

# Index

A B C D E F G H I J L M O P R S T U V W

All Classes and Interfaces | All Packages | Serialized Form

## A

**AccessDeniedException** - Exception Class in `com.mapmyjourney.backend.exception`  
Excepción para acceso denegado (falta de permisos).

**AccessDeniedException(String)** - Constructor for exception class `com.mapmyjourney.backend.exception.AccessDeniedException`

**AccessDeniedException(String, Throwable)** - Constructor for exception class `com.mapmyjourney.backend.exception.AccessDeniedException`

**Activity** - Class in `com.mapmyjourney.backend.model`  
Entidad que representa una actividad en un itinerario de viaje.

**Activity()** - Constructor for class `com.mapmyjourney.backend.model.Activity`

**ACTIVITY** - Enum constant in enum class `com.mapmyjourney.backend.model.ActivityType`

**ActivityController** - Class in `com.mapmyjourney.backend.controller`  
Controlador REST para gestionar las actividades del itinerario de un viaje.

**ActivityController()** - Constructor for class `com.mapmyjourney.backend.controller.ActivityController`

**ActivityDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para la entidad Activity.

**ActivityDTO()** - Constructor for class `com.mapmyjourney.backend.dto.ActivityDTO`

**ActivityRepository** - Interface in `com.mapmyjourney.backend.repository`  
Repositorio para la entidad Activity.

**ActivityService** - Class in `com.mapmyjourney.backend.service`  
Servicio para gestionar las operaciones de las actividades.

**ActivityService()** - Constructor for class `com.mapmyjourney.backend.service.ActivityService`

**ActivityType** - Enum Class in `com.mapmyjourney.backend.model`  
Enum que representa los tipos de actividades disponibles en un itinerario.

**addCorsMappings(CorsRegistry)** - Method in class `com.mapmyjourney.backend.config.CorsConfig`  
Configura los parámetros CORS para todas las rutas.

**addMember(Long, AddMemberRequestDTO)** - Method in class `com.mapmyjourney.backend.controller.TripMemberController`  
1.

**AddMemberRequestDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para agregar un nuevo miembro a un viaje.

**AddMemberRequestDTO()** - Constructor for class `com.mapmyjourney.backend.dto.AddMemberRequestDTO`

**addMemberToTrip(Long, Long)** - Method in class `com.mapmyjourney.backend.service.TripService`  
Agrega un usuario a un viaje con rol VIEWER.

**addMemberToTrip(Long, Long, TripMemberRole)** - Method in class `com.mapmyjourney.backend.service.TripMemberService`  
Agrega un nuevo miembro a un viaje.

**ADMIN** - Enum constant in enum class `com.mapmyjourney.backend.model.enums.UserRole`

**authenticate(String, String)** - Method in class `com.mapmyjourney.backend.service.UserService`  
6.

**authenticationManager(HttpSecurity)** - Method in class `com.mapmyjourney.backend.security.SecurityConfig`  
Configura el AuthenticationManager con DaoAuthenticationProvider.

**authHealth()** - Method in class `com.mapmyjourney.backend.controller.HealthController`  
Verifica que el servicio de autenticación está disponible

## B

**BackendApplication** - Class in `com.mapmyjourney.backend`  
Clase principal para iniciar la aplicación Spring Boot.

**BackendApplication()** - Constructor for class `com.mapmyjourney.backend.BackendApplication`

## C

**changeMemberRole(Long, Long, ChangeMemberRoleRequestDTO)** - Method in class `com.mapmyjourney.backend.controller.TripMemberController`  
4.

**changeMemberRole(Long, Long, TripMemberRole)** - Method in class `com.mapmyjourney.backend.service.TripMemberService`  
Cambia el rol de un miembro en un viaje.

**ChangeMemberRoleRequestDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para cambiar el rol de un miembro en un viaje.

**ChangeMemberRoleRequestDTO()** - Constructor for class `com.mapmyjourney.backend.dto.ChangeMemberRoleRequestDTO`

com.mapmyjourney.backend - package com.mapmyjourney.backend

com.mapmyjourney.backend.config - package com.mapmyjourney.backend.config

com.mapmyjourney.backend.controller - package com.mapmyjourney.backend.controller

com.mapmyjourney.backend.dto - package com.mapmyjourney.backend.dto

com.mapmyjourney.backend.exception - package com.mapmyjourney.backend.exception

com.mapmyjourney.backend.model - package com.mapmyjourney.backend.model

com.mapmyjourney.backend.model.enums - package com.mapmyjourney.backend.model.enums

com.mapmyjourney.backend.repository - package com.mapmyjourney.backend.repository

com.mapmyjourney.backend.security - package com.mapmyjourney.backend.security

com.mapmyjourney.backend.service - package com.mapmyjourney.backend.service

**CorsConfig** - Class in com.mapmyjourney.backend.config

Configuración de CORS (Cross-Origin Resource Sharing) Permite que el frontend pueda hacer solicitudes al backend CORS permite solicitudes desde: - http://localhost:4200 (desarrollo Angular local) - http://localhost:3000 (desarrollo Node local) - https://mapmyjourney-4w93.onrender.com (producción)

**CorsConfig()** - Constructor for class com.mapmyjourney.backend.config.CorsConfig

**corsConfigurationSource()** - Method in class com.mapmyjourney.backend.security.SecurityConfig

Configura CORS para permitir solicitudes desde el frontend en localhost y dominios Render.

