Universidad Nacional Mayor de San Marcos

Facultad de Ingeniería de Sistemas e Informática

E.P. de Ingeniería de Software

PEP: DOCUMENTACIÓN FRONTEND

Integrantes

Calle Huamantinco, Luis Eduardo	22200255
Calongos Jara, Leonid	22200272
Flores Cóngora, Paolo Luis	22200232
Matthew Alexandre, Pariona Molina	22200235
Moreno Zevallos Eva Lucía	20200277
Luján Vila Frank José	12200058

Curso: Gestión de la Configuración del Software

Docente: Wong Portillo, Lenis Rossi

INDICE

1. **Introducción**

El proyecto ProfeSoft es una plataforma web destinada a los estudiantes de la Facultad de Ingeniería de Sistemas e Informática (FISI) de la UNMSM. Permite a los usuarios registrados evaluar a los profesores, así como participar en discusiones sobre las evaluaciones. Esta plataforma está diseñada para ser accesible, intuitiva y segura, con soporte para múltiples dispositivos y navegadores.

El frontend de este proyecto es una plataforma web que consta de múltiples plantillas para diferentes funcionalidades. Cada plantilla está compuesta por archivos HTML, CSS y JavaScript que trabajan juntos para crear una experiencia coherente. Las tecnologías utilizadas incluyen HTML para la estructura, CSS para el diseño y estilo, y JavaScript para la interactividad.

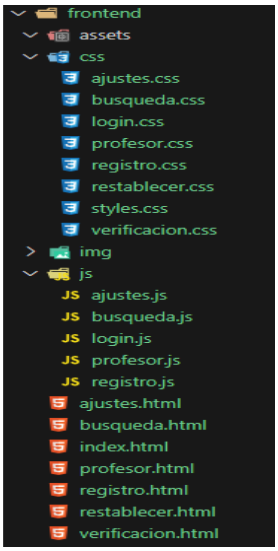
2. Objetivos del Proyecto Frontend

El objetivo del frontend es proporcionar una interfaz intuitiva y fácil de navegar que permita a los estudiantes:

- Registrar e iniciar sesión: Sistema de autenticación exclusivo para estudiantes de la UNMSM.
- Evaluar profesores: Calificar y comentar sobre los profesores de manera pública o anónima.
- Ver y filtrar información de profesores: Acceder a las calificaciones y comentarios existentes y filtrarlos por curso.
- Participar en discusiones: Añadir respuestas y comentarios a las evaluaciones ya existentes.

3. Estructura General del Proyecto

La estructura de frontend para el proyecto está organizado de la siguiente manera:



- **HTML: Estructura y Propósito**

  El HTML (HyperText Markup Language) es el núcleo de la estructura del proyecto web. Actúa como la columna vertebral de cada página, definiendo la disposición y organización del contenido. En el contexto de este proyecto, cada archivo HTML tiene el propósito de proporcionar una estructura clara y semántica para diferentes secciones del sitio web.

  Funciones principales de los archivos HTML en el proyecto:

  - **Estructura de la página:** Organiza elementos como encabezados, párrafos, formularios, botones, y menús en una disposición lógica que guía al usuario a través de la interfaz.

  - **Elementos semánticos:** Emplea etiquetas semánticas (<header>, <main>, <section>, <footer>) para definir la intención de cada parte de la página, mejorando la accesibilidad y SEO.

  - **Conexión con otros recursos:** Incluye enlaces a archivos CSS para estilizar el contenido y scripts JavaScript para agregar interactividad. Además, también enlaza imágenes y otros recursos multimedia, asegurando que todos los elementos necesarios se carguen correctamente.

  Por ejemplo, en el archivo index.html para la página de login, se define la estructura del formulario de inicio de sesión con etiquetas <form>, <input>, y <button>, asegurando que todos los campos estén organizados de manera accesible y fácil de usar.

- ## CSS: Diseño y Estilos

  El CSS (Cascading Style Sheets) se utiliza para dar estilo y diseño visual al proyecto web. Controla la apariencia de los elementos definidos en el HTML, permitiendo que las páginas sean atractivas, consistentes y funcionales.

  Características clave del uso de CSS en el proyecto:

  - **Estilos globales**: Define reglas generales para la tipografía, colores, márgenes, y diseños que se aplican en todo el sitio. Esto asegura una experiencia de usuario consistente en cada sección del proyecto.

  - **Estilos específicos:** Archivos CSS adicionales para personalizar la apariencia de cada página o componente. Por ejemplo, login.css puede incluir reglas específicas para el diseño del formulario de inicio de sesión, asegurando que los campos sean atractivos y accesibles.

