Diagrama de secuencia: Inicio de sesión

Caminos alternativos y excepciones:

1. Credenciales incorrectas: Firebase devuelve un error y la app muestra un mensaje al usuario.
2. Fallo de conexión: La app muestra un mensaje de "Error de conexión. Intenta nuevamente."
3. Usuario no ingresa datos: La app no envía la solicitud y marca los campos vacíos.

Diagrama de secuencia

1. Actor: Usuario.
2. Componentes: Login.js, Firebase Authentication.

Flujo principal

* Usuario → Login.js: Ingresa email y contraseña.
* Login.js → Firebase: Solicitud de autenticación.
* Firebase → Login.js: Respuesta exitosa con token.
* Login.js → Usuario: Redirige al Dashboard.

Caminos alternativos:

Si las credenciales son incorrectas:

* Firebase → Login.js: Devuelve un error de autenticación.
* Login.js → Usuario: Muestra "Credenciales incorrectas."

Si hay un fallo de conexión:

* Firebase no responde.
* Login.js → Usuario: Muestra "Error de conexión. Intenta nuevamente."

Si los datos están incompletos:

* Login.js → Usuario: Muestra "Completa todos los campos."

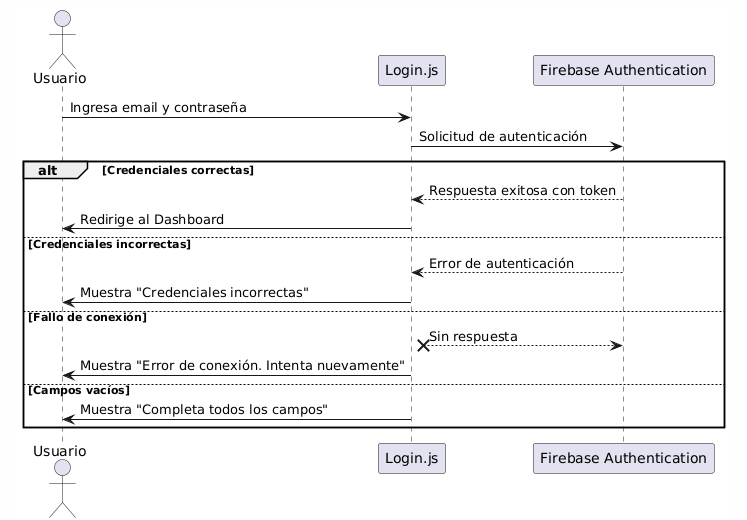


Diagrama de secuencia: registro de alimentos

Caminos alternativos y excepciones

1. Usuario deja campos vacíos o ingresa datos inválidos: La app valida y marca los errores.
2. Fallo en la conexión al backend: La app reintenta hasta 3 veces antes de notificar el error.
3. Fallo al almacenar en la base de datos: El backend devuelve un error y la app notifica al usuario.

Diagrama de Secuencia

1. Actor: Usuario.
2. Componentes: RegisterFood.js, Backend, Base de datos.

Flujo principal

* Usuario → RegisterFood.js: Completa los campos y presiona "Guardar".
* RegisterFood.js → Backend: Envía datos.
* Backend → Base de datos: Almacena los datos.
* Base de datos → Backend: Confirma éxito.
* Backend → RegisterFood.js: Respuesta exitosa.
* RegisterFood.js → Usuario: Muestra "Registro exitoso."

Caminos alternativos

Si los datos están vacíos/incorrectos:

* RegisterFood.js → Usuario: Muestra "Completa los campos requeridos."

Si hay un fallo de conexión:

* Backend no responde.
* RegisterFood.js → Usuario: Muestra "Error de conexión. Intenta nuevamente."

Si la base de datos falla:

* Base de datos → Backend: Devuelve un error.
* Backend → RegisterFood.js: Notifica fallo.
* RegisterFood.js → Usuario: Muestra "Error al guardar los datos."

