

Informe Final Desafío II

Marx Eusdav Lopez Montero, Daniel Antonio Londoño Godoy

Facultad De Ingeniería, Universidad De Antioquia

Informática II: 2598521

28 de mayo de 2025

Introducción.

El presente informe se documenta el desarrollo de un sistema de gestión de alojamientos, reservaciones, anfitriones y huéspedes llamado *UdeAStay*, basado en el paradigma de la Programación Orientada a Objetos. El objetivo principal del proyecto radica en una solución funcional y modular que permita registrar, consultar y administrar información relacionada con alojamientos turísticos.

El sistema se construyó utilizando bibliotecas estándar, organizando el código en múltiples archivos que separan claramente la lógica de este. Los datos se almacenaron en archivos de texto, y su lectura y escritura se realiza mediante flujos de entrada y salida. Además, se implementaron múltiples clases tales como: Alojamiento, Reserva, Huésped, Anfitrión y Fecha, cada uno con sus propios atributos y métodos, lo que permite una representación concisa del problema.

Se optó por el uso de estructuras dinámicas de datos, organizando la información en arreglos dinámicos de punteros para entidades como anfitriones, alojamientos y huéspedes. En el caso específico de las reservas, se utiliza una matriz dinámica (Reserva ***reservas), lo que facilita la gestión tabular y la manipulación eficiente de registros.

Metodología.

Análisis previo.

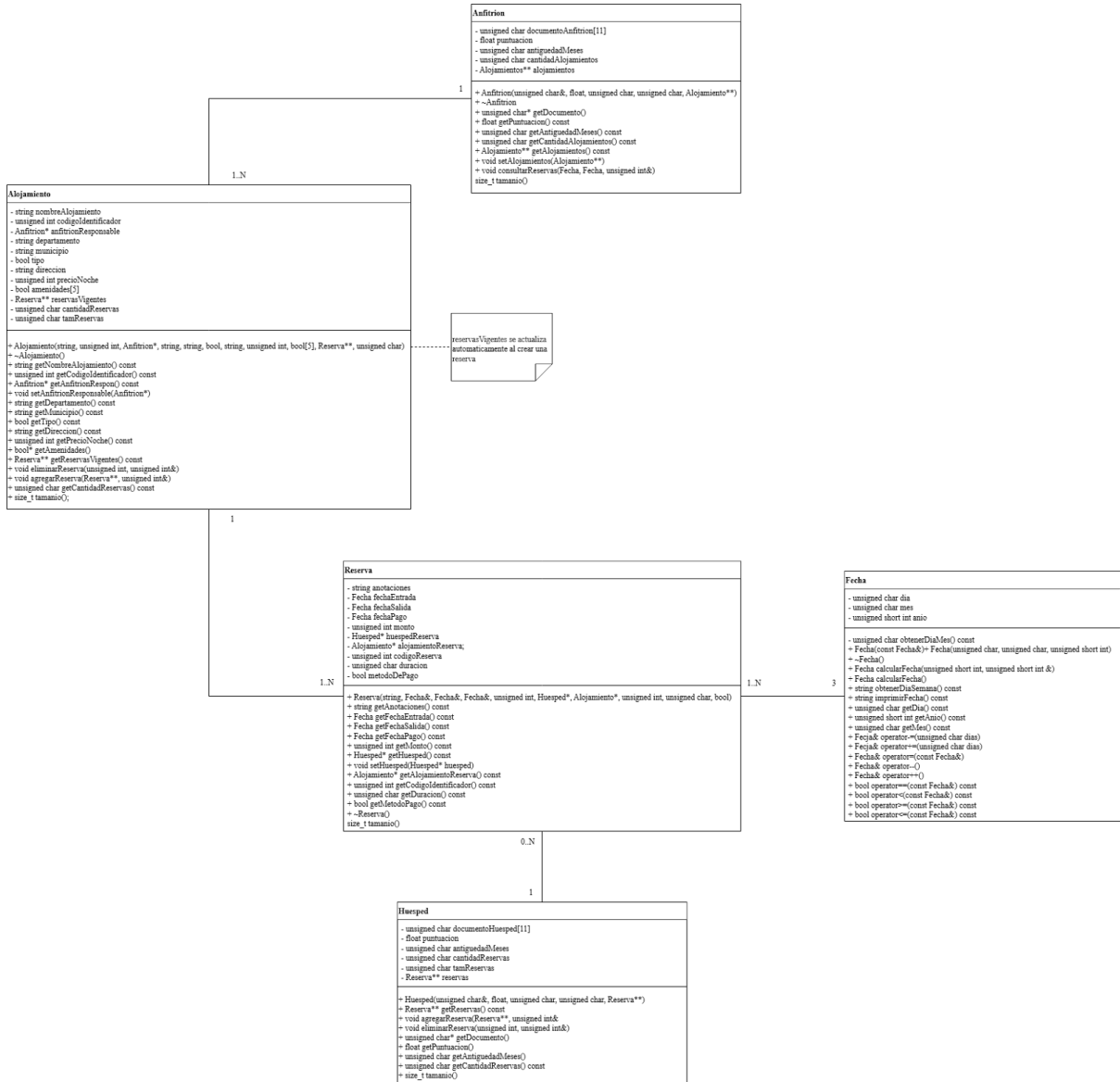
Se delimitaron cinco clases principales: Anfitrión, Huésped, Alojamiento, Reserva y Fecha. Aunque tanto huéspedes como anfitriones comparten atributos como la identificación y el tiempo de permanencia en la plataforma, sus funciones son esencialmente diferentes.

