**Especificación de requisitos de software**

**Proyecto:**

# TURNERO PARA INSTITUCION MÉDICA

2022

.

# Ficha del documento

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Revisión** | **Autor** | **Verificado dep. Calidad.** |
| 20/06/2023 | 20/06/2023 | **GRUPO N°91**    Benítez Alejandro  Olima Sebastián  Verónica Nievas  Lamia Natalia |  |

# Contenido

**FICHA DEL DOCUMENTO 3**

**CONTENIDO 4**

**1 INTRODUCCIÓN 6**

|  |  |
| --- | --- |
| **1.1** | **Propósito** |
| **1.2** | **Alcance** |
| **1.3** | **Personal involucrado** |
| **1.4** | **Definiciones, acrónimos y abreviaturas** |
| **1.5** | **Referencias** |
| **1.6** | **Resumen** |
| **2** | **DESCRIPCIÓN GENERAL** |
| **2.1** | **Perspectiva del producto** |
| **2.3** | **Características de los usuarios** |
| **2.4** | **Restricciones** |
| **3** | **REQUISITOS ESPECÍFICOS** |

**17**

|  |  |
| --- | --- |
| 3.2.1 | Product Backlog |
| **3.3** | **Sprints** |
| 3.3.1 | Sprint 1 |
| 3.3.2 | Spint 2 |

17

|  |
| --- |
| 3.3.3. Sprint 3 |
| 3.3.4. Spint 4 |

# 1 Introducción

Este documento es una Especificación de Requisitos Software (ERS) para el Sistema de información para la gestión de procesos y control de inventarios. Esta especificación se ha estructurado basándose en las directrices dadas por el estándar IEEE Práctica Recomendada para Especificaciones de Requisitos Software ANSI/IEEE 830, 1998.

## 1.1 Propósito

El presente documento tiene como propósito definir las especificaciones funcionales, para el desarrollo de un sistema de información web que permitirá: desarrollar una página web que tiene como destino de aplicación el ámbito de la salud y será planteada como un turnero para instituciones médicas, que permitirá a los usuarios registrarse como pacientes o prestadores de servicios de salud y acceder a la posibilidad de gestionar turnos de atención por medios digitales, centralizando la información de especialidades, días y horarios disponibles, direcciones y cobertura de obras sociales y abonar los servicios solicitados, tales como: consulta, estudios, evaluaciones médicos y análisis de laboratorio.

## 1.2 Alcance

Esta especificación de requisitos está dirigida al usuario del sistema, correspondiente al ámbito de salud, y servirá para la autogestión del usuario, desde esta página se podrá:

* Registrarse como usuario o institución.
* Reservar turno por especialidad.
* Sugerir disponibilidad de días y horarios por especialista.

• Abonar los servicios solicitados.

Además permitirá al administrador de la página web llevar un control ordenado de sus servicios y pacientes.

## 1.2 Personal involucrado

|  |  |
| --- | --- |
| **Nombre** | Benítez Alejandro |
| **Rol** | Administrador / Desarrollador |
| **Categoría Profesional** | Estudiante |
| **Responsabilidad** | Organización, gestión, desarrollo y mantenimiento del software. |
| **Información de contacto** | alebenitez.89@gmail.com |

|  |  |
| --- | --- |
| **Nombre** | Olima Sebastián |
| **Rol** | Administrador / Desarrollador |
| **Categoría Profesional** | Estudiante |
| **Responsabilidad** | Organización, gestión, desarrollo y mantenimiento del software. |
| **Información de contacto** | edificarvcp@gmail.com |

|  |  |
| --- | --- |
| **Nombre** | Verónica Nievas |
| **Rol** | Administrador / Desarrollador |
| **Categoría Profesional** | Estudiante |
| **Responsabilidad** | Organización, gestión, desarrollo y mantenimiento del software. |
| **Información de contacto** | veronievasguardia@gmail.com |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Nombre** | Lamia Natalia |
| **Rol** | Administrador / Desarrollador |
| **Categoría Profesional** | Estudiante |
| **Responsabilidad** | Organización, gestión, desarrollo y mantenimiento del software. |
| **Información de contacto** | natalialamia@yahoo.com.ar |

|  |  |
| --- | --- |
| **Nombre** | Barrionuevo Eric |
| **Rol** | Administrador / Desarrollador |
| **Categoría Profesional** | Estudiante |
| **Responsabilidad** | Organización, gestión, desarrollo y mantenimiento del software. |
| **Información de contacto** | tecsamuel25@gmail.com |