**corsFilter()** - Method in class com.mapmyjourney.backend.config.WebConfig

Bean de CorsFilter - Ejecuta ANTES que Security Filters Esto es crítico para manejar preflight correctamente

**createActivity(Long, CreateActivityRequestDTO)** - Method in class com.mapmyjourney.backend.controller.ActivityController

Crea una nueva actividad en un viaje.

**createActivity(Long, CreateActivityRequestDTO)** - Method in class com.mapmyjourney.backend.service.ActivityService

Crea una nueva actividad en un viaje.

**CreateActivityRequestDTO** - Class in com.mapmyjourney.backend.dto

DTO para crear una nueva actividad.

**CreateActivityRequestDTO()** - Constructor for class com.mapmyjourney.backend.dto.CreateActivityRequestDTO

**createExpense(Long, ExpenseCreateRequestDTO)** - Method in class com.mapmyjourney.backend.controller.ExpenseController  
1.

**createExpense(Long, ExpenseCreateRequestDTO, Long)** - Method in class com.mapmyjourney.backend.service.ExpenseService  
1.

**createSplit(Long, CreateSplitRequestDTO)** - Method in class com.mapmyjourney.backend.controller.ExpenseSplitController  
1.

**createSplit(Long, Long, BigDecimal, BigDecimal)** - Method in class com.mapmyjourney.backend.service.ExpenseSplitService  
Crea una nueva división de gasto.

**CreateSplitRequestDTO** - Class in com.mapmyjourney.backend.dto

DTO para crear una nueva división de gasto.

**CreateSplitRequestDTO()** - Constructor for class com.mapmyjourney.backend.dto.CreateSplitRequestDTO

**createTrip(TripCreateRequestDTO)** - Method in class com.mapmyjourney.backend.controller.TripController  
1.

**createTrip(TripCreateRequestDTO, Long)** - Method in class com.mapmyjourney.backend.service.TripService  
Crea un nuevo viaje.

**CUSTOM** - Enum constant in enum class com.mapmyjourney.backend.model.enums.ExpenseSplitType

**customOpenAPI()** - Method in class com.mapmyjourney.backend.config.OpenApiConfig

Crea la configuración personalizada de OpenAPI.

**CustomUserDetailsService** - Class in com.mapmyjourney.backend.security

Servicio para cargar detalles de usuario desde la base de datos.

**CustomUserDetailsService()** - Constructor for class com.mapmyjourney.backend.security.CustomUserDetailsService

## D

**daoAuthenticationProvider()** - Method in class com.mapmyjourney.backend.security.SecurityConfig  
Configura el proveedor de autenticación DAO.

**DatabaseSeeder** - Class in com.mapmyjourney.backend.config

Executes the SQL statements contained in data.sql when the database is empty.

**DatabaseSeeder(JdbcTemplate, DataSource, Environment)** - Constructor for class com.mapmyjourney.backend.config.DatabaseSeeder

**deleteActivity(Long, String)** - Method in class com.mapmyjourney.backend.controller.ActivityController  
Elimina una actividad.

**deleteActivity(Long, String)** - Method in class com.mapmyjourney.backend.service.ActivityService  
Elimina una actividad.

**deleteExpense(Long, Long)** - Method in class com.mapmyjourney.backend.controller.ExpenseController

5.

**deleteExpense(Long, Long)** - Method in class `com.mapmyjourney.backend.service.ExpenseService`  
Elimina un gasto.

**deleteSplit(Long)** - Method in class `com.mapmyjourney.backend.service.ExpenseSplitService`  
Elimina una división de gasto.

**deleteSplit(Long, Long)** - Method in class `com.mapmyjourney.backend.controller.ExpenseSplitController`  
6.

**deleteTrip(Long)** - Method in class `com.mapmyjourney.backend.controller.TripController`  
6.

**deleteTrip(Long, Long)** - Method in class `com.mapmyjourney.backend.service.TripService`  
Elimina un viaje.

**deleteUser(Long)** - Method in class `com.mapmyjourney.backend.controller.UserController`  
7.

**deleteUser(Long)** - Method in class `com.mapmyjourney.backend.service.UserService`  
5.

**doFilterInternal(HttpServletRequest, HttpServletResponse, FilterChain)** - Method in class `com.mapmyjourney.backend.security.JwtAuthenticationFilter`

**DuplicateResourceException** - Exception Class in `com.mapmyjourney.backend.exception`  
Excepción para operaciones duplicadas (ej: usuario ya existe).