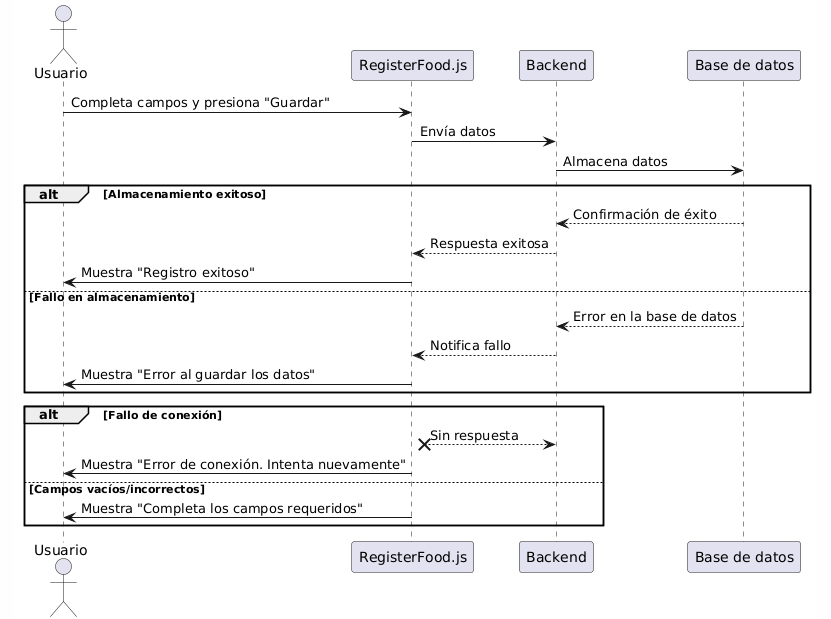


Diagrama de secuencia: Dashboard de macronutrientes

Caminos alternativos y excepciones:

1. Fallo al obtener datos del backend: La app muestra un mensaje de "Error de conexión."
2. Datos incompletos o corruptos: La app muestra "Datos no disponibles."
3. Usuario intenta alternar vistas sin datos cargados: Muestra "Cargando datos."

Diagrama de Secuencia

1. Actor: Usuario.
2. Componentes: Dashboard.js, Backend, Base de datos.

Flujo principal:

* Usuario → Dashboard.js: Abre la pantalla.
* Dashboard.js → Backend: Solicita datos de macronutrientes.
* Backend → Base de datos: Consulta los datos.
* Base de datos → Backend: Devuelve los datos.
* Backend → Dashboard.js: Respuesta exitosa.
* Dashboard.js → Usuario: Muestra el Dashboard actualizado.

Caminos alternativos

Si hay un fallo de conexión:

* Backend no responde.
* Dashboard.js → Usuario: Muestra "Error de conexión."

Si los datos están incompletos:

* Base de datos → Backend: Devuelve datos inconsistentes.
* Backend → Dashboard.js: Notifica fallo.
* Dashboard.js → Usuario: Muestra "Datos no disponibles."

Si el usuario intenta alternar vistas:

* Dashboard.js → Usuario: Muestra "Cargando datos."

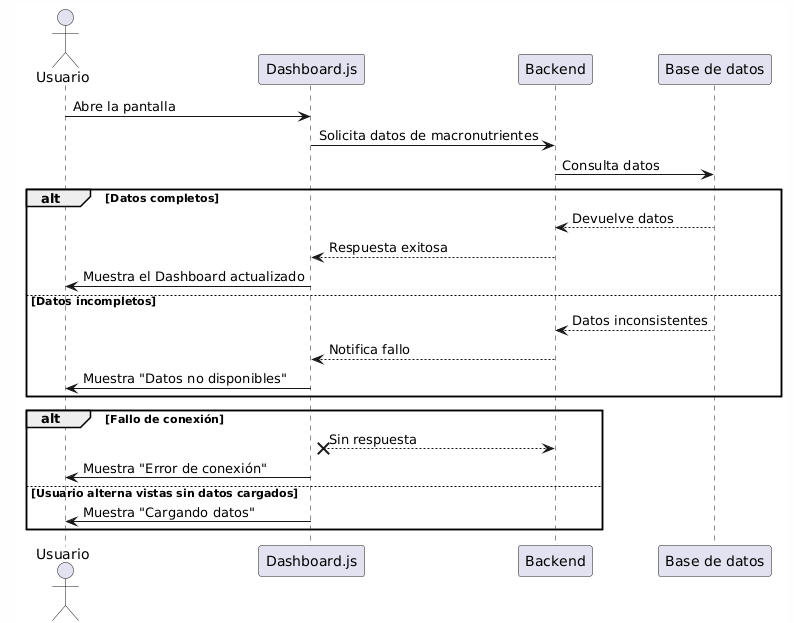


Diagrama de secuencia: Generar recomendaciones

Caminos alternativos y excepciones:

1. Fallo en la lógica del backend: El sistema devuelve un error y la app muestra un mensaje.
2. Usuario intenta acceder sin datos suficientes: Muestra "No hay datos suficientes para generar recomendaciones."

Diagrama de Secuencia

1. Actor: Usuario.
2. Componentes: Dashboard.js, Backend, Base de datos.

Flujo principal:

* Usuario → Dashboard.js: Presiona "Recomendaciones."
* Dashboard.js → Backend: Solicita datos.
* Backend → Base de datos: Consulta datos de consumo.
* Base de datos → Backend: Devuelve los datos.
* Backend → Dashboard.js: Genera y devuelve recomendaciones.
* Dashboard.js → Usuario: Muestra las recomendaciones.