**1.3 Definiciones, acrónimos y abreviaturas**

|  |  |
| --- | --- |
| ***Nombre*** | ***Descripción*** |
| **Usuario** | Persona que usará el sistema para gestionar procesos |
| **SIS-I** | Sistema de Información Web para la Gestión de Procesos Administrativos y Académicos |
| **ERS** | Especificación de Requisitos Software |
| **RF** | Requerimiento Funcional |
| **RNF** | Requerimiento No Funcional |
| **FTP** | Protocolo de Transferencia de Archivos |
| **Moodle** | Aula Virtual |

## 1.4 Referencias

|  |  |  |
| --- | --- | --- |
| **Título del Documento** |  | **Referencia** |
| Standard IEEE 830 - 1998 | IEEE |  |

**1.6. Resumen**

Breve resumen de las secciones del documento.

# 2 Descripción general

## 2.1 Perspectiva del producto

El sistema para la gestión de turnos será un producto diseñado para trabajar en entornos WEB, lo que permitirá su utilización de forma rápida y eficaz, además de centralizar la información sobre profesionales de la salud y disponibilidad de días y horarios para elección del paciente, además de poder realizar el abono de los servicios solicitados desde la página por parte de los usuarios.

## 2.2Características de los usuarios

|  |  |
| --- | --- |
| **Tipo de usuario** | Administrador |
| **Formación** | Manejo de herramientas informáticas |
| **Actividades** | Control y manejo del sistema en general |
| **Tipo de usuario** | Visitante |
| **Formación** | Manejo de entornos web |
| **Actividades** | Observa e indaga información de la web, se registra y gestiona sus turnos. |

*.*

## 2.3 Restricciones

* Interfaz para ser usada con internet a través de cualquier dispositivo capaz de conectarse a una red.
* Lenguajes y tecnologías en uso: HTML5, CSS3, Bootstrap, Phyton, JavaScript, MySQL , Typescript, Angular, Django.

# 3 Requisitos específicos

## Product Backlog

#U01 Como gerente/administrador de una institución médica quiero un sistema de autogestión de los pacientes para agilizar la reserva de turnos y otras tareas similares de la institución.

#U02 Como usuario general (paciente) quiero ingresar a una página principal donde se informen los servicios de la Institución médica y la posibilidad de gestionar turnos por la aplicación web.

#U03 Como usuario paciente quiero poder registrarme para gestionar turnos de manera online según mis preferencias de profesional, horarios y cobertura de obra social.

“#US04: Como usuario me gustaría recibir un recordatorio de mi turno agendado con fecha y hora 24hs. antes de mismo, con posibilidad de confirmar, reprogramar o cancelar.”.

**BACKEND (crud)**

#U05 Como empleado/a administrativo de una institución médica quiero poder registrar, modificar, controlar disponibilidad de turnos y eliminar diferentes profesionales de la salud por especialidad, disponibilidad horaria, lugar de atención y cobertura de obra social en el sistema.

(Relacionar al modelo spa de angular) ejemplo seba

#US06 Como administrador quisiera agregar un nuevo método de pago para abonar las consultas médicas. E-commerce

#US07: Como usuario paciente quiero poder ingresar en la página web, ingresar y modificar datos de perfil, efectuar pagos de forma segura, y cancelar un turno asignado o reprogramar. (Relacionar con el #U05)

**Interfaces**.

**Interfaz de usuario. VER**

Para que exista facilidad de uso del sistema, la interfaz constará con un conjunto de objetos como: tarjetas, botones, cuadros de texto, etiquetas de texto, tablas, imágenes, íconos, listas de opciones, menús desplegables, entre otros. REVISAR OBJETOS.

**Interfaz de hardware y software.**

Al hablar de hardware nos referimos a los componentes físicos con los que debemos contar para el uso correcto del sistema, en este caso requerimos de una computadora con todos sus periféricos y acceso a internet.