**DuplicateResourceException(String)** - Constructor for exception class  
`com.mapmyjourney.backend.exception.DuplicateResourceException`

**DuplicateResourceException(String, Throwable)** - Constructor for exception class  
`com.mapmyjourney.backend.exception.DuplicateResourceException`

## E

**EDITOR** - Enum constant in enum class `com.mapmyjourney.backend.model.enums.TripMemberRole`

**EQUAL** - Enum constant in enum class `com.mapmyjourney.backend.model.enums.ExpenseSplitType`

**ErrorResponseDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para respuestas de error de la API.

**ErrorResponseDTO()** - Constructor for class `com.mapmyjourney.backend.dto.ErrorResponseDTO`

**ExampleController** - Class in `com.mapmyjourney.backend.controller`  
Controlador de ejemplo para Swagger

**ExampleController()** - Constructor for class `com.mapmyjourney.backend.controller.ExampleController`

**existsByEmail(String)** - Method in interface `com.mapmyjourney.backend.repository.UserRepository`  
Verifica si existe un usuario con ese email.

**existsByTripIdAndActivityDate(Long, LocalDate)** - Method in interface `com.mapmyjourney.backend.repository.ActivityRepository`  
Verifica si existe una actividad para un viaje y una fecha.

**Expense** - Class in `com.mapmyjourney.backend.model`  
Entidad que representa un gasto en un viaje.

**Expense()** - Constructor for class `com.mapmyjourney.backend.model.Expense`

**ExpenseController** - Class in `com.mapmyjourney.backend.controller`  
Controlador REST para gestionar gastos en viajes.

**ExpenseController()** - Constructor for class `com.mapmyjourney.backend.controller.ExpenseController`

**ExpenseCreateRequestDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para crear un nuevo gasto.

**ExpenseCreateRequestDTO()** - Constructor for class `com.mapmyjourney.backend.dto.ExpenseCreateRequestDTO`

**ExpenseDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para información completa de un gasto.

**ExpenseDTO()** - Constructor for class `com.mapmyjourney.backend.dto.ExpenseDTO`

**ExpenseRepository** - Interface in `com.mapmyjourney.backend.repository`

**ExpenseService** - Class in `com.mapmyjourney.backend.service`

**ExpenseService()** - Constructor for class `com.mapmyjourney.backend.service.ExpenseService`

**ExpenseSplit** - Class in `com.mapmyjourney.backend.model`  
Entidad que representa la división de un gasto entre un usuario.

**ExpenseSplit()** - Constructor for class `com.mapmyjourney.backend.model.ExpenseSplit`

**ExpenseSplitController** - Class in `com.mapmyjourney.backend.controller`  
Controlador REST para gestionar divisiones de gastos.

**ExpenseSplitController()** - Constructor for class `com.mapmyjourney.backend.controller.ExpenseSplitController`

**ExpenseSplitDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para una división de gasto.

**ExpenseSplitDTO()** - Constructor for class `com.mapmyjourney.backend.dto.ExpenseSplitDTO`

**ExpenseSplitRepository** - Interface in `com.mapmyjourney.backend.repository`

**ExpenseSplitService** - Class in `com.mapmyjourney.backend.service`

Servicio para gestionar las divisiones de gastos.

**ExpenseSplitService()** - Constructor for class `com.mapmyjourney.backend.service.ExpenseSplitService`

**ExpenseSplitType** - Enum Class in `com.mapmyjourney.backend.model.enums`

Enum que define cómo se divide un gasto entre los participantes.

**extractEmail(String)** - Method in class `com.mapmyjourney.backend.security.JwtTokenProvider`

Extrae el email (subject) de un token JWT válido.

## F

**filterChain(HttpSecurity)** - Method in class `com.mapmyjourney.backend.security.SecurityConfig`

Configura la cadena de filtros de seguridad HTTP.

**findAllByTripId(Long)** - Method in interface `com.mapmyjourney.backend.repository.TripMemberRepository`

Obtiene todos los miembros de un viaje.

**findAllByUserId(Long)** - Method in interface `com.mapmyjourney.backend.repository.TripRepository`

Obtiene todos los viajes donde el usuario es miembro.

**findByEmail(String)** - Method in interface `com.mapmyjourney.backend.repository.UserRepository`

Busca un usuario por su email.

**findByExpenseId(Long)** - Method in interface `com.mapmyjourney.backend.repository.ExpenseSplitRepository`

Obtiene todas las divisiones de un gasto.

**findByIdAndTripId(String, Long)** - Method in interface `com.mapmyjourney.backend.repository.ActivityRepository`

Obtiene una actividad por ID y verifica que pertenece a un viaje específico.

**findByParticipantIdAndPaidFalse(Long)** - Method in interface `com.mapmyjourney.backend.repository.ExpenseSplitRepository`

Obtiene todas las divisiones no pagadas de un usuario.

**findByTripCode(String)** - Method in interface `com.mapmyjourney.backend.repository.TripRepository`

Busca un viaje por su código único (para invitaciones).

**findByTripId(Long)** - Method in interface `com.mapmyjourney.backend.repository.ActivityRepository`

Obtiene todas las actividades de un viaje, ordenadas por fecha y hora.

**findByTripId(Long)** - Method in interface `com.mapmyjourney.backend.repository.ExpenseRepository`

Obtiene todos los gastos de un viaje.

**findByTripIdAndDate(Long, LocalDate)** - Method in interface `com.mapmyjourney.backend.repository.ActivityRepository`

Obtiene las actividades de un viaje para una fecha específica.

