Deusto
Facultad de Ingeniería
Ingeniaritza Fakultatea

Proyecto web colaborativo: E3 + E4

Asignatura	Ingeniería Web
Curso	Curso 2023/2024
Fecha	25/04/2024
Tipo de entrega	Grupos de 3 personas * En caso de no poneros de acuerdo, se harán los grupos de forma aleatoria.
Fecha límite de entregas	• E2 → 03/05/2024 a las 19:00 • E3+E4 → 28/05/2024 a las 15:00

¿Cómo documentar el proyecto web?

La documentación de los proyectos deberá contar, como mínimo, con la siguiente información:

1. Explicación general del proyecto en Django.
    - Descripción de los escenarios y funcionalidades planteadas.
    - Descripción del modelo de datos diseñado.
    - Descripción de las plantillas utilizadas para las interfaces web (descripción de la estructura HTML y estilos CSS).
    - Descripción de la estructura del proyecto web (ficheros correspondientes a cada componente de la arquitectura con Django)
    - Descripción de las funcionalidades añadidas implementadas.
2. Explicación general del proyecto en JavaScript.
    - Descripción de la estructura del proyecto web.
    - Descripción de las funcionalidades implementadas.
3. Conclusiones.
4. Bibliografía.

Dicha información se organizará en un documento que cumpla con lo establecido en el apartado “Comunicación Escrita” de ALUD.

Deusto
Facultad de Ingeniería
Ingeniaritza Fakultatea

Criterios de Evaluación

Entregable 3

Se evaluarán los siguientes aspectos:

- **4 puntos:** Ampliación de funcionalidades en Python. Escoge las funcionalidades que quieras de las listadas a continuación (máximo 4 puntos):
    - (2 puntos) Envío de emails desde la aplicación
    - (2 puntos) Subida de ficheros al servidor mediante <input type="file"> y mostrarlos en una página (tienen que poder descargarse)
    - (2 puntos) Autenticación y registro de usuarios. La aplicación no mostrará las páginas si el usuario no se ha autenticado previamente.
    - (2 puntos) Buscador de registros, ofreciendo al usuario al menos la posibilidad de filtrar por dos campos distintos.
    - (1 punto) Buscador simple de registros por un único campo.
    - (1 punto) Registro de mensajes (informativos, avisos, errores) mediante Logger
    - (1 punto) Paginación en tablas/listados de los resultados de una tabla.

  * Se trata de un apartado de investigación/ampliación. El objetivo es, partiendo de los conocimientos adquiridos en clase, demostrar la capacidad de ampliar vuestros conocimientos trabajando en equipo.
  * Si las funcionalidades mínimas requeridas en la E2 no se encuentran funcionando correctamente, la puntuación de este apartado será de 0.
  * En caso de implementar más de dos funcionalidades, se tendrá en cuenta de forma positiva (pero no se podrán obtener más de 4 puntos en este apartado).

- **(3 puntos)** Implementar las siguientes funcionalidades JavaScript (escoger 3 de ellas, en caso de implementar más de 3 opciones se tendrán en cuenta las 3 con mejor puntuación):
    - (1p) Crear un evento al hacer click en un botón o enlace que produzca el cambio (aumentar/disminuir) del tamaño de los elementos de texto (<h1>, <h2>, <p>, ...).
    - (1p) Validar campos de un formulario antes de su envío al servidor, impidiendo el envío si no se supera la validación (p. ej: número dentro de un rango determinado, comprobar que se ha introducido el mismo valor en dos campos, permitir únicamente contraseñas que contienen ciertos caracteres, impedir que se introduzca un nombre de usuario perteneciente a una lista de palabras reservadas, ...). Mostrar el mensaje de error tras la validación.
        - Las validaciones tienen que aportar valor, es decir, no se contabilizarán validaciones que ya se pueden hacer de forma simple mediante HTML (campo vacío o required, email válido, campo númerico dentro de un rango, etc.).
    - (1p) Autocalcular un campo de un formulario (p. ej: generar el campo de username a partir del nombre y apellidos del usuario, hacer una operación matemática con el valor de dos campos, seleccionar automáticamente el

valor de un campo en función de otra selección previa, calcular un valor numérico cambiar la unidad seleccionada, convertir los caracteres seleccionados a mayúsculas, ...)
    - (1p) Generar contenido HTML a partir de un array de datos (el array puede incluir objetos, strings, ...) y añadirlo al DOM (p. ej: incluir los valores de un elemento de tipo <select>, crear un menú generando cada elemento a partir de los valores de un array, ...).
    - (1p) Capturar un evento en el DOM y producir un cambio en el estilo/contenido de la página (p. ej: mostrar/ocultar un bloque al hacer click en “expandir información”, mostrar una alerta si el usuario realiza una acción determinada,...)

  * Para aprobar la entrega es necesario demostrar que habéis adquirido un mínimo de conocimientos de JS, por lo que se requiere obtener la mitad de los puntos (1’5 puntos) de esta parte para aprobar la entrega .
  ** Es posible proponer la implementación de otras funcionalidades que impliquen captura de eventos y modificación del DOM. En caso de querer proponer alguna que no figure en los puntos anteriores, tendréis que escribir un email explicando la propuesta antes del 17/05/2023 a las 14:00.