**Funciones. MODELO CRUD (CREAR, LEER, ACTUALIZAR, ELIMINAR)**

**Requerimientos funcionales.**

**RF1. Administrar especialistas.**

Para cumplir con esta función el sistema debe permitir que se realice lo siguiente:

* **Crear especialidad**. Mediante un formulario se ingresarán los datos que corresponden con la información de una nueva especialidad y los profesionales a cargo.
* **Consultar especialistas**. Por medio de esta función se podrá ver los profesionales que ya han sido registrados en el sistema y se podrá acceder a todos o a cada uno según la necesidad del usuario.
* **Modificar especialistas**. Esta función permite al usuario del sistema actualizar la información de los profesionales registrados.
* **Eliminar especialistas**. A través de esta función, el usuario del sistema podrá eliminar los registros de profesionales si así lo requiere.

**RF2. Administrar pacientes**

Para llevar a cabo esta función el sistema debe cumplir con lo siguiente:

* **Crear paciente**. Mediante un formulario se ingresarán los datos que corresponden con la información de un nuevo paciente, datos personales, contacto y obra social.
* **Consultar paciente.** Por medio de esta función se podrá ver los pacientes, con los que trabaja la institución, que ya han sido registrados en el sistema y se podrá acceder a todos o a cada uno según la necesidad del usuario del sistema.
* **Modificar paciente.** Esta función permite al usuario del sistema actualizar la información de los pacientes registrados.
* **Eliminar paciente.** A través de esta función, el usuario del sistema podrá eliminar los registros de pacientes si así lo requiere.

**RF3. Administrar los turnos de atención médica.**

Para llevar a cabo con esta función se debe considerar que el sistema permita:

* **Generar turno.** Esta función se llevará a cabo cada vez que el paciente seleccione entre turnos disponibles.
* **Consultar turno.** Por medio de esta función el usuario del sistema tiene acceso a todos los turnos que han sido reservados.  
  **Modificar turno.** Esta función permite reprogramar un turno prestablecido.
* **Cancelar turno.** Esta función permite eliminar el registro de un turno solicitado.

**RF4. Administrar los pagos que son realizadas por los usuarios a través de la página web.**

La Figura A.2.3 es el diagrama de actividades que muestra el proceso de pago.

Requerimientos no funcionales.

RNF1. Lenguaje de programación.

RNF2. Base de datos.

RNF3. Restricciones de funcionamiento.

Para el acceso al sistema se contemplará tres tipos de usuarios.

* El administrador tendrá acceso a todas las funcionalidades.
* El empleado administrativo que solo tendrá acceso limitado al sistema según las indicaciones del administrador.
* El usuario que autogestionará el turno de manera online.

## Sprints 0

Semana 1:

-Se comparte disponibilidad horaria para coordinar reuniones de equipo.

-Creación de un canal en Discord para usar las posibilidades que brinda esta herramienta a fin de facilitar la comunicación de una forma ordenada y más eficiente que en WhatsApp.

-Cada participante trabaja de forma particular con el documento Standard IEEE 830, para luego compartir perspectivas y aunar criterios generales.

-Definición de roles en reunión programada.

### Proyecto Integrador

**Especificación de requisitos de software**



|  |  |
| --- | --- |
| **N° de sprint** | 0 |
| **Sprint Backlog** | SPRINT 0  Fecha Inicio = 17/09/2022 - Fecha de Fin = 03/10/2022    Pasos a seguir de acuerdo a las Fases del Ciclo de Vida de un Software: ANÁLISIS     * Definir requerimientos que dejarán reflejados en el IEEE830. * Plantear Historias de Usuarios y Tareas dependientes de las US para incorporarlas en el repositorio remoto GitHub. (Issues y Milestones) - Tener en cuenta la redacción adecuada para las US y nomenclatura, ej “#US01 Como usuario quiero ingresar al carrito para poder comprar”.   (Luego trasladarlas al Product Backlog del Project).   * Definir tareas dentro de las Historias de Usuario (GITHUB) ej dentro de las ISSUES #TK01 importar repositorio. |
| **Responsabilidades**  **2022** | *Scrum Master*:  **Benítez Alejandro**: Organizador general, reuniones, equipo, tareas, planificación.  **Olima Sebastián:** Organización y desarrollo de contenidos.  **Verónica Nievas:** Organización y desarrollo de contenidos y reuniones de equipo..  **Bordón Matías León:** Organización y desarrollo de contenidos y foros de discusión.  **Lamia Natalia: D**esarrollo de contenidos.  **Barrionuevo Eric:** Desarrollo de contenidos.  **Brito Nicolás:** Desarrollo de contenidos.  **Rodríguez Yamila:** Organización y desarrollo de contenidos.  **Dagna Gabriel:** Desarrollo de contenidos.  **Bordón Matías:** Desarrollo de contenidos. |
| **Calendario** | Semana 1:   * Se comparte disponibilidad horaria para coordinar reuniones de equipo. * Creación de un canal en Discord para usar las posibilidades que brinda esta herramienta a fin de facilitar la comunicación de una forma ordenada y más eficiente que en WhatsApp. * Cada participante trabaja de forma particular con el documento Standard IEEE 830, para luego compartir perspectivas y aunar criterios generales. * Se definen roles en reunión de equipo. Semana 2: * Revisión de consignas del sprint0. * Revisión de nuevo material sobre formulario IEEE |