**findByTripIdAndDateRange(Long, LocalDate, LocalDate)** - Method in interface

`com.mapmyjourney.backend.repository.ActivityRepository`

Obtiene las actividades de un viaje para un rango de fechas.

**findByTripIdAndDayIndex(Long, Integer)** - Method in interface `com.mapmyjourney.backend.repository.ActivityRepository`

Obtiene las actividades de un viaje para un día índice específico.

**findByTripIdAndUserId(Long, Long)** - Method in interface `com.mapmyjourney.backend.repository.TripMemberRepository`

Obtiene el rol de un usuario en un viaje.

**findMaxOrderByTripIdAndDate(Long, LocalDate)** - Method in interface `com.mapmyjourney.backend.repository.ActivityRepository`

Obtiene el máximo orden de una actividad para una fecha.

## G

**generateToken(String)** - Method in class `com.mapmyjourney.backend.security.JwtTokenProvider`

Genera un token JWT para un email específico.

**generateToken(Authentication)** - Method in class `com.mapmyjourney.backend.security.JwtTokenProvider`

Genera un token JWT a partir de una Authentication de Spring.

**getActivitiesByDate(Long, LocalDate)** - Method in class `com.mapmyjourney.backend.controller.ActivityController`

Obtiene las actividades de un viaje para una fecha específica.

**getActivitiesByDate(Long, LocalDate)** - Method in class `com.mapmyjourney.backend.service.ActivityService`

Obtiene las actividades de un viaje para una fecha específica.

**getActivitiesByTrip(Long)** - Method in class `com.mapmyjourney.backend.controller.ActivityController`

Obtiene todas las actividades de un viaje.

**getActivitiesByTrip(Long)** - Method in class `com.mapmyjourney.backend.service.ActivityService`

Obtiene todas las actividades de un viaje.

**getActivity(Long, String)** - Method in class `com.mapmyjourney.backend.controller.ActivityController`

Obtiene una actividad específica.

**getActivity(Long, String)** - Method in class `com.mapmyjourney.backend.service.ActivityService`

Obtiene una actividad específica.

**getBudget()** - Method in class `com.mapmyjourney.backend.dto.TripCreateRequestDTO`

**getCurrentPassword()** - Method in class `com.mapmyjourney.backend.dto.UserUpdateRequestDTO`

**getDescription()** - Method in class `com.mapmyjourney.backend.dto.TripCreateRequestDTO`

**getDescription()** - Method in enum class `com.mapmyjourney.backend.model.enums.TripMemberRole`

**getDestination()** - Method in class `com.mapmyjourney.backend.dto.TripCreateRequestDTO`

**getDisplayName()** - Method in enum class `com.mapmyjourney.backend.model.ActivityType`

**getDisplayName()** - Method in enum class com.mapmyjourney.backend.model.enums.ExpenseSplitType

**getDisplayName()** - Method in enum class com.mapmyjourney.backend.model.enums.UserRole

**getEmail()** - Method in class com.mapmyjourney.backend.dto.LoginRequestDTO

**getEmail()** - Method in class com.mapmyjourney.backend.dto.UserCreateRequestDTO

**getEmail()** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**getEndDate()** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**getExpense(Long, Long)** - Method in class com.mapmyjourney.backend.controller.ExpenseController  
3.

**getExpenseByld(Long)** - Method in class com.mapmyjourney.backend.service.ExpenseService  
3.

**getExpenseSplits(Long)** - Method in class com.mapmyjourney.backend.controller.ExpenseSplitController  
2.

**getExpenseSplits(Long)** - Method in class com.mapmyjourney.backend.service.ExpenseSplitService  
Obtiene todas las divisiones de un gasto.

**getExpirationTime()** - Method in class com.mapmyjourney.backend.security.JwtTokenProvider  
Retorna el tiempo de expiración del token en milisegundos.

**getImageUrl()** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**getMember(Long, Long)** - Method in class com.mapmyjourney.backend.controller.TripMemberController  
3.

**getMember(Long, Long)** - Method in class com.mapmyjourney.backend.service.TripMemberService  
Obtiene un miembro específico de un viaje.

**getMyTrips(int, int)** - Method in class com.mapmyjourney.backend.controller.TripController  
3.

**getName()** - Method in class com.mapmyjourney.backend.dto.UserCreateRequestDTO

**getName()** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**getNewPassword()** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**getNewPasswordConfirm()** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**getPassword()** - Method in class com.mapmyjourney.backend.dto.LoginRequestDTO

**getPassword()** - Method in class com.mapmyjourney.backend.dto.UserCreateRequestDTO

**getProfilePicture()** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**getSplit(Long)** - Method in class com.mapmyjourney.backend.service.ExpenseSplitService  
Obtiene una división específica por ID.

**getSplit(Long, Long)** - Method in class com.mapmyjourney.backend.controller.ExpenseSplitController  
3.

**getStartDate()** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**getTitle()** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**getTotalPendingDebt(Long)** - Method in class com.mapmyjourney.backend.controller.ExpenseSplitController  
8.

**getTotalPendingDebt(Long)** - Method in class com.mapmyjourney.backend.service.ExpenseSplitService  
Calcula el total de deudas pendientes de un usuario.

