**Tecnológico de Monterrey**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Sistema de Reservación e Inventario Mostla**

Documento de Especificación del Diseño del Software

**Happy Bits**

A01191954 Cristina Jimenez

A01281296 Federico Huguet

A01251219 Juan Pablo Ferreira Rodríguez

A01400214 Ana Clarissa Miranda Peña

**Control de Versiones**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción del cambio** | **Autor(es)** | **Revisor(es)** |
| 07/Sep/2019 | v1.0 | Versión Inicial | Cristina Jimenez  Federico Huguet  Juan Pablo Ferreira Rodríguez  Ana Clarissa Miranda Peña | Cristina Jimenez |
| 27/Sep/2019 | v2.0 | Correcciones y sugerencias del profesor Rafael Salazar  Se agregó el apartado de “Contenido del documento”  Se agregaron los párrafos de honestidad. | Cristina Jimenez  Juan Pablo Ferreira | Juan Pablo Ferreira Rodríguez  Cristina Jimenez |
| 20/Octubre/2019 | V3.0 | Update de los wireframes | Cristina Jimenez | Cristina Jimenez |
| 19/Noviembre/2019 | V4.0 | Update de la paletta de colores a utilizar | Cristina Jimenez | Cristina Jimenez |

# 

**Tabla de Contenidos**

[**1 Introducción**](#_cbs5mmoc84os) **4**

[1.1 Propósito](#_l8s6tehbkfzm) 4

[1.2 Alcance](#_g2dtn9k5d0nw) 4

[1.3 Contexto](#_y3cq9ml4wv) 4

[1.4 Resumen](#_77i4e37ez5er) 5

[**2 Referencias**](#_mg0v6w2sdrpf) **5**

[**3 Glosario**](#_gopk39b3mwb7) **5**

[**4 Contenido detallado**](#_i34tw4gnqjaj) **6**

[4.1 Identificación de involucrados](#_x9r3houfo8ob) 6

[4.2 Preocupaciones de diseño](#_793wwsn4cdif) 7

[4.3 Perspectivas](#_jp6vy6t74oae) 7

[4.3.1 Perspectiva [PD001] - Developer](#_n2v13pkff9br) 7

[4.3.1.1 Modelo de datos](#_r7wk6m94il43) 7

[4.3.1.2 Diagramas de clase](#_r7wk6m94il43) 9

[4.3.1.3 Arquitectura](#_e9dg6zgtik8b) 9

[4.3.1.4 Diagramas de contexto](#_r7wk6m94il43) 10

[4.3.2 Perspectiva [PD002] - Deployment](#_vuh0ys6fwmhi) 10

[4.3.2.1 Instalar dependencias en máquina](#_mbpx1wdb5x28) 10

[4.3.2.2 Clonar repositorio](#_sv2qkut7dv8) 10

[4.3.2.3 Instalar dependencias en proyecto](#_14rntcoetr19) 10

[4.3.2.4 Crear base de datos](#_vbf0l2e4fa72) 11

[4.3.2.5 Configurar variables de ambiente](#_ppj0ejk3i7u7) 11

[4.3.2.6 Preparar base de datos](#_ks9y1x1mi7io) 12

[4.3.2.7 Ejecutar proceso del servidor](#_9pgt659lxjt5) 12

[4.3.3 Perspectiva [PD003] - Usuario general](#_q9y67s5yv9fv) 12

[4.3.3.1 Guía de uso](#_gmhd9zq09hpu) 12

[4.3.3.2 Wireframes](#_gmhd9zq09hpu) 13

[4.3.3.3 Casos de uso](#_l33lh0xy1luh) 19

[4.3.3.4 Guía de estilo](#_v90kaaqa4vjl) 21

[**5. Anexos**](#_j91bf03tpe5l) **22**

[5.1 Párrafo de honestidad](#_ntfvv08fl336) 22

# 

# 1 Introducción

## 1.1 Propósito

Mostla es una entidad perteneciente al Tecnológico de Monterrey que se enfoca en hacer disponibles dispositivos de alta tecnología a alumnos y profesores de la institución, con el fin de promover el conocimiento de las nuevas tendencias tecnológicas y fomentar su uso en la educación. La organización actualmente cuenta con dificultad en la organización de préstamos a colaboradores del Tecnológico debido a que no cuentan con un sistema central para mantener el inventario y el registro de préstamos y devoluciones del equipo, por lo que buscan que se desarrolle un sistema que les facilite el proceso, tanto en el Campus Monterrey como potencialmente en módulos de diferentes campus a medida que se vaya extendiendo su presencia a ellos.

El propósito de este documento es detallar la especificación del diseño del sistema que se desarrollará por el equipo Happy Bits para los laboratorios Mostla del Tecnológico de Monterrey durante el semestre Ago-Dic 2019. El sistema a desarrollar tiene como propósito facilitar el control de inventario de equipos tecnológicos propiedad de Mostla, así como permitir un control centralizado de los préstamos de éstos.

## 1.2 Alcance

El alcance del proyecto abarca desde el diseño de una plataforma que resuelva las necesidades funcionales especificadas por Mostla, hasta el desarrollo de dicha plataforma, su documentación, y su entrega al equipo de la organización como una plataforma en funcionamiento y en cumplimiento con los requerimientos del cliente. El alcance del sistema abarca desde el registro de equipo tecnológico en la base de datos, hasta permitir la reservación y seguimiento de préstamos de éste, con la opción para los administradores de actualizar información de éste, así como consultar reportes de disponibilidad y préstamos, entre otras funcionalidades administrativas.

## 1.3 Contexto

El sistema desarrollado en este proyecto funcionará en un ambiente académico donde tanto maestros y alumnos pueden necesitar ciertos materiales prestados para sus proyectos educativos. En cuanto a los usuarios del sistema, estos serán colaboradores del Tec de Monterrey, donde los anteriormente mencionados son quienes accederán al sistema en busca del material que requiera, y donde empleados del campus, más específicamente de Mostla, accederán al sistema para administrar aquellos préstamos que los anteriores llegasen a solicitar. Es decir, las acciones que se realicen en el sistema se verán, en cierta manera, sujetas a la aceptación del personal de Mostla como organización con fines educativos. Además, el sistema se desarrolla con la visión de ser utilizado en los distintos campus del Tec según Mostla se expanda a ellos, y de permitir el préstamo de propiedades digitales si esto fuera una posibilidad para la organización en el futuro.

## 1.4 Resumen

El sistema será utilizado en los campus del Tecnológico de Monterrey por alumnos para reservar el equipo que necesiten por el periodo requerido siempre y cuando esté disponible. Asimismo, los administradores serán capaces de agregar y remover equipos del inventario.

# 2 Referencias

IEEE 1016-2009: Standard for Information Technology - Systems Design - Software Design Description

# 3 Glosario

* **Mostla**: La palabra en lenguaje nàhuatl significa “mañana”. El un espacio para profesores y estudiantes del Tecnológico de Monterrey en el que se puede conocer y experimentar con las tecnologías más innovadoras.
* **Equipo**: se refiere a cualquier dispositivo o aparato que sea administrado por Mostla, y que sea considerado para su inventario, independientemente si se encuentra disponible para reservación.
* **Administrador general:** El usuario que estará a cargo del sistema, de mantener actualizada la base de datos y solucionar cualquier problema.
* **Administrador de campus:** Usuario que estará a cargo de la disponibilidad del equipo y gestionar las reservas intercampus.
* **Alumno:** Usuario estudiante del Tecnológico de Monterrey.
* **Campus:** Sedes del Tecnológico de Monterrey.

