Update Transaction

User	Frontend	JWTFilter	TransactionController	TransactionService	TransactionRepository	SharedAccessRepository	Database

User → Frontend: Edit transaction

Frontend → JWTFilter: PUT /api/transactions/[id] + JWT
{amount: 150, categoryId: "...", description: "Updated"}

JWTFilter → JWTFilter: validateToken(jwt)

JWTFilter → TransactionController: request + userId

TransactionController → TransactionService: update(userId, transactionId, updateDTO)

TransactionService → TransactionRepository: findById(transactionId)

TransactionRepository → Database: SELECT * FROM transactions WHERE id = ?

Database → TransactionRepository: transaction

alt [Transaction not found]

TransactionService → TransactionController: throw NotFoundException

TransactionController → Frontend: 404 Not Found

Frontend → User: Transaction not found

[Transaction exists]

TransactionService → SharedAccessRepository: findByUserAndResource(userId, 'WALLET', transaction.walletId)

SharedAccessRepository → Database: SELECT * FROM shared_access WHERE...

Database → SharedAccessRepository: sharedAccess

alt [No access to wallet]

TransactionService → TransactionController: throw ForbiddenException

TransactionController → Frontend: 403 Forbidden

Frontend → User: Access denied

[Has access (OWNER or MEMBER)]

alt [User is not transaction creator AND User is MEMBER]

TransactionService → TransactionController: throw ForbiddenException

TransactionController → Frontend: 403 Forbidden

Frontend → User: You can only edit your own transactions

[User is creator OR User is OWNER]

TransactionService → TransactionService: updateFields(transaction, updateDTO)

TransactionService → TransactionRepository: save(transaction)

TransactionRepository → Database: UPDATE transactions SET amount=?, category_id=?, description=?, updated_at=NOW() WHERE id=?

Database → TransactionRepository: updatedTransaction

TransactionService → TransactionController: TransactionDTO

TransactionController → Frontend: 200 OK

Frontend → User: Transaction updated