**1. Desarrolla una página web simple que se ejecute en Front-End, utilizando la versión actual de HTML.**

Hemos desarrollado varias páginas HTML, como index.html, login.html, registro.html, etc., todas utilizando la versión más reciente de HTML5. Estas páginas contienen estructuras HTML básicas y avanzadas para cumplir con los requisitos del proyecto.

**2. Integra el Framework BootStrap de CSS para la maquetación de sus páginas responsivas.**

Hemos integrado Bootstrap en nuestras páginas, como se evidencia en los archivos HTML que cargan archivos CSS y JavaScript de Bootstrap para la maquetación responsiva. Ejemplo de inclusión de Bootstrap en index.html:

html

Copiar código

<!-- Bootstrap CSS -->

<link href="{% static 'myapp/vendor/bootstrap/css/bootstrap.min.css' %}" rel="stylesheet">

**3. Utiliza lenguaje de programación de JavaScript para agregar funcionalidades de Front-End con la finalidad de mejorar la carga de sus páginas web.**

Hemos añadido funcionalidades de JavaScript en el archivo script.js para mejorar la interactividad y la experiencia del usuario en nuestras páginas web. Esto incluye el manejo de eventos y la manipulación del DOM.

**4. Programa funcionalidades y validaciones en el ingreso de datos, tanto en formatos como en reglas de negocio en Front-End, incorporando librerías de JavaScript, como por ejemplo JQuery.**

Para la validación de formularios y otras funcionalidades de Front-End, hemos utilizado JavaScript. Un ejemplo de esto es la validación de formularios de registro y login en login.html y registro.html.

**5. Desarrolla la programación de un requerimiento de una página web, en Front-End, para el consumo de servicios web, dejando un respaldo en la plataforma de versionamiento GitHub.**

El archivo script.js incluye código para consumir un archivo JSON y actualizar dinámicamente el contenido de la sección de promociones en index.html. Este proyecto está versionado en GitHub.

**6. Describe las capas de presentación, negocios y datos de un proyecto web utilizando el Framework Django.**

**Capa de Presentación:** Consiste en los archivos HTML, CSS y JavaScript. Son los que el usuario final ve e interactúa.

**Capa de Negocios:** Incluye la lógica de la aplicación, implementada en los archivos Python en Django, como views.py, donde se manejan las solicitudes del usuario y se generan respuestas adecuadas.

**Capa de Datos:** Maneja la interacción con la base de datos. En Django, esto se realiza a través de los modelos definidos en models.py y la base de datos SQLite.

**7. Configura una aplicación web para levantar un servidor y una base de datos, utilizando el framework Django.**

Hemos configurado y levantado un servidor Django con una base de datos SQLite. La configuración está definida en settings.py.

**8. Genera una aplicación web utilizando un framework que permita programar en Back-End teniendo accesos a datos.**

La aplicación web está generada utilizando Django, permitiendo el acceso y la manipulación de datos a través de modelos, vistas y templates.

**9. Realiza un CRUD de acuerdo con los requerimientos del proyecto semestral, integrando las distintas capas de sus aplicaciones.**

Se han desarrollado vistas y formularios en Django para permitir la creación, lectura, actualización y eliminación de registros de usuarios, manejando así el CRUD completo.

**10. Diferencia autenticación y autentificación, utilizando la técnica correcta para desarrollar los accesos a sitios privados de sus proyectos.**

**Autenticación:** Verificación de la identidad del usuario (login). **Autorización:** Determinación de qué recursos puede acceder el usuario autenticado.

Se ha implementado autenticación usando login\_view y registro en views.py.

**11. Desarrolla código que utilice algún método de cifrado para almacenar datos sensibles ingresados en sus aplicaciones web en la base de datos de sus proyectos.**

En el archivo models.py, se utiliza el módulo de autenticación de Django que cifra automáticamente las contraseñas del usuario antes de almacenarlas en la base de datos.

**12. Describe el funcionamiento de la tecnología cliente-servidor, como así mismo elemento básico de TCP/IP, diferenciando lo que se puede hacer en Front-End y Back-End.**

**Cliente-Servidor:** El cliente realiza solicitudes a un servidor que responde con los recursos solicitados. La comunicación se lleva a cabo a través del protocolo HTTP, que funciona sobre TCP/IP.

**Front-End:** Todo lo que el usuario interactúa directamente (HTML, CSS, JS).

**Back-End:** Maneja la lógica del servidor, la base de datos y la autenticación del usuario (Django en este caso).

Con esto, cumplimos todos los requerimientos de la rúbrica para el proyecto semestral.