# 

# 4 Contenido detallado

## 4.1 Identificación de involucrados

|  |  |  |
| --- | --- | --- |
| Nombre | Rol | Intereses/Responsabilidades |
| José Antonio Hernández García | Cliente | Tener un MVP de un sistema de reservación de equipo de tecnología del Tec de Monterrey. Proporcionar al equipo de desarrollo con la base de datos de inventario y ayudar a definir el Product Backlog |
| Cristina Jiménez | Desarrollador Front-end,  Product Owner | Representar los intereses del cliente y comunicarles al equipo de desarrollo.  Escribir las historias de usuario.  Administrador de Github y la documentación del proyecto.  Diseñar y desarrollar las vistas de la aplicación web. |
| Federico Huguet | Desarrollador Backend, Administrador del proyecto | Desarrollar los servicios(end-points) que serán utilizados en el sistema. Dividir correctamente las tareas para que se cumplan las metas a tiempo. |
| Clarissa Miranda | Desarrollador Front-end, Scrum Master | Diseñar y desarrollar las vistas de la aplicación web.  Gestionar los eventos de Scrum. |
| Juan Pablo Ferreira | Desarrollador Back-end, Administración de la configuración | Mantener consistencia en los archivos del proyecto y su configuración. Desarrollar lógica interna del sistema mediante la cual se actualizará la base de datos según sea necesario. |

## 4.2 Preocupaciones de diseño

|  |  |
| --- | --- |
| **Preocupación** | **Descripción** |
| Definición y acceso a servicios | Los usuarios deben estar autenticados y autorizados para ciertos servicios. |
| Utilización de recursos | La optimización de los recursos facilitará y mejorará la gestión del proyecto para los administradores |
| Usabilidad del sistema | Debe de ser un sistema fácil de comprender para el usuario. No debe de tomarle mucho tiempo entender cómo realizar una reservación. |
| Lógica procedural | La lógica y los algoritmos deben ser entendibles y escalables para poder dar el servicio a otros campus. |

## 4.3 Perspectivas

### **4.3.1 Perspectiva [PD001] - Developer**

#### 4.3.1.1 Modelo de datos

|  |
| --- |
| **Grupo**  idGrupo  idUbicación  Metadata (descripción)  Categorí**a**  Cantidad  **PK**(idGrupo)  Metadata  Titulo  Marca  Numero de serie  Descripcion  Especificacion  Observaciones campos      **Equipo**  idGrupo  idEquipo  Disponibilidad  **PK**(idGrupo, idEquipo, idStatus)  **FK**(idGrupo) from Grupo(idGrupo)  **FK**(idStatus) from Status(idStatus)      **Permisos**  idStatus  Descripcion  **PK**(idStatus)      **Reservacion**  idGrupo  idEquipo  PK (idGrupo, idEquipo)  **FK**(idGrupo, idEquipo) from Equipo      **Ubicación**  idUbicacion  Estado  Ciudad  Campus  Propietario  **PK**(idUbicacion)        **Recordatorios**  idGrupo  idEquipo  idRecordatorio  Actividad  Fecha  **PK**(idGrupo, idEquipo, idRecordatorio)  **FK**(idGrupo, idEquipo) from Equipo(idGrupo, idEquipo)      **Reservacion**  idGrupo  idEquipo  idUsuario  fechaInicio  fechaTermino  PK (idGrupo, idEquipo, fechaInicio)  **FK**(idGrupo, idEquipo) from Equipo      **Comentarios**  idUsuario  idComentario  idGrupo  Fecha  Comentario  **PK**(idUsuario, idComentario)  **FK**(idUsuario) from Usuario (idUsuario)  **FK**(idGrupo) from Grupo (idGrupo)      **Usuario**  IdUsuario  Nombre  Apellido  **PK**(idUsuario)      **Permisos**  idUsuario  idPermiso  **PK**(idUsuario, idPermiso)  **FK**(idUsuario) from Usuario (idUsuario)  **FK**(idPermiso) from Permiso (idPermiso)      **Permiso**  idPermiso  Descripción  **PK**(idPermiso) |

#### 4.3.1.2 Diagramas de clase

#### 4.3.1.3 Arquitectura

El estilo de la arquitectura del software será MVC, utilizando el framework Laravel para desarrollar la aplicación web y la base de datos se conectará con PostgreSQL. En MVC, el Modelo representa los datos que maneja el sistema, la Vista es la interfaz del usuario y el Controlador gestiona el flujo de información.

<https://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-vista-controlador-mvc.html>

#### 4.3.1.4 Diagramas de contexto

### **4.3.2 Perspectiva [PD002] - Deployment**

#### 4.3.2.1 Instalar dependencias en máquina

Antes de poder deployar el sistema en una máquina, es necesario asegurarnos que ésta cuente con todo el software requerido por Laravel para poderse ejecutar. Según la documentación de Laravel (<https://laravel.com/docs/5.8/installation>), es necesario contar con los siguientes paquetes:

* PHP >= 7.1.3
* BCMath PHP Extension
* Ctype PHP Extension
* JSON PHP Extension
* Mbstring PHP Extension
* OpenSSL PHP Extension
* PDO PHP Extension
* Tokenizer PHP Extension
* XML PHP Extension

Es importante también instalar Composer en la máquina, ya que éste se utilizará para manejar las dependencias del proyecto. Las instrucciones para instalar Composer pueden ser encontradas aquí <https://getcomposer.org/doc/00-intro.md>. Nótese que es recomendado realizar el deployment de la aplicación en un sistema Unix-like para facilitar la ejecución de sus componentes.

Si no se cuenta con algún administrador de bases de datos en el sistema, MySQL en este caso, también es necesario instalarlo para poder deployar la aplicación.

#### 4.3.2.2 Clonar repositorio

El siguiente paso es clonar o copiar los archivos del repositorio a nuestra máquina. Éste se puede encontrar en la liga <https://github.com/ProyectoIntegrador2018/MostlaInventario/>, y para realizar este paso solo es necesario descargar y extraer los archivos a nuestra máquina, o clonar el repositorio a la misma usando Git, a través del comando git clone, como se indica en el sitio del repositorio.

#### 4.3.2.3 Instalar dependencias en proyecto

Una vez que se cuente con los archivos del proyecto, y se tenga Composer instalado en la máquina (Importante), se puede utilizar el comando composer install desde una terminal en el directorio raíz del proyecto. Esto descargará las versiones requeridas de diferentes paquetes utilizados en el proyecto a su máquina y agregará estos al proceso de autoload de los paquetes de éste.

#### 4.3.2.4 Crear base de datos

Para poder ejecutar el sistema es primero necesario contar con una base de datos que éste pueda utilizar para guardar su información. Para este proyecto, se está utilizando MySQL, por lo que es recomendado que se utilice el mismo en su máquina cuando se haga el deploy. El primer paso para preparar la base de datos para el deploy, asumiendo que ya se cuenta con el software instalado y se cuenta con un usuario con permisos suficientes (en caso de que no sea así, ver <https://kyup.com/tutorials/create-new-user-grant-permissions-mysql/> ), es crear la base de datos. Esto se puede hacer desde la consola de MySQL (a la cual se puede acceder usando el comando mysql desde la terminal). Desde dentro de la terminal de MySQL, utilizar el comando create database mostla;. Ésto creará una base de datos llamada “mostla” en su sistema. El nombre de ésta no es relevante para la ejecución, pero se utilizará en este ejemplo para demostrar la configuración necesaria.

#### 4.3.2.5 Configurar variables de ambiente

Dentro del directorio raíz del proyecto, se encuentra un archivo llamado “.env.example”. Éste es una plantilla (con valores predeterminados) de un archivo que define las variables del ambiente. El primer paso para configurar éstas es crear una copia de este archivo y titularlo “.env”. Dentro de éste, ahora, se deben actualizar las siguientes variables:

Variables requeridas:

|  |  |
| --- | --- |
| DB\_DATABASE | El valor de esta variable es el nombre de la base de datos creada en el paso anterior. En este caso, es “mostla”, según el ejemplo de la sección anterior, pero puede variar si se utilizó otro nombre en el paso anterior. |
| DB\_USERNAME | El nombre del usuario que el sistema debe utilizar para acceder a la base de datos. |
| DB\_PASSWORD | El password del usuario que el sistema debe utilizar para acceder a la base de datos. |
| GOOGLE\_ID | Estas variables no vienen por default en el archivo, hay que agregarlas. El valor de estas variables son aquellos dados por la API de Google cuando se registra una cuenta para utilizar el servicio de login a través de Google. |
| GOOGLE\_SECRET |
| GOOGLE\_URL |

Variables relevantes:

|  |  |
| --- | --- |
| APP\_NAME | El nombre que se desea que el sistema tenga en la máquina. El efecto de esta variable es únicamente visual, ya que es el nombre que se desplegará en algunas partes del sistema. |
| APP\_ENV | El valor de esta variable puede ser “local”, “staging” o “production” según cual sea el modo en el que se hace el deploy. |
| APP\_DEBUG | Si el valor de esta variable es “true”, se mostrará al usuario información del desarrollo cuando el sistema se encuentre con algún problema. Si la aplicación se encuentra en producción, es preferible mantener esto en “false” para evitar mostrar información sensible a usuarios externos. |
| APP\_URL | Éste es el URL a través del cual uno accede a esta instancia del sistema (puede ser localhost o algún dominio o dirección IP). Esto sirve de que cuando el sistema tenga que generar links para entregar al usuario, se utilice la dirección adecuada. |

Un último paso para configurar estas variables es ejecutar el comando php artisan key:generate desde la terminal en el directorio raíz del proyecto. Esto generará la llave secreta con la que se encriptan ciertos datos utilizados en el sistema, y la agregará en archivo “.env” automáticamente.

