de fotografías para ser contratado. |
| **Prioridad** Media | **Estimación:** 1 días |
| **Criterios de Aceptación**   * El usuario fotógrafo puede fotos al portafolio. * Los usuarios organizadores pueden ver el portafolio del usuario a la hora de contratarlos. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Generar QR de sus Eventos** | |
| **Id** 8 | **Descripción:** Como usuario organizador quiero poder generar códigos QR para mis eventos para que los invitados puedan escanearlos y acceder a información sobre el evento. |
| **Prioridad** Media | **Estimación:** 0.5 días |
| **Criterios de Aceptación**   * El usuario puede generar un código QR para un evento en particular. * El código QR incluye la asociación del usuario al evento como la posibilidad para acceder a la información básica. * Los códigos QR son fácilmente imprimibles para poder ser colocados en lugares visibles en el evento. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Login App Móvil** | |
| **Id** 11 | **Descripción:** Como usuario quiero poder iniciar sesión en la aplicación móvil con mi cuenta de usuario invitado para poder acceder a la información de mis eventos. |
| **Prioridad** Media | **Estimación:** 1.5 días |
| **Criterios de Aceptación**   * El usuario puede iniciar sesión en la aplicación móvil con sus credenciales de usuario. * La aplicación móvil valida la información de inicio de sesión del usuario. * Una vez iniciada sesión, el usuario tiene acceso a la información de sus eventos. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Perfil de Usuario / Móvil** | |
| **Id** 13 | **Descripción:** Como usuario invitado quiero poder acceder a mi perfil en la aplicación móvil para ver mi información personal. |
| **Prioridad** Media | **Estimación:** 1 días |
| **Criterios de Aceptación**   * El usuario puede acceder a su perfil en la aplicación móvil. * El usuario puede ver su información personal, como su nombre, correo electrónico y foto. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

## **3.3. Sprint Review**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Inicio** | **Duración** | **Actividad** | **Sprint Review** | **Responsable** |
| 08:00 | 15 min. | Objetivo del Sprint | En este sprint se enfocó en implementar las funcionalidades principales del software PicStore y se lograron cumplir todas las tareas del Sprint Backlog de manera satisfactoria. | Product Owner |
| 08:15 | 25 min. | Revisión de los requisitos principales | Todos los criterios de aceptación se cumplieron y se lograron los objetivos planteados para el sprint. | Product Owner |
| 08:50 | 25 min. | Estado del Sprint | El equipo cumplió con todas las tareas del Sprint Backlog y se lograron las implementaciones básicas del software de manera satisfactoria. | Scrum Master |
| 09:15 | 20 min. | Demostración | Se presentaron las funcionalidades implementadas en el sprint en una demostración del software PicStore, donde se mostró su correcto funcionamiento y cumplimiento de los criterios de aceptación. | Product Owner |
| 09:35 | 10 min. | Feedback | El equipo expresó su satisfacción por el trabajo realizado y se sugirió la inclusión de más pruebas unitarias en el desarrollo del software para asegurar su calidad y evitar futuros errores. | Scrum Master |
| 09:45 | 5 min. | Conclusiones | El sprint fue un éxito, con un buen ritmo de trabajo y colaboración entre el equipo. Se espera seguir mejorando en áreas específicas, como la inclusión de más pruebas unitarias en el desarrollo del software. | Scrum Master |

## **3.3. Sprint Retrospective**

Después de la reunión, se concluyó que:

**Lo que fue bien:**

* Se logró cumplir con todas las tareas del Sprint Backlog.
* La comunicación y colaboración del equipo fue buena.
* Se implementaron correctamente las funcionalidades principales del software.

**Lo que se puede mejorar:**

* Incluir más pruebas unitarias en el desarrollo del software.
* Evitar la procrastinación y trabajar en las tareas asignadas en tiempo y forma.
* Optimizar el uso de herramientas de gestión de proyectos para una mejor organización del trabajo.

**Acciones concretas a tomar:**

* Establecer un plan para incluir más pruebas unitarias en el desarrollo del software.
* Realizar una reunión diaria breve para monitorear el progreso del trabajo y evitar la procrastinación.
* Revisar y optimizar el uso de las herramientas de gestión de proyectos utilizadas por el equipo.

# **4. SPRINT 2**

## **4.1. Sprint Backlog**

|  |  |
| --- | --- |
| **Sprint Backlog** | |
| **4** | Gestionar Fotógrafo/ Evento |
| **6** | Subir Fotografías / Evento |
| **7** | Ver Eventos / Fotos Vinculadas |
| **9** | Implementación de IA |
| **10** | Generar Notificaciones |
| **12** | Gestión de Eventos y Foto / Móvil |
| **14** | Gestión de Foto Compradas/ Móvil |
| **15** | Comprar Foto / Móvil |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sprint Backlog - Tareas** | | | | | |
| **Numero de Sprint:** 2 | | **Tiempo Programado:** 13 días | | | |
| **Fecha Inicio:** 26/04/2023 | | **Fecha de Finalización:** 08/05/2023 | | | |
| **Objetivo:** Implementar nuevas funcionalidades en el software PicStore | | | | | |
| **ID** | **Tarea** | **Tipo** | **Estimación** | **Responsable** | **Estado** |
| 1 | Implementar gestión de Fotógrafos y sus eventos | Implementación | 2 Días | Leo Mogiano | Completado |
| 2 | Implementar la subida fotografías para fotógrafo | Diseño  Implementación | 2Días | Leo Mogiano | Completado |
| 3 | Implementar para invitados puedan ver eventos y fotos asociadas | Diseño  Implementación | 1 Día | Leo Mogiano | Completado |
| 4 | Implementar detección facial utilizando AWS Rekognition | Implementación | 2 Días | Leo Mogiano | Completado |
| 5 | Generar notificaciones cuando se contrata un fotógrafo o invitado que aparece en una foto | Implementación | 2 Días | Leo Mogiano | Completado |
| 6 | Implementar gestión de eventos y fotos en dispositivos móviles | Implementación  Diseño | 1 Días | Leo Mogiano | Completado |
| 7 | Implementar visualización de fotos compradas en dispositivos móviles | Implementación  Diseño | 1 Día | Leo Mogiano | Completado |
| 8 | Implementar la funcionalidad de compra de fotos para invitados en móvil | Implementación  Diseño | 2 Día | Leo Mogiano | Completado |

## **4.2. Historias de Usuario**

|  |  |
| --- | --- |
| **Gestionar Fotógrafo/ Evento** | |
| **Id** 4 | **Descripción:** Como organizador quiero poder contratar a los fotógrafos y asociarlos al evento después de que ellos acepten |
| **Prioridad** Alta | **Estimación:** 2 días |
| **Criterios de Aceptación**   * El usuario puede acceder a su perfil en la aplicación móvil. * El usuario puede ver su información personal, como su nombre, correo electrónico y foto. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Subir Fotografías / Evento** | |
| **Id** 6 | **Descripción:** Como fotógrafo quiero poder subir fotografías para mis eventos para que los invitados identificarlos mediante IA si es privado o público para puedan verlas. |
| **Prioridad** Alta | **Estimación:** 2 días |
| **Criterios de Aceptación**   * El fotógrafo puede seleccionar las fotografías que desea subir para su evento. * Las fotografías se cargan correctamente en el sistema y están disponibles para los invitados mediante IA. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Ver Eventos / Fotos Vinculadas** | |
| **Id** 7 | **Descripción:** Como invitado quiero poder ver los eventos y las fotos asociadas para poder ver las fotos tomadas en un evento específico. |
| **Prioridad** Media | **Estimación:** 1 días |
| **Criterios de Aceptación**   * El invitado puede ver los eventos a los que ha sido invitado. * El invitado puede ver las fotos asociadas a cada evento. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Implementación de IA** | |
| **Id** 9 | **Descripción:** Como administrador quiero integrar una solución de inteligencia artificial (AI) para la detección facial para poder detectar los invitados en las fotos tomadas en los eventos. |
| **Prioridad** Alta | **Estimación:** 2 días |
| **Criterios de Aceptación**   * El sistema utiliza la solución de AI para detectar invitados en las fotos tomadas en los eventos. * Los invitados son notificados cuando aparecen en una foto. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Generar Notificaciones** | |
| **Id** 10 | **Descripción:** Como administrador quiero poder generar notificaciones para los usuarios para informarles sobre la contratación de un fotógrafo o cuando aparecen en una foto, para mantenerlos informados sobre la actividad en la plataforma. |
| **Prioridad** Alta | **Estimación:** 2 días |
| **Criterios de Aceptación**   * Los usuarios reciben notificaciones cuando se contrata un fotógrafo o cuando aparecen en una foto. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Gestión de Eventos y Foto / Móvil** | |
| **Id** 12 | **Descripción:** Como usuario quiero poder ver la información de los eventos y las fotos en mi dispositivo móvil para poder acceder a la plataforma desde cualquier lugar. |
| **Prioridad** Media | **Estimación:** 1 días |
| **Criterios de Aceptación**   * Los usuarios pueden acceder a la información de los eventos y las fotos desde su dispositivo móvil. | |
| **Desarrolladores a Cargo** Leo Mogiano | |
| **Gestión de Fotos Compradas/ Móvil** | |
| **Id** 14 | **Descripción:** Como usuario invitado quiero poder gestionar y ver las fotos que he comprado en la aplicación móvil. |
| **Prioridad** Alta | **Estimación:** 1 días |
| **Criterios de Aceptación**   * El usuario puede ver las fotos que ha comprado en la aplicación móvil. * El usuario puede descargar las fotos compradas en la aplicación móvil. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

|  |  |
| --- | --- |
| **Comprar Foto / Móvil** | |
| **Id** 14 | **Descripción:** Como usuario invitado quiero poder comprar una foto desde la aplicación móvil para poder obtener una copia de la misma. |
| **Prioridad** Alta | **Estimación:** 2 días |
| **Criterios de Aceptación**   * El usuario debe poder ver una lista de fotos disponibles para comprar desde la aplicación móvil. * El usuario debe poder descargar la foto comprada después de realizar el pago. | |
| **Desarrolladores a Cargo** Leo Mogiano | |

