|  |
| --- |
|  |

**Especificación de requisitos de software**

**Proyecto: A.M.P.A Complejo Deportivo**

|  |
| --- |
|  |

Abril de 2023

**Ficha del documento**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Revisión** | **Autor** | **Verificado dep. calidad.** |
| 18/04/23 | ALPHA 1.0 | DevCord Inspiradores Digitales |  |

Documento validado por las partes en fecha:

|  |  |
| --- | --- |
| Por el cliente | Por la empresa suministradora |
|  |  |
| Fdo. D./ Dña | Fdo. D./Dña |

**Contenido**

* **Introducción**

*La introducción de la Especificación de requisitos de software (SRS) debe proporcionar una vista general de la SRS. Debe incluir el objetivo, el alcance, las definiciones y acrónimos, las referencias, y la vista general del SRS.*

* **Propósito**

El propósito de este documento es presentar los requisitos para la API web del complejo deportivo "A.M.P.A", desarrollada por el equipo de software DevCord. El documento describe los requisitos funcionales y no funcionales, interfaces de usuario, interfaces de software, comunicación con otros sistemas, restricciones de diseño e implementación, así como cualquier otro requisito necesario para el correcto funcionamiento de la API.

* **Alcance**

La API web de A.M.P.A es la interfaz que permite a los clientes interactuar con la plataforma web y realizar diversas tareas, tales como la compra de productos deportivos, la reserva de servicios como alquileres de canchas, y la adquisición de membresías del club, entre otras. Este documento cubre los requisitos para la API web de A.M.P.A, desarrollada por el equipo de software DevCord, y se centra en las características y funcionalidades de la API que permiten a los clientes interactuar con el complejo deportivo.

* **Personal Involucrado**

|  |  |
| --- | --- |
| Nombre | Pardo, Catriel Ignacio |
| Rol | Colaborador |
| Responsabilidades | Desarrollo web general |
| Información de contacto | Cartup90@gmail.com |
| ------------------------------ | -------------------------------------------------------------------------------- |
| Nombre | Ledesma Sofia Gimena |
| Rol | Colaboradora |
| Responsabilidades | Desarrollador de software full stack |
| Información de contacto | ledesmasofiagimena49@gmail.com |
| ------------------------------ |  |
| Nombre | Nadia Soledad Brizuela |
| Rol | Colaboradora |
| Responsabilidades | desarrollo web |
| Información de contacto | sole.b\_17@hotmail.com |
| ------------------------------ |  |
| Nombre | Mateo Joel Lopez |
| Rol | Colaborador |
| Responsabilidades | Desarrollo Full Stack |
| Información de contacto | thefilesystem1024@gmail.com |
| ------------------------------ |  |
| Nombre | Godoy Maria Eugenia |
| Rol | Colaboradora |
| Responsabilidades | Desarrollo Frontend |
| Información de contacto | mariaeugeniagodoy8@gmail.com |
| ------------------------------ |  |
| Nombre | Pardo Lesa Nahuel Cruz |
| Rol | Scrum Master |
| Responsabilidades | Desarrollo web general |
| Información de contacto | nahuel.pardo74@gmail.com |
| —--------------------------- |  |
| Nombre | Lautaro Brian Torres |
| Rol | Colaborador |
| Responsabilidades | Desarrollo Web Full Stack |
| Información de contacto | lautioptimus123@gmail.com |

* **Definiciones, acrónimos y abreviaturas**

A continuación, se presentan las definiciones de algunos términos y conceptos relevantes para la comprensión de este documento:

- API: Interfaz de Programación de Aplicaciones, por sus siglas en inglés.  
- A.M.P.A: Asociación Mutual de Programadores Argentinos, el complejo deportivo para el que se desarrolla la API.

- DevCord: El equipo de desarrollo de software encargado del desarrollo de la API web de A.M.P.A.

- Cliente: Usuario final que utiliza la API web de A.M.P.A.

- Plataforma web: La interfaz web que permite a los clientes interactuar con la API de A.M.P.A.