#### 4.3.2.6 Preparar base de datos

Antes de poder ejecutar la aplicación, es necesario crear las tablas requeridas en la base de datos. Para hacer esto, primero hay que ejecutar el comando php artisan migrate:install desde la terminal en el directorio raíz del proyecto. Esto preparará las tablas necesarias que requiere Laravel para ejecutar las migraciones definidas que crearán todas las tablas requeridas del proyecto. Una vez hecho lo anterior, se pueden correr las migraciones con el comando php artisan migrate --seed. Esto creará las tablas del proyecto y además generará algunos registros iniciales que son requeridos para la ejecución del sistema.

#### 4.3.2.7 Ejecutar proceso del servidor

El último paso para deployar el sistema de manera local es ejecutar el comando php artisan serve desde la terminal en el directorio raíz del proyecto. Esto creará una instancia local del sistema, a la cual se podrá acceder mediante la dirección http://localhost:8000, o aquella que regrese el comando al ejecutarlo en caso de que ésta sea diferente. Si se quiere especificar el puerto en el que se ejecutará el proceso, se puede hacer agregando la opción --port=#### al final del comando anterior, donde #### es el número de puerto que se desea utilizar. En caso de querer deployar la aplicación de manera pública, es preferible utilizar Nginx o Apache para ejecutar el proceso del servidor.

