Evaluación Final - Base de Datos II

Laboratorio Doofenshmirtz desea una aplicación web que les permita gestionar su catálogo de exámenes y servicios. En este sentido, desean que la aplicación permita:

1) **Gestionar exámenes y servicios**: En esta opción, se podrá crear, modificar y eliminar un examen o servicio. De cada uno de ellos se saben sus datos básicos: un código identificador único, una categoría a la que pertenecen, el tipo de muestra del mismo, el precio (en Bs) y las indicaciones pertenecientes al examen o servicio. Las indicaciones también podrán crearse, modificarse y eliminarse, y la aplicación deberá estar en capacidad de permitir al usuario seleccionar una de las indicaciones ya creadas al crear un examen, o bien, crear una nueva si llegase a ser necesario.

2) **Gestionar categorías**: Esta opción permitirá al usuario crear, modificar y eliminar categorías de exámenes. Por cada categoría se contará con su nombre y descripción, y esta categoría es la que seleccionará el usuario cuando esté registrando el examen o servicio.

3) **Consultar catálogo:** Esta opción permitirá al usuario listar todos los exámenes y servicios, pudiendo filtrarlos por su categoría o por tipo de muestra. Este listado deberá ofrecerse en una tabla, que por cada examen indicará su código, nombre y precio, además de las opciones:

Consultar, modificar y eliminar. Además de esto, cada examen deberá poder ser consultado de manera individual.

4) **Ver reporte**: Esta opción generará un reporte, en el cual se podrán visualizar:

- Cuántos exámenes hay registrados en cada categoría.

- Cuál es la indicación de examen más común.

- Una lista que agrupe los exámenes por precio según el intervalo de: 1 - 100 bs, 101 - 200 bs, 201 - 300 bs, 301 - 500 bs, 501+ bs. Esta lista deberá indicar cuántos exámenes hay en cada intervalo.

Esta aplicación deberá cumplir con los siguientes criterios técnicos para poder ser evaluada:

1) Ser una aplicación web, desarrollada utilizando Flask, Jinja2 y PyMongo, además de una librería de CSS a su gusto (**obligatorio**).

2) Utilizar git en un repositorio remoto, con un servicio como GitHub o GitLab. (**obligatorio**)

3) Estar desplegada en un servicio en línea, como puede ser PythonAnywhere, y contactar con MongoDB Atlas en una capa gratuita. (**opcional - puntos adicionales**)

4) Utilizar una librería del tipo ChartJS para la visualización de la data en los reportes (**opcional - puntos adicionales**).

5) Ser atractiva visualmente, contar con mensajes de éxito y error, con las validaciones correspondientes, etc. (**evidentemente obligatorio**)

Entrega

La entrega será realizada a través del repositorio de GitHub o GitLab, el cual deberá ser compartido con mi cuenta (@AbnerSaavedra), y se tomará como entregada la versión final del repositorio al jueves 16-05-2024 a las 11:59 p.m. Sólo se solicita entrega del repositorio con el código, por lo que no hace falta informes ni diagramas.

El día viernes 17-05-2024 se realizará una defensa presencial en el laboratorio. Para esta defensa será obligatorio asistir con una diapositiva que refleje su solución al problema, además de dar un tour por el proyecto. **La defensa es estrictamente obligatoria**, y faltar a la misma de manera injustificada acarreará la pérdida de los puntos del proyecto.

Ponderación

- Uso de herramientas (MongoDB, Flask, Jinja2, GitHub): **10 pts.**

- Diseño de colecciones: **10 pts**.

- Gestión de exámenes, servicios y categorías: **7.5 pts.**

- Consulta de catálogo: **7.5 pts.**

- Reportes: **5 pts.**

- Implementación de un sistema de autenticación con usuario y contraseña sencillo: ***5 pts.***

- Calidad del código: **5 pts.**

TOTAL: 50 pts.

***Adicionales*:**

- Despliegue con herramientas cloud (PythonAnywhere, MongoDB Atlas): ***10 pts.***

- Uso de librería ChartJS: ***10 pts.***

***Total adicional*:** 20 pts.