* **Referencias**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Referencia** | **Título** | **Ruta** | **Fecha** | **Autor** |
| IEEE | Standard IEEE 830 - 1998 |  | 18/04/23 | DevCord |
|  |  |  |  |  |

* **Resumen**

Este documento consta de tres secciones. En la primera sección se realiza una introducción al mismo y se proporciona una visión general del propósito y alcance del proyecto. También el personal involucrado. En la segunda sección del documento se realiza una descripción general del producto, con el fin de conocer las principales funciones que este debe o no realizar, como así también las características de los usuarios. Por último, la tercera sección del documento es aquella en la que se definen detalladamente los requisitos que debe satisfacer el sistema

* **Descripción general**
* **Perspectiva del producto**

El sitio web A.M.P.A será diseñado para gestionar de una forma eficaz y rápida a los proveedores para el ámbito comercial, y también a clientes, como para alquilar las instalaciones del complejo deportivo, o adquirir algún producto desde la compra y venta de accesorios, sin tener la necesidad de una interacción física, operando desde su computadora y/o celular.

* **Características de los usuarios**

|  |  |
| --- | --- |
| Tipo de usuario | Administrador |
| Formación | Manejo de herramientas informáticas |
| Actividades | Control y manejo del sistema en general |
| ----------------------------------- | -------------------------------------------------------------------------------- |
| Tipo de usuario | Registrado |
| Formación | Manejo de entornos web |
| Actividades | 1. Observar servicios y productos del complejo.  2. Reservar el uso de alguna de las instalaciones del complejo.  3. Poder cancelar mis reservas.  4. Poder efectuar compra de accesorios.  5. Poder contactarme con la empresa |
| ----------------------------------- |  |
| Tipo de usuario | No registrado |
| Formación | Manejo de entornos web |
| Actividades | 1. Observar servicios y productos del complejo.  2. Poder registrarse y ponerse en contacto con la empresa. |
|  |  |

* **Restricciones**

1. Interfaz para ser usada con internet (solamente o puede ser de escritorio).

2. Lenguajes y tecnologías en uso: HTML, CSS, Bootstrap, JavaScript, Python, MySQL, Django, Angular.

* **Requisitos específicos**
* [#US01: Como usuario deseo tener soporte técnico las 24hs del día, por si algo no funciona como corresponde desde la Web o app móvil](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/72).
* #US02: Como usuario deseo poder pagar con cualquier tipo de tarjeta, sea crédito o débito, transferencias bancarias o pago efectivo.
* [#US03: Como usuario me gustaría poder alquilar las instalaciones o servicios brindados con varios meses de anticipación y congelar la tarifa.](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/74)
* [#US04: Como usuario me gustaría poder loguearme y crear mi usuario con los datos de mi cuenta google o de facebook. Sin tener que llenar datos.](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/75)
* [#US05: Como usuario, quiero poder ver las descripciones detalladas de los productos, incluyendo imágenes y especificaciones, para tomar una decisión informada sobre mi compra.](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/76)
* #US06; Como usuario deseo recibir notificaciones con los torneos próximos.
* [#US07: Como usuario me gustaría ver las últimas noticias, torneos y eventos próximos.](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/78)
* [#US08: Como usuario deseo contar con el alquiler del equipamiento para cada deporte y reservarlo desde la app.](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/79)
* [#US09: Como usuario, quiero poder agregar productos a mi carrito de compras y poder modificarlo fácilmente antes de la compra.](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/80)
* [#US10: Como usuario, quiero realizar búsquedas de productos o servicios fácilmente para encontrar lo que necesito de manera rápida y sencilla.](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/81)
* [#US11: Como usuario, quiero poder compartir productos que me gustan en las redes sociales y enviarlos a amigos y familiares.](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/82)
* [#US12: Como dueño de la página quiero tener servicios que consuman una API rest de prueba para poder realizar peticiones http.](https://github.com/NahuelPardo89/Proyecto-FullStack/issues/104)
* **Sprint Backlog**

#TK01: Crear documento IEEE830

#TK02: Cargar US y TK

#TK03: Realizar diagrama base de datos

#TK04: Realizar diagrama de clases

#TK05: Realizar Diagrama de casos de uso

#TK06: Crear sección chat con atención al cliente en vivo: (Ejemplo: Burbuja de WhatsApp flotante en la esquina inferior)

#TK07: Codificación de página de pago, e integrar plataforma de pagos en linea (Mercadopago o TodoPago) y transferencia bancaria o pago efectivo en Oficinas del polideportivo A.M.P.A..

#TK08: Codificación Página SERVICIOS, agregar instalaciones y productos

#TK09: Codificación página REGISTRO y LOGIN, integración de registro e inicio de sesión con cuenta de Google/Facebook. Opción de registrarse con correo electrónico.

#TK10: Agregar descripción y especificaciones de cada producto e instalación.

#TK11: Funcionalidad de notificaciones para eventos, torneos próximos

#TK12: Codificación Sidebar con información sobre los eventos, torneos próximos a disputarse.