### **4.3.3 Perspectiva [PD003] - Usuario general**

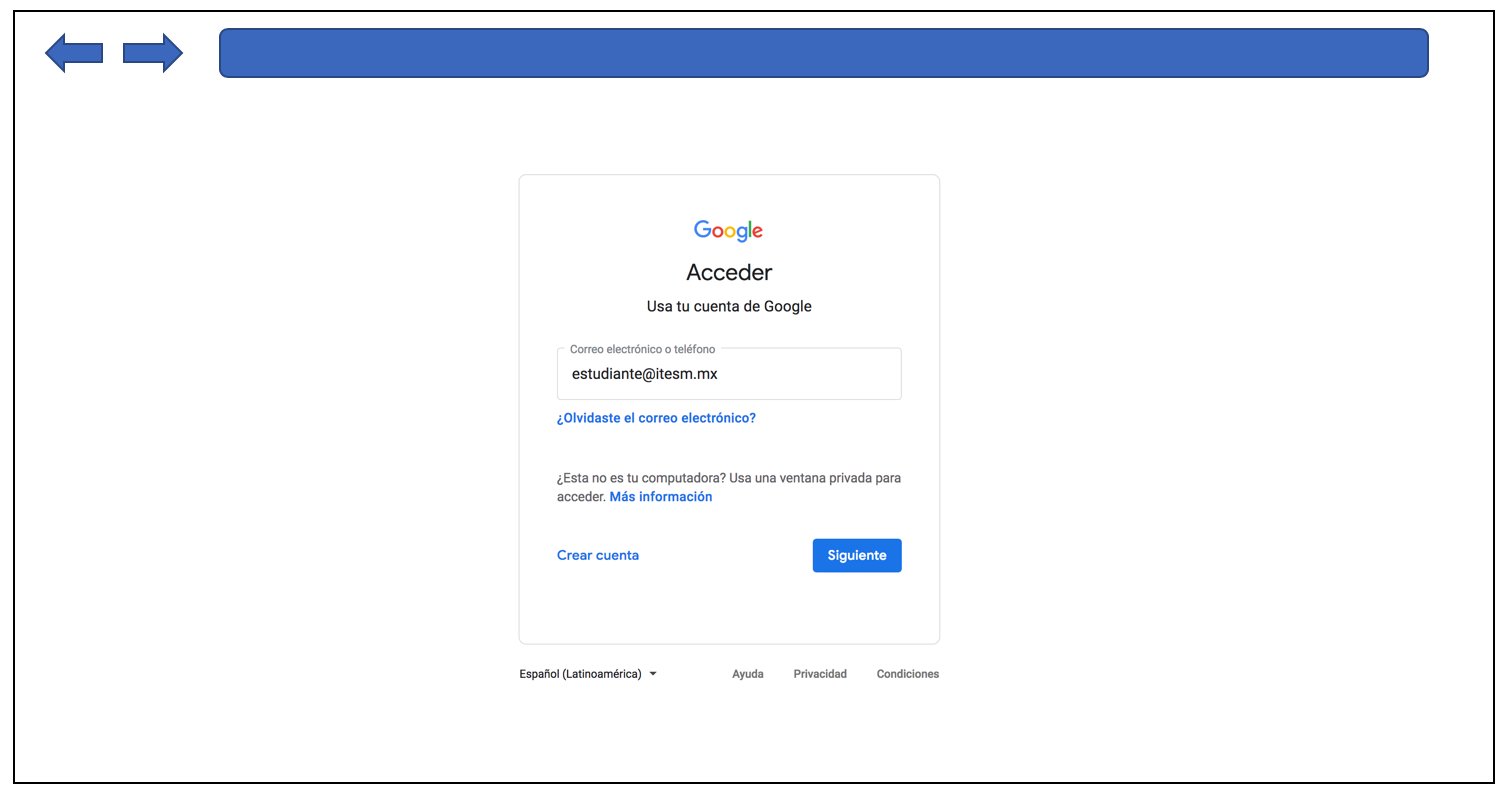
#### 4.3.3.1 Guía de uso

#### 4.3.3.2 Wireframes

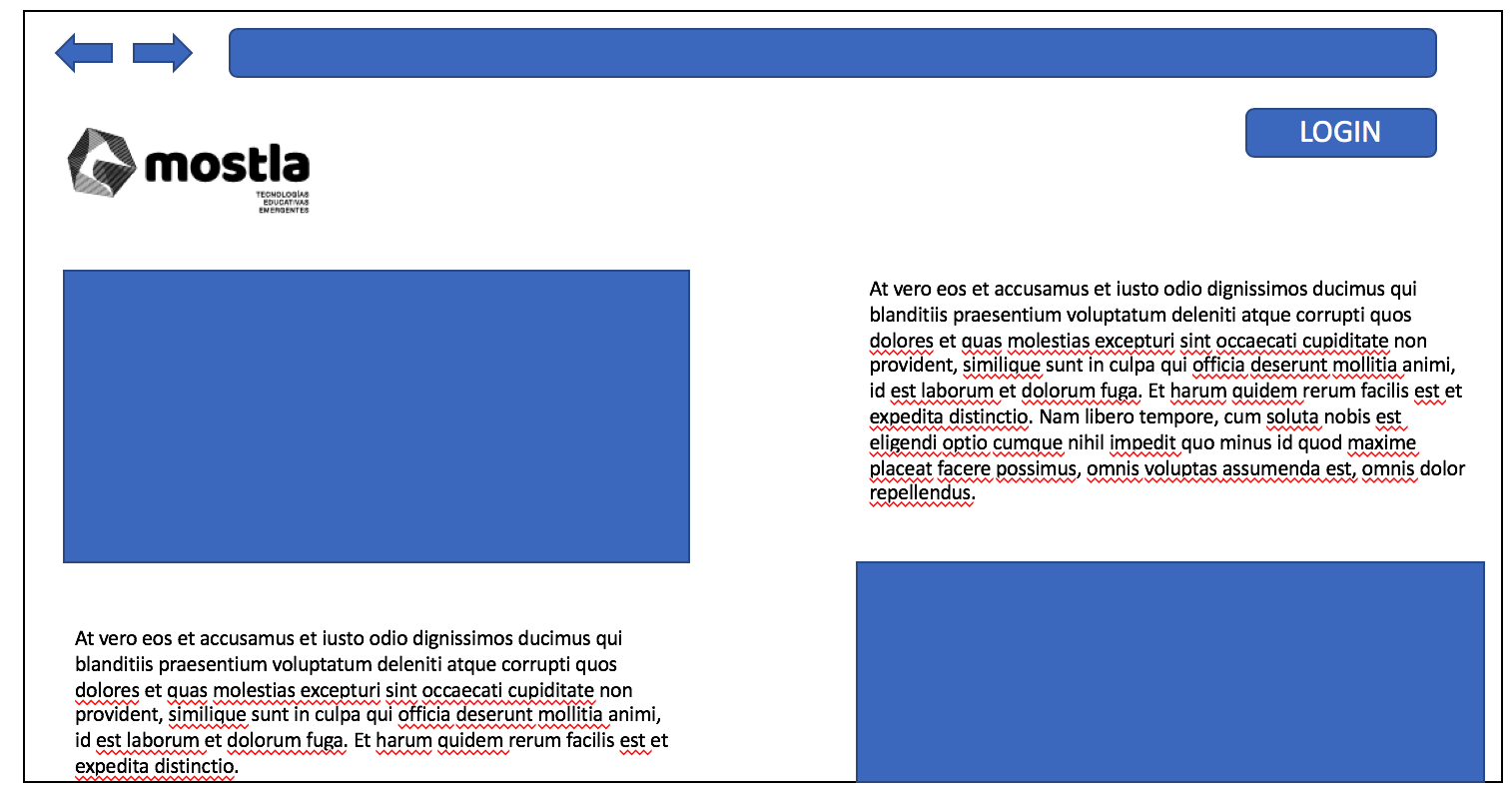
HU-1 Usuario General- Mostrar equipos



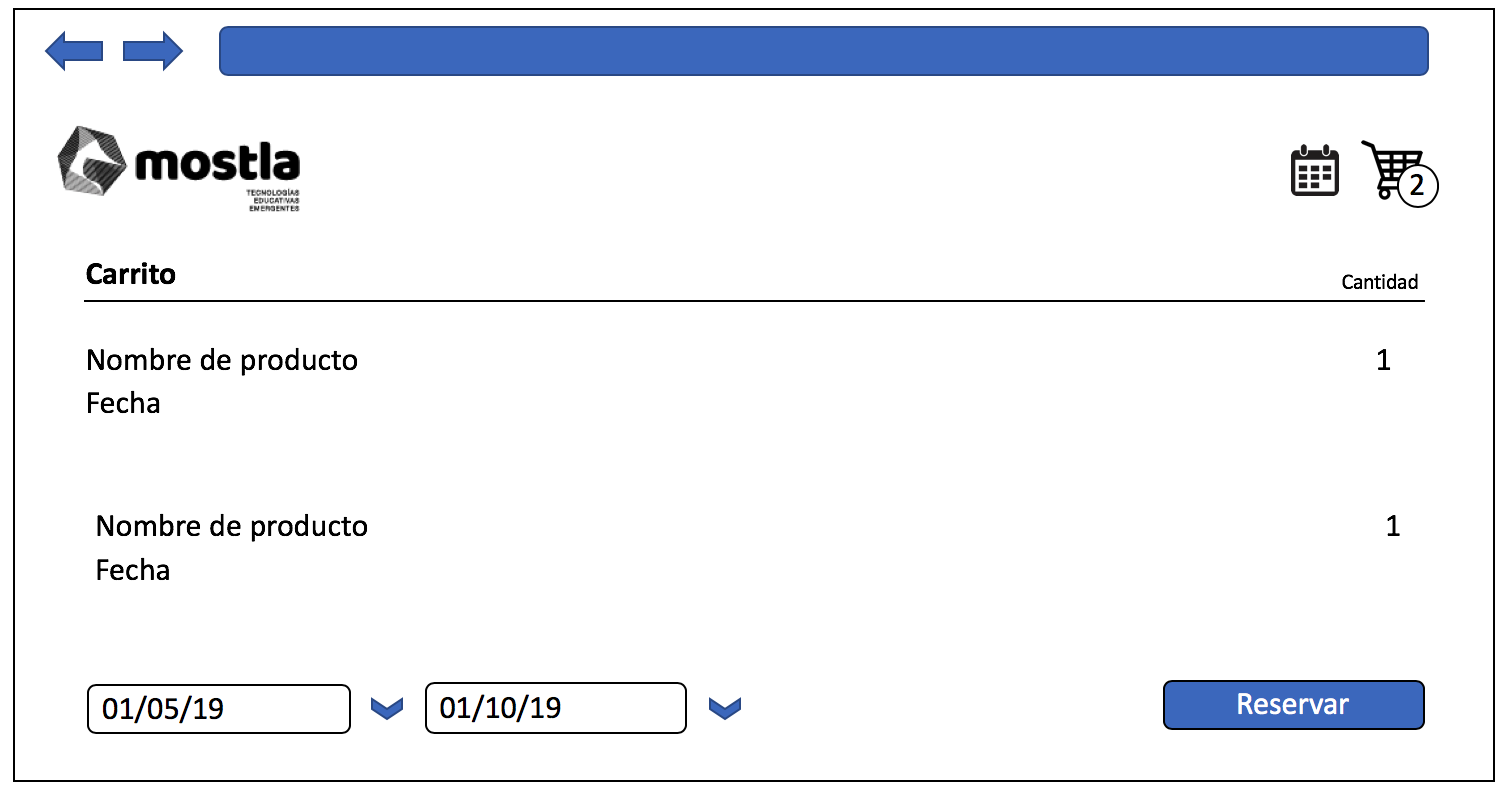
HU-4 Usuario General- Ingresar a la pagina a traves de Google sign in



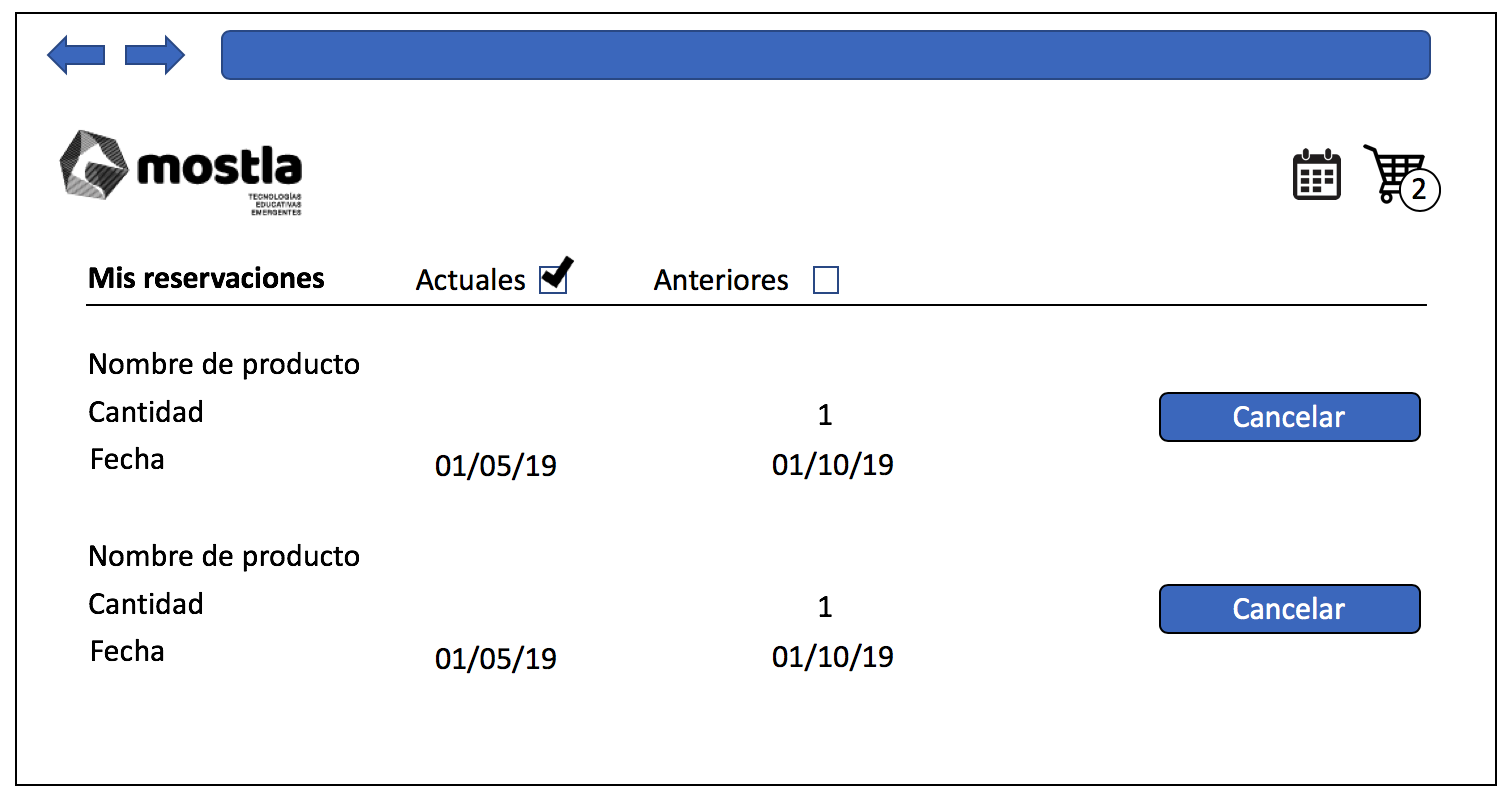
HU-5 Usuario General- Ingresar a traves de la landing page



