# Delete role

## Overview

This API allows a user with the right ("Delete role") to delete a role from the system.

This API has an important business logic:

1. System roles (e.g. "ADMIN", "LAB_MANAGER", "CUSTOM_ROLE"...) cannot be deleted.
2. If the role being deleted is still assigned to any user, the system will automatically move those users to the default role "CUSTOM_ROLE".
3. The system will then delete the RolePrivileges and finally delete the role itself.

## API Specification

| API | URL |
| --- | --- |
| DELETE | /api/role/delete/{roleCode} |
| Permission | [AuthorizeByPrivilege("Delete role")] |

## Request sample

| Field | Description | Data Type | Examples |
| --- | --- | --- | --- |

| | | | |
|---|---|---|---|
| roleCode | RoleCode of the role to be deleted. | string | LAB_USER |

# Response sample

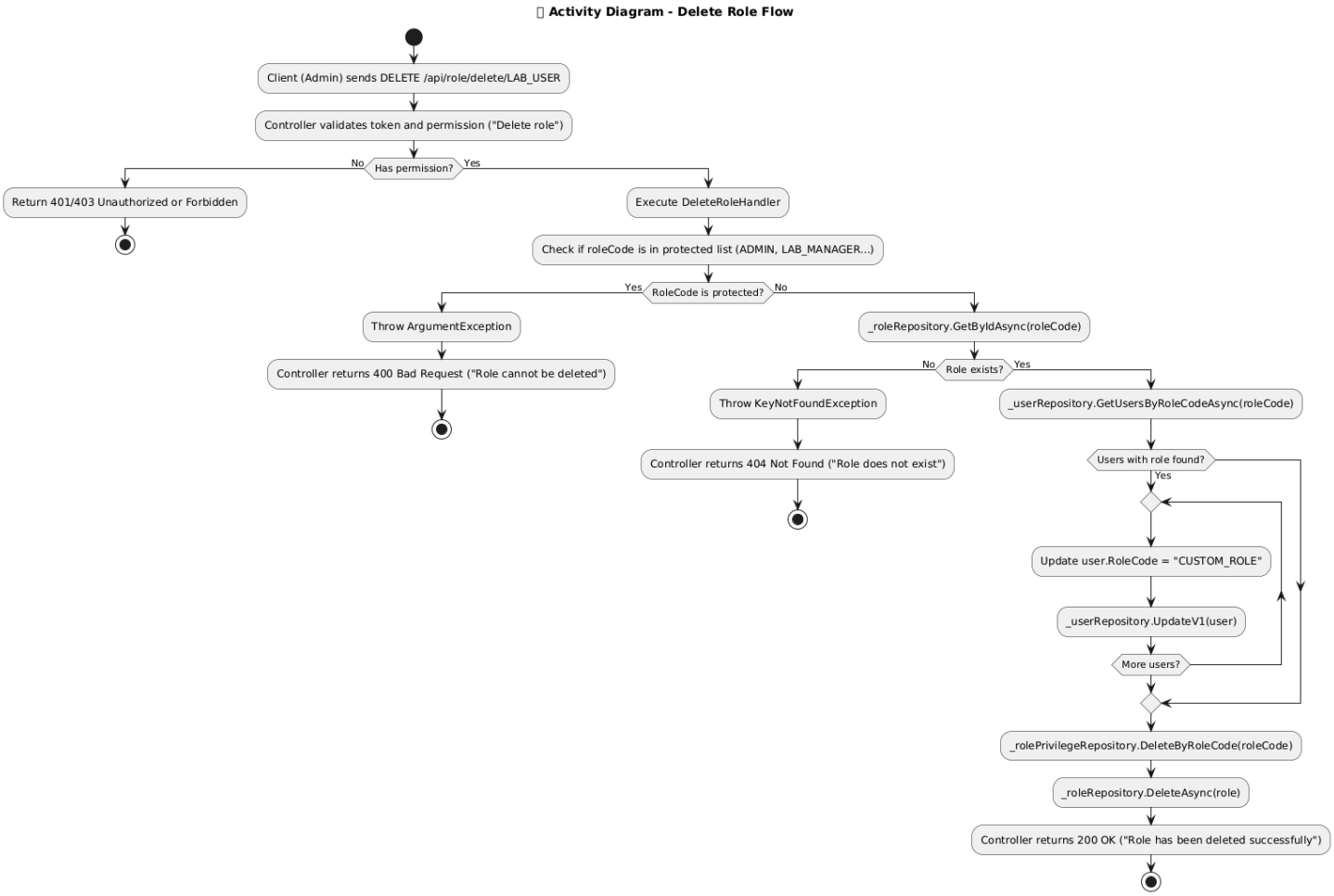

```
{
 "statusCode": 200,
 "message": "Role has been deleted successfully.",
 "data": null,
 "responsedAt": "2025-10-30T10:30:00.123Z"
}
```


