Desafio #2: UdeaStay

Nikolas Geovanny Ortega Suarez   
  
Keiner torres

Informatica 2

Augusto Enrique Salazar

ingeniería en telecomunicaciones

Universidad de Antioquia

17/07/2025

**Análisis y desarrollo de solución(13 de mayo/ 17 de mayo):**En el desafío 2 un cliente nos propone un proyecto de nombre UdeaStay la cual es una pagina de inmobiliaria y alquiler de alojamientos. Esta debe contar con diferentes características especificas basada en el correcto funcionamiento y uso por parte de un cliente y de un anfitrión (en este caso el anfitrión tendría las cualidades de un admin en la página), cada uno disfrutara de distintas características aunque parecidas como credenciales que los identificaran, vale la aclaración de que todo dato sobre alojamientos huéspedes y anfitriones será dado por el clientes( los profesores)   
  
  
**características importantes:**

Del proyecto se esperan funcionalidades tales que:

* Acceso mediante credenciales tanto de huéspedes como de anfitriones
* Gestión eficiente de los datos
* Búsqueda y creación de reservas (para los huéspedes)
* Cancelación de reservas
* Mantenimiento de un archivo histórico de reservaciones
* Medición del rendimiento del código(para métodos puramente académicos)

**Limitantes o retos:**

El equipo toma como retos o limitaciones situaciones como el no saber el formato en el que vendrán los archivos del dataset proporcionado por los profesores, pensamos que es uno de los retos  
  
otro reto en cuanto al desarrollo es el limitante de uso de la STL para estructuras de datos, es decir estamos supeditados al uso de arreglos dinámicos para el manejo de los diferentes datos del programa haciendo uso de la memoria dinámica siempre viéndolo desde un punto de vista de eficiencia y no redundancia  
  
  
**consideraciones personales:**

Como decisión de grupo y decisión en vista de los requerimientos, procuraremos evitar el uso de temascomo herencia que a pesar de ayudar a la eficiencia delprograma para evitar la redundancia, no es un tema que manejemos en u totalidad además no lo hemos visto en clase, del mismo modo descartamos cualquier uso de métodos virtuales en vita de que no son necesarios y al igual que la herencia son temas que no se han manejado, procuramos mantener básica la arquitectura del programa obviamente cumpliendo los estamentos propuestos por los profesores haciendo solo uso de string como una útil y necesaria herramienta   
  
  
**aclaraciones:**

El diagrama de clases propuesto y su orden obedecen a una visión inicial del problema, esta sujeto a posibles cambios ya que ver el programa y su complejidad completa si una mínima etapa de desarrollo puede ser contraproducente o así lo asumimos, esperamos señirnos a lo propuesto y que lo cambios sean de bajo impacto por ejemplo no implementar mas clases sino usar las ya propuestas que naces de nuestro análisis inicial

**Diagrama de clases (INICIAL):**  
**Diagrama

El contenido generado por IA puede ser incorrecto.**

**Actualización (23/05/2025):**

El desarrollo del desafio se ha dado utilizando QT y el repositorio de git hub, como decisión del arquitecto del diagrama de clases se separo el desarrollo en dos Branch, una enfocada en las clases: anfitrión, huésped y reservaciones.  
la otra enfocada en las clases: fecha, alojamientos y SistemaUdeaStay (que sería el cerebro de todo), esto con el fin de tener un trabajo conjunto pero con reglas establecidas, y mantener la equidad de trabajo.  
  
en el diagrama de clases hasta el momento se hizo un cambio en la clase fecha, que ahora incluye un método “CalcularNombreDiaSemana” a fin de cumplir con el formato de "nombre Día, día 'de' nombreMes 'del' año" )

Se ha definido el formato de los archivos de datos basados en el modelo CSV basado en la separación lógica por medio de comas, considerando 5 archivos:

* Alojamientos.csv
* Anfitriones.csv
* Huéspedes.csv
* Reservaciones.csv
* Histórico.csv

**Respecto al no uso de la STL:**

Para manejar los datos se prohíbe la STL por ende optamos por el uso de arreglos dinámicos, además de el uso de componentes de la biblioteca estándar que no son para manejo de datos por si mismas como **string** para el manejo de cadenas, **iomanip** para formateo y  **sstream** para la construcción de los strings.