Documentatie API - Endpointuri CRUD

# 1. Endpointuri din ApplicationController.java

## POST /application/create

Descriere: Creează o aplicație nouă.

Metodă HTTP: POST

Exemplu JSON:

{"nume": "Aplicație 1", "status": "În progres"}

Răspuns: Status 201: Aplicația a fost creată cu succes.

Exemplu răspuns:

{"application": {"id": 3, "nume": "Aplicație 1", "status": "În progres"}}

**GET /applications**

Descriere: Obține o listă cu toate aplicațiile.

Metodă HTTP: GET

Răspuns: Status 200: Listă cu aplicații.

Exemplu răspuns:

{"application": [{"id": 1, "nume": "Aplicație 1", "status": "În progres"}, {"id": 2, "nume": "Aplicație 2", "status": "Finalizată"}]}

**GET /application/{applicationId}**

Descriere: Obține o aplicație pe baza ID-ului.

Metodă HTTP: GET

Parametri: applicationId: ID-ul aplicației (Path variable).

Răspuns: Status 200: Detalii aplicație.

Exemplu răspuns:

{"application": {"id": 1, "nume": "Aplicație 1", "status": "În progres"}}

## PUT /application/update

Descriere: Actualizează o aplicație existentă.

Metodă HTTP: PUT

Exemplu JSON:

{"id": 1, "nume": "Aplicație 1", "status": "Finalizată"}

Răspuns: Status 200: Aplicația a fost actualizată cu succes.

Exemplu răspuns:

{"application": {"id": 1, "nume": "Aplicație 1", "status": "Finalizată"}}

## DELETE /application/{applicationId}

Descriere: Șterge o aplicație pe baza ID-ului.

Metodă HTTP: DELETE

Parametri: applicationId: ID-ul aplicației (Path variable).

Răspuns: Status 200: Aplicația a fost ștearsă cu succes.

Exemplu răspuns:

{"message": "Application deleted successfully", "status": "OK"}

# 2. Endpointuri din EmployeeApplicationController.java (CRUD pentru aplicațiile angajaților)

## POST /employee/application/create

Descriere: Creează o aplicație pentru un angajat.

Metodă HTTP: POST

Parametri: Body: JSON cu detalii despre aplicația angajatului.

Exemplu JSON:

{"employeeId": 1, "applicationId": 3, "status": "În procesare"}

Răspuns: Status 201: Aplicația angajatului a fost creată cu succes.

Exemplu răspuns:

{"employeeApplication": {"id": 3, "employeeId": 1, "applicationId": 3, "status": "În procesare"}}

## GET /employee/applications

Descriere: Obține o listă cu toate aplicațiile angajaților.

Metodă HTTP: GET

Răspuns: Status 200: Listă cu aplicațiile angajaților.

Exemplu răspuns:

{"employeeApplication": [{"id": 1, "employeeId": 1, "applicationId": 3, "status": "În procesare"}, {"id": 2, "employeeId": 2, "applicationId": 5, "status": "Finalizată"}]}

## GET /employee/application/{employeeApplicationId}

Descriere: Obține o aplicație a unui angajat pe baza ID-ului aplicației angajatului.

Metodă HTTP: GET

Parametri: employeeApplicationId: ID-ul aplicației angajatului (Path variable).

Răspuns: Status 200: Detalii aplicație angajat.

Exemplu răspuns:

{"employeeApplication": {"id": 1, "employeeId": 1, "applicationId": 3, "status": "În procesare"}}

## PUT /employee/application/update

Descriere: Actualizează o aplicație a unui angajat existent.

Metodă HTTP: PUT

Parametri: Body: JSON cu detalii actualizate pentru aplicația angajatului.

Exemplu JSON:

Răspuns: Status 200: Aplicația angajatului a fost actualizată cu succes.

Exemplu răspuns:

{"employeeApplication": {"id": 3, "employeeId": 1, "applicationId": 3, "status": "Finalizată"}}

## DELETE /employee/application/{employeeApplicationId}

Descriere: Șterge aplicația unui angajat pe baza ID-ului aplicației angajatului.

Metodă HTTP: DELETE

Parametri: employeeApplicationId: ID-ul aplicației angajatului (Path variable).