Proyecto Integrador

**Especificación de requisitos de software**

|  |  |
| --- | --- |
|  | * decisiones en torno a quien estará dirigida la aplicación web (cliente), considerando que sea para una única institución de salud * ejemplos de requerimientos y de historias de usuario. * redacción de historia de usuario disparadora. Compromisos: pensar requerimientos e historias de usuarios a partir del material teórico y completar formulario compartido por Yami en wsp, pensar y proponer nombre de proyecto y nombre de la empresa desarrolladora, reunirnos nuevamente el viernes para elegir las mejores ideas que se propongan en el formulario , definir mapa del sitio a partir de los requerimientos elegidos y revisar que faltaría para completar el sprint0 Link de documento para completar |
| **Inconvenientes: Falto mayor participación activa en el proceso y decisiones generales, como así también mayor fluidez en la comunicación grupal** | |

Proyecto integrador E-Commerce

Sprint 0

agregar fecha de inicio

AGREGAR LOS REQUERIMIENTOS QUE FALTAN

- Revisar documentación IEEE830 si está completa y retomar desde allí para esta nueva etapa seguir documentando los avances. (realizado y modificado el 22/4/23).

- Revisar la estructura web Semántica, responsive (RWD) pendiente

Pasos a seguir de acuerdo a las Fases del Ciclo de Vida de un Software:

ANÁLISIS

Definir requerimientos para el nuevo módulo a desarrollar e-commerce (colocarlos en el Product Backlog del Project), a su vez revisar si han cumplimentado todos los requerimientos previos, realizando mejoras del mismo. (Pendiente).

Plantear Historias de Usuarios y Tareas dependientes de las US para incorporarlas en el repositorio remoto GitHub. (Issues y Milestones) - Tener en cuenta la redacción adecuada para las US y nomenclatura, ej “#US01 Como usuario quiero ingresar al carrito para poder comprar”. (En Proceso)

Definir tareas dentro de las Historias de Usuario (GITHUB) ej dentro de las ISSUES #TK01 revisar IEEE830.

**TURNERO WEB 2023 –Proyecto Final**

#### **1.1. Sprint 0**

***Fecha Inicio = 10/04/2023 - Fecha de Fin = 21/04/2023***

Se les solicitará el proyecto que han trabajado y aplicado todos los contenidos dados en el módulo de la cursada previa (HTML5, CSS3, BOOTSTRAP, JAVASCRIPT) para extraer recursos necesarios y asi continuar desde ese punto con las nuevas solicitudes.

*La “consultora ISPC”, solicita  en esta oportunidad, incorporar un****módulo de e-commerce para comercializar sus productos y/o servicios online****.*

**Antes de comenzar**

* Definir **Scrum Master inicial** y registrarse cada integrante faltante dentro del repositorio a trabajar.
* Revisar **documentación IEEE830** si está completa y retomar desde allí para esta nueva etapa seguir documentando los avances.
* Revisar la estructura web Semántica, responsive (RWD)

***Pasos a seguir de acuerdo a las Fases del Ciclo de Vida de un Software:***

**ANÁLISIS**

1. Definir **requerimientos**para el nuevo módulo a desarrollar e-commerce (colocarlos en el ***Product Backlog del Project)***, a su vez revisar si han cumplimentado todos los requerimientos previos, realizando mejoras del mismo
2. Plantear **Historias de Usuarios y Tareas** dependientes de las US para incorporarlas en el repositorio remoto GitHub. (***Issues y Milestones***) - Tener en cuenta la redacción adecuada para las US y nomenclatura, ej “#US01 Como usuario quiero ingresar al carrito para poder comprar”
3. Definir tareas dentro de las Historias de Usuario (GITHUB) ej dentro de las ISSUES  *#TK01 revisar IEEE830*.