## **4.3. Sprint Review**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Inicio** | **Duración** | **Actividad** | **Sprint Review** | **Responsable** |
| 08:00 | 15 min. | Objetivo del Sprint | Implementar nuevas funcionalidades en el software PicStore. Se lograron cumplir todas las tareas del Sprint Backlog de manera satisfactoria. | Product Owner |
| 08:15 | 25 min. | Revisión de los requisitos principales | Se cumplieron todos los criterios de aceptación y se mejoró la experiencia de los usuarios de PicStore. Terminando con aspectos de app móvil. | Product Owner |
| 08:50 | 25 min. | Estado del Sprint | Se destacó la colaboración y comunicación entre el equipo durante el sprint, lo que permitió cumplir con los objetivos de manera efectiva y en el tiempo planificado. Se lograron las implementaciones básicas del software de manera satisfactoria, cumpliendo con todas las tareas del Sprint Backlog. | Scrum Master |
| 09:15 | 20 min. | Demostración | Se presentaron las funcionalidades implementadas en el sprint en una demostración del software PicStore, donde se mostró su correcto funcionamiento y cumplimiento de los criterios de aceptación. | Product Owner |
| 09:35 | 10 min. | Feedback | Se discutieron áreas de mejora, como la inclusión de más pruebas unitarias, y se compartieron ideas para futuras mejoras. | Scrum Master |
| 09:45 | 5 min. | Conclusiones | El sprint fue un éxito con un buen ritmo de trabajo y colaboración entre el equipo. Se espera seguir mejorando en áreas específicas. | Scrum Master |

## **4.4. Sprint Retrospective**

Después de la reunión, se concluyó que:  
**Lo que fue bien:**

• El equipo logró cumplir con todas las tareas del Sprint Backlog.

• La colaboración y comunicación entre los miembros del equipo fue excelente.

• Se entregó el producto final en el tiempo estipulado y con una buena calidad.

**Lo que se puede mejorar:**

• Se notó una falta de motivación en algunos momentos, lo que ralentizó el progreso del trabajo.

• Se podría haber optimizado aún más el proceso de control de calidad.

**Acciones concretas a tomar:**

• Realizar una revisión exhaustiva del proceso de control de calidad y hacer mejoras en el proceso donde sea necesario.

# **5. MODELO C4**

## **5.1. Nivel 1: Diagrama de Contexto**

## D:\mmogiano\Downloads\Capa2 - Página 1 (1).png**5.2. Nivel 2: Diagrama de Contenedor**

## **5.2. Nivel 3: Diagrama de** **Componentes**

## **5.4. Modelo de Datos**

**SCRIPT DATABASE**

CREATE TABLE `eventos` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `nombre` varchar(255) NOT NULL,

  `tipo` varchar(255) NOT NULL,

  `lugar` varchar(255) NOT NULL,

  `imagen` varchar(255) *DEFAULT* NULL,

  `fecha` date NOT NULL,

  `hora` time NOT NULL,

  `direccion` varchar(255) NOT NULL,

  `org\_id` int(11) *DEFAULT* NULL,

  `created\_at` timestamp NULL *DEFAULT* NULL,

  `updated\_at` timestamp NULL *DEFAULT* NULL

)

CREATE TABLE `fotografias` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `foto` varchar(255) NOT NULL,

  `foto\_c` varchar(255) *DEFAULT* NULL,

  `autor\_id` int(11) *DEFAULT* NULL,

  `precio` double(8,2) *DEFAULT* NULL,

  `tipo` varchar(255) *DEFAULT* NULL,

  `evento\_id` bigint(20) UNSIGNED NOT NULL,

  `created\_at` timestamp NULL *DEFAULT* NULL,

  `updated\_at` timestamp NULL *DEFAULT* NULL

)

CREATE TABLE `foto\_user` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `estado` varchar(255) *DEFAULT* NULL,

  `tarjeta\_id` int(11) *DEFAULT* NULL,

  `foto\_id` bigint(20) UNSIGNED NOT NULL,

  `user\_id` bigint(20) UNSIGNED NOT NULL,

  `created\_at` timestamp NULL *DEFAULT* NULL,

  `updated\_at` timestamp NULL *DEFAULT* NULL

)

CREATE TABLE `personal\_access\_tokens` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `tokenable\_type` varchar(255) NOT NULL,

  `tokenable\_id` bigint(20) UNSIGNED NOT NULL,

  `name` varchar(255) NOT NULL,

  `token` varchar(64) NOT NULL,

  `abilities` text *DEFAULT* NULL,

  `expires\_at` timestamp NULL *DEFAULT* NULL,

  `last\_used\_at` timestamp NULL *DEFAULT* NULL,

  `created\_at` timestamp NULL *DEFAULT* NULL,

  `updated\_at` timestamp NULL *DEFAULT* NULL

)

CREATE TABLE `portafolios` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `foto` varchar(255) NOT NULL,

  `user\_id` bigint(20) UNSIGNED NOT NULL,

  `created\_at` timestamp NULL *DEFAULT* NULL,

  `updated\_at` timestamp NULL *DEFAULT* NULL

)

CREATE TABLE `suscripciones` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `nombre` varchar(255) NOT NULL,

  `precio` double(8,2) NOT NULL,

  `logo` varchar(255) NOT NULL,

  `funciones` longtext CHARACTER SET utf8mb4 *COLLATE* utf8mb4\_bin NOT NULL *CHECK* (json\_valid(`funciones`)),

  `created\_at` timestamp NULL *DEFAULT* NULL,

  `updated\_at` timestamp NULL *DEFAULT* NULL

)

CREATE TABLE `tarjetas` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `numero` varchar(255) NOT NULL,

  `nombre` varchar(255) NOT NULL,

  `fecha\_v` date NOT NULL,

  `saldo` double(8,2) NOT NULL,

  `gasto` double(8,2) NOT NULL,

  `cvc` varchar(255) NOT NULL,

  `user\_id` bigint(20) UNSIGNED NOT NULL,

  `created\_at` timestamp NULL *DEFAULT* NULL,

  `updated\_at` timestamp NULL *DEFAULT* NULL

)

CREATE TABLE `users` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `name` varchar(255) NOT NULL,

  `email` varchar(255) NOT NULL,

  `password` varchar(255) NOT NULL,

  `ci` varchar(255) *DEFAULT* NULL,

  `phone` bigint(20) *DEFAULT* NULL,

  `location` varchar(255) *DEFAULT* NULL,

  `about\_me` varchar(255) *DEFAULT* NULL,

  `rol` varchar(255) *DEFAULT* NULL,

  `photo1` varchar(255) *DEFAULT* NULL,

  `photo2` varchar(255) *DEFAULT* NULL,

  `remember\_token` varchar(100) *DEFAULT* NULL,

  `created\_at` timestamp NULL *DEFAULT* NULL,

  `updated\_at` timestamp NULL *DEFAULT* NULL

)

CREATE TABLE `user\_evento` (

  `id` bigint(20) UNSIGNED NOT NULL,

  `estado` varchar(255) *DEFAULT* NULL,

  `user\_id` bigint(20) UNSIGNED NOT NULL,

  `evento\_id` bigint(20) UNSIGNED NOT NULL,

  `created\_at` timestamp NULL *DEFAULT* NULL,

  `updated\_at` timestamp NULL *DEFAULT* NULL

)

ALTER TABLE `eventos`

  ADD *PRIMARY KEY* (`id`);

ALTER TABLE `fotografias`

  ADD *PRIMARY KEY* (`id`),

  ADD KEY `fotografias\_evento\_id\_foreign` (`evento\_id`);

ALTER TABLE `foto\_user`

  ADD *PRIMARY KEY* (`id`),

  ADD KEY `foto\_user\_foto\_id\_foreign` (`foto\_id`),

  ADD KEY `foto\_user\_user\_id\_foreign` (`user\_id`);

ALTER TABLE `personal\_access\_tokens`

  ADD *PRIMARY KEY* (`id`),

  ADD UNIQUE KEY `personal\_access\_tokens\_token\_unique` (`token`),

  ADD KEY `personal\_access\_tokens\_tokenable\_type\_tokenable\_id\_index` (`tokenable\_type`,`tokenable\_id`);

ALTER TABLE `portafolios`

  ADD *PRIMARY KEY* (`id`),

  ADD KEY `portafolios\_user\_id\_foreign` (`user\_id`);

ALTER TABLE `suscripciones`

  ADD *PRIMARY KEY* (`id`);

ALTER TABLE `tarjetas`

  ADD *PRIMARY KEY* (`id`),

  ADD KEY `tarjetas\_user\_id\_foreign` (`user\_id`);