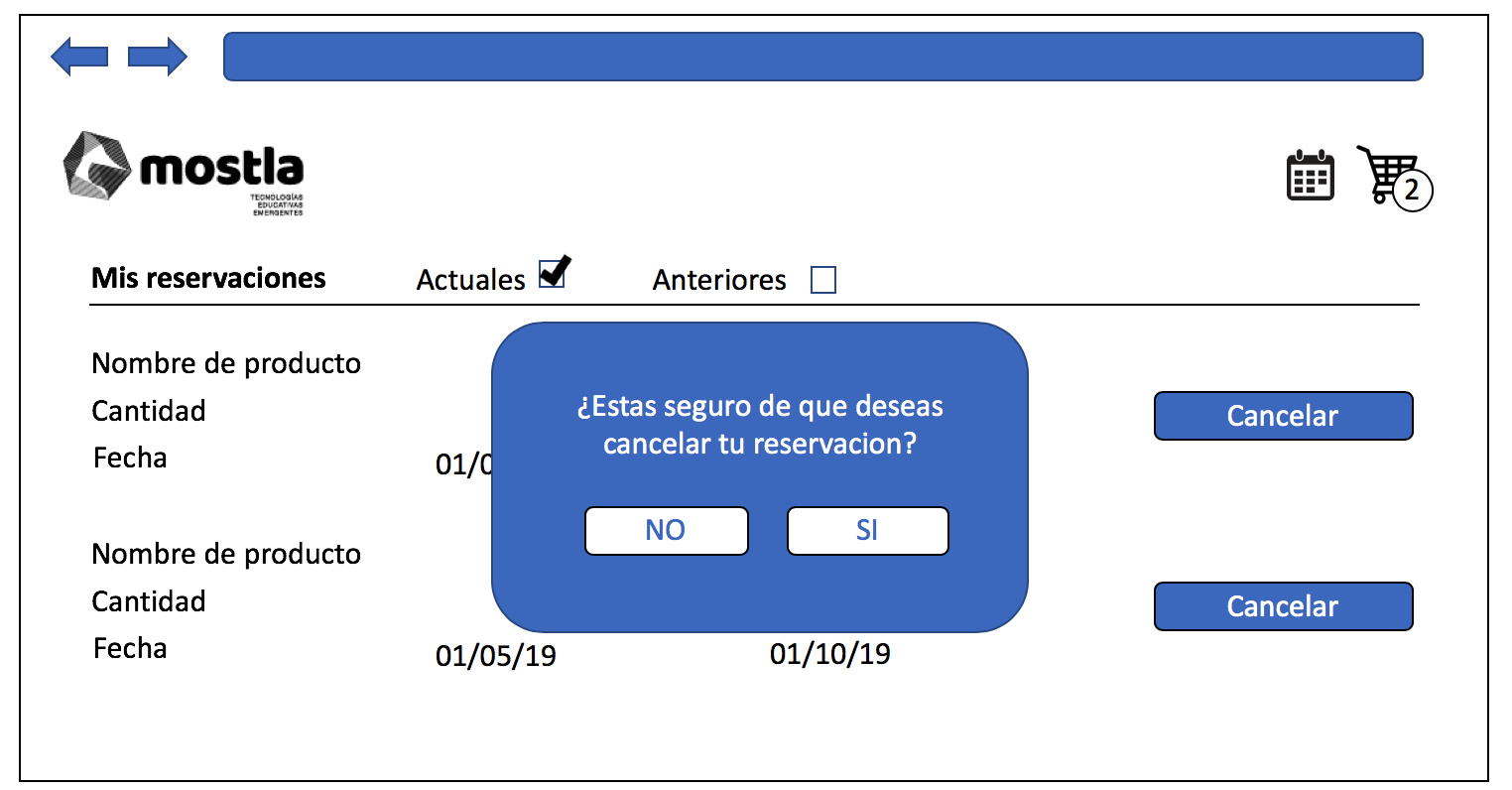
HU-6 Usuario General- Hacer una reservacion



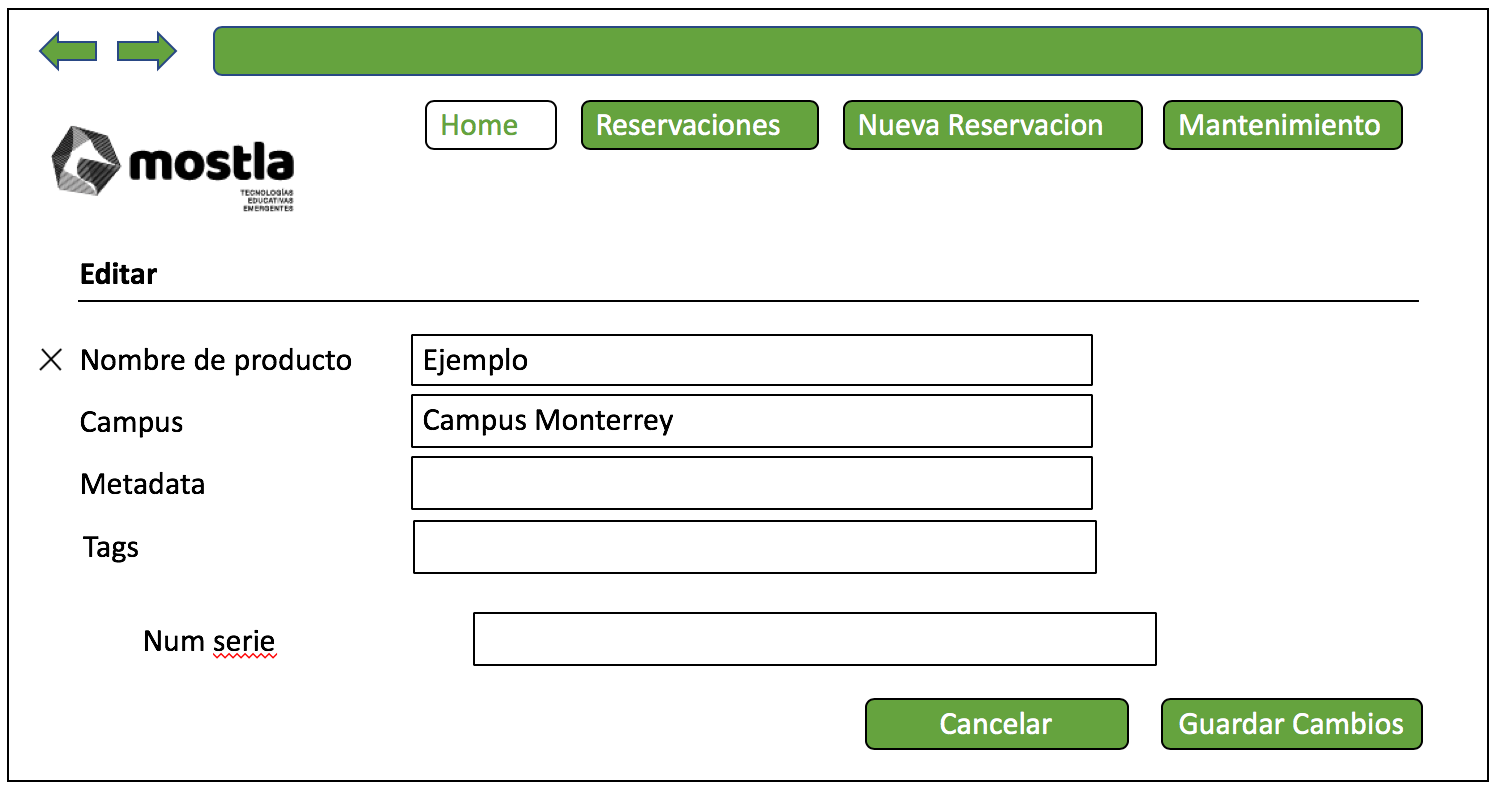
HU-7 Usuario General- Ver historial de reservaciones