# Validation

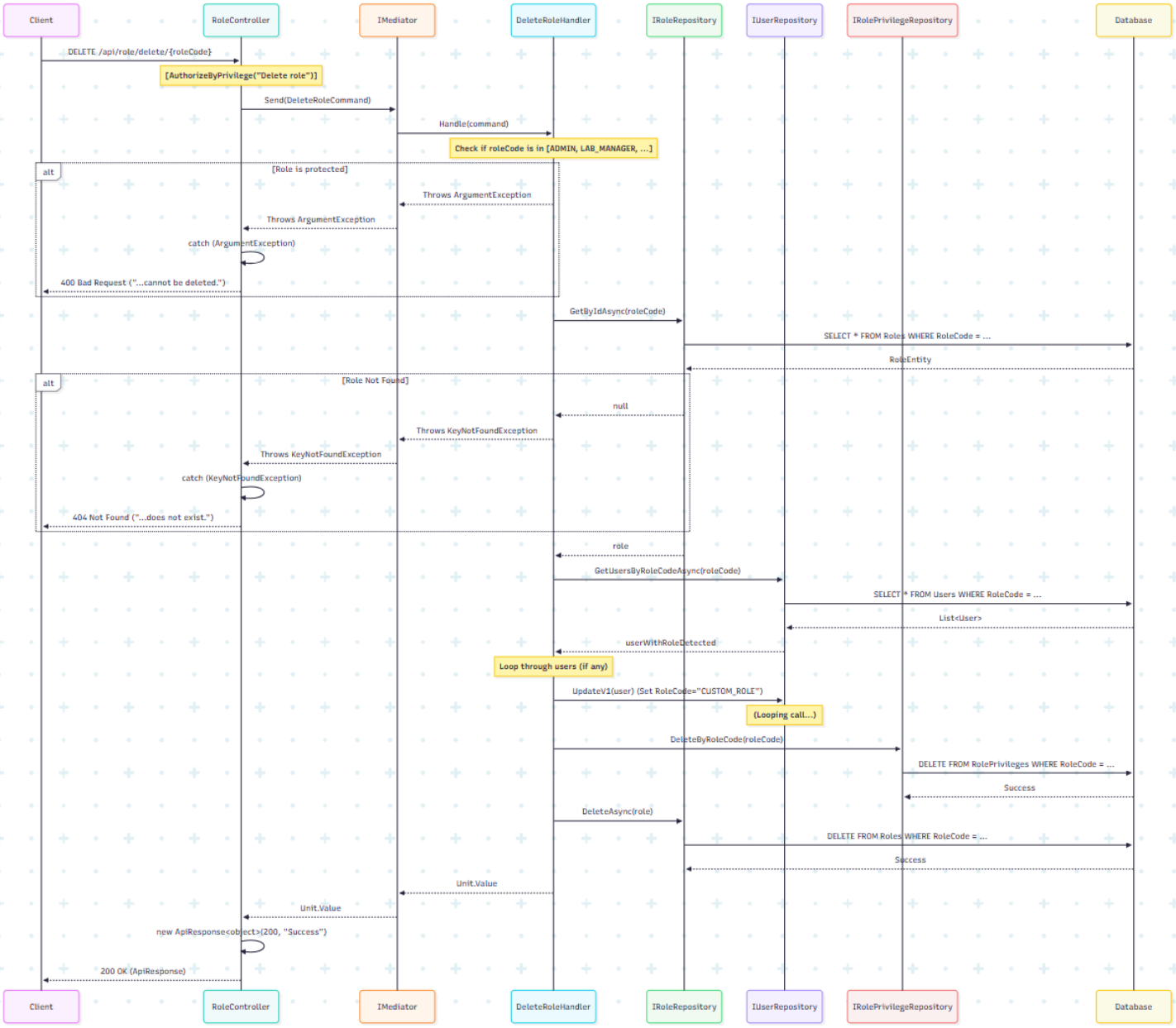
| Status code | Description | Examples |
|---|---|---|
| 400 | Operational error: Attempting to delete a system role. | {<br>"statusCode": 400,<br>"message": "Role with code ADMIN cannot be deleted.", ...} |
| 404 | Error: RoleCode not found in database. | {<br>"statusCode": 404,<br>"message": "Role with code XYZ does not exist.", ...} |

| | | |
|---|---|---|
| 401 | Unauthorized. User is not logged in. | Error: response status is 401 |
| 403 | Forbidden. The user is logged in but does not have the "Delete role" permission. | Error: response status is 403 |

# Activity Diagram




# Sequence Diagram


Client
RoleController
IMediator
DeleteRoleHandler
IRoleRepository
IUserRepository
IRolePrivilegeRepository
Database
DELETE /api/role/delete/{roleCode}
[AuthorizeByPrivilege("Delete role")]
Send(DeleteRoleCommand)
Handle(command)
Check if roleCode is in [ADMIN, LAB_MANAGER, ...]
alt
[Role is protected]
Throws ArgumentException
Throws ArgumentException
catch (ArgumentException)
400 Bad Request ("...cannot be deleted.")
GetByIdAsync(roleCode)
SELECT * FROM Roles WHERE RoleCode = ...
RoleEntity
alt
[Role Not Found]
null
Throws KeyNotFoundException
Throws KeyNotFoundException
catch (KeyNotFoundException)
404 Not Found ("...does not exist.")
role
GetUsersByRoleCodeAsync(roleCode)
SELECT * FROM Users WHERE RoleCode = ...
List<User>
userWithRoleDetected
Loop through users (if any)
UpdateV1(user) (Set RoleCode="CUSTOM_ROLE")
(Looping call...)
DeleteByRoleCode(roleCode)
DELETE FROM RolePrivileges WHERE RoleCode = ...
Success
DeleteAsync(role)
DELETE FROM Roles WHERE RoleCode = ...
Success
Unit.Value
Unit.Value
new ApiResponse<object>(200, "Success")
200 OK (ApiResponse)