- **(2 puntos)** Funcionalidades JavaScript para cargar y/o almacenar datos utilizando Fetch. Escoger una de las siguientes opciones:
    - Cargar datos y modificar el DOM mediante JavaScript: llamada a API de Django utilizando Fetch para obtener datos y mostrar los valores modificando el DOM (p. ej: ampliar los detalles de un registro que se muestran al usuario).
    - Envío de datos de un formulario mediante AJAX para su almacenamiento en BBDD y posterior visualización del resultado (p. ej: cambiar el estado de un registro de “en proceso” a “completado” y mostrarlo en pantalla sin necesidad de recarga de la página).
- **(1 punto)** Calidad del código entregado (estructura correcta, uso de las técnicas adecuadas que se han visto en clase, limpieza, comentarios en el código, ausencia de errores en la sintaxis o durante la ejecución de la aplicación,...)
- **Penalizaciones**
    - (1 punto) El número de commits realizados es mínimo o no se han realizado commits de forma regular (deben figurar commits todas las semanas no vacacionales) y por todos los integrantes del equipo.
    - (2 puntos) El código no es limpio, no está bien formateado u ordenado, la implementación es muy mejorable (por ejemplo, unas plantillas tienen herencia y otras no, hay código repetido que podría reutilizarse mediante funciones, apenas se utilizan Vistas Basadas en Clases, al editar campos de un formulario se pierden valores,…).
    - (2 puntos) El código contiene errores de sintaxis, de ejecución (de Python o JavaScript, aunque sea en la consola), no se controla si saltan errores al dejar vacío algún campo, ...
    - (1 punto) El número de registros de prueba es insuficiente o no es relevante según el negocio del que se trate.

Para obtener la puntuación completa de las funcionalidades de Javascript, se tendrá en cuenta la calidad de la solución aportada. Ejemplos:

- La funcionalidad se consigue pero el código no está organizado en funciones, realiza acciones innecesarias o mejorables, etc.
- La funcionalidad se consigue pero de forma muy mejorable (por ejemplo: se hace una validación y el resultado sale en un “alert” en lugar de construir un mensaje en HTML dentro de la página).
- La funcionalidad implementada es muy básica o no tiene sentido (al hacer un click, mostrar un contenido que no aporta nada, o su contenido es muy básico, desentona con los estilos de la página,...
    - Ejemplo de funcionalidad sin sentido: comprobar que un campo de texto está vacío → no tiene sentido porque se puede conseguir añadiendo “required”.

En caso de que el proyecto no se pueda cargar correctamente (mediante el comando *runserver*) y probar, la calificación será automáticamente cero.

Prestad atención a los requerimientos y los indicadores, por favor, y a la fecha límite de entrega.

Entregable 4

Es importante prestar atención en los siguientes aspectos de la entrega ya que se tendrán en cuenta en la calificación:

- La documentación incluye el contenido recogido en el punto “¿Cómo documentar el proyecto web?” y contiene un índice que enlaza correctamente a cada apartado.
- El contenido está organizado en distintos apartados señalados en el documento “Apartados de una documentación” disponible en ALUD.
- Las explicaciones incluyen diagramas UML para facilitar la lectura y los diagramas empleados reflejan correctamente su contenido.
- Se cuida la calidad en la redacción, con párrafos bien redactados, siguiendo un orden lógico y expresando de forma clara las ideas o conceptos.
- El formato empleado es el indicado y **se cuida la ortografía**. Debe estar escrito en tercera persona y el enfoque debe ser de manual técnico, no contando la historia de su desarrollo.

La extensión máxima de la documentación será de 20 hojas (en caso de necesitar extenderse más, se permite incluir diagramas o capturas de pantalla como anexos).

Normas de entrega

- **Fecha y hora límite: 28/05/2024 a las 15:00. No se admitirán entregas posteriores y no se realizarán excepciones (una entrega posterior se calificará con un cero).**
- Cread un archivo comprimido (se admiten ZIP, GZIP o TGZ, RAR) con el prefijo IW seguido del nombre del equipo (EQUIPO1, EQUIPO2, EQUIPO3, ...) y del sufijo

E3E4, por ejemplo: IW-EQUIPO1-E3E4.zip (o en lugar de ".zip" la extensión del compresor que hayáis utilizado).

- El archivo comprimido debe contener:
    - El proyecto de Django al completo para ejecutar. Incluida la base de datos, la cual deberá contener registros de prueba simulando una situación de uso real.
    - La URL del repositorio de Github (en caso de ser un repositorio privado, dar acceso a jvadillo@deusto.es).
    - Un archivo requirements.txt con el contenido del comando “pip freeze”.
    - La documentación para el E4 (un único documento en formato PDF).
- Se debe subir a la entrega habilitada en ALUD.
- Al margen de la entrega comentada, **cada equipo tendrá que completar el siguiente formulario (enlace)**, donde se indican las funcionalidades implementadas en cada apartado.