**getTotalSplits()** - Method in class com.mapmyjourney.backend.model.Expense  
Calcula el total de los splits.

**getTripByCode(String)** - Method in class com.mapmyjourney.backend.controller.TripController  
4.

**getTripByCode(String)** - Method in class com.mapmyjourney.backend.service.TripService  
Obtiene un viaje por su código (para invitaciones).

**getTripByld(Long)** - Method in class com.mapmyjourney.backend.controller.TripController  
2.

**getTripByld(Long)** - Method in class com.mapmyjourney.backend.service.TripService  
Obtiene un viaje por ID.

**getTripExpenses(Long)** - Method in class com.mapmyjourney.backend.service.ExpenseService  
2.

**getTripExpenses(Long, int, int)** - Method in class com.mapmyjourney.backend.controller.ExpenseController  
2.

**getTripMembers(Long)** - Method in class com.mapmyjourney.backend.controller.TripMemberController  
2.

**getTripMembers(Long)** - Method in class com.mapmyjourney.backend.service.TripMemberService  
Obtiene todos los miembros de un viaje.

**getUserByEmail(String)** - Method in class com.mapmyjourney.backend.controller.UserController  
4.

**getUserByEmail(String)** - Method in class com.mapmyjourney.backend.service.UserService



3.  
**getUserById(Long)** - Method in class `com.mapmyjourney.backend.controller.UserController`  
3.  
**getUserById(Long)** - Method in class `com.mapmyjourney.backend.service.UserService`  
2.  
**getUserIdByEmail(String)** - Method in class `com.mapmyjourney.backend.service.UserService`  
Obtiene el ID del usuario a partir del email.  
**getUserPendingDebts(Long)** - Method in class `com.mapmyjourney.backend.controller.ExpenseSplitController`  
7.  
**getUserPendingDebts(Long)** - Method in class `com.mapmyjourney.backend.service.ExpenseSplitService`  
Obtiene todas las deudas pendientes de un usuario.  
**getUserTrips(Long)** - Method in class `com.mapmyjourney.backend.service.TripService`  
Obtiene todos los viajes del usuario.  
**GlobalExceptionHandler** - Class in `com.mapmyjourney.backend.config`  
Manejador global de excepciones para la API REST.  
**GlobalExceptionHandler()** - Constructor for class `com.mapmyjourney.backend.config.GlobalExceptionHandler`

## H

**handleAccessDenied(AccessDeniedException)** - Method in class `com.mapmyjourney.backend.config.GlobalExceptionHandler`  
Maneja excepciones de acceso denegado (403).  
**handleAuthenticationException(AuthenticationException)** - Method in class `com.mapmyjourney.backend.config.GlobalExceptionHandler`  
Maneja excepciones generales de autenticación (401).  
**handleBadCredentials(BadCredentialsException)** - Method in class `com.mapmyjourney.backend.config.GlobalExceptionHandler`  
Maneja excepciones de credenciales inválidas (401).  
**handleDuplicateResource(DuplicateResourceException)** - Method in class `com.mapmyjourney.backend.config.GlobalExceptionHandler`  
Maneja excepciones de recurso duplicado (409).  
**handleGeneralException(Exception)** - Method in class `com.mapmyjourney.backend.config.GlobalExceptionHandler`  
Maneja todas las excepciones no capturadas (500).  
**handleResourceNotFound(ResourceNotFoundException)** - Method in class `com.mapmyjourney.backend.config.GlobalExceptionHandler`  
Maneja excepciones de recurso no encontrado (404).  
**handleValidationException(ValidationException)** - Method in class `com.mapmyjourney.backend.config.GlobalExceptionHandler`  
Maneja excepciones de validación.  
**handleValidationExceptions(MethodArgumentNotValidException)** - Method in class `com.mapmyjourney.backend.config.GlobalExceptionHandler`  
Maneja excepciones de validación de argumentos (ej: @Valid fallido).  
**hasPermission(TripMemberRole)** - Method in enum class `com.mapmyjourney.backend.model.enums.TripMemberRole`  
Verifica si este rol tiene un permiso específico.  
**hasPermission(TripMemberRole)** - Method in class `com.mapmyjourney.backend.model.TripMember`  
Verifica si este miembro tiene el permiso requerido.  
**hasRole(Long, Long, TripMemberRole)** - Method in class `com.mapmyjourney.backend.service.TripMemberService`  
Verifica si un usuario tiene un rol específico en un viaje.  
**health()** - Method in class `com.mapmyjourney.backend.controller.HealthController`  
Verifica que el servidor backend está disponible  
**HealthController** - Class in `com.mapmyjourney.backend.controller`  
Controlador para verificar el estado de los servicios del backend Usado por la página de estado para comprobar que el backend está disponible  
**HealthController()** - Constructor for class `com.mapmyjourney.backend.controller.HealthController`  
**hello()** - Method in class `com.mapmyjourney.backend.controller.ExampleController`  
**helloName(String)** - Method in class `com.mapmyjourney.backend.controller.ExampleController`

## I

**isMemberOfTrip(Long, Long)** - Method in class `com.mapmyjourney.backend.service.TripMemberService`  
Verifica si un usuario es miembro de un viaje.  
**isOwner()** - Method in class `com.mapmyjourney.backend.model.TripMember`  
Verifica si este miembro es el propietario del viaje.