ALTER TABLE `users`

  ADD *PRIMARY KEY* (`id`),

  ADD UNIQUE KEY `users\_email\_unique` (`email`),

  ADD UNIQUE KEY `users\_ci\_unique` (`ci`);

ALTER TABLE `user\_evento`

  ADD *PRIMARY KEY* (`id`),

  ADD KEY `user\_evento\_user\_id\_foreign` (`user\_id`),

  ADD KEY `user\_evento\_evento\_id\_foreign` (`evento\_id`);

ALTER TABLE `eventos`

  MODIFY `id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT;

ALTER TABLE `fotografias`

  MODIFY `id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT;

ALTER TABLE `foto\_user`

  MODIFY `id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT;

ALTER TABLE `personal\_access\_tokens`

  MODIFY `id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT;

ALTER TABLE `portafolios`

  MODIFY `id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT;

ALTER TABLE `suscripciones`

  MODIFY `id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT;

ALTER TABLE `tarjetas`

  MODIFY `id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT;

ALTER TABLE `users`

  MODIFY `id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT;

ALTER TABLE `user\_evento`

  MODIFY `id` bigint(20) UNSIGNED NOT NULL AUTO\_INCREMENT;

ALTER TABLE `fotografias`

  ADD *CONSTRAINT* `fotografias\_evento\_id\_foreign` *FOREIGN KEY* (`evento\_id`) *REFERENCES* `eventos` (`id`) *ON DELETE CASCADE*;

ALTER TABLE `foto\_user`

  ADD *CONSTRAINT* `foto\_user\_foto\_id\_foreign` *FOREIGN KEY* (`foto\_id`) *REFERENCES* `fotografias` (`id`) *ON DELETE CASCADE*,

  ADD *CONSTRAINT* `foto\_user\_user\_id\_foreign` *FOREIGN KEY* (`user\_id`) *REFERENCES* `users` (`id`) *ON DELETE CASCADE*;

 `portafolios`

ALTER TABLE `portafolios`

  ADD *CONSTRAINT* `portafolios\_user\_id\_foreign` *FOREIGN KEY* (`user\_id`) *REFERENCES* `users` (`id`) *ON DELETE CASCADE*;

ALTER TABLE `tarjetas`

  ADD *CONSTRAINT* `tarjetas\_user\_id\_foreign` *FOREIGN KEY* (`user\_id`) *REFERENCES* `users` (`id`) *ON DELETE CASCADE*;

ALTER TABLE `user\_evento`

  ADD *CONSTRAINT* `user\_evento\_evento\_id\_foreign` *FOREIGN KEY* (`evento\_id`) *REFERENCES* `eventos` (`id`) *ON DELETE CASCADE*,

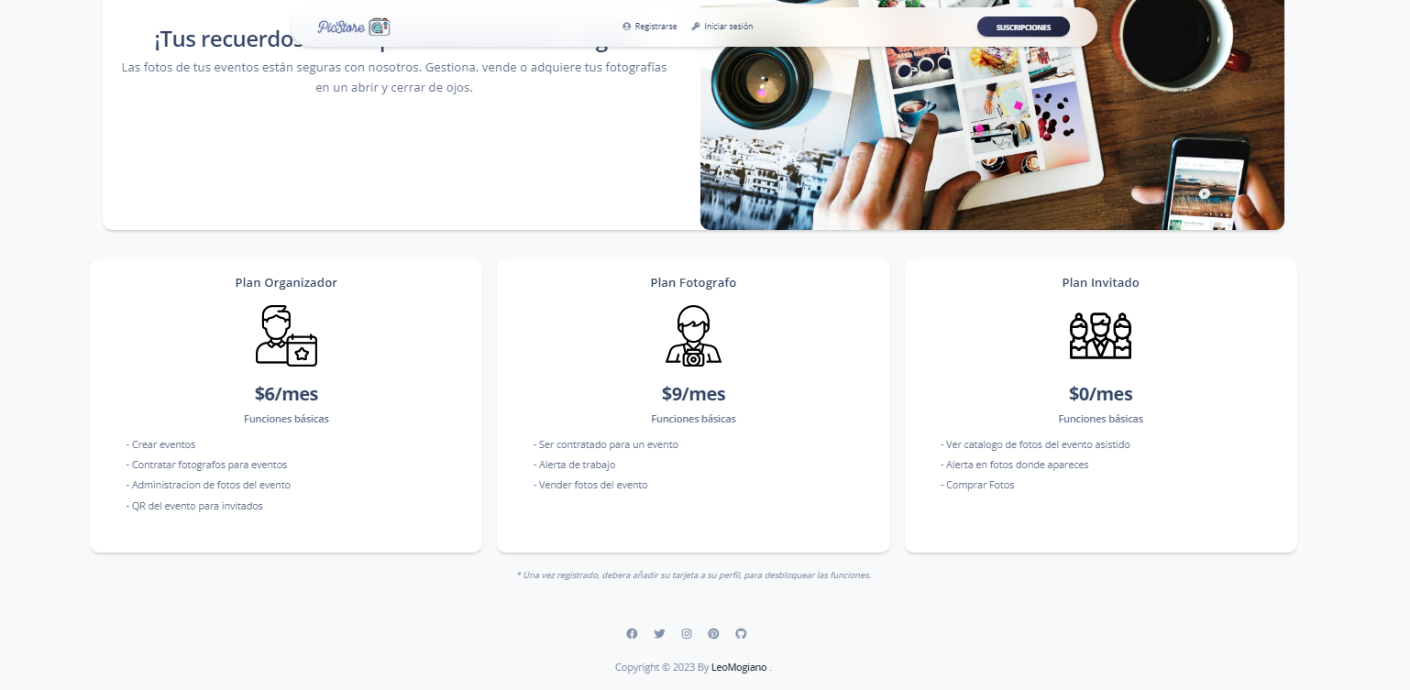
  ADD *CONSTRAINT* `user\_evento\_user\_id\_foreign` *FOREIGN KEY* (`user\_id`) *REFERENCES* `users` (`id`) *ON DELETE CASCADE*;

COMMIT;