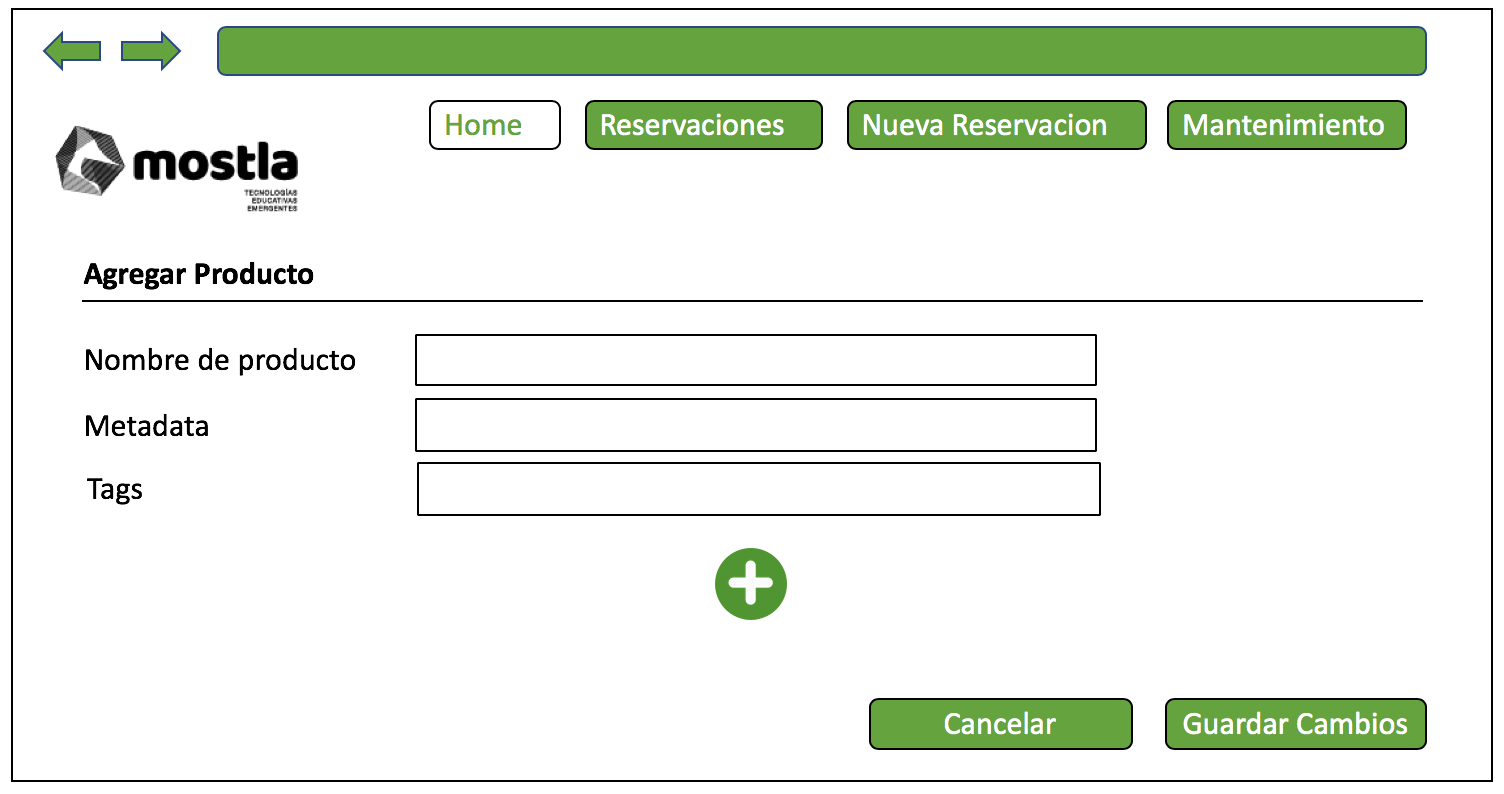
HU-8 Usuario General- Cancelar una reservacion

