

| **Nombres:**  Ricardo Arriaga Ortegón  Dana Elena Zertuche Castro | **Correos:**  [AL03110119@tecmilenio.mx](mailto:AL03110119@tecmilenio.mx)  [AL02991408@tecmilenio.mx](mailto:AL02991408@tecmilenio.mx) |
| --- | --- |
| **Materia:** Desarrollo Full Stack | **Maestro:** Juan Manuel Cepeda Barragán |
| **Tarea:** Actividad 4 | **Fecha:** viernes 14 de febrero del 2025 |

**Etapa de Planificación del Proyecto SimpleClothing**

**Definición del Proyecto**

El proyecto **SimpleClothing** consiste en el desarrollo de un sitio web full-stack para una marca mexicana de ropa urbana unisex. El objetivo principal es crear una plataforma digital que permita a la marca mostrar su catálogo de productos, gestionar inventarios y ofrecer una experiencia de usuario atractiva y funcional. El sitio está dirigido a un público joven y moderno interesado en moda urbana y unisex.

**Funcionalidades Principales:**

1. **Página de Inicio:** Presentación de la marca y destacados del catálogo.
2. **Catálogo de Productos:** Visualización de prendas organizadas por categorías (hombres, mujeres, unisex).
3. **Gestión de Inventario:** Panel administrativo para agregar, editar o eliminar productos.
4. **Diseño Responsive:** Adaptabilidad del sitio a dispositivos móviles y desktop.

**Stack Tecnológico**

* **Front-end:** HTML, CSS y JavaScript para la estructura, diseño e interactividad del sitio.
* **Back-end:** Node.js con Express.js para la lógica del servidor.
* **Base de Datos:** MongoDB para almacenar información flexible como catálogos de productos.
* **Despliegue:** Firebase para hosting, Firestore para la base de datos en tiempo real, y Firebase Functions para la lógica del backend.

**Diseño de la Interfaz de Usuario (UI)**

Se ha creado un wireframe que define la estructura visual del sitio:

* **Página de Inicio:** Banner destacado con las últimas colecciones y sección de productos populares.
* **Catálogo:** Cuadrícula con imágenes, nombres y precios de cada prenda.
* **Diseño:** Minimalista y moderno, alineado con la identidad de la marca.

**Arquitectura de la Base de Datos**

La base de datos en MongoDB se ha diseñado para almacenar información de productos, incluyendo:

* **ID único:** Identificador de cada producto.
* **Nombre:** Nombre de la prenda.
* **Descripción:** Detalles como materiales, tallas y colores disponibles.
* **Precio:** Precio de venta.
* **Categoría:** Tipo de producto (hombres, mujeres, unisex).

**Planificación del Desarrollo**

El proyecto se ha dividido en las siguientes fases:

1. **Desarrollo del Front-end:** Creación de las páginas principales (inicio, catálogo) utilizando HTML, CSS y JavaScript.
2. **Configuración del Back-end:** Implementación de la lógica para gestionar productos y conectarse a la base de datos.
3. **Integración Front-end y Back-end:** Conexión de la interfaz de usuario con la base de datos mediante solicitudes HTTP.
4. **Pruebas y Depuración:** Verificación del funcionamiento del sitio en diferentes dispositivos y navegadores.
5. **Despliegue en Firebase:** Configuración del hosting y publicación del sitio.

**Configuración de Firebase en el Proyecto SimpleClothing**

**1. Inicialización del Proyecto en Firebase**

1. **Instalación de Firebase CLI:**

npm install -g firebase-tools

1. **Autenticación en Firebase:**

firebase login

1. **Inicialización de Firebase en el Proyecto:**

firebase init

* + Seleccionar: Firestore, Functions, Hosting y Storage.
  + Configurar reglas de seguridad y directorios.

1. **Configuración de Firestore:**
   * Definir reglas de seguridad en firestore.rules.
   * Crear índices en firestore.indexes.json.
2. **Configuración de Functions:**
   * Elegir JavaScript como lenguaje.
   * Instalar dependencias con npm install.
3. **Configuración de Hosting:**
   * Especificar la carpeta public como directorio de archivos estáticos.
   * Configurar integración con GitHub para CI/CD.
4. **Configuración de Storage:**
   * Definir reglas de seguridad en storage.rules.

**2. Despliegue del Proyecto**

1. **Despliegue en Firebase Hosting:**

firebase deploy

* + URL pública proporcionada por Firebase.