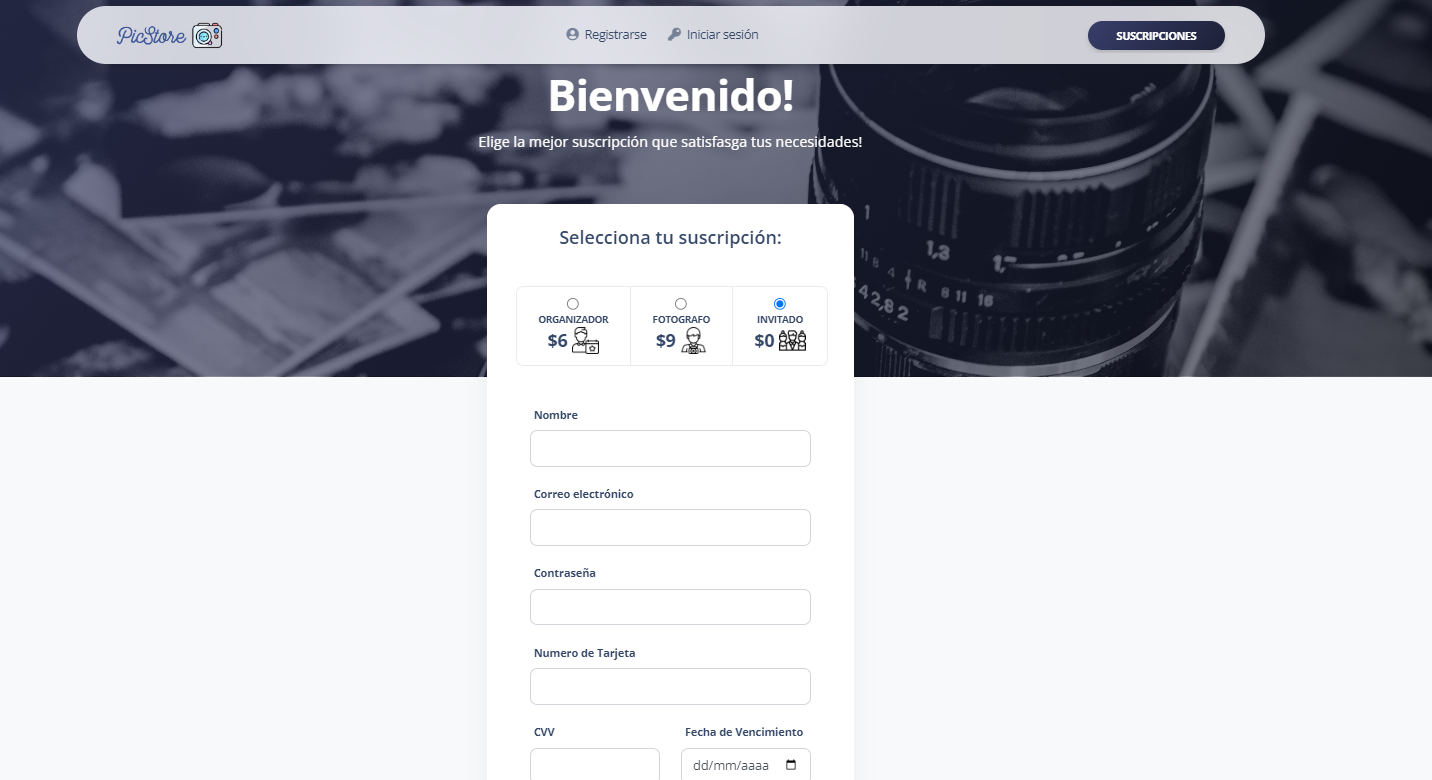
# **6. GUÍA DE SOFTWARE**

## **6.1. Navegación de Página Principal**

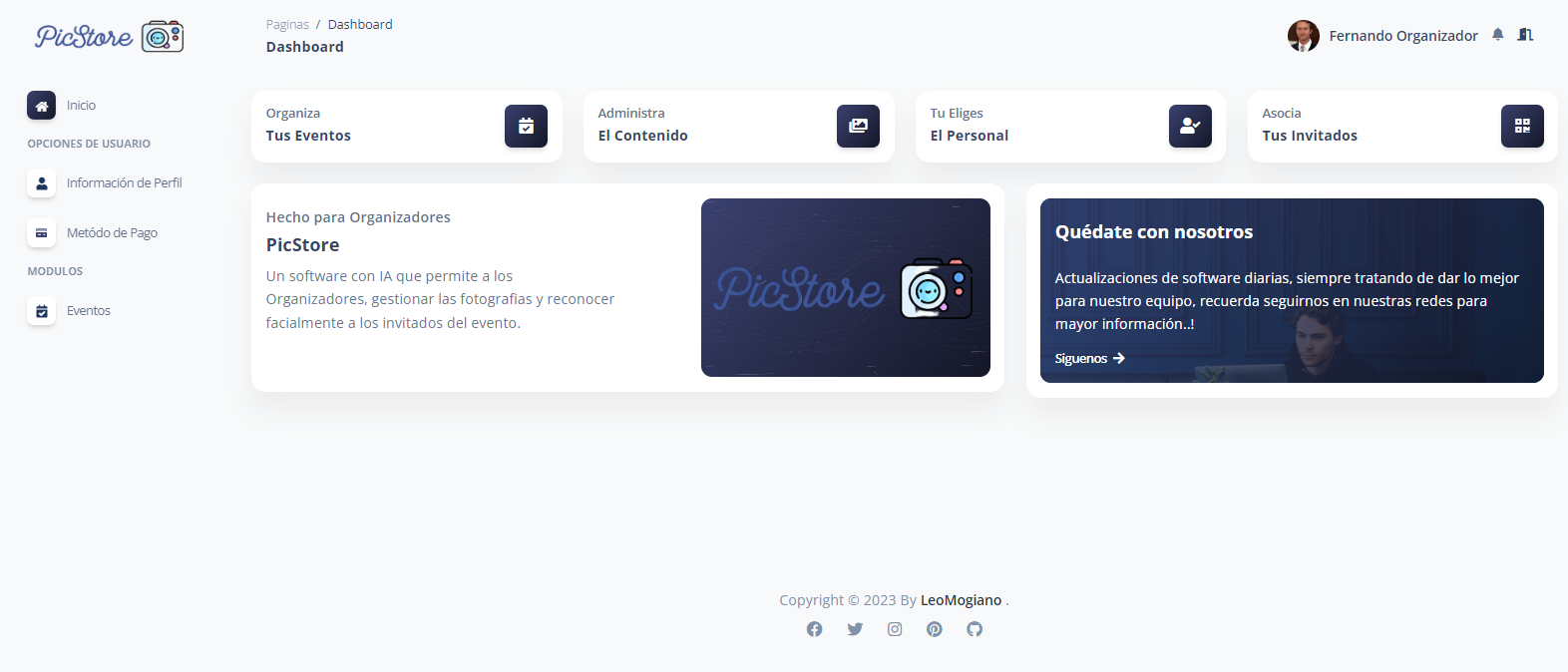
El usuario podrá navegar en la sección principal para obtener mayor información.



## **6.2. Registrase**

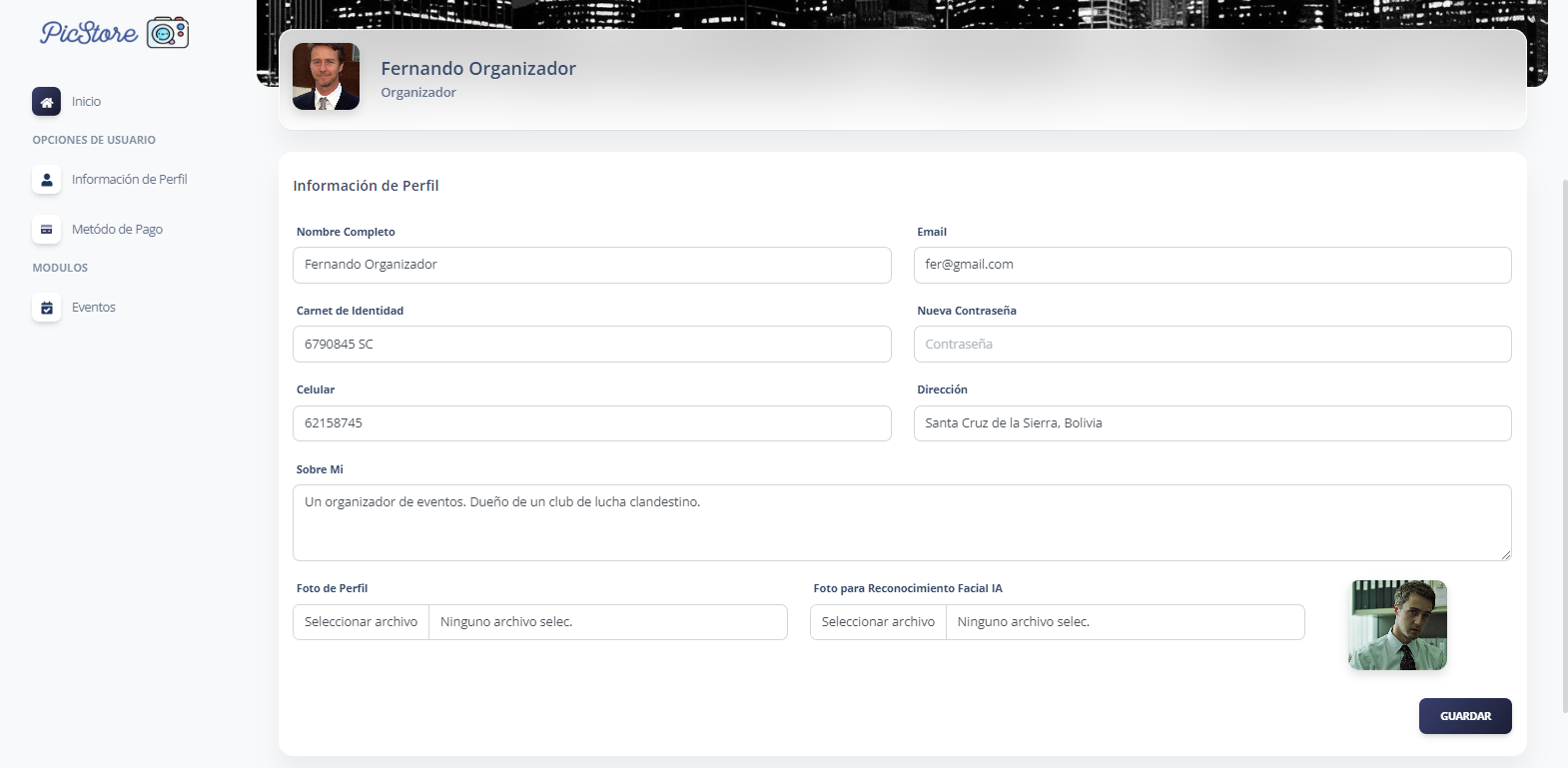
El usuario podrá elegir el tipo de suscripción de paga de acuerdo a sus necesidades, permitiendo registrase como invitado, organizador u fotógrafo.