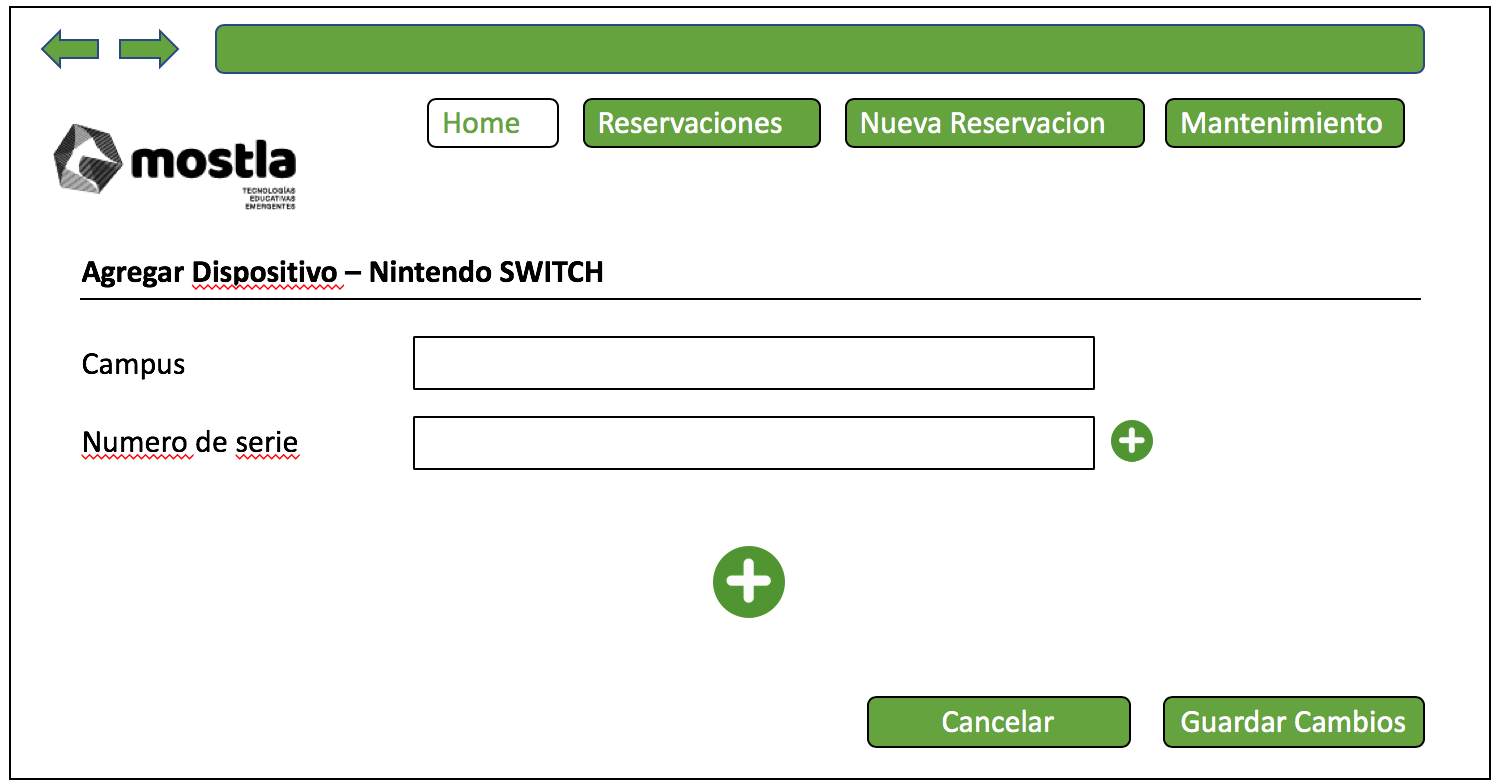


HU-9 Editar producto

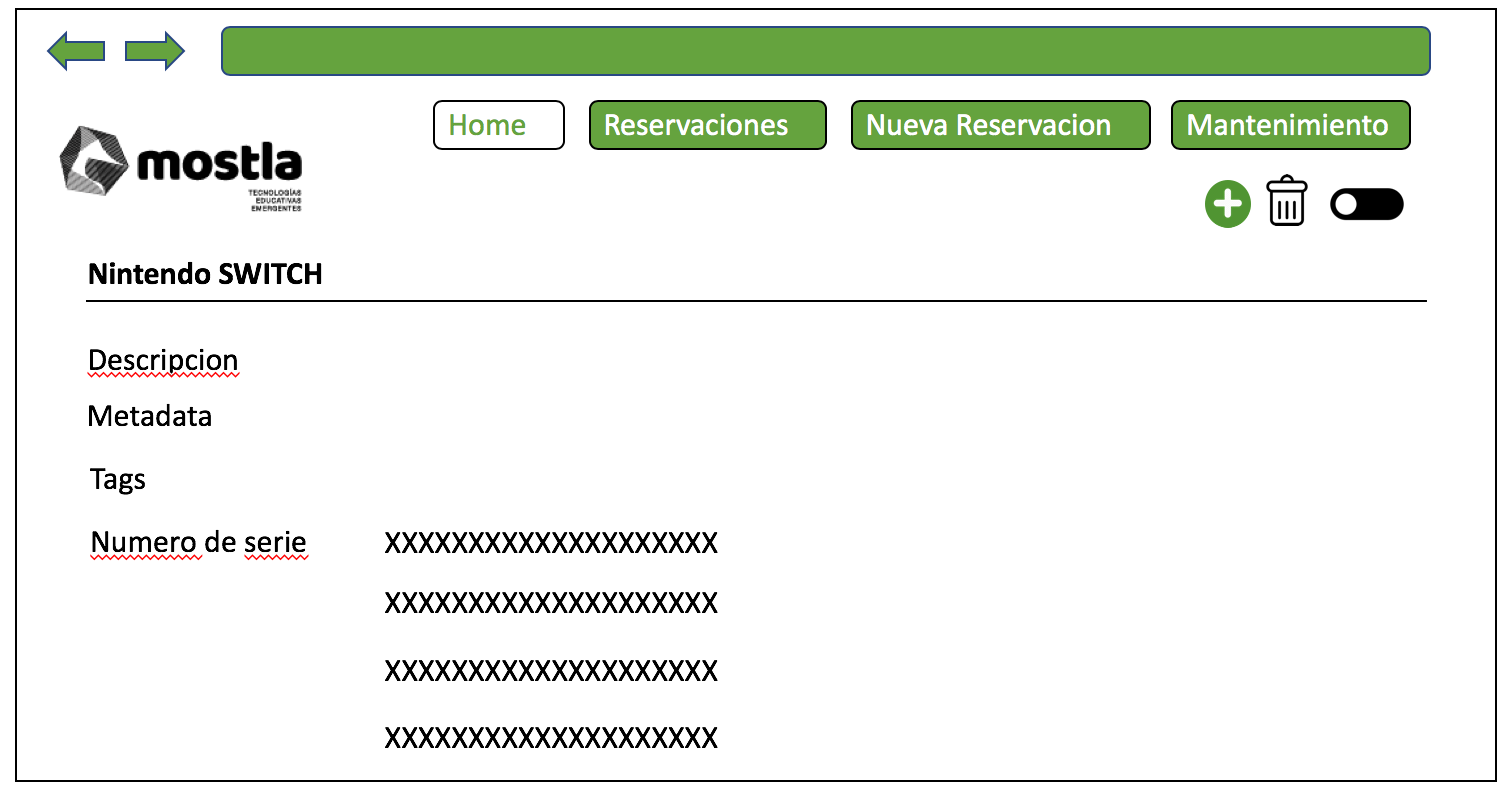


HU-10 Agregar producto

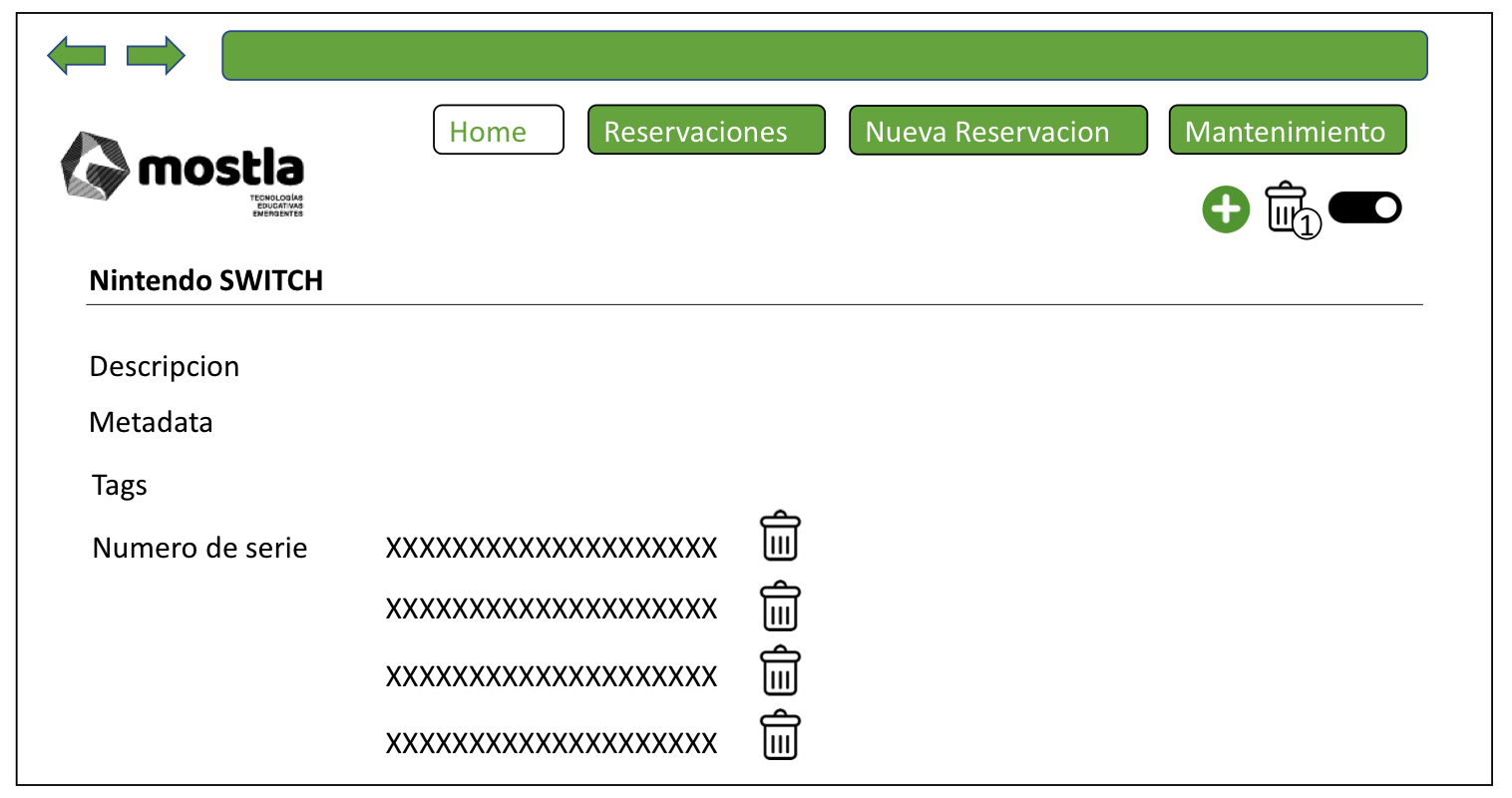


HU-11 Agregar un dispositivo de un tipo de producto

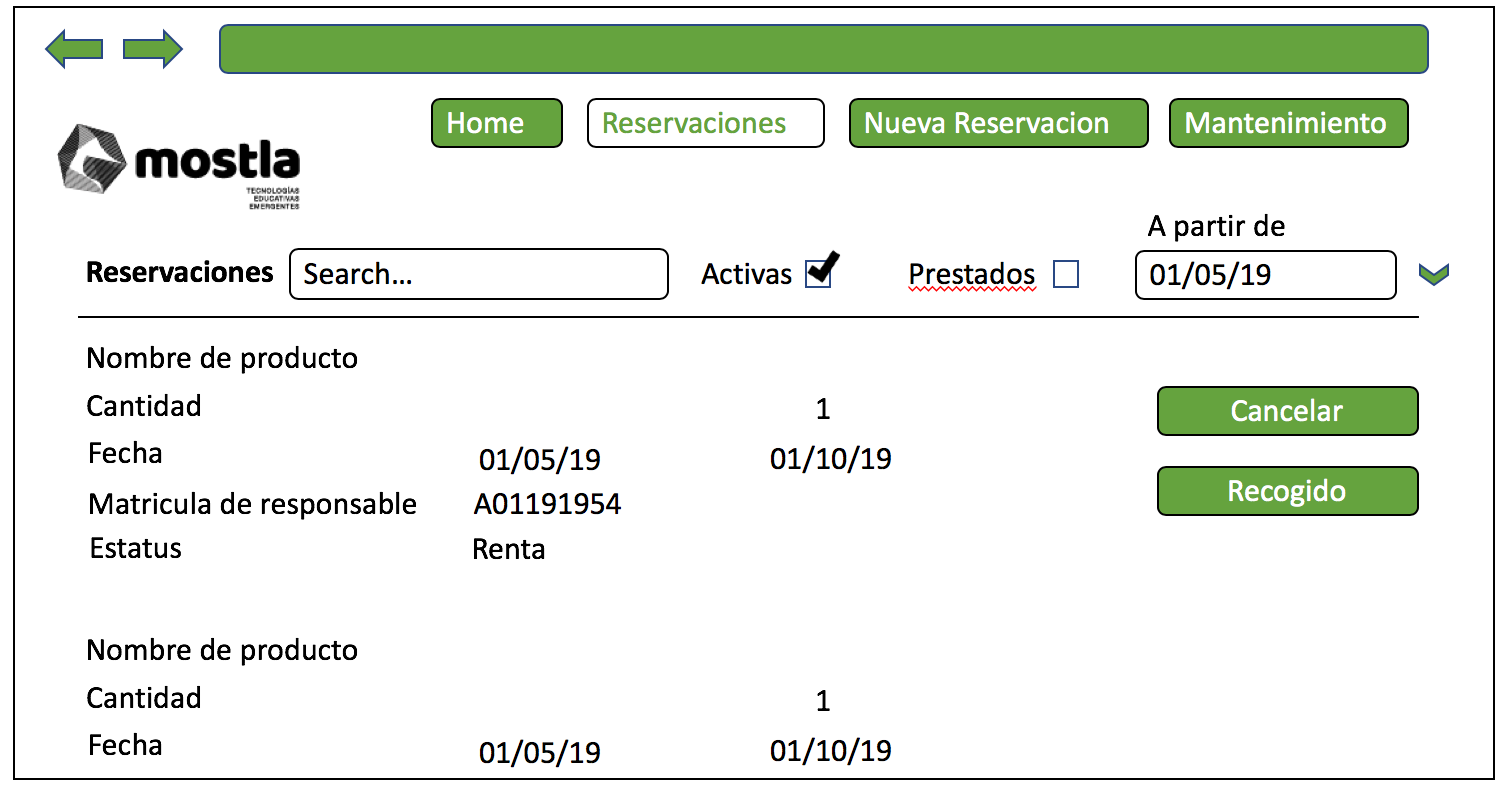
HU-12 Ver producto



HU-13 Eliminar producto

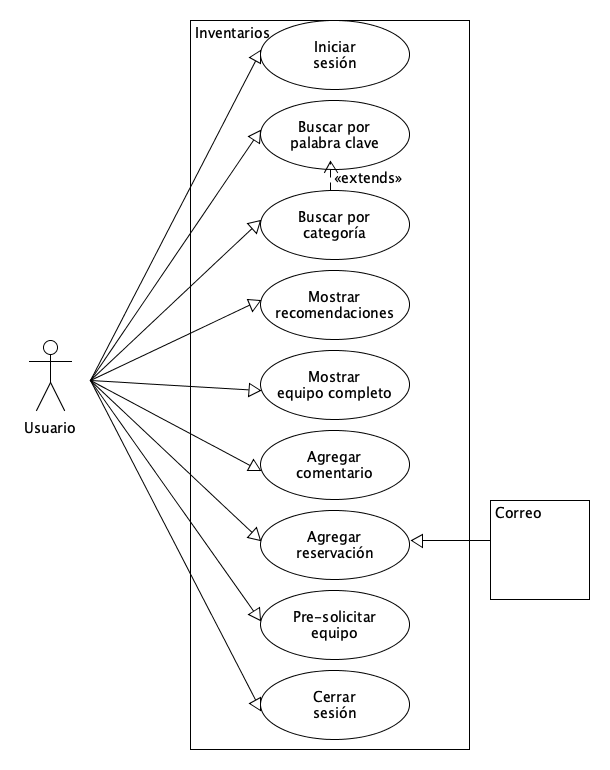


HU-15 Ver reservaciones hechas

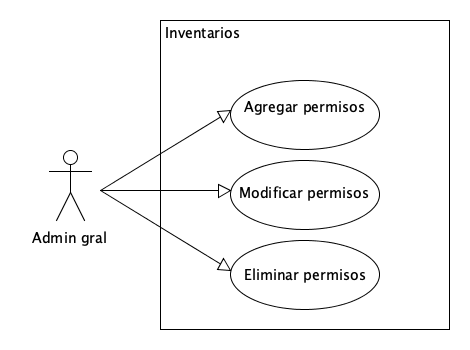


#### 4.3.3.3 Casos de uso

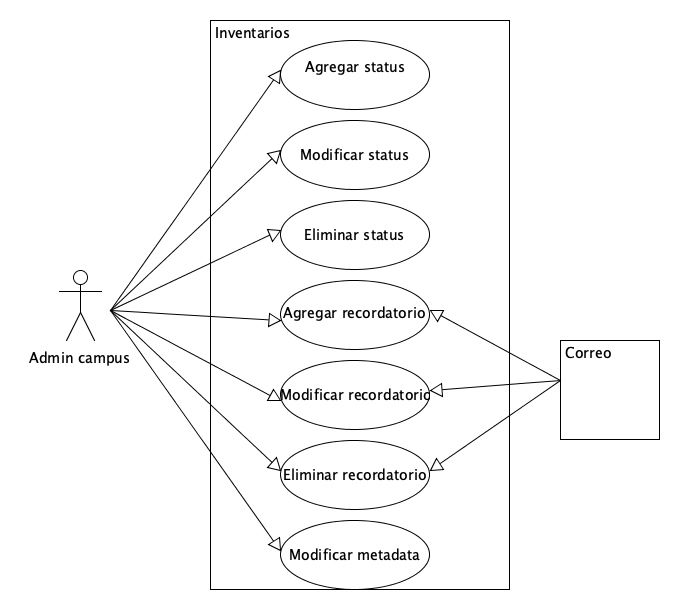
[CU001] Usuario General



Los usuarios generales que tengan el acceso básico a la plataforma, podrán tener la capacidad de realizar los casos de uso que se describen el diagrama anterior.

[CU002] Usuario Administrador

Los usuarios administradores tendrán los accesos básico a la plataforma además de tener la capacidad de realizar los casos de uso que se describen el diagrama anterior.

[CU003] Usuario Administrador del Campus 

El usuario administrador del campus es el usuario con mayor acceso en el sistema. Este es capaz de realizar los casos de uso de un usuario regular, de un administrador general y además llevar a cabo los casos de uso que se muestran en el diagrama anterior.

#### 4.3.3.4 Guía de estilo

los colores que proponemos utilizar para el sistema a desarrollar son los siguientes:

<https://rizzo.lonelyplanet.com/styleguide/design-elements/colours>

# **5. Anexos**

## 5.1 Párrafo de honestidad

"Por medio de este párrafo yo **Cristina Jiménez** certifico que soy el autor intelectual del material que arriba entregó y que no realicé una copia de ninguna otra persona. En aquellos casos en los que tuve que incluir material de otra persona, reporté su apropiada referencia."

"Por medio de este párrafo yo **Federico Huguet** certifico que soy el autor intelectual del material que arriba entregó y que no realicé una copia de ninguna otra persona. En aquellos casos en los que tuve que incluir material de otra persona, reporté su apropiada referencia."

"Por medio de este párrafo yo **Clarissa Miranda** certifico que soy el autor intelectual del material que arriba entregó y que no realicé una copia de ninguna otra persona. En aquellos casos en los que tuve que incluir material de otra persona, reporté su apropiada referencia."

"Por medio de este párrafo yo **Juan Pablo Ferreira** certifico que soy el autor intelectual del material que arriba entregó y que no realicé una copia de ninguna otra persona. En aquellos casos en los que tuve que incluir material de otra persona, reporté su apropiada referencia."