**1.2. Sprint 1**

Fecha Inicio = 24/04/2023 -  Fecha de Fin = 07/05/2023

**Punto inicial de Implementación FRONTEND**

Convertir los archivos .html del proyecto previo, en una **SPA en Angular** con módulos (***Layouts/Shared y Pages*** son un ejemplo) con sus *components*correspondientes. (**HOME, REGISTRO, LOGIN, DASHBOARD**)

Crear **módulos y componentes** para la tienda virtual o ecommerce. (PRODUCTO o SERVICIOS, PRODUCTO INDIVIDUAL, COMPRA).

Incorporar la navegabilidad de la aplicación mediante **Routing**con Angular.

**Diseño del BACKEND**

Subir en la carpeta documentación las tablas actualizadas de la **base de datos**:

* **Diseño conceptual** al que ya hemos visto como DER: Diagrama Entidad-Relación.
* **Diseño Lógico** que consta de la Normalización y el Modelo Relacional.
* **Diseño Físico** que es el script legible de la DB en MySQL.
* Crear un **Diagrama de Clases y Casos de Uso** para facilitar el modelado en POO.

**GESTION DE PROYECTO**

Subir al repo grupal en GitHub, mediante comandos GIT desde el local, pueden crearla en una branch por desarrollador para que cada uno tenga una copia, o directamente crear una branch **FEATURE**, para actualizar los cambios que no se encuentran aún en condiciones de incorporarse a la branch **DEVELOP**.

En GitHub, mediante la sección WIKI deben reflejar los avances individuales y de equipo en relación a Scrum (Ceremonias, roles, historias de usuarios, etc). Actualizar IEEE830.

* **“FEATURE”** son las branchs en las que se trabaja la mayor parte del tiempo. Cuando no se utilizan más se borran.
* **“DEVELOP"** es la branch donde se mantiene la versión de desarrollo más estable y reciente. La que es candidata para pasar a ser la principal (**MAIN**).
* La branch **“MAIN” o “MASTER”** es la que suele usarse para mantener las versiones estables (productivas) de nuestros proyectos. Listas para salir a producción. Se supone que en cada cierre de Sprint, queda lo solicitado en esa condición, por lo que:
  + *Para el cierre de cada SPRINT****si****deberá realizarse un MERGE a la rama****MAIN/MASTER****como constancia que está completo dicho Sprint para la****Review****(en producción)*

*Estas son algunas CONVENCIONES más usadas para trabajar con branchs en proyectos de desarrollo de software*

**1.3. Sprint 2**

Fecha Inicio = 08/05/2023 -  Fecha de Fin = 21/05/2023

**Frontend**

El goal de este sprint es construir un frontend dinámico. Por ello, se requiere crear servicios que provean los datos a los componentes. Para lograrlo, los servicios deben  consumir una API Rest de prueba (Ej. <https://www.npmjs.com/package/json-server>).

Se requiere:

* 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.

**Backend**

* 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.

**GESTION DE PROYECTO**

Subir al repo grupal en GitHub, mediante comandos GIT desde el local, pueden crearla en una branch por desarrollador para que cada uno tenga una copia, o directamente crear una branch **FEATURE**, para actualizar los cambios que no se encuentran aún en condiciones de incorporarse a la branch **DEVELOP**.

En GitHub, mediante la sección WIKI deben reflejar los avances individuales y de equipo en relación a Scrum (Ceremonias, roles, historias de usuarios, etc). Actualizar IEEE830.