## **6.3. Dashboard**

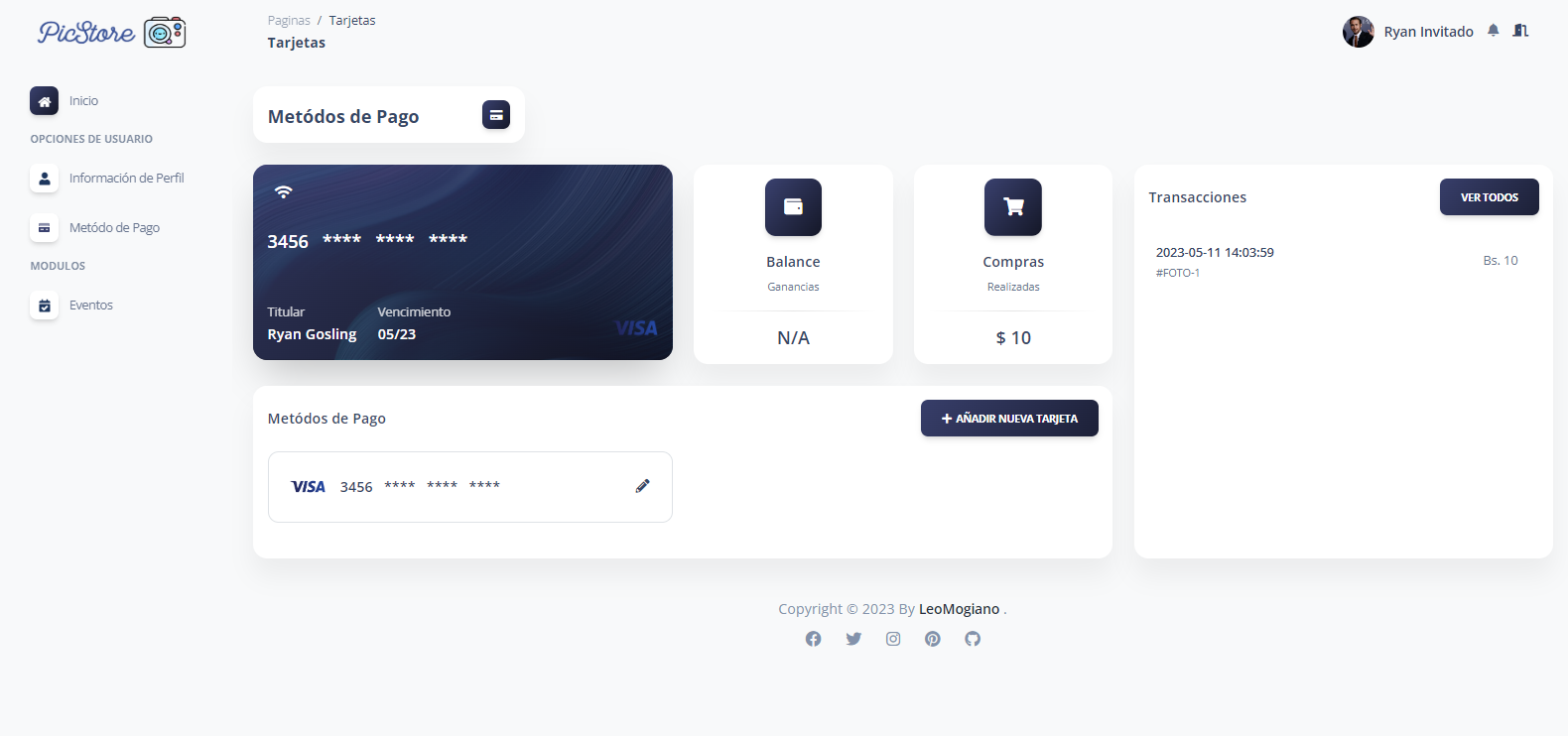
El usuario podrá tener acceso a distintas funcionalidades que se listan en la vista inicial, todo de acuerdo al tipo de rol o suscripción que opto.

## **6.4. Perfil de Usuario**

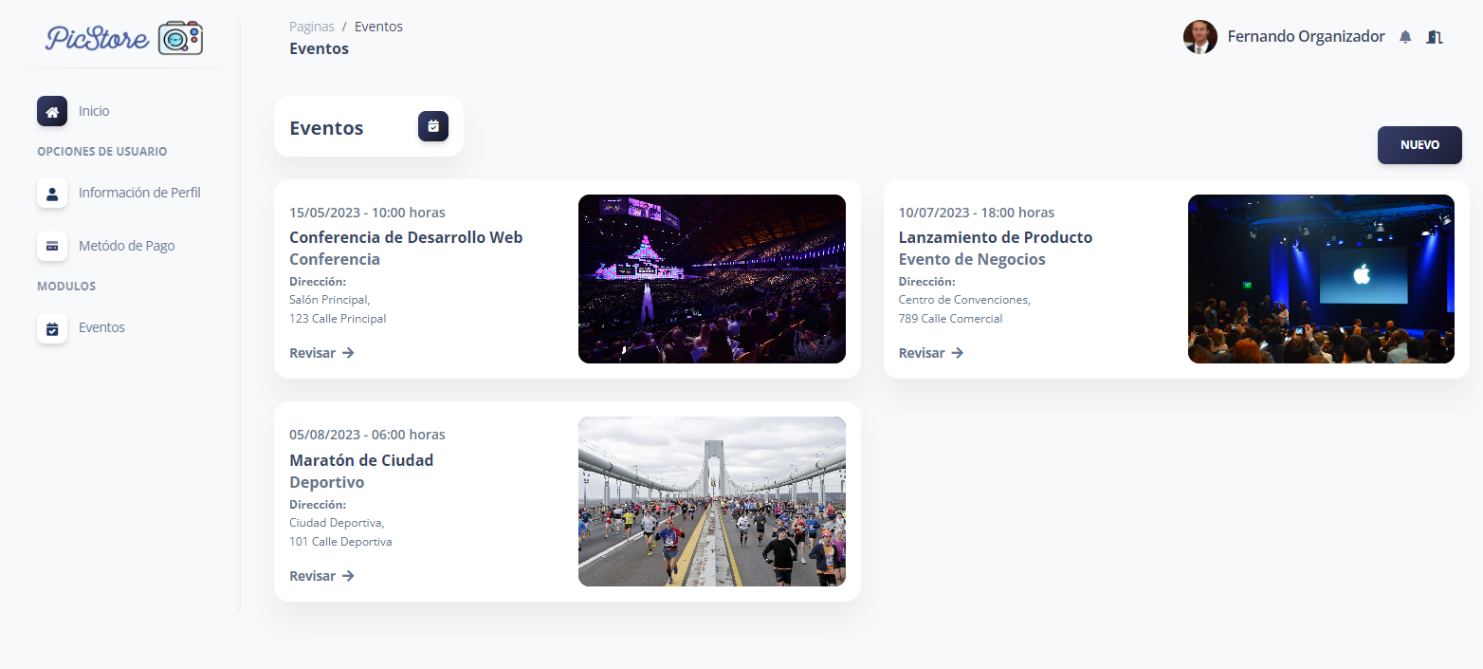
Mediante el módulo de Perfil de Usuario, podremos gestionar nuestra información personal.