## J

**JwtAuthenticationFilter** - Class in `com.mapmyjourney.backend.security`  
**JwtAuthenticationFilter()** - Constructor for class `com.mapmyjourney.backend.security.JwtAuthenticationFilter`  
**JwtTokenProvider** - Class in `com.mapmyjourney.backend.security`  
Proveedor de tokens JWT.  
**JwtTokenProvider()** - Constructor for class `com.mapmyjourney.backend.security.JwtTokenProvider`

## L

**leaveTrip(Long)** - Method in class `com.mapmyjourney.backend.controller.TripMemberController`

6.

**loadUserByUsername(String)** - Method in class `com.mapmyjourney.backend.security.CustomUserDetailsService`

**login(LoginRequestDTO)** - Method in class `com.mapmyjourney.backend.controller.UserController`

2.

**LoginRequestDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para solicitud de login.

**LoginRequestDTO()** - Constructor for class `com.mapmyjourney.backend.dto.LoginRequestDTO`

**LoginRequestDTO(String, String)** - Constructor for class `com.mapmyjourney.backend.dto.LoginRequestDTO`

**LoginResponseDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para respuesta de login (contiene el token JWT y datos del usuario).

**LoginResponseDTO(String, Long, UserDTO)** - Constructor for class `com.mapmyjourney.backend.dto.LoginResponseDTO`

**LoginResponseDTO(String, String, Long, UserDTO)** - Constructor for class `com.mapmyjourney.backend.dto.LoginResponseDTO`

## M

**main(String[])** - Static method in class `com.mapmyjourney.backend.BackendApplication`  
Método principal que arranca la aplicación Spring Boot.

**MANUAL** - Enum constant in enum class `com.mapmyjourney.backend.model.enums.ExpenseSplitType`

**markAsPaid(Long)** - Method in class `com.mapmyjourney.backend.service.ExpenseSplitService`  
Marca una división como pagada.

**markAsUnpaid(Long)** - Method in class `com.mapmyjourney.backend.service.ExpenseSplitService`  
Marca una división como no pagada.

**markSplitAsPaid(Long)** - Method in class `com.mapmyjourney.backend.service.ExpenseService`

4.

**markSplitAsPaid(Long, Long)** - Method in class `com.mapmyjourney.backend.controller.ExpenseSplitController`

4.

**markSplitAsUnpaid(Long, Long)** - Method in class `com.mapmyjourney.backend.controller.ExpenseSplitController`

5.

## O

**onUpdate()** - Method in class `com.mapmyjourney.backend.model.Activity`

**onUpdate()** - Method in class `com.mapmyjourney.backend.model.Expense`

**onUpdate()** - Method in class `com.mapmyjourney.backend.model.User`

**OpenApiConfig** - Class in `com.mapmyjourney.backend.config`  
Configuración de OpenAPI/Swagger para documentación automática de la API REST.

**OpenApiConfig()** - Constructor for class `com.mapmyjourney.backend.config.OpenApiConfig`

**OWNER** - Enum constant in enum class `com.mapmyjourney.backend.model.enums.TripMemberRole`

## P

**passwordEncoder()** - Method in class `com.mapmyjourney.backend.security.SecurityConfig`  
Define el encoder de contraseñas (BCrypt).

**PERCENTAGE** - Enum constant in enum class `com.mapmyjourney.backend.model.enums.ExpenseSplitType`

## R

**registerUser(UserCreateRequestDTO)** - Method in class `com.mapmyjourney.backend.controller.UserController`

1.

**registerUser(UserCreateRequestDTO)** - Method in class `com.mapmyjourney.backend.service.UserService`

1.

**registerUserAndAuthenticate(UserCreateRequestDTO)** - Method in class `com.mapmyjourney.backend.service.UserService`  
Registra un nuevo usuario y lo autentica automáticamente.

**removeMember(Long, Long)** - Method in class `com.mapmyjourney.backend.controller.TripMemberController`

5.

**removeMemberFromTrip(Long, Long)** - Method in class `com.mapmyjourney.backend.service.TripMemberService`  
Elimina un miembro de un viaje.

**reorderActivities(Long, LocalDate, List<String>)** - Method in class `com.mapmyjourney.backend.controller.ActivityController`  
Reordena las actividades de una fecha específica.

**reorderActivities(Long, LocalDate, List<String>)** - Method in class `com.mapmyjourney.backend.service.ActivityService`  
Reordena las actividades de una fecha.

**ResourceNotFoundException** - Exception Class in `com.mapmyjourney.backend.exception`  
Excepción base para errores de recursos no encontrados.

