Capacitación Inicial 2018

Entrega Diseño del modelo “Ratatoullie”

**Documento de pre-entrega**

Grupo 04

Gonzalez Jose Federico

Leguizamón Anahí

Valdez Alejandro

**Enunciado – Alcance**

Se desea realizar el diseño de un sitio web de recomendación de restaurantes.

La plataforma mantiene la información de restaurantes y usuarios.

Solución propuesta:

El sistema deberá mostrar los restaurantes que estén registrados al mismo.

Se proveerá en principio de 3 tipos de usuarios registrados al sistema.

Cada usuario tiene una función específica dentro del sistema, detallada a continuación:

* Usuario Normal: la función que este puede realizar es la de votar un restaurante, menú y platos. Puede añadir comentarios a los restaurantes. También podrá denunciar a otros usuarios explicando los motivos de la denuncia, para más tarde ser visto por el Administrador el cual tomará las decisiones que cree necesarias.
* Usuario Responsable: puede agregar un restaurante al sistema (incluyendo todos los datos necesarios). También podrá realizar comentarios y/o votaciones a otros restaurantes pero con una restricción, la cual consiste en que no podrá comentar/votar a restaurantes que se encuentre dentro del radio de 1km de la ubicación de sus restaurantes. Al igual que los usuario normal, podrá realizar denuncias de otros usuarios.
* Usuario Administrador: su función será la de administrar el sistema, controlar a los usuarios de tipo normal y responsables (en caso de ser necesario), así como también de las publicaciones que estos realicen.
* Usuario-No-Registrado: se contempla este caso, lo único que podrá ver este son los restaurantes mejor rankeados que aparecerán en la pantalla de inicio, así como también su información básica.

Los usuarios Responsables pueden tener desde 1 hasta muchos (N) restaurantes registrados.

Un restaurante solo pertenece a un usuario. Los restaurantes pueden tener de 1 a muchos( N) menúes. Los menús tienen de 1 a muchos(N) platos. Los menúes pertenecen a 1 o varios(N) restaurantes. Los platos pueden estar en 1 o varios(N) menúes.

Cada restaurante, menú o plato puede ser votado por varios usuarios. Los usuarios pueden votar al mismo restaurante 1 sola vez, considerando un nuevo voto sobre este como una modificación del anterior voto. Esto no se aplica al menú y a los platos ya que estos se puede repetir en diferentes restaurantes.

Los usuarios son rankeados de acuerdo a la cantidad de comentarios. En principio se consideran 3 categorías (visitantes, comensales y gourmets). Solo podrá tener una sola categoría. Las categorías pueden estar en varios usuarios a la vez.

Los usuarios registrados podrán realizar búsquedas de restaurantes con varios filtros los cuales se podrán aplicar de forma simultánea.

* Basados en la ubicación del usuario: se tomará la ubicación actual de forma automática, o se podrá ingresar un destino. El límite de la búsqueda dependerá del tamaño del rango que se le asigne.
* Basados en la cantidad de votos: se le podrá aplicar un rango que diga desde que cantidad de votos hasta que cantidad.
* Recomendaciones basadas en amigos del usuario: la búsqueda mostrará los restaurantes recomendados por el/los “amigos” del usuario seleccionado/s (de no seleccionar ninguno, muestra todas las recomendaciones).
* Según etiquetas predefinidas.

Cada restaurante puede tener desde 0 beneficios hasta muchos(N). Cada beneficio cuenta con alguna particularidad especial. Como ejemplos: el resaltado de letras, mejor posicionamiento en las listas, etc.

Con respecto al primer acercamiento a lo que es una interfaz gráfica del sistema, se propone una página inicial con una lista de restaurantes, más un apartado para loguearse. Una vez ingresado se muestra otra página la cual dispondrá de un buscador y un menú de navegación con las funcionalidades que el tipo de usuario logueado requiera.

**Diagrama de clases**

Archivo anexado a la carpeta del drive.

Cuestiones de Diseño:

Se decide implementar la clase System mediante el uso del patrón de diseño Singleton. Esto se debe a que solo necesitamos una instancia de System para de esta forma evitar la inconsistencia de datos (tener 2 o más instancias produciría que un dato que se modifique en una no se vea reflejado en las otras) y poder tener un mayor control de acceso (generar un control estricto sobre cómo y cuándo acceder a ella).

Para que el usuario pueda configurar el tipo de notificación que se desea recibir se decide implementarlo a través del patrón Strategy. Se le envía un comentario y de acuerdo al tipo de este se toma la estrategia necesaria para devolver el comentario deseado. También se prevé un futuro cambio en tipos de usuarios, ya sea agregación o eliminación de algún tipo, y así facilitar la adaptación del nuevo requerimiento pedido.

La clase CommentFilter ocupando el rol de Estrategia del patrón, mientras que sus hijos el de EstrategiaConcreta, siendo estos los que resolverán las cuestiones planteadas.

Aclaración: por el momento los tipos de usuario de decidieron resolver a través de un método en la clase User, que retorne el tipo de usuario según la cantidad de comentarios.

**Diagrama/s de secuencia (si fue/ron requerido/s)**

Archivo anexado a la carpeta del drive.