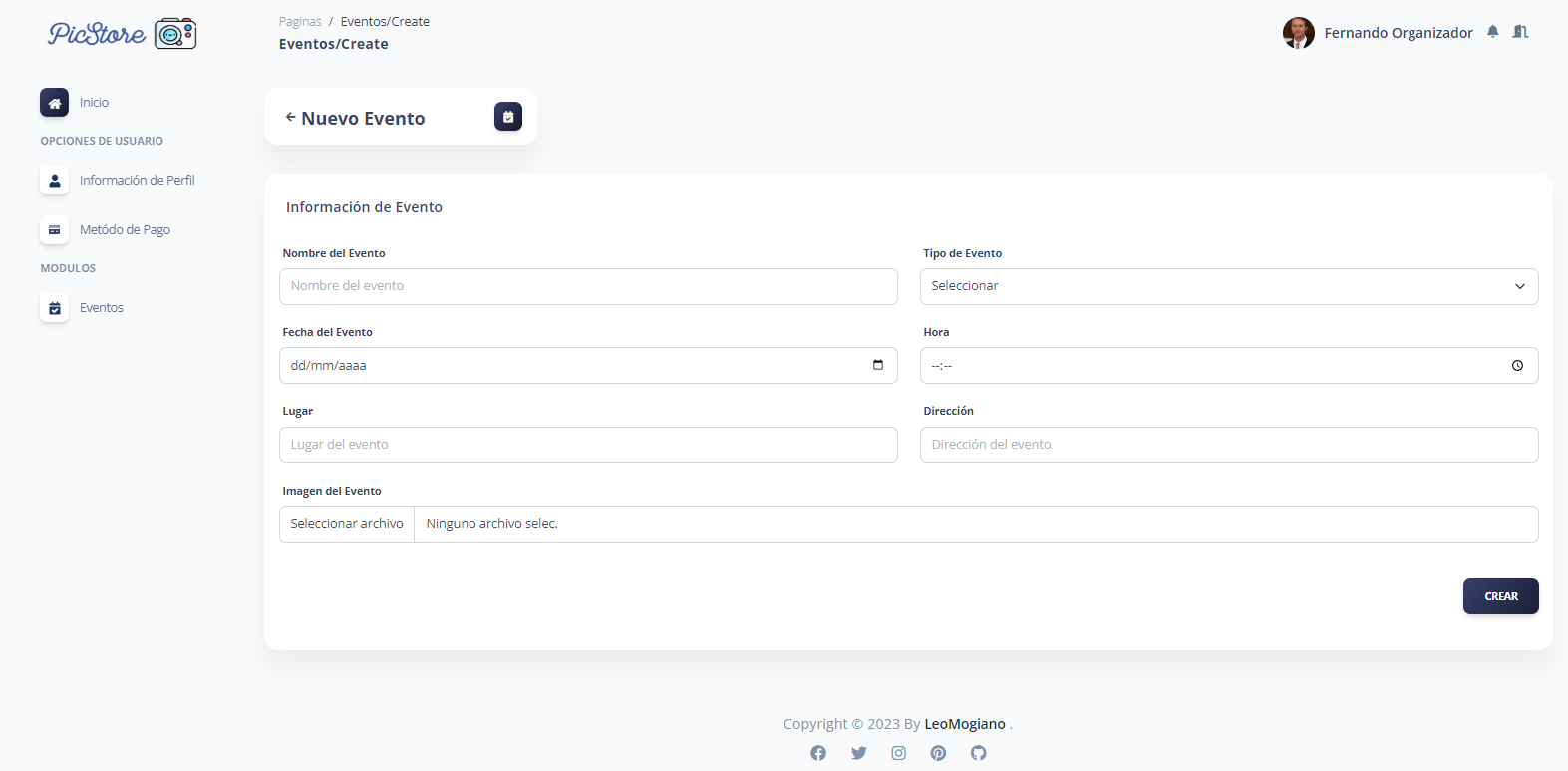
## **6.5. Métodos de Pago**

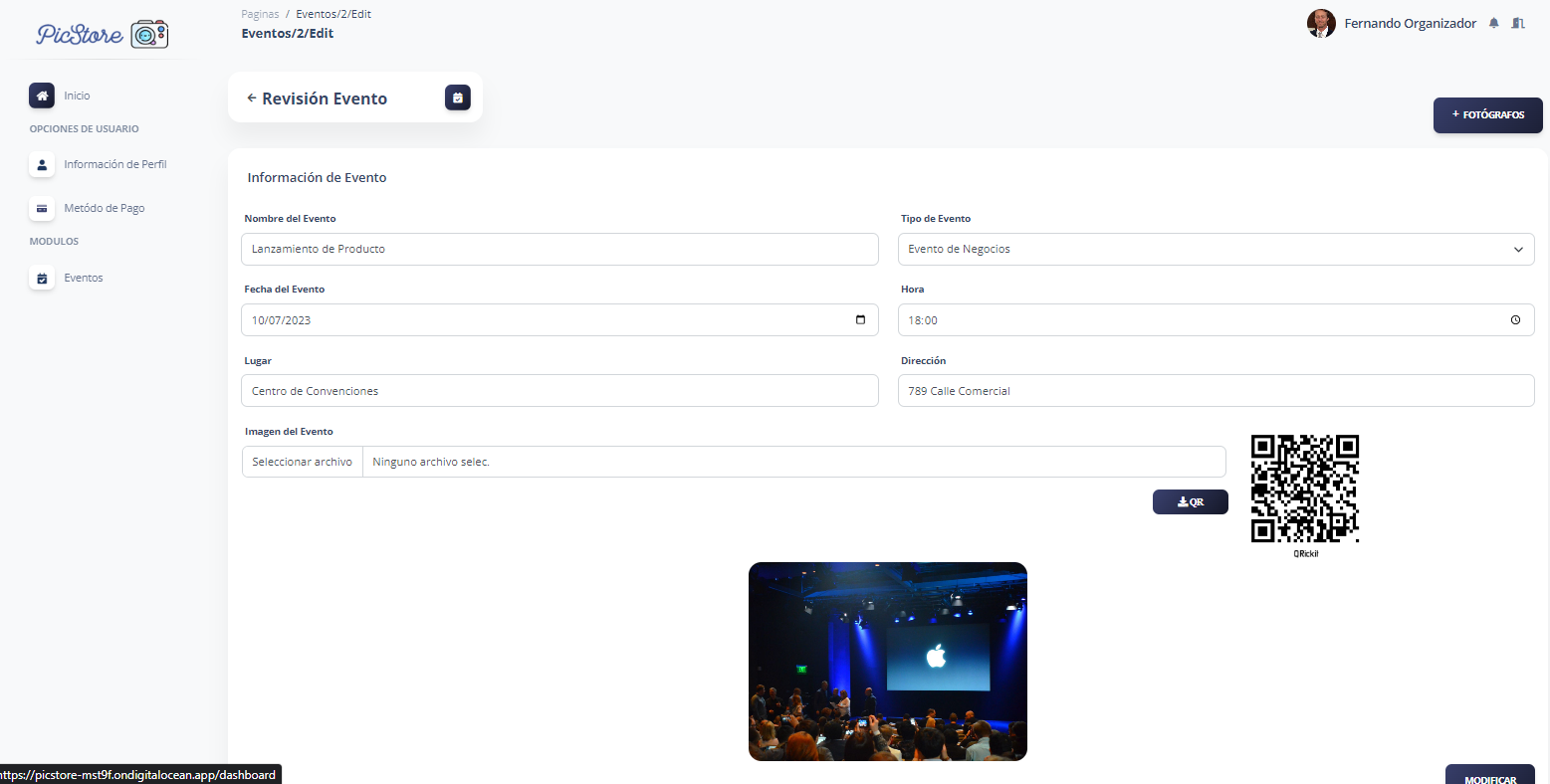
****Aquí se podrá gestionar los métodos de Pago a la hora de realizar la compra de fotos u obtener las ganancias como fotógrafo de la venta de fotos.

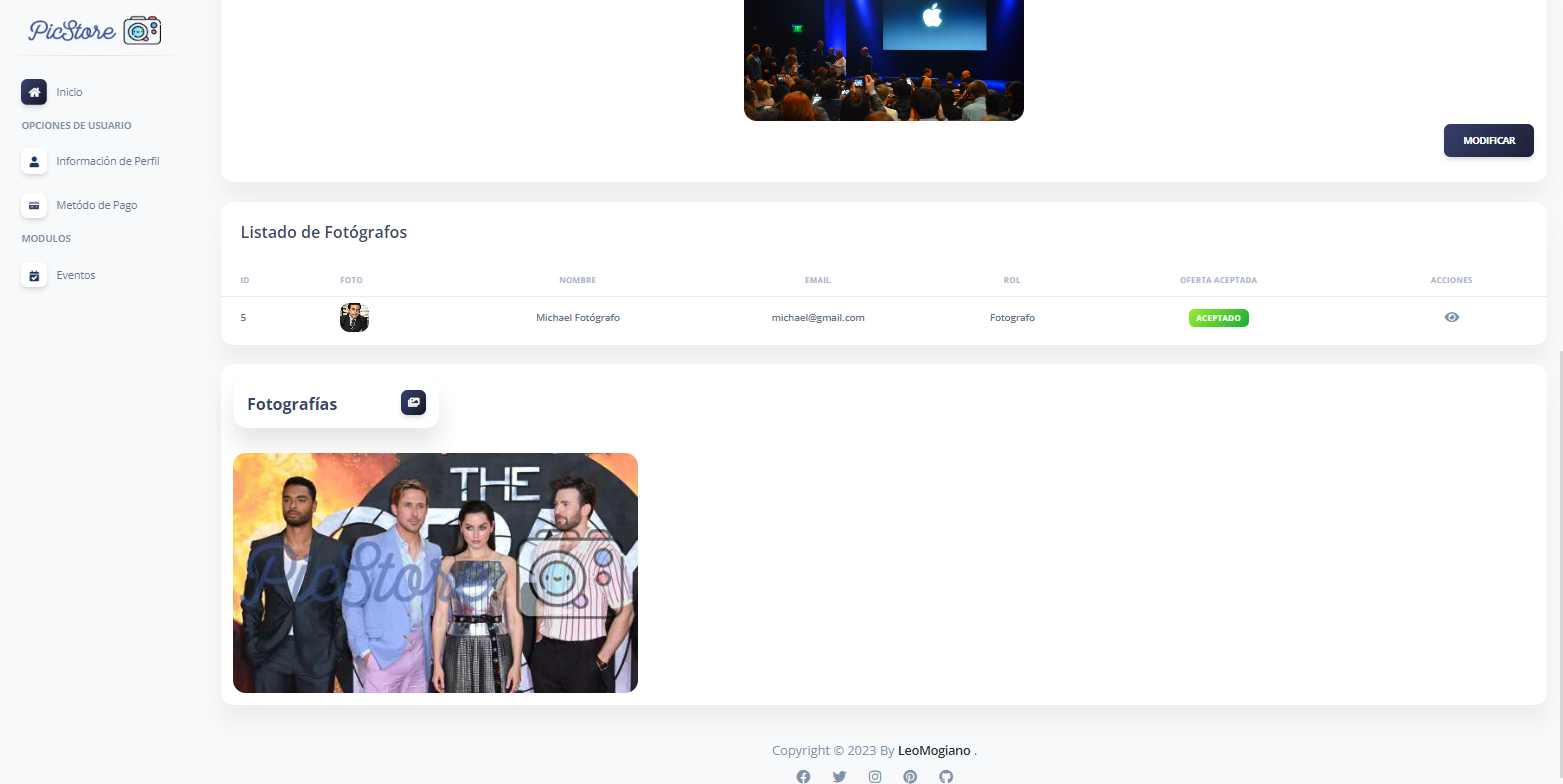
## **6.6. Eventos**

Independiente mente del usuario se podrá ingresar al módulo usuario, permitiéndoles crear nuevos eventos para los organizadores o visualizar eventos asociados para los invitados o fotógrafos contratados.

