Entregable # 2 – Arquitectura MVC + Servicios + DI + Docker

1. Implementación en Django – adicional para entregable 2

Instrucciones para el arquitecto de usabilidad (no puede asignársele esa responsabilidad al arquitecto del proyecto pasado):

- Como arquitecto de usabilidad usted será responsable de verificar que todo el sistema desarrollado cumpla con buenos estándares de usabilidad y deberá asignar a los desarrolladores correspondientes los cambios que considere pertinentes.
- Aspectos para tener en cuenta como arquitecto de usabilidad:
  - Que todas las vistas de la aplicación manejen la misma estructura visual. Que se utilicen los mismos colores, fondos, tipos de letra, etc.
  - Que los formularios estén bien diseñados y contengan una estructura coherente.
  - Que los formularios no se vacíen (y toque volverlos a llenar) si se encuentran errores.
  - Que los campos de los formularios estén bien diseñados (campos de textos, campos de selección, radio buttons, textarea, etc).
  - Que la aplicación tenga botones de fácil acceso (menú principal, menús laterales, pie de página correspondiente, etc).
  - OPCIONAL: Que la aplicación cuente con un sistema de navegación (buscar en Google “breadcumbs navigation”).
  - OPCIONAL: que la aplicación se vea bien, tanto en computador como en celular.
- Utilice GitHub projects para controlar las actividades y tareas del equipo en cuanto a pendientes de usabilidad.
- Lectura recomendada: https://www.quicksprout.com/website-usability/

Instrucciones globales (entregable #2):

- Implemente todo lo que se pidió para el entregable #1 (si es que aún no lo ha hecho / ver entregable anterior).
- Haga todas las correcciones que el docente le notificó tanto en las sustentaciones pasadas, como en el documento entregado a cada equipo. Si no realiza las correcciones de la entrega pasada, no se revisará la nueva entrega.
- Actualice el diagrama de clases y el diagrama de arquitectura con las recomendaciones pertinentes, y verifique que hay consistencia entre los diagramas y el código elaborado. **Nota:** en el diagrama de arquitectura se debe poder ver como aplicaciones externas se comunican con los servicios implementados.

- En esta entrega el sistema se deberá implementar en 2 idiomas (recuerde, nada de textos “quemados” ni en controladores, ni en vistas, siempre use LANG).
- Implemente dos pruebas unitarias simples (no tiene que instalar nada, Django ya lo soporta).
- Implemente un servicio web donde usted provea en formato JSON, información relevante de su aplicación.
    - Por ejemplo: (i) lista de productos en stock en venta, incluyendo el enlace directo a la visualización de cada producto; (ii) lista de estudiantes con su nota acumulada; (iii) lista de rutinas de gimnasio, (iv) listado de gafas en promoción, (v) lista de mascotas para adoptar, etc.
    - El equipo siguiente deberá consumir ese servicio, y mostrarlo en la aplicación de ese equipo.
    - Por ejemplo: El equipo 1 provee un servicio donde despliega información de sus productos. El equipo 2 consume ese servicio, crea un nuevo controlador y vista, y crea una ruta: /productos-aliados y ahí despliega la información que consumió del servicio del Equipo 1. Adicionalmente, el equipo 2 debe proveer un servicio, que deberá ser consumido por el equipo 3, y así sucesivamente.
- Dentro de su aplicación, consuma el servicio del equipo precedente (ver explicación anterior).
- Dentro de su aplicación, consuma un servicio de una compañía tercera (ya sea Google, Facebook, un API en la web, etc). Por ejemplo, el equipo 3, podría consumir un servicio para mostrar el clima actual en Medellín, y desplegarlo en la parte superior de la cabecera de la aplicación.
- Implemente una inversión de dependencias. **Debe existir una interfaz y dos clases concretas**. Sugerencias de inversiones de dependencias:
    - Generar reportes en PDF y Excel.
    - Cargar info importante desde una API o quemado en código.
    - Simular pagos con cheques (mostrar un PDF con la info del cheque para pagar), o pagos donde le reste dinero a la cuenta del usuario (simulando que el usuario tiene un atributo con su dinero disponible).
    - Etc.
- Despliegue la aplicación con Docker (en GCP).
- OPCIONAL Implemente un sistema de nombre de dominio simplificado (que no toque acceder por la IP de Google) -> utilice dominios gratis .tk http://www.dot.tk/.
- Implemente mejoras que considere pertinentes, paginación, carga de archivos, mejora en los sistemas de fakers, menús laterales, un sistema de banner en la página principal, entre otros.
- Todos los participantes deben codificar (tener commits registrados en el repositorio). Si un estudiante no tiene commits tiene cero en la entrega.

Instrucciones de entrega para el arquitecto principal (debe ser uno diferente al de la primera entrega):

- Suba el proyecto a la cuenta que se le brindó en GCP (despliéguelo con Docker).
- Finalmente comparta tanto el link del repositorio, como el dominio .tk con el docente antes de la fecha de finalización de entrega.
- El día de la sustentación, deberán desplegar la instancia de GCP para poder navegarla durante la sustentación.

NOTA IMPORTANTE: el día antes de la entrega (ni el mismo día de la entrega) ni el monitor ni el docente solucionarán dudas.