**Package** com.mapmyjourney.backend.controller

# Class ExpenseSplitController

java.lang.Object
    com.mapmyjourney.backend.controller.ExpenseSplitController

```
@RestController
@RequestMapping("/expenses")
public class ExpenseSplitController
extends Object
```

Controlador REST para gestionar divisiones de gastos.

## Constructor Summary

**Constructors**

| Constructor | Description |
|---|---|
| ExpenseSplitController() | |

## Method Summary

**All Methods** | **Instance Methods** | **Concrete Methods**

| Modifier and Type | Method | Description |
|---|---|---|
| org.springframework.http.ResponseEntity<ExpenseSplitDTO> | createSplit(Long expenseId, @Valid CreateSplitRequestDTO request) | 1. |
| org.springframework.http.ResponseEntity<Void> | deleteSplit(Long expenseId, Long splitId) | 6. |
| org.springframework.http.ResponseEntity<List<ExpenseSplitDTO>> | getExpenseSplits(Long expenseId) | 2. |
| org.springframework.http.ResponseEntity<ExpenseSplitDTO> | getSplit(Long expenseId, Long splitId) | 3. |
| org.springframework.http.ResponseEntity<BigDecimal> | getTotalPendingDebt(Long userId) | 8. |
| org.springframework.http.ResponseEntity<List<ExpenseSplitDTO>> | getUserPendingDebts(Long userId) | 7. |
| org.springframework.http.ResponseEntity<ExpenseSplitDTO> | markSplitAsPaid(Long expenseId, Long splitId) | 4. |
| org.springframework.http.ResponseEntity<ExpenseSplitDTO> | | |

markSplitAsUnpaid(Long expenseId, Long splitId)

5.

### Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

## Constructor Details

### ExpenseSplitController

```
public ExpenseSplitController()
```

## Method Details

### createSplit

```
@PreAuthorize("hasRole(\'USER\')")
@PostMapping("/{expenseId}/splits")
public org.springframework.http.ResponseEntity<ExpenseSplitDTO> createSplit
(@PathVariable
 Long expenseId,
 @Valid @RequestBody
 @Valid CreateSplitRequestDTO request)
```

1. Crea una nueva división de gasto. POST /api/expenses/{expenseId}/splits

### getExpenseSplits

```
@PreAuthorize("hasRole(\'USER\')")
@GetMapping("/{expenseId}/splits")
public org.springframework.http.ResponseEntity<List<ExpenseSplitDTO>> getExpenseSplits(@PathVariable
                                                                                        Long expenseId)
```

2. Obtiene todas las divisiones de un gasto. GET /api/expenses/{expenseId}/splits

### getSplit

```
@PreAuthorize("hasRole(\'USER\')")
@GetMapping("/{expenseId}/splits/{splitId}")
public org.springframework.http.ResponseEntity<ExpenseSplitDTO> getSplit(@PathVariable
                                                                         Long expenseId,
                                                                         @PathVariable
                                                                         Long splitId)
```

3. Obtiene una división específica. GET /api/expenses/{expenseId}/splits/{splitId}

### markSplitAsPaid

```
@PreAuthorize("hasRole(\'USER\')")
@PutMapping("/{expenseId}/splits/{splitId}/pay")
public org.springframework.http.ResponseEntity<ExpenseSplitDTO> markSplitAsPaid(@PathVariable
                                                                               Long expenseId,
                                                                               @PathVariable
                                                                               Long splitId)
```

4. Marca una división como pagada. PUT /api/expenses/{expenseId}/splits/{splitId}/pay

### markSplitAsUnpaid

```
@PreAuthorize("hasRole(\'USER\')")
@PutMapping("/{expenseId}/splits/{splitId}/unpay")
```

```
public org.springframework.http.ResponseEntity<ExpenseSplitDTO> markSplitAsUnpaid(@PathVariable
                                                                                   Long expenseId,
                                                                                   @PathVariable
                                                                                   Long splitId)
```

5. Marca una división como no pagada. PUT /api/expenses/{expenseId}/splits/{splitId}/unpay

### deleteSplit

```
@PreAuthorize("hasRole(\'USER\')")
@DeleteMapping("/{expenseId}/splits/{splitId}")
public org.springframework.http.ResponseEntity<Void> deleteSplit(@PathVariable
                                                                 Long expenseId,
                                                                 @PathVariable
                                                                 Long splitId)
```

6. Elimina una división de gasto. DELETE /api/expenses/{expenseId}/splits/{splitId}

### getUserPendingDebts

```
@PreAuthorize("hasRole(\'USER\')")
@GetMapping("/users/{userId}/pending-debts")
public org.springframework.http.ResponseEntity<List<ExpenseSplitDTO>> getUserPendingDebts(@PathVariable
                                                                                          Long userId)
```

7. Obtiene las deudas pendientes de un usuario. GET /api/users/{userId}/pending-debts

### getTotalPendingDebt

```
@PreAuthorize("hasRole(\'USER\')")
@GetMapping("/users/{userId}/total-debt")
public org.springframework.http.ResponseEntity<BigDecimal> getTotalPendingDebt(@PathVariable
                                                                               Long userId)
```

8. Calcula la deuda total de un usuario. GET /api/users/{userId}/total-debt