#### **1. Análisis del problema y consideraciones para la alternativa de solución propuesta**

UdeAStay es una plataforma para la gestión de alojamientos tipo homestay. El sistema debe permitir la administración de alojamientos por parte de anfitriones y facilitar la reserva de estos por parte de los usuarios huéspedes.

Las consideraciones principales fueron:

* **No uso de STL**, únicamente tipos primitivos y string.
* Lectura de archivos **línea por línea**.
* Uso de **memoria dinámica**.
* Separación clara entre **administrador** y **usuario** (rama paralela).
* Manejo y comparación de fechas usando una clase fecha.

#### **2. Esquema de desarrollo (tareas definidas)**

**Diseño de clases básicas**:

* + administrador
  + lugar
  + fecha
  + usuarios

**Implementación del login único** desde un archivo administradores.txt y usuarios.txt.

**Manejo de archivos**:

* + lugares.txt: información de alojamientos.
  + reservas\_activas.txt: reservas vigentes.
  + historico.txt: reservas expiradas.

**Funcionalidades para el administrador**:

* + Visualización de alojamientos propios.
  + Visualización y cancelación de reservas.
  + Actualización del histórico con verificación de fechas.

**Funcionalidades para el usuario**.

* + Hacer reservas..
  + Visualización y cancelación de reservas.

#### **3. Algoritmos implementados**

* **Lectura filtrada de lugares** por documento del administrador.
* **Comparación de fechas** usando sobrecarga de operadores en la clase fecha.
* **Filtrado y copia de líneas entre archivos** con reserva dinámica.
* **Transformación de fecha a formato largo**: "Martes, 15 de mayo del 2025".

#### **4. Problemas de desarrollo afrontados**

* El uso restringido de STL obligó a diseñar mecanismos manuales de parsing (getline, separadores, contadores).
* Control de errores de lectura de fechas a partir de strings (sscanf generaba errores inesperados).
* Manejo cuidadoso de memoria dinámica (evitar fugas).
* Evitar colisiones y conflictos entre ramas (main vs administrador).
* Validación de campos en archivos con múltiples administradores y reservas.

#### **5. Evolución de la solución y consideraciones**

* Inicialmente, se pensó cargar todos los alojamientos, pero se refinó para que solo se cargaran los del administrador actual.
* Se eliminó el uso de constantes estáticas para tamaños de arreglos, sustituyéndolas por conteo previo dinámico.
* El menú principal fue reestructurado para que todo el flujo del programa esté dentro de un bucle principal, permitiendo múltiples logins.
* La clase fecha se convirtió en una herramienta crucial para la validación de historial y futura gestión de reservas de usuarios.