#TK13: Opción de agregar equipamiento para cada deporte, cobrando una tarifa fija por el alquiler del mismo.

#TK14: Codificar Página de Carrito, funcionalidad de agregar productos e instalaciones al carrito de compras. Botones para eliminar o modificar cantidad de productos

#TK15: Desarrollo de función de búsqueda para productos.

#TK16: Funcionalidad de compartir productos en redes sociales.

#TK17: Codificación Modulo Home: Login-Register.

#TK18: Codificación Modulo Home: Contacto.

#TK19: Codificación Modulo Dashboard: Cliente-Empleado.

#TK20: Codification Modulo Shared: Navbar.

#TK21: Codification Modulo Shared: Footer.

#TK22: Codification Modulo Home: About Us.

#TK23: Codificación Modulo Reservas: Instalaciones-Agenda.

#TK24: Codificación Modulo Productos: Productos-Carrito.

#TK25: Crear el servicio de Login

#TK26: Crear el servicio de registro

#TK27: Crear la tienda virtual

#TK28: Crear proyecto en Django

#TK29: Crear app en Django

#TK30: Generar la tabla productos

#TK31: Crear un superusuario

-#TK32: Generar un CRUD en el panel de administración

#TK33: Cargar productos y usuarios

#TK34: Inyectar a los componentes los correspondientes servicios a fin de que los componentes puedan acceder a los datos y mostrarlos en pantalla.

#TK35: Incorporar la programación reactiva (asincrónica) mediante observables para la comunicación entre servicios y componentes.

-#TK36: Agregar funcionalidades a la página carrito

#TK37: Crear el servicio que consume la api (json) utilizando servicio de HttpClient

#TK38: Crear modelo usuario

#TK39: Crear serial layer usuario

#TK40: Crear vista login

#TK41: Crear vista registro

#TK42: Crear vista logout

#TK43 Crear formulario reactivo para login

#TK44: Crear formulario reactivo para registro

#TK45: Django- crear modelo producto

#TK46: Django- crear modelo Categoría

#TK47: Django- crear serial layser de productos

#TK48: Django- crear serial layser de categorías

#TK49: Django- crear vista para hacer el CRUD de productos

#TK50: Django- Crear vista para hacer el CRUD de categorías

#TK51: Angular – crear formulario reactivo para cargar productos

#TK52: Angular – crear formulario reactivo para editar productos

#TK53: Angular – crear formulario reactivo para cargar categorías

#TK54: Angular – crear formulario reactivo para editar categorías

#TK55: Crear servicio para conectarse a la API Rest para realizar el CRUD

#TK56 Crear dashboard admin Instalaciones

#TK57 Crear dashboard admin Reservas

#TK58 Conexion back y front dashboard user reservas

#TK59 Conexión back y front dashboard admin Instalaciones y reservas

#TK60 Codificación página reservas con conección back y front formulario reservas

#TK61 Crear dashboard cliente

#TK62 Crear modelo, serializer y vista carrito

#TK63 Crear modelo, serilaizer y vista Factura

#TK64 Crear vista para pasarela de pagos(backend)

#TK65 Crear componente y servicio carrito

#TK66 Crear componente y servicio Pasarela de pagos

#TK67 Agregar permisos a las vistas de Django (Admin- islogged)

* **Requisitos funcionales**
* RF1 - Registro de usuarios: La API debe permitir el registro de usuarios nuevos en la plataforma web, solicitando la información necesaria para el proceso de registro (nombre, correo electrónico, contraseña, etc.).
* RF2 - Inicio de sesión: La API debe permitir a los usuarios iniciar sesión en la plataforma web, solicitando las credenciales de acceso (correo electrónico y contraseña).
* RF3 - Compra de productos deportivos: La API debe permitir a los usuarios realizar compras de productos deportivos a través de la plataforma web, seleccionando los productos deseados y proporcionando la información necesaria para el proceso de compra (dirección de envío, información de pago, etc.).
* RF4 - Reserva de servicios deportivos: La API debe permitir a los usuarios reservar servicios deportivos como alquiler de canchas, solicitar entrenamientos personales, y reservar clases en línea o presenciales.
* RF5 - Adquisición de membresías: La API debe permitir a los usuarios adquirir membresías para el club deportivo, proporcionando la información necesaria para el proceso de adquisición (tipo de membresía, duración, información de pago, etc.).
* RF6 - Gestión de la cuenta de usuario: La API debe permitir a los usuarios gestionar su cuenta de usuario, incluyendo la actualización de la información de perfil, cambio de contraseña, entre otros.
* RF7 - Consulta de información del complejo deportivo: La API debe permitir a los usuarios consultar información relevante sobre el complejo deportivo, incluyendo horarios, ubicación, servicios disponibles, y cualquier otra información relevante para los usuarios.
* RF8 - Comunicación con los usuarios: La API debe permitir a los usuarios recibir notificaciones de la plataforma web, incluyendo confirmaciones de compra, recordatorios de citas, y cualquier otra información relevante para los usuarios.