El anfitrión es responsable de la administración de los alojamientos mientras que el huésped se enfoca en gestionar sus reservas. La clase Alojamiento concentra toda la información relevante de una propiedad, incluyendo su ubicación, precio, el conjunto de amenidades y calendario de reservas, lo cual permite evitar duplicación de datos en otras clases.

Por otro lado, la clase Reserva se encarga de validar la disponibilidad de los alojamientos y registrar las operaciones de pago, funciones que requieren una lógica independiente.

Finalmente, la clase Fecha actúa como soporte para el manejo de operaciones temporales, facilitando el cálculo de periodos de reservas y disponibilidad sin comprometer la lógica interna de las clases Reserva o Alojamiento.

De acuerdo con el análisis anterior, se diseñó el diagrama de clases de la siguiente manera:



Implementación.

Se recalcó el trabajo modular, por consiguiente, se describirá brevemente el propósito de cada archivo principal:

- **main.cpp:** Contiene la función principal del programa y coordina la interacción entre las diferentes clases. Gestiona el menú de opciones para anfitriones y huéspedes.
- **huesped.h / huesped.cpp:** Representa al usuario que realiza reservas en los alojamientos. Contiene información personal básica y gestiona sus reservas, permitiendo la interacción con el sistema para consultar y administrar sus estadías.
- **anfitrión.h/anfitrión.h:** Gestiona alojamientos y consulta reservas asociadas a sus propiedades.
- **alojamiento.h / alojamiento.cpp:** Representa propiedades con ubicación, amenidades y precios, además de manejar la relación con las reservas.
- **reserva.h / reserva.cpp:** Contiene la lógica de la clase Reserva, que permite a los huéspedes reservar alojamientos disponibles, calculando fechas y precios.
- **fecha.h / fecha.cpp:** Implementa la clase Fecha, utilizada como apoyo para la comparación de fechas y cálculo de duraciones entre reservas.
- **miscelaneos.h / miscelaneos.cpp:** Incluye funciones auxiliares para manejo de archivos, validaciones o presentación de datos en pantalla.

Los archivos del sistema se organizaron en formato .txt con campos separados por comas y guiones. Esta elección se debe a su simplicidad y facilidad para ser procesados, sin requerir librerías adicionales o estructuras muy complejas. Las comas funcionan como separadores principales entre campos, lo que permite una lectura y escritura ordenada de cada registro. Por otro lado, los guiones se utilizan para delimitar subcampos o listas dentro de un mismo campo.

Además, el manejo de datos del proyecto se dividió en cinco archivos principales: alojamientos, huéspedes, anfitriones, reservas y un archivo histórico. Mantener archivos independientes reduce la complejidad de las operaciones de lectura/escritura, mejora la claridad del almacenamiento, y permite que cada módulo del programa acceda únicamente a la información necesaria, optimizando el uso de memoria y facilitando la escalabilidad futura del sistema.

La matriz de reservas permite una estructura organizada, facilitando el acceso y la actualización de registros sin necesidad de copiar estructuras completas en cada operación. Se emplea una matriz dinámica de punteros para gestionar las reservas, lo que permite expandir la capacidad cuando es necesario sin comprometer la eficiencia. Para otras entidades como anfitriones, alojamientos y huéspedes, se optó por arreglos dinámicos de punteros, en lugar de matrices, debido a su flexibilidad en la gestión de memoria. Estos arreglos permiten ampliar o reducir la cantidad de elementos de manera controlada, sin desperdicio innecesario de recursos ni necesidad de iteraciones adicionales.

Flujo del programa.

Descripción detallada de la lógica o flujo del programa

Al iniciar, el sistema carga la información desde archivos de texto y la almacena en arreglos dinámicos de punteros (Alojamiento **, Huesped **, Anfitrión **) y en una matriz dinámica de reservas (Reserva ***reservas).

Ingreso y autenticación:

El programa presenta un menú inicial que pide la credencial del usuario para distinguir entre anfitrión o huésped.

Flujo para Huésped:

El huésped puede consultar los alojamientos disponibles y realizar reservas. Para ello, el sistema:

- Solicita los parámetros de búsqueda
- Recorre la matriz de alojamientos y verifica la disponibilidad usando la clase Fecha y el calendario de reservas de cada alojamiento.
- Permite seleccionar un alojamiento disponible y registra una nueva reserva, actualizando la matriz de reservas y las fechas ocupadas del alojamiento.
- Calcula el costo total basado en el precio por noche y duración, registrando el pago.
- Permite al huésped cancelar reservas existentes, actualizando la información en memoria y sincronizando con los archivos.

Flujo para Anfitrión:

El anfitrión administra sus alojamientos y puede revisar las reservas existentes:

- Consulta su lista de alojamientos a través de su arreglo dinámico
- Visualiza detalles como ubicación, precio y fechas reservadas.
- Consulta reservas en un rango de fechas para evaluar ocupación y gestionar disponibilidad.

Operaciones comunes :

La clase Fecha se utiliza para comparar, validar y calcular intervalos de tiempo entre reservas, facilitando la verificación de solapamientos y el control de disponibilidad. La clase Reserva gestiona la información de pago y duración de la reserva, asegurando que los registros se mantengan coherentes.

Manejo actualizaciones:

Después de cada operación que modifica los datos, los arreglos dinámicos de punteros

y la matriz de reservas se vuelcan a los archivos de texto mediante funciones de guardado. Esto garantiza la persistencia y permite recuperar la información en futuras ejecuciones.