Sequence Diagram for Employee Controller

Client | JwtAuthFilter | SecurityConfig | EmployeeController | EmployeeService | EmployeeRepository | Database

Create Employee (Authenticated User)

Client → JwtAuthFilter: POST /api/v1/employees/create
JwtAuthFilter → SecurityConfig: Check @PreAuthorize("isAuthenticated()")

alt [Not Authenticated]
JwtAuthFilter → Client: HTTP 401 Unauthorized

[Authenticated]
SecurityConfig → EmployeeController: Allow Request
EmployeeController → EmployeeService: createEmployee(EmployeeDTO)
EmployeeService → EmployeeService: validateEmployee()
EmployeeService → EmployeeRepository: existsByEmail()
EmployeeRepository → Database: SELECT COUNT(*)
Database ⇠ EmployeeRepository: count
EmployeeService → EmployeeRepository: create(EmployeeDTO)
EmployeeRepository → Database: INSERT Employee
Database ⇠ EmployeeRepository: Employee data
EmployeeRepository ⇠ EmployeeService: EmployeeResponse
EmployeeService ⇠ EmployeeController: EmployeeResponse
EmployeeController → Client: HTTP 201 Created

Get Employee (Authenticated User)

Client → JwtAuthFilter: GET /api/v1/employees/{id}
JwtAuthFilter → SecurityConfig: Check @PreAuthorize("isAuthenticated()")

alt [Not Authenticated]
JwtAuthFilter → Client: HTTP 401 Unauthorized

[Authenticated]
SecurityConfig → EmployeeController: Allow Request
EmployeeController → EmployeeService: getEmployee(id)
EmployeeService → EmployeeRepository: findById(id)
EmployeeRepository → Database: SELECT Employee
Database ⇠ EmployeeRepository: Employee data
EmployeeRepository ⇠ EmployeeService: Optional<EmployeeResponse>
EmployeeService ⇠ EmployeeController: Optional<EmployeeResponse>
EmployeeController → Client: HTTP 200 OK

Update Employee (Admin Only)

Client → JwtAuthFilter: PUT /api/v1/employees/update/{id}
JwtAuthFilter → SecurityConfig: Check @PreAuthorize("hasRole('ADMIN')")

alt [Not Admin]
JwtAuthFilter → Client: HTTP 403 Forbidden

[Is Admin]
SecurityConfig → EmployeeController: Allow Request
EmployeeController → EmployeeService: updateEmployee(id, EmployeeDTO)
EmployeeService → EmployeeService: validateEmployee()
EmployeeService → EmployeeRepository: update(id, EmployeeDTO)
EmployeeRepository → Database: UPDATE Employee
Database ⇠ EmployeeRepository: Updated data
EmployeeRepository ⇠ EmployeeService: Optional<EmployeeResponse>
EmployeeService ⇠ EmployeeController: Optional<EmployeeResponse>
EmployeeController → Client: HTTP 200 OK

Delete Employee (Admin Only)

Client → JwtAuthFilter: DELETE /api/v1/employees/delete/{id}
JwtAuthFilter → SecurityConfig: Check @PreAuthorize("hasRole('ADMIN')")

alt [Not Admin]
JwtAuthFilter → Client: HTTP 403 Forbidden

[Is Admin]
SecurityConfig → EmployeeController: Allow Request
EmployeeController → EmployeeService: deleteEmployee(id)
EmployeeService → EmployeeRepository: delete(id)
EmployeeRepository → Database: DELETE Employee
Database ⇠ EmployeeRepository: Result
EmployeeRepository ⇠ EmployeeService: Boolean
EmployeeService ⇠ EmployeeController: Boolean
EmployeeController → Client: HTTP 200 OK

Client | JwtAuthFilter | SecurityConfig | EmployeeController | EmployeeService | EmployeeRepository | Database