  - **Diseño adaptativo:** Se emplean técnicas CSS como media quieres para asegurar que el sitio se vea bien en dispositivos de diferentes tamaños (móviles, tabletas, y escritorios). Esto garantiza que la experiencia del usuario sea óptima sin importar el dispositivo que utilice.

  Por ejemplo, ajustes.css contiene estilos para botones de alternancia y menús desplegables que mejoran la experiencia visual y la usabilidad de la página de ajustes.

- ## JavaScript: Interactividad y Lógica

  El JavaScript añade interactividad y dinamismo a las páginas web. En el proyecto, los scripts JS se utilizan para gestionar la lógica y las funciones que requieren interacción del usuario.

  Funciones principales del uso de JavaScript en el proyecto:

  - **Validación de formularios:** Por ejemplo, login.js verifica que los usuarios hayan ingresado datos válidos antes de enviar el formulario. Esto asegura que el usuario complete todos los campos requeridos y proporciona mensajes de error útiles si hay algún problema.

  - **Actualización dinámica del contenido:** Scripts como busqueda.js permiten que las páginas se actualicen con nuevos resultados sin tener que recargarse completamente, proporcionando una experiencia de usuario más fluida.

- **Manejo de eventos:** Captura eventos del usuario, como clics en botones o cambios en los campos de entrada, y responde a estos eventos para mejorar la funcionalidad del sitio. Por ejemplo, ajustes.js podría ajustar las preferencias del usuario en tiempo real cuando selecciona una nueva configuración.

El uso de JavaScript asegura que el sitio no solo sea estático, sino que responda activamente a las acciones de los usuarios, mejorando la experiencia de navegación.

- ## Assets

Los assets son todos los recursos adicionales que el proyecto necesita para funcionar correctamente y brindar una experiencia completa al usuario. Estos incluyen archivos CSS y JavaScript, fuentes, íconos y otros elementos multimedia.

Organización de assets en el proyecto:

- **CSS y JS:** Almacenados en carpetas organizadas (/css, /js) para mantener el proyecto estructurado y facilitar el mantenimiento. Cada sección de la aplicación tiene sus propios archivos CSS y JS para simplificar la gestión de código.

- **Fuentes e íconos:** Aseguran que la tipografía sea atractiva y coherente, mientras que los íconos mejoran la navegación visual. Los assets pueden incluir archivos de fuentes personalizados y librerías de íconos, como FontAwesome o Material Icons.

- **Otros recursos multimedia:** Vídeos, gráficos, y archivos PDF que podrían ser necesarios para complementar el contenido de la plataforma. Estos assets están optimizados para que no ralenticen la carga del sitio.

- ## Img:

La carpeta img contiene todas las imágenes utilizadas en el proyecto, que pueden incluir logotipos, íconos, fotos de perfil, y otros elementos visuales que enriquecen la experiencia del usuario.

Buenas prácticas para la gestión de imágenes:

- **Optimización de imágenes:** Las imágenes se optimizan para reducir el tamaño del archivo sin perder calidad visual. Esto es crucial para asegurar que las páginas se carguen rápidamente y que la experiencia del usuario no se vea afectada.

- **Uso semántico y descripciones alternativas:** Al incluir imágenes en el HTML, se utilizan atributos alternativos para proporcionar descripciones accesibles. Esto no solo mejora la accesibilidad para usuarios con discapacidades visuales, sino que también optimiza el SEO del sitio.

- **Organización por tipos de contenido:** Las imágenes se organizan en subcarpetas dentro de /img, como logos, profesores, iconos, para mantener el proyecto limpio y facilitar el acceso.

Por ejemplo, la carpeta img/fisi podría almacenar fotos de la FISI que se muestran en el fondo de algunas plantillas, mientras que img/logos almacena el logotipo principal de la plataforma para uso en el encabezado y el pie de página.

4. Detalles de Funcionalidad por Plantilla

Login

- **index.html**:
  - Define la estructura del formulario de inicio de sesión con campos de entrada para el nombre de usuario y la contraseña.
  - Incluye enlaces a styles.css, login.css, y login.js.
  - Contiene etiquetas para mensajes de error que se muestran dinámicamente según la validación.
- **styles.css**:
  - Proporciona estilos básicos que aseguran consistencia en la tipografía y diseño de toda la aplicación.
- **login.css**:
  - Ajusta el diseño del formulario de inicio de sesión, asegurando una apariencia clara y profesional.
  - Estiliza botones, campos de entrada, mensajes de error, y el contenedor principal.
- **login.js**:
  - Implementa validaciones de campos, asegurando que el usuario haya completado todos los datos requeridos antes de enviar.