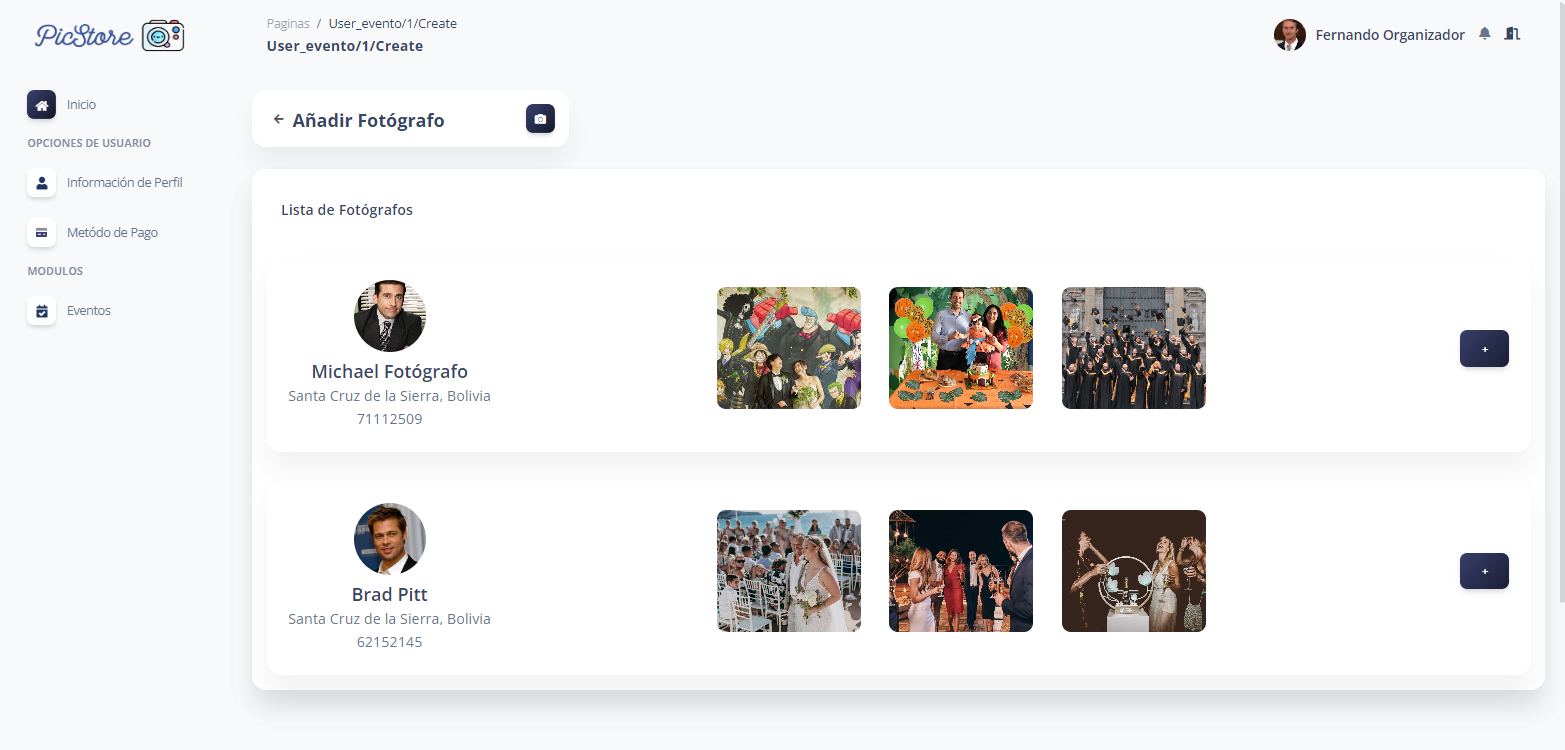
Como se puede observar en la vista de crear evento, como usuario organizador podremos crear un nuevo evento especificando los aspectos más importantes del mismo.



Y en la Revisión de Evento, nosotros podremos consultar detalles importantes del mismo evento como también las fotos adjudicadas al mismo, caso dado que el usuario invitado este asociado o el usuario trabajador este contratado.



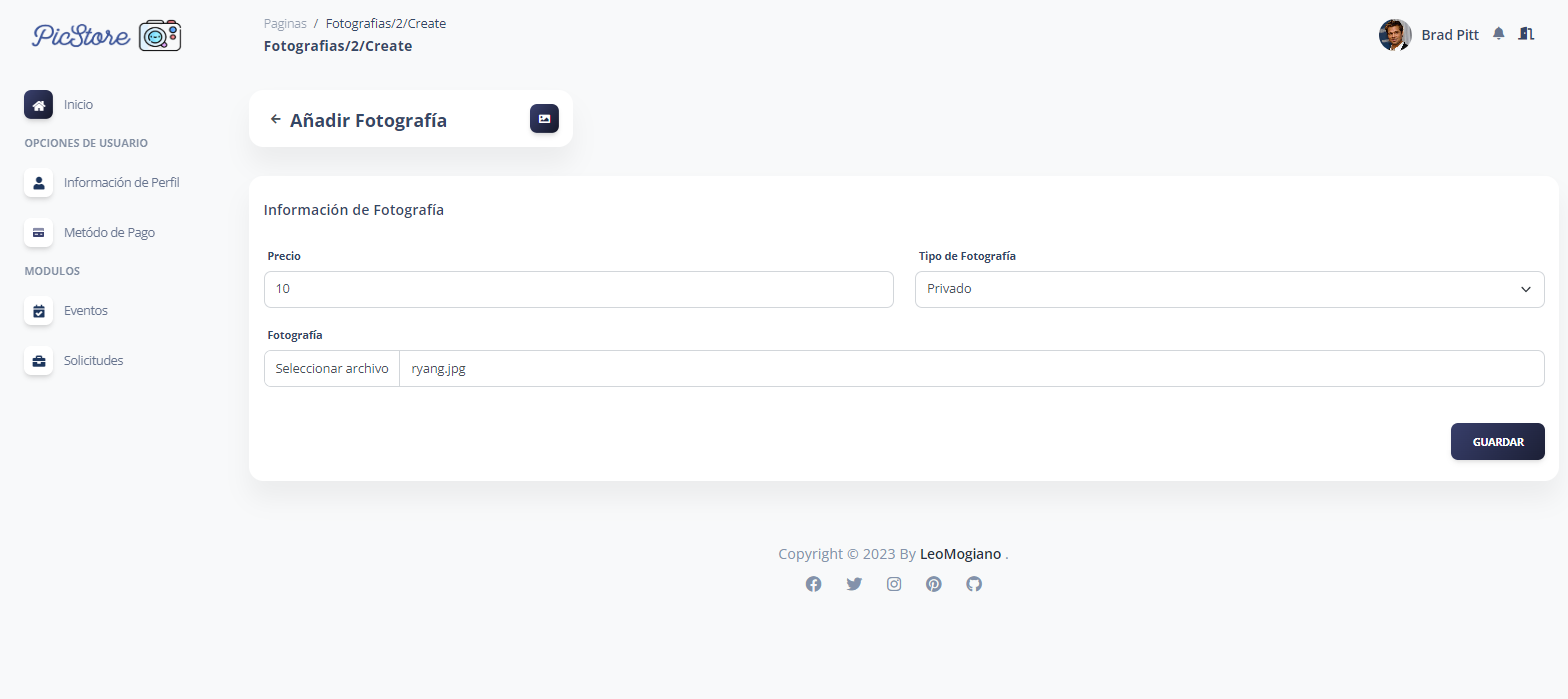
## **6.7. Contrato de Fotógrafos**

A través de la Revisión de Evento, cuando somos organizadores, se nos permitirá mandar una solicitud de trabajo a los fotógrafos.

