# Prueba - Roommates

- Para realizar esta prueba debes haber estudiado previamente todo el material disponible en el LMS correspondiente al módulo.
- Una vez terminada la prueba, comprime la carpeta que contiene el desarrollo de los requerimientos solicitados y sube el *.zip* en el LMS.
- Puntaje total: 10 puntos
- Desarrollo prueba:
    - La prueba se debe resolver de manera Individual
    - Para la realización de la prueba necesitarás el archivo disponible en el LMS llamado *Apoyo Prueba - Roommates.*

## Habilidades a evaluar

- Manipular archivos con File System
- Manejar errores
- Construir una API RESTful
- Manejar códigos de estado HTTP
- Utilizar paquetes de npm

## Descripción

Es bien sabido que entre las mejores recomendaciones que un programador amateur puede recibir para mejorar sus habilidades es “crear aplicaciones”, sin darle tanta importancia a la temática a elaborar, sino que solo basta con un problema para desarrollar una solución digital.

En esta prueba deberás crear un servidor con Node que sirva una interfaz HTML que tendrás a disposición en el **Apoyo Prueba - Roommates** y cuya temática está basada en el registro de gastos entre roommates.

Además deberás servir una API REST que permita hacer lo siguiente:

- Almacenar roommates nuevos ocupando random user.
- Devolver todos los roommates almacenados.
- Registrar nuevos gastos.
- Devolver el historial de gastos registrados.
- Modificar la información correspondiente a un gasto.

- Eliminar gastos del historial.

A continuación se muestra una imagen con la interfaz que deberás devolver en la ruta raíz del servidor:

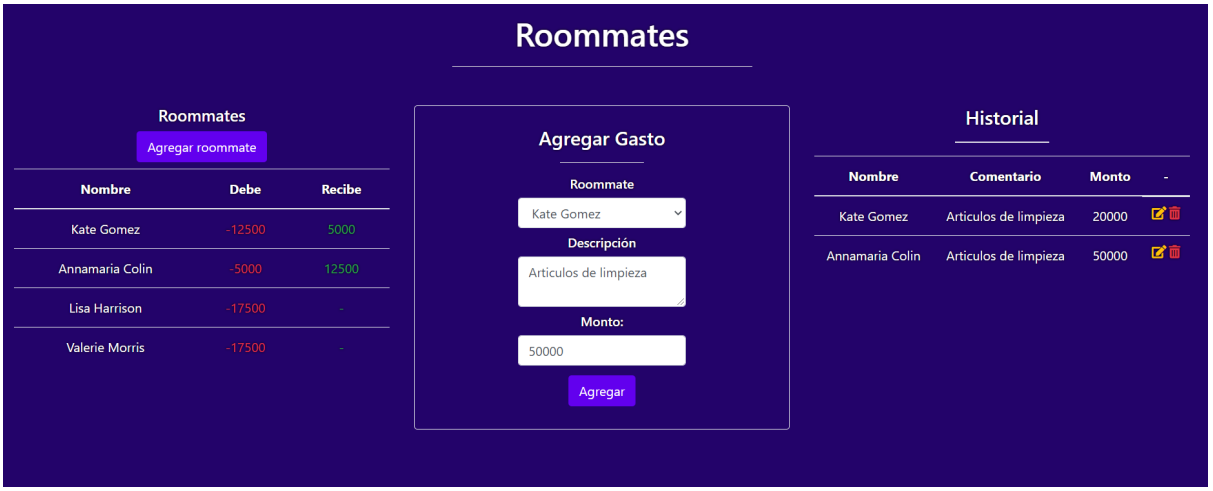


Imagen 1. Interfaz.
Fuente: Desafío Latam

Rutas que debes crear en tu servidor:

- **/ GET:** Debe devolver el documento HTML disponibilizado en el apoyo.
- **/roommate POST:** Almacena un nuevo roommate ocupando random user.
- **/roommate GET:** Devuelve todos los roommates almacenados.
- **/gastos GET:** Devuelve el historial con todos los gastos registrados.
- **/gasto PUT:** Edita los datos de un gasto.
- **/gasto DELETE:** Elimina un gasto del historial.

## Requerimientos

1. Ocupar el módulo File System para la manipulación de archivos alojados en el servidor **(3 Puntos)**

2. Capturar los errores para condicionar el código a través del manejo de excepciones. **(1 Punto)**

3. El botón "Agregar roommate" de la aplicación cliente genera una petición POST (sin payload) esperando que el servidor registre un nuevo roommate random con la API randomuser, por lo que debes preparar una ruta **POST /roommate** en el servidor que ejecute una función asíncrona importada de un archivo externo al del servidor (la función debe ser un módulo), para obtener la data de un nuevo usuario y la acumule en un JSON (roommates.json).

   El objeto correspondiente al usuario que se almacenará debe tener un id generado con el paquete UUID. **(2 Puntos)**

4. Crear una API REST que contenga las siguientes rutas:

   a. **GET /gastos:** Devuelve todos los gastos almacenados en el archivo gastos.json.

   b. **POST /gasto:** Recibe el payload con los datos del gasto y los almacena en un archivo JSON (gastos.json).

   c. **PUT /gasto**: Recibe el payload de la consulta y modifica los datos almacenados en el servidor (gastos.json).

   d. **DELETE /gasto:** Recibe el id del gasto usando las Query Strings y la elimine del historial de gastos (gastos.json).

   e. **GET /roommates:** Devuelve todos los roommates almacenados en el servidor (roommates.json)

   Se debe considerar recalcular y actualizar las cuentas de los roommates luego de este proceso. **(3 Puntos)**

5. Devolver los códigos de estado HTTP correspondientes a cada situación. **(1 Punto)**

6. Enviar un correo electrónico a todos los roommates cuando se registre un nuevo gasto. Se recomienda agregar a la lista de correos su correo personal para verificar esta funcionalidad. **(Opcional)**