* **“FEATURE”** son las branchs en las que se trabaja la mayor parte del tiempo. Cuando no se utilizan más se borran.
* **“DEVELOP"** es la branch donde se mantiene la versión de desarrollo más estable y reciente. La que es candidata para pasar a ser la principal (**MAIN**).
* La branch **“MAIN” o “MASTER”** es la que suele usarse para mantener las versiones estables (productivas) de nuestros proyectos. Listas para salir a producción. Se supone que en cada cierre de Sprint, queda lo solicitado en esa condición, por lo que:
  + *Para el cierre de cada SPRINT****si****deberá realizarse un MERGE a la rama****MAIN/MASTER****como constancia que está completo dicho Sprint para la****Review****(en producción)*

*Estas son algunas CONVENCIONES más usadas para trabajar con branchs en proyectos de desarrollo de Sofware*

**1.4. Sprint 3**

Fecha Inicio = 22/05/2023 -  Fecha de Fin = 04/06/2023

**Requerimientos en Incremento al producto**

**Frontend:**

* 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.
* **Opcional:** Agregar rutas protegidas a la navegación en base al usuario autenticado/autorizado. Ej. para acceder al tablero de control o bien, para acceder al registro de productos/servicios.

**Backend**

Definición de la Api Rest y los endpoints necesarios para las funcionalidades de:

* 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).
* Tablero de control (Gestión) o Dashboard con módulos planteados en la maquetación del FrontEnd según proyecto:
  + **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.

**GESTION DE PROYECTO**

Subir al repo grupal en GitHub, mediante comandos GIT desde el local, pueden crearla en una branch por desarrollador para que cada uno tenga una copia, o directamente crear una branch **FEATURE**, para actualizar los cambios que no se encuentran aún en condiciones de incorporarse a la branch **DEVELOP**.

En GitHub, mediante la sección WIKI deben reflejar los avances individuales y de equipo en relación a Scrum (Ceremonias, roles, historias de usuarios, etc). Actualizar IEEE830.

* **“FEATURE”** son las branchs en las que se trabaja la mayor parte del tiempo. Cuando no se utilizan más se borran.
* **“DEVELOP"** es la branch donde se mantiene la versión de desarrollo más estable y reciente. La que es candidata para pasar a ser la principal (**MAIN**).
* La branch **“MAIN” o “MASTER”** es la que suele usarse para mantener las versiones estables (productivas) de nuestros proyectos. Listas para salir a producción. Se supone que en cada cierre de Sprint, queda lo solicitado en esa condición, por lo que:
  + *Para el cierre de cada SPRINT****si****deberá realizarse un MERGE a la rama****MAIN/MASTER****como constancia que está completo dicho Sprint para la****Review****(en producción)\*\*\**

*Estas son algunas CONVENCIONES más usadas para trabajar con branchs en proyectos de desarrollo de software.*

**\*\*\* NOTA:** para agilizar la vista del proyecto y simular una presentación de una "Demo" será necesario que el grupo completo realice una presentación del producto (MVP) con las funcionalidades incorporadas hasta el momento del cierre de Sprint.

**1.5. Sprint 4**

Fecha Inicio = 05/06/2023 -  Fecha de Fin = 18/06/2023

**Requerimientos  Incremento del producto:**

**Funcionalidades completas (front + back integrado)**

* *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

**GESTION DE PROYECTO**

Subir al repo grupal en GitHub, mediante comandos GIT desde el local, pueden crearla en una branch por desarrollador para que cada uno tenga una copia, o directamente crear una branch **FEATURE**, para actualizar los cambios que no se encuentran aún en condiciones de incorporarse a la branch **DEVELOP**.

En GitHub, mediante la sección WIKI deben reflejar los avances individuales y de equipo en relación a Scrum (Ceremonias, roles, historias de usuarios, etc). Actualizar IEEE830.

* **“FEATURE”** son las branchs en las que se trabaja la mayor parte del tiempo. Cuando no se utilizan más se borran.
* **“DEVELOP"** es la branch donde se mantiene la versión de desarrollo más estable y reciente. La que es candidata para pasar a ser la principal (**MAIN**).
* La branch **“MAIN” o “MASTER”** es la que suele usarse para mantener las versiones estables (productivas) de nuestros proyectos. Listas para salir a producción. Se supone que en cada cierre de Sprint, queda lo solicitado en esa condición, por lo que:
  + *Para el cierre de cada SPRINT****si****deberá realizarse un MERGE a la rama****MAIN/MASTER****como constancia que está completo dicho Sprint para la****Review****(en producción)*

*Estas son algunas CONVENCIONES más usadas para trabajar con branchs en proyectos de desarrollo de software.*

**\*\*\* NOTA:** para agilizar la vista del proyecto y simular una presentación de una "Demo" será necesario que el grupo completo realice una presentación del producto final con las funcionalidades incorporadas hasta el momento del cierre de Sprint.