- Proporciona mensajes de error si los campos no son válidos, como "usuario o contraseña incorrectos".
- Maneja el envío del formulario, asegurando que la entrada se procese correctamente para iniciar sesión.

Ajustes

- **ajustes.html**:
  - Contiene la estructura para la página de ajustes, incluyendo secciones para cambiar la contraseña, preferencias de tema, y notificaciones.
  - Incluye referencias a styles.css, ajustes.css, y ajustes.js.
- **ajustes.css**:
  - Estiliza controles de usuario específicos, como sliders para cambiar el brillo, menús desplegables para seleccionar opciones, y botones de alternancia para activar/desactivar configuraciones.
- **ajustes.js**:
  - Maneja la interactividad, como el cambio de temas en tiempo real, permitiendo a los usuarios ver cómo se aplican las preferencias sin recargar la página.
  - Guarda las preferencias del usuario localmente o mediante una API para que se recuerden las configuraciones en futuras sesiones.

Búsqueda

- **busqueda.html**:
  - Estructura la página para incluir un campo de búsqueda y un área para mostrar los resultados.
  - Se enlaza con styles.css, busqueda.css, y busqueda.js.
- **busqueda.css**:
  - Define estilos para la barra de búsqueda, filtros adicionales, y el diseño de la lista de resultados.
  - Estiliza los resultados para que sean claros y fáciles de leer.
- **busqueda.js**:
  - Captura la entrada del usuario y envía la consulta a un servicio de búsqueda.
  - Maneja la respuesta, mostrando los resultados dinámicamente en la página sin necesidad de recargarla.

- Implementa sugerencias en tiempo real a medida que el usuario escribe.

Registro

- **registro.html**:
  - Define la estructura del formulario de registro, incluyendo campos para el nombre de usuario, correo electrónico, contraseña, y confirmación de contraseña.
  - Incluye enlaces a styles.css, registro.css, y registro.js.
- **styles.css**:
  - Mantiene una base de diseño unificada para el proyecto.
- **registro.css**:
  - Estiliza los campos del formulario de registro, haciendo que la experiencia de usuario sea intuitiva y clara.
  - Estilos específicos para mensajes de error y confirmaciones, asegurando que los usuarios puedan identificar fácilmente qué correcciones se necesitan.
- **registro.js**:
  - Realiza validaciones, como comprobar que la contraseña cumpla con los requisitos mínimos de seguridad y que el correo electrónico esté en el formato correcto.
  - Valida que las contraseñas coincidan y proporciona mensajes de error claros si no es así.
  - Envía los datos a la API para crear una nueva cuenta si todo es correcto.

Restablecer Contraseña

- **restablecer.html**:
  - Proporciona un formulario para que el usuario introduzca su correo electrónico y reciba un enlace para restablecer la contraseña.
  - Se enlaza con styles.css y restablecer.css.
- **styles.css**:
  - Asegura consistencia en la apariencia general de la interfaz.
- **restablecer.css**:
  - Personaliza el diseño del formulario de restablecimiento de contraseña, incluyendo estilos para mensajes de confirmación y errores.

Verificación

- **verificacion.html**:
    - Estructura la página para mostrar instrucciones sobre cómo verificar la cuenta del usuario.
    - Se enlaza con styles.css y verificacion.css.
- **styles.css**:
    - Define la base del diseño general para todas las páginas.
- **verificacion.css**:
    - Añade estilos específicos para mostrar mensajes de verificación y estado, asegurando que los usuarios puedan completar el proceso de manera intuitiva.

Perfil del Profesor

- **profesor.html**:
    - Proporciona la estructura para mostrar información detallada del perfil del profesor, incluyendo biografía, materias enseñadas, y horarios.
    - Se enlaza con styles.css, profesor.css, y profesor.js.
- **profesor.css**:
    - Estiliza el perfil del profesor, asegurando que la información se muestre de forma clara y profesional.
- **profesor.js**:
    - Permite la edición de la información del perfil y actualiza dinámicamente los datos mostrados en la página.
    - Gestiona la interacción del usuario, como enviar mensajes al profesor o programar citas.

5. Recursos Adicionales

Las imágenes y otros recursos están organizados para asegurar una carga eficiente y fácil mantenimiento. Los activos se almacenan en una carpeta específica (img/) y se optimizan para mejorar la velocidad de carga de la página.