**SPRINT**

|  |  |
| --- | --- |
| ***N° de sprint*** | ***00*** |
| *Sprint Backlog* | *#TK01: Crear documento IEEE830*  *#TK02: Cargar US y TK*  *#TK03: Realizar diagrama base de datos*  *#TK04: Realizar diagrama de clases*  *#TK05: Realizar Diagrama de casos de uso* |
| *Responsabilidades* | ● *Registrar la Especificación de Requerimientos mediante la documentación IEEE830 (subirlo en una carpeta de GitHub en la rama main).*  ● *Git/GitHub : Instalación y registración*  ● *Crear Project estilo Kanban con incorporación de Historias de Usuarios, tareas, e incidencias.*  ● *Idea de mapa del Sitio*  ● *Llevar registro de meetings y toda la info necesaria dentro de la Wiki de GitHub.* |
| *Calendario* | ***Fecha Inicio = 10/04/2023 - Fecha de Fin = 21/04/2023*** |
| *Inconvenientes:* |  |

|  |  |
| --- | --- |
| ***N° de sprint*** | ***01*** |
| *Sprint Backlog* | #TK17: Codificación Modulo Home: Login-Register.  #TK18: Codificación Modulo Home: Contacto.  #TK19: Codificación Modulo Dashboard: Cliente-Empleado.  #TK20: Codificación Modulo Shared: Navbar.  #TK21: Codificación Modulo Shared: Footer.  #TK22: Codificación Modulo Home: About Us.  #TK23: Codificación Modulo Reservas: Instalaciones-Agenda.  #TK24: Codificación Modulo Productos: Productos-Carrito. |
| *Responsabilidades* | ● *Convertir los archivos .html del proyecto previo, en una SPA en Angular con módulos y con sus components correspondientes*  ● *Incorporar la navegabilidad de la aplicación mediante Routing con Angular.*  ● *Subir en la carpeta documentación las tablas actualizadas de la base de datos*  ● *Crear una branch DEVELOP para actualizar los cambios que no se encuentran aún en condiciones de incorporarse a la branch MAIN.*  ● *Llevar registro de meetings y toda la info necesaria dentro de la Wiki de GitHub.* |
| *Calendario* | ***Fecha Inicio = 24/04/2023 - Fecha de Fin = 07/05/2023*** |
| *Inconvenientes:* |  |

|  |  |
| --- | --- |
| ***N° de sprint*** | ***02*** |
| *Sprint Backlog* | #TK25: Crear el servicio de Login  #TK26: Crear el servicio de registro  #TK27: Crear la tienda virtual  #TK28: Crear proyecto en Django  #TK29: Crear app en Django  #TK30: Generar la tabla productos  #TK31: Crear un superusuario  -#TK32: Generar un CRUD en el panel de administración  #TK33: Cargar productos y usuarios  #TK34: Inyectar a los componentes los correspondientes servicios a fin de que los componentes puedan acceder a los datos y mostrarlos en pantalla.  #TK35: Incorporar la programación reactiva (asincrónica) mediante observables para la comunicación entre servicios y componentes.  -#TK36: Agregar funcionalidades a la página carrito  #TK37: Crear el servicio que consume la api (json) utilizando servicio de HttpClient |
| *Responsabilidades* | ● *Crear servicios que consuman un API Rest de prueba. Para realizar las peticiones http, utilizar el servicio de HttpClient..*  ● *Inyectar a los componentes los correspondientes servicios a fin de que los componentes puedan acceder a los datos y mostrarlos en pantalla. Incluye enlace de datos (Data binding).*  ● *Incorporar la programación reactiva (asincrónica) mediante observables para la comunicación entre servicios y componentes.*  ● *Establecer el modelo de base de datos del primer sprint en Django usando clases.*  ● *Establecer el sistema de usuarios en Django-Admin (registro, login y roles)*  ● *Crear un ABM de productos/servicios utilizando el panel de control de Django.* |
| *Calendario* | ***Fecha Inicio = 08/05/2023 - Fecha de Fin = 21/05/2023*** |
| *Inconvenientes:* |  |