**ResourceNotFoundException(String)** - Constructor for exception class `com.mapmyjourney.backend.exception.ResourceNotFoundException`

**ResourceNotFoundException(String, Throwable)** - Constructor for exception class  
com.mapmyjourney.backend.exception.ResourceNotFoundException

**run(String...)** - Method in class com.mapmyjourney.backend.config.DatabaseSeeder

## S

**SecurityConfig** - Class in com.mapmyjourney.backend.security  
Configuración de Spring Security con JWT y CORS

**SecurityConfig()** - Constructor for class com.mapmyjourney.backend.security.SecurityConfig

**setBudget(BigDecimal)** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**setCurrentPassword(String)** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**setDescription(String)** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**setDestination(String)** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**setEmail(String)** - Method in class com.mapmyjourney.backend.dto.LoginRequestDTO

**setEmail(String)** - Method in class com.mapmyjourney.backend.dto.UserCreateRequestDTO

**setEmail(String)** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**setEndDate(LocalDate)** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**setImageUrl(String)** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**setName(String)** - Method in class com.mapmyjourney.backend.dto.UserCreateRequestDTO

**setName(String)** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**setNewPassword(String)** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**setNewPasswordConfirm(String)** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**setPassword(String)** - Method in class com.mapmyjourney.backend.dto.LoginRequestDTO

**setPassword(String)** - Method in class com.mapmyjourney.backend.dto.UserCreateRequestDTO

**setProfilePicture(String)** - Method in class com.mapmyjourney.backend.dto.UserUpdateRequestDTO

**setStartDate(LocalDate)** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**setTitle(String)** - Method in class com.mapmyjourney.backend.dto.TripCreateRequestDTO

## T

**toString()** - Method in class com.mapmyjourney.backend.dto.LoginRequestDTO

**TRANSITION** - Enum constant in enum class com.mapmyjourney.backend.model.ActivityType

**Trip** - Class in com.mapmyjourney.backend.model  
Entidad que representa un viaje colaborativo.

**Trip()** - Constructor for class com.mapmyjourney.backend.model.Trip

**TripController** - Class in com.mapmyjourney.backend.controller  
Controlador REST para gestionar viajes.

**TripController()** - Constructor for class com.mapmyjourney.backend.controller.TripController

**TripCreateRequestDTO** - Class in com.mapmyjourney.backend.dto  
DTO para crear un nuevo viaje.

**TripCreateRequestDTO()** - Constructor for class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**TripCreateRequestDTO(String, String, String, String, LocalDate, LocalDate, BigDecimal)** - Constructor for class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**TripCreateRequestDTO(String, String, String, LocalDate, LocalDate, BigDecimal)** - Constructor for class com.mapmyjourney.backend.dto.TripCreateRequestDTO

**TripDTO** - Class in com.mapmyjourney.backend.dto  
DTO para información completa de un viaje.

**TripDTO()** - Constructor for class com.mapmyjourney.backend.dto.TripDTO

**TripMember** - Class in com.mapmyjourney.backend.model  
Entidad que representa la relación entre un Usuario y un Viaje.

**TripMember()** - Constructor for class com.mapmyjourney.backend.model.TripMember



**TripMemberController** - Class in `com.mapmyjourney.backend.controller`  
Controlador REST para gestionar miembros de viajes.

**TripMemberController()** - Constructor for class `com.mapmyjourney.backend.controller.TripMemberController`

**TripMemberDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para información de un miembro de un viaje.

**TripMemberDTO()** - Constructor for class `com.mapmyjourney.backend.dto.TripMemberDTO`

**TripMemberRepository** - Interface in `com.mapmyjourney.backend.repository`

**TripMemberRole** - Enum Class in `com.mapmyjourney.backend.model.enums`  
Enumeración que define los roles de un usuario dentro de un viaje.

**TripMemberService** - Class in `com.mapmyjourney.backend.service`  
Servicio para gestionar la membresía de usuarios en viajes.

**TripMemberService()** - Constructor for class `com.mapmyjourney.backend.service.TripMemberService`

**TripRepository** - Interface in `com.mapmyjourney.backend.repository`

**TripService** - Class in `com.mapmyjourney.backend.service`

**TripService()** - Constructor for class `com.mapmyjourney.backend.service.TripService`

**tripsHealth()** - Method in class `com.mapmyjourney.backend.controller.HealthController`  
Verifica que el servicio de viajes está disponible

## U

**updateActivity(Long, String, UpdateActivityRequestDTO)** - Method in class `com.mapmyjourney.backend.controller.ActivityController`  
Actualiza una actividad existente.

**updateActivity(Long, String, UpdateActivityRequestDTO)** - Method in class `com.mapmyjourney.backend.service.ActivityService`  
Actualiza una actividad existente.

**UpdateActivityRequestDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para actualizar una actividad existente.

**UpdateActivityRequestDTO()** - Constructor for class `com.mapmyjourney.backend.dto.UpdateActivityRequestDTO`

**updateExpense(Long, ExpenseCreateRequestDTO, Long)** - Method in class `com.mapmyjourney.backend.service.ExpenseService`  
Actualiza un gasto existente.

**updateExpense(Long, Long, ExpenseCreateRequestDTO)** - Method in class `com.mapmyjourney.backend.controller.ExpenseController`  
4.

**updateSplitAmount(Long, BigDecimal)** - Method in class `com.mapmyjourney.backend.service.ExpenseSplitService`  
Actualiza el monto de una división.

**updateTrip(Long, TripCreateRequestDTO)** - Method in class `com.mapmyjourney.backend.controller.TripController`  
5.

**updateTrip(Long, TripCreateRequestDTO, Long)** - Method in class `com.mapmyjourney.backend.service.TripService`  
Actualiza un viaje existente.