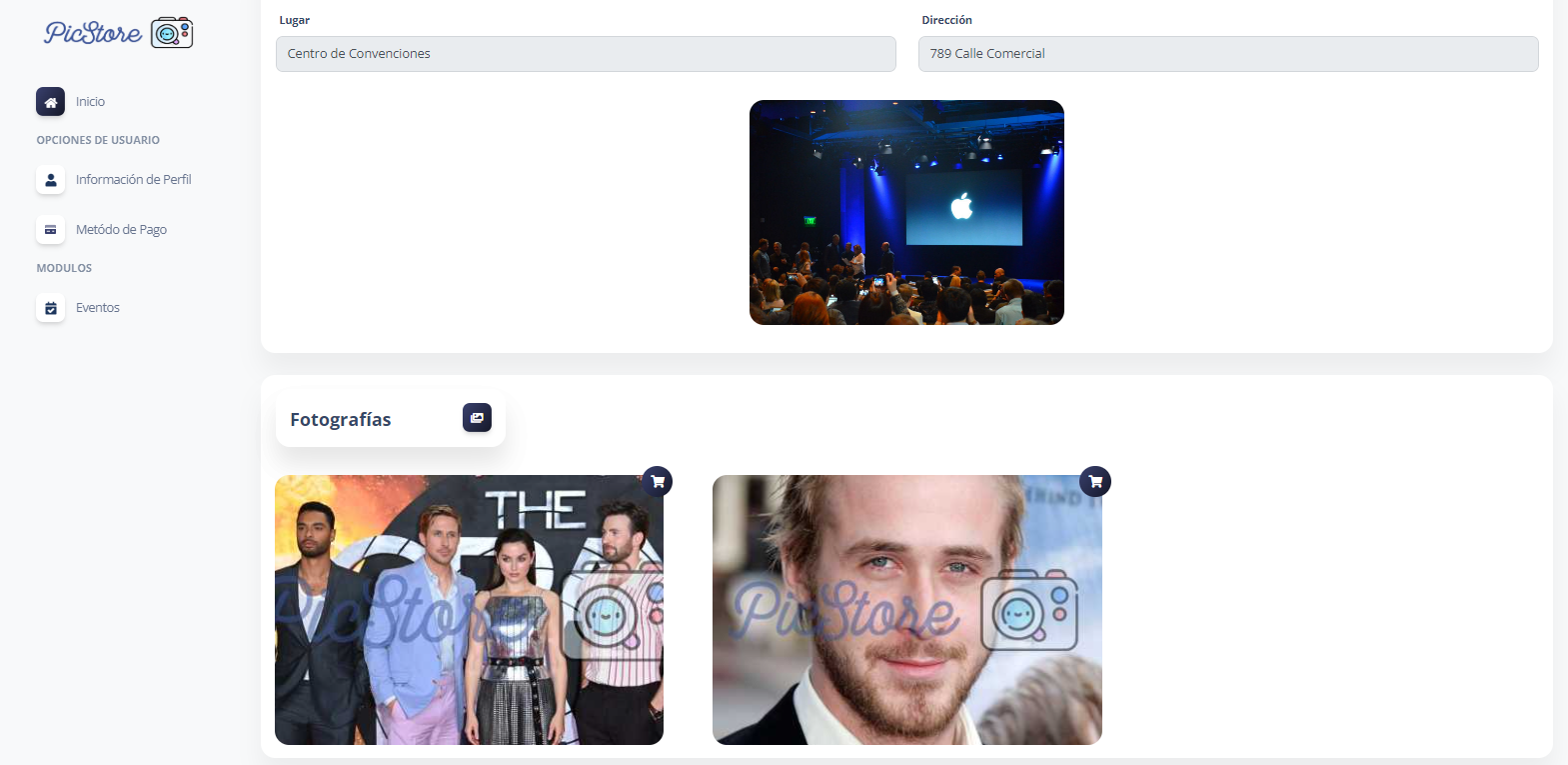
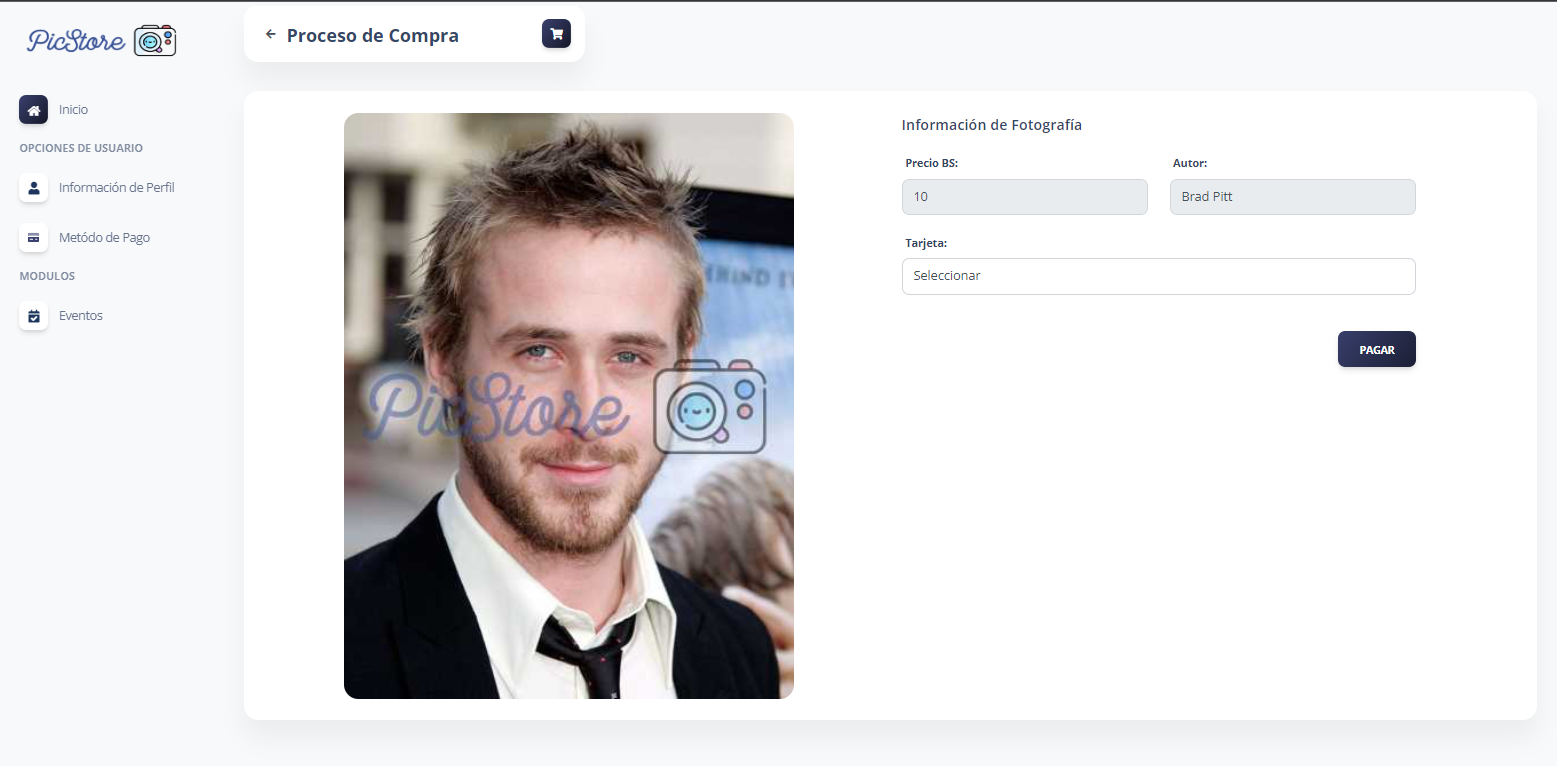
## **6.8. Solicitudes de Trabajo**

Mediante el módulo Solicitudes de Trabajo, solo a los fotógrafos se les permitirá aceptar o rechazar solicitudes de trabajo de parte de los eventos.

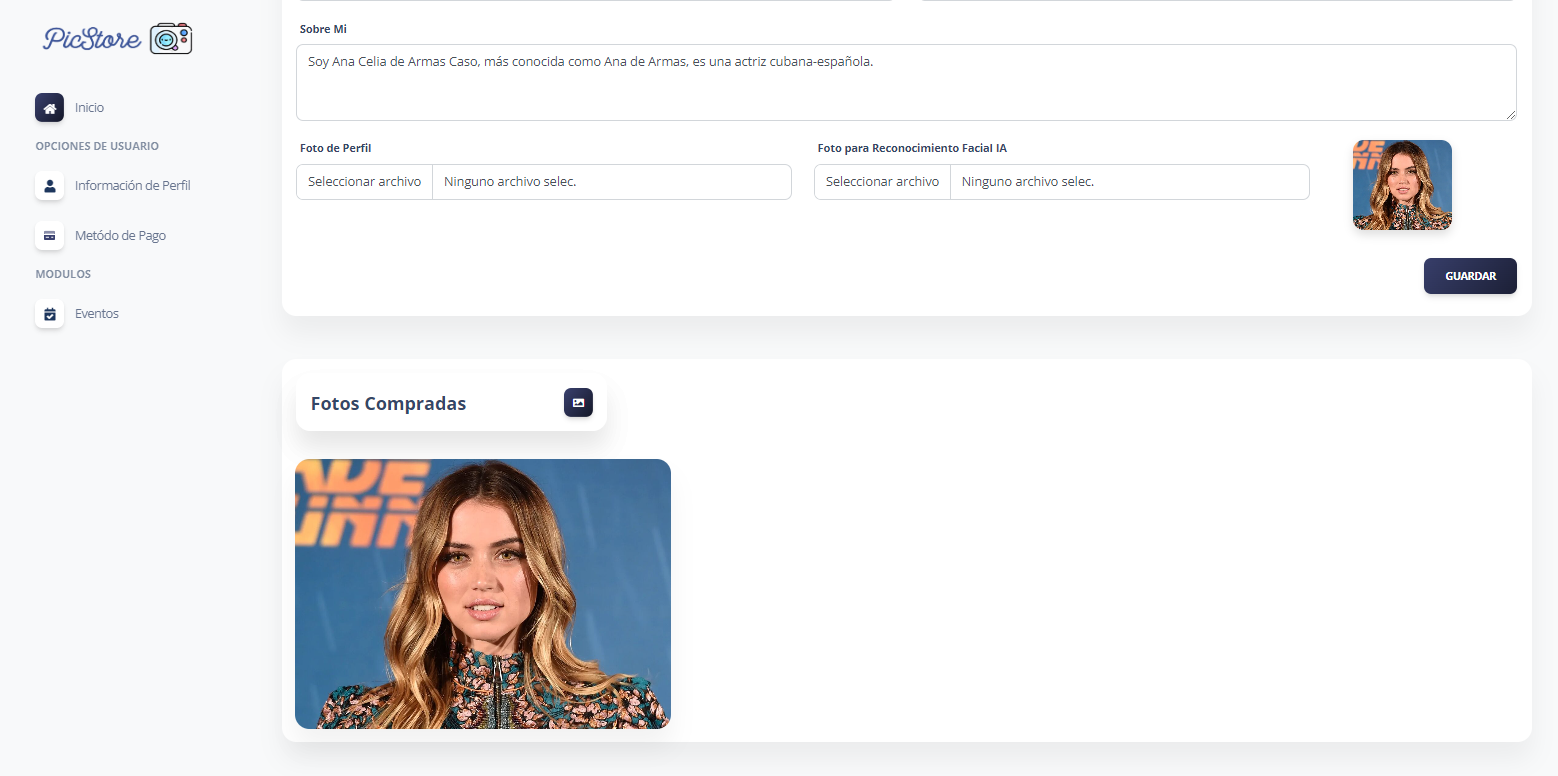
## **6.9. Subida de Fotos**

Mediante la Revisión de Evento, cuando somos fotógrafos, se nos otorga la posibilidad de subir fotos en caso de ser contratado y haber aceptado la solicitud.

## **6.10. Compra de Fotos**

Mediante la Revisión de Evento, en caso de ser Invitado, se le dará la posibilidad de adquirir fotografías ya sea, las fotografías sean pública o privadas y el usuario esté vinculado al evento.  


## **6.11. Fotos Compradas**

A través de del módulo de Información de Perfil, podremos ver todas las fotos que hemos adquirido.