1. **Integración con GitHub:**
   * Configurar GitHub Actions para despliegue automático.

**3. Configuración de la Base de Datos (Firestore)**

* Crear una base de datos Firestore en la consola de Firebase.
* Configurar reglas de seguridad para acceso controlado.

**4. Configuración de Firebase Storage**

* Almacenar archivos como imágenes de productos.
* Definir reglas de seguridad en storage.rules.

**5. Uso de Firebase Functions**

* Implementar funciones en la nube para lógica del backend (actualización de inventario, procesamiento de pedidos).

**Requerimientos Funcionales**

**1. Autenticación y Autorización**

* **Registro de Usuarios:** Los usuarios pueden registrarse con nombre, correo electrónico y contraseña (hash con bcrypt).
* **Inicio de Sesión:** Los usuarios pueden iniciar sesión y recibir un JWT.
* **Protección de Rutas:** Rutas sensibles protegidas con middleware JWT.

**2. Gestión de Productos**

* **CRUD de Productos:** Crear, leer, actualizar y eliminar productos.
* **Atributos de Producto:** Nombre, descripción, precio, categoría, URL de la imagen.

**3. Vistas Estáticas**

* **Página de Inicio:** Lista de productos disponibles.
* **Página de Login y Registro:** Formularios para autenticación.

**4. API RESTful**

* **Endpoints de Autenticación:** /api/auth/register, /api/auth/login.
* **Endpoints de Productos:** /api/products (GET, POST, PUT, DELETE).

**5. Pruebas Unitarias**

* **Herramientas:** Jest y Supertest.
* **Pruebas:** Registro, inicio de sesión, CRUD de productos, protección de rutas.

**6. Despliegue**

* **Plataforma SaaS:** Render o Vercel.
* **CI/CD:** GitHub Actions para automatización.

**Requerimientos No Funcionales**

**1. Seguridad**

* **JWT:** Para autenticación.
* **Protección de Rutas:** Middleware para rutas sensibles.

**2. Escalabilidad**

* **MongoDB:** Base de datos escalable.
* **Diseño Modular:** Backend organizado en módulos.

**3. Rendimiento**

* **Optimización de Consultas:** Uso de índices en MongoDB.
* **Middlewares:** Validación y manejo de errores.

**4. Mantenibilidad**

* **Código Limpio:** Documentado y organizado.
* **Pruebas Unitarias:** Facilitan futuras actualizaciones.

**Diagrama de Entidad-Relación (ERD)**

PRODUCTO

id (PK)

nombre (String)

descripcion (String)

precio (float)

categoria (String)

imagenURL (varchar)

USUARIO

id (PK)

nombre (String)

email (varchar)

contrasena (varchar)

rol (String)

**Conclusión**

La planificación y configuración del proyecto **SimpleClothing** han permitido definir claramente los objetivos, funcionalidades y tecnologías a utilizar. Con un enfoque en la usabilidad, seguridad y escalabilidad, el sitio web busca posicionar a la marca en el mercado digital, ofreciendo una experiencia atractiva y funcional para sus usuarios. El uso de herramientas como MongoDB y Firebase garantiza un desarrollo eficiente y un despliegue escalable.

**Resumen de Tecnologías y Herramientas Utilizadas**

* **Front-end:** HTML, CSS, JavaScript.
* **Back-end:** Node.js, Express.js, MongoDB.
* **Despliegue:** Firebase (Hosting, Firestore, Storage, Functions).
  + **Firebase CLI**: Para gestionar la configuración y el despliegue del proyecto.
  + **Firestore**: Base de datos NoSQL para almacenar información de productos.
  + **Firebase Hosting**: Para desplegar el sitio web de manera rápida y segura.
  + **Firebase Storage**: Para almacenar archivos como imágenes de productos.
  + **Firebase Functions**: Para implementar lógica del backend en la nube.
* **Pruebas:** Jest, Supertest.
* **CI/CD:** GitHub Actions: Para automatizar el despliegue del sitio web.

**Conclusión**

La integración de Firebase en el proyecto SimpleClothing permitió implementar un backend robusto y escalable, así como desplegar el sitio web de manera eficiente. Firebase simplificó la gestión de la base de datos, el almacenamiento de archivos y la implementación de funciones en la nube, lo que resultó en un desarrollo más rápido y eficaz.