**updateUser(Long, UserCreateRequestDTO)** - Method in class `com.mapmyjourney.backend.controller.UserController`  
5.

**updateUser(Long, UserCreateRequestDTO)** - Method in class `com.mapmyjourney.backend.service.UserService`  
4.

**updateUserProfile(Long, UserUpdateRequestDTO)** - Method in class `com.mapmyjourney.backend.controller.UserController`  
6.

**updateUserProfile(Long, UserUpdateRequestDTO)** - Method in class `com.mapmyjourney.backend.service.UserService`  
Actualiza el perfil de un usuario existente.

**User** - Class in `com.mapmyjourney.backend.model`  
Entidad que representa un usuario en el sistema.

**User()** - Constructor for class `com.mapmyjourney.backend.model.User`

**USER** - Enum constant in enum class `com.mapmyjourney.backend.model.enums.UserRole`

**UserController** - Class in `com.mapmyjourney.backend.controller`  
Controlador REST para gestionar usuarios.

**UserController()** - Constructor for class `com.mapmyjourney.backend.controller.UserController`

**UserCreateRequestDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para crear/registrarse un nuevo usuario.

**UserCreateRequestDTO()** - Constructor for class `com.mapmyjourney.backend.dto.UserCreateRequestDTO`

**UserCreateRequestDTO(String, String, String)** - Constructor for class `com.mapmyjourney.backend.dto.UserCreateRequestDTO`

**UserDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para información pública de un usuario.

**UserDTO()** - Constructor for class `com.mapmyjourney.backend.dto.UserDTO`

**UserRepository** - Interface in `com.mapmyjourney.backend.repository`

**UserRole** - Enum Class in `com.mapmyjourney.backend.model.enums`

Enum que define los roles de los usuarios en el sistema.

**UserService** - Class in `com.mapmyjourney.backend.service`

**UserService()** - Constructor for class `com.mapmyjourney.backend.service.UserService`

**usersHealth()** - Method in class `com.mapmyjourney.backend.controller.HealthController`  
Verifica que el servicio de usuarios está disponible

**UserUpdateRequestDTO** - Class in `com.mapmyjourney.backend.dto`  
DTO para actualizar datos de un usuario existente.

**UserUpdateRequestDTO()** - Constructor for class `com.mapmyjourney.backend.dto.UserUpdateRequestDTO`

**UserUpdateRequestDTO(String, String, String)** - Constructor for class `com.mapmyjourney.backend.dto.UserUpdateRequestDTO`

V

**validateDatesAndUpdate()** - Method in class `com.mapmyjourney.backend.model.Trip`  
Valida que la fecha de fin sea posterior a la de inicio y actualiza timestamp.

**validateToken(String)** - Method in class `com.mapmyjourney.backend.security.jwtTokenProvider`  
Valida que el token JWT sea válido.

**ValidationException** - Exception Class in `com.mapmyjourney.backend.exception`  
Excepción para errores de negocio (ej: viaje con fechas inválidas).

**ValidationException(String)** - Constructor for exception class `com.mapmyjourney.backend.exception.ValidationException`

**ValidationException(String, Throwable)** - Constructor for exception class `com.mapmyjourney.backend.exception.ValidationException`

**valueOf(String)** - Static method in enum class `com.mapmyjourney.backend.model.ActivityType`  
Returns the enum constant of this class with the specified name.

**valueOf(String)** - Static method in enum class `com.mapmyjourney.backend.model.enums.ExpenseSplitType`  
Returns the enum constant of this class with the specified name.

**valueOf(String)** - Static method in enum class `com.mapmyjourney.backend.model.enums.TripMemberRole`  
Returns the enum constant of this class with the specified name.

**valueOf(String)** - Static method in enum class `com.mapmyjourney.backend.model.enums.UserRole`  
Returns the enum constant of this class with the specified name.

**values()** - Static method in enum class `com.mapmyjourney.backend.model.ActivityType`  
Returns an array containing the constants of this enum class, in the order they are declared.

**values()** - Static method in enum class `com.mapmyjourney.backend.model.enums.ExpenseSplitType`  
Returns an array containing the constants of this enum class, in the order they are declared.

**values()** - Static method in enum class `com.mapmyjourney.backend.model.enums.TripMemberRole`  
Returns an array containing the constants of this enum class, in the order they are declared.

**values()** - Static method in enum class `com.mapmyjourney.backend.model.enums.UserRole`  
Returns an array containing the constants of this enum class, in the order they are declared.

**verifyUserInTrip(Long, Long)** - Method in class `com.mapmyjourney.backend.service.TripService`  
Verifica que un usuario pertenece a un viaje.

**VIEWER** - Enum constant in enum class `com.mapmyjourney.backend.model.enums.TripMemberRole`

W

**WebConfig** - Class in `com.mapmyjourney.backend.config`  
Configuración Web - CORS Global Este filtro CORS se ejecuta ANTES de todos los filtros de seguridad, asegurando que el preflight (OPTIONS) sea procesado correctamente.

**WebConfig()** - Constructor for class `com.mapmyjourney.backend.config.WebConfig`