Caminos alternativos:

Si no hay datos suficientes:

* Base de datos → Backend: Devuelve error.
* Backend → Dashboard.js: Notifica fallo.
* Dashboard.js → Usuario: Muestra "No hay datos suficientes."

Si el backend falla:

* Backend → Dashboard.js: Devuelve error.
* Dashboard.js → Usuario: Muestra "Error al generar recomendaciones."

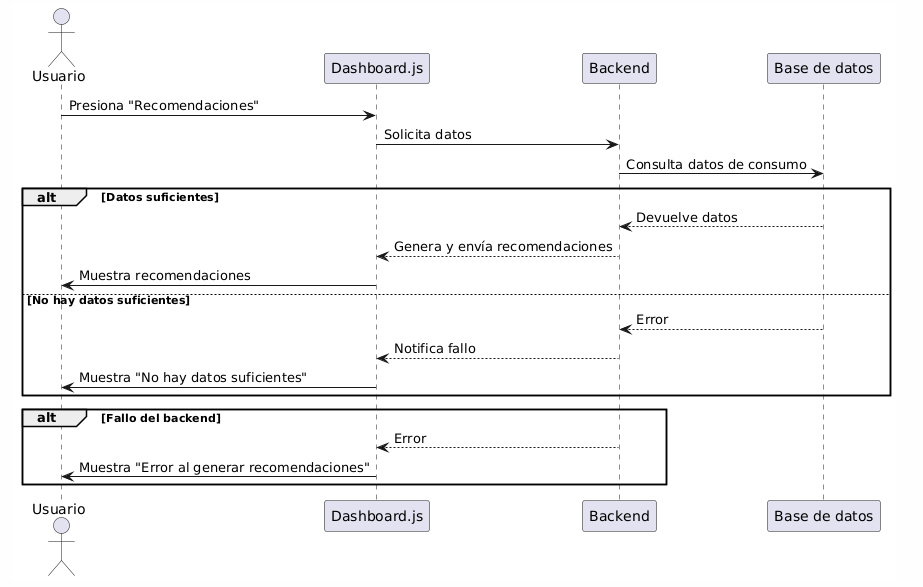


Diagrama de secuencia: Creación de recetas

Flujo Principal

1. El Usuario accede a la pantalla RecipeCreation.js.
2. Ingresa el nombre de la receta y selecciona los ingredientes necesarios de una lista.

Verificación de Ingredientes:

1. La pantalla RecipeCreation.js envía la lista de ingredientes seleccionados al Backend.
2. El Backend consulta la Base de datos para verificar si todos los ingredientes existen.

Almacenamiento de la Receta:

Si todos los ingredientes existen, el Backend:

1. Envía una solicitud para guardar la receta en la Base de datos.
2. Recibe una confirmación de éxito.

La pantalla RecipeCreation.js informa al usuario con el mensaje: "Receta creada exitosamente."

Caminos Alternativos

Ingredientes Faltantes: Uno o más ingredientes seleccionados por el usuario no están registrados en la base de datos.

Flujo:

1. La Base de datos devuelve una lista con los ingredientes faltantes.
2. El Backend informa a RecipeCreation.js sobre los ingredientes que no existen.
3. RecipeCreation.js muestra un mensaje al usuario: "Faltan ingredientes: [lista de ingredientes]."
4. El usuario puede optar por registrar los ingredientes faltantes antes de continuar.

Fallo de conexión con la base de datos:

El Backend no puede comunicarse con la base de datos por problemas de red o servidor.

Flujo:

1. El Backend devuelve un error de conexión a RecipeCreation.js.
2. RecipeCreation.js muestra el mensaje: "Error de conexión. Intente nuevamente."
3. El sistema permite que el usuario reintente la operación después de un tiempo.

Caminos de excepción: El usuario intenta guardar una receta sin seleccionar ningún ingrediente.

Flujo:

1. RecipeCreation.js detecta que no hay ingredientes seleccionados.
2. Muestra el mensaje: "Debe seleccionar al menos un ingrediente para guardar la receta."

Fallo al Almacenar la Receta: Un problema ocurre mientras se intenta guardar la receta en la base de datos.

Flujo:

1. La Base de datos devuelve un error al Backend.
2. El Backend informa a RecipeCreation.js sobre el fallo.

RecipeCreation.js muestra el mensaje: "No se pudo guardar la receta. Intente nuevamente."

