UNIVERSIDAD TECNOLÓGICA NACIONAL

Facultad Regional Buenos Aires

Diseño de Sistemas

2024

Docentes: Ezequiel Escobar, Lucas Saclier, Nicolas Gustavo Contreras y Pablo Sabatino

Trabajo de a pares: Cuidandonos

Sede: Campus	**Curso: K3052/K3152**

Legajo	**Apellido y Nombre**
203.528-5	**Covalchuk, Cecilia Denisse**
204.176-5	**Yedid, Victoria Sol**

Patrones de diseño utilizados en la resolución del trabajo práctico

Para gestionar las reacciones ante incidentes, hemos adoptado el **patrón de diseño Strategy.** Esta elección se basa en dos motivos principales: la posibilidad de que el usuario modifique la configuración de las reacciones y la flexibilidad para agregar nuevas reacciones en el futuro. Para implementar el patrón Strategy, hemos creado una interfaz llamada "Reaccion", que es implementada por diversas clases representando las reacciones específicas, como "EnviarAlerta", "RealizarLlamada" y "EsperarFalsaAlarma". Cada una de estas clases actúa de manera polimórfica, ya que todas comparten el mismo método "reaccionar(Viaje)". Esta característica permite que el usuario configure la respuesta deseada ante un incidente intercambiando fácilmente entre diferentes reacciones disponibles. Además, si se desea agregar una nueva reacción, simplemente se debe crear una nueva clase que represente dicha reacción y hacer que implemente la interfaz "Reaccion". De esta manera, la nueva reacción se integra sin problemas con las existentes, manteniendo la capacidad de configuración del usuario y la polimorficidad del sistema.

Para integrar la API REST "Distance Matrix API" es fundamental aplicar el **patrón de diseño Adapter**. La elección se justifica porque la API ya está desarrollada y debemos asegurar la compatibilidad entre su interfaz y nuestro sistema. Cuando trabajamos con APIs externas, no podemos modificar su código fuente, lo que nos obliga a adaptar nuestro diseño para que sea compatible con la API existente. El patrón Adapter nos permite precisamente eso: adaptar la interfaz de la API para que se ajuste a las necesidades y estructura de nuestro sistema, evitando así posibles conflictos de implementación.