|  |  |
| --- | --- |
| ***N° de sprint*** | ***03*** |
| *Sprint Backlog* | #TK38: Crear modelo usuario  #TK39: Crear serial layer usuario  #TK40: Crear vista login  #TK41: Crear vista registro  #TK42: Crear vista logout  #TK43 Crear formulario reactivo para login  #TK44: Crear formulario reactivo para registro  #TK45: Django- crear modelo producto  #TK46: Django- crear modelo Categoría  #TK47: Django- crear serial layser de productos  #TK48: Django- crear serial layser de categorías  #TK49: Django- crear vista para hacer el CRUD de productos  #TK50: Django- Crear vista para hacer el CRUD de categorías  #TK51: Angular – crear formulario reactivo para cargar productos  #TK52: Angular – crear formulario reactivo para editar productos  #TK53: Angular – crear formulario reactivo para cargar categorías  #TK54: Angular – crear formulario reactivo para editar categorías  #TK55: Crear servicio para conectarse a la API Rest para realizar el CRUD |
| *Responsabilidades* | ● *Incorporar formularios reactivos de Registro y Login para la autenticación y formularios para la manipulación (Alta) de servicios/productos/suscripción, según el caso con sus respectivas validaciones.*  Diseño de los formularios:  Deberán estar conformados por **input**types acorde al dato que se va a ingresar (<https://developer.mozilla.org/es/docs/Web/HTML/Element/input>) - Deberán informar al usuario mediante mensajes personalizados (Validaciones) - Deberán respetar los colores semánticos que propone Bootstrap, <https://getbootstrap.com/docs/5.0/utilities/borders/> Por ej.   1. <span class="border border-success"></span> 2. <span class="border border-danger"></span>   ● *Conexión con el backend para el registro e inicio de sesión.*  ● *Conexión con backend para almacenar los datos de los productos/servicios y detalles de producto/servicios en la DB a través del modelo en Django.*  ● *Registro de usuarios.*  ● *Login de usuarios.*  ●*Registro / Alta de al menos 3 productos/servicios/suscripción ( solo desde el rol Administrador, mediante su panel de control).*   * ***El rol ADMIN****pueda ver la lista de usuarios registrados y el listado de los productos/servicios/suscripción según corresponda.* * ***El rol USUARIO****registrado solo que pueda ver su Dashboard para editar su perfil.*   ● *Los datos serán consumidos desde la Base de datos MySQL.*  **OPCIONAL**: Token validación del login con JWT. |
| *Calendario* | ***Fecha Inicio = 22/05/2023 - Fecha de Fin = 04/06/2023*** |
| *Inconvenientes:* |  |
| ***N° de sprint*** | ***04*** |
| *Sprint Backlog* | #TK56 Crear dashboard admin Instalaciones  #TK57 Crear dashboard admin Reservas  #TK58 Conexion back y front dashboard user reservas  #TK59 Conexión back y front dashboard admin Instalaciones y reservas  #TK60 Codificación página reservas con conección back y front formulario reservas  #TK61 Crear dashboard cliente  #TK62 Crear modelo, serializer y vista carrito  #TK63 Crear modelo, serilaizer y vista Factura  #TK64 Crear vista para pasarela de pagos(backend)  #TK65 Crear componente y servicio carrito  #TK66 Crear componente y servicio Pasarela de pagos  #TK67 Agregar permisos a las vistas de Django (Admin- islogged) |
| *Responsabilidades* | ● Tablero de control para usuarios: *Mostrar catálogo (mínimo 3 productos o servicios o suscripción). Seleccionar para comprar. Simular Pago con API Mercado Pago o similar*  <https://www.mercadopago.com.ar/developers/es/docs#online-payments>  ●*Tablero de control administrador CRUD básico funcional de producto y cliente.*  ● *Actualizar documentación IEEE830 - tablero Kanban (project) y Wiki.*  ● *Actualizado todo el proyecto en GitHub como respaldo*  ● *Demo final en video con participación de todo el equipo describiendo todas las funcionalidades. (quien no explique en el mismo alguna de las funcionalidades/código/gestión del proyecto, no será tenido en cuenta para calificación)*  ● *OPCIONAL: Sitio publicado en Servidor* |
| *Calendario* | ***Fecha Inicio = 05/06/2023 - Fecha de Fin = 18/06/2023*** |
| *Inconvenientes:* |  |