Răspuns: Status 200: Aplicația angajatului a fost ștearsă cu succes.

Exemplu răspuns:

{"message": "Employee application deleted successfully", "status": "OK"}

# 3. Endpointuri din EmployeeController.java (CRUD pentru angajați)

## POST /employee/add

Descriere: Creează un angajat nou.

Metodă HTTP: POST

Exemplu JSON:

{"nume": "Ion", "prenume": "Popescu", "varsta": 35, "departament": "IT"}

Răspuns: Status 201: Angajatul a fost creat cu succes.

Exemplu răspuns:

{"employee": {"id": 3, "nume": "Ion", "prenume": "Popescu", "varsta": 35, "departament": "IT"}}

## GET /employees

Descriere: Obține o listă cu toți angajații.

Metodă HTTP: GET

Parametri: Nu sunt necesari parametri.

Răspuns: Status 200: Listă cu angajați.

Exemplu răspuns:

{"employee": [{"id": 1, "nume": "Ion", "prenume": "Popescu", "varsta": 35, "departament": "IT"}]}

## GET /employee/{employeeId}

Descriere: Obține un angajat pe baza ID-ului.

Metodă HTTP: GET

Parametri: employeeId: ID-ul angajatului (Path variable).

Răspuns: Status 200: Detalii angajat.

Exemplu răspuns:

{"employee": {"id": 1, "nume": "Ion", "prenume": "Popescu", "varsta": 35, "departament": "IT"}}

## PUT /employee/update

Descriere: Actualizează un angajat existent.

Metodă HTTP: PUT

Parametri: Body: JSON cu detalii actualizate pentru angajat.

Răspuns: Status 200: Angajatul a fost actualizat cu succes.

Exemplu răspuns:

{"employee": {"id": 3, "nume": "Ion", "prenume": "Popescu", "varsta": 35, "departament": "IT"}}

## DELETE /employee/{employeeId}

Descriere: Șterge un angajat pe baza ID-ului.

Metodă HTTP: DELETE

Parametri: employeeId: ID-ul angajatului (Path variable).

Răspuns: Status 200: Angajatul a fost șters cu succes.

Exemplu răspuns:

{"message": "Employee deleted successfully", "status": "OK"}

# 4. Endpointuri din TaskController.java (CRUD pentru task-uri)

## POST /task/add

Descriere: Creează un task nou.

Metodă HTTP: POST

Parametri: Body: JSON cu detalii despre task.

Exemplu JSON:

{"nume": "Task 1", "status": "În desfășurare"}

Răspuns: Status 201: Task-ul a fost creat cu succes.

Exemplu răspuns:

{"task": {"id": 3, "nume": "Task 1", "status": "În desfășurare"}}

## GET /tasks

Descriere: Obține o listă cu toate task-urile.

Metodă HTTP: GET

Parametri: Nu sunt necesari parametri.

Răspuns: Status 200: Listă cu task-uri.

Exemplu răspuns:

{"task": [{"id": 1, "nume": "Task 1", "status": "În desfășurare"}, {"id": 2, "nume": "Task 2", "status": "Finalizat"}]}

## GET /task/{taskId}

Descriere: Obține un task pe baza ID-ului.

Metodă HTTP: GET

Parametri: taskId: ID-ul task-ului (Path variable).

Răspuns: Status 200: Detalii task.

Exemplu răspuns:

{"task": {"id": 1, "nume": "Task 1", "status": "În desfășurare"}}

## PUT /task/update

Descriere: Actualizează un task existent.

Metodă HTTP: PUT

Parametri: Body: JSON cu detalii actualizate pentru task.

Răspuns: Status 200: Task-ul a fost actualizat cu succes.

Exemplu răspuns:

{"task": {"id": 3, "nume": "Task 1", "status": "Finalizat"}}

## DELETE /task/{taskId}

Descriere: Șterge un task pe baza ID-ului.

Metodă HTTP: DELETE

Parametri: taskId: ID-ul task-ului (Path variable).

Răspuns: Status 200: Task-ul a fost șters cu succes.

Exemplu răspuns:

{"message": "Task deleted successfully", "status": "OK"}

Nu-mi functioneaza endpoint-urile din pacate, insa asta ar fi logica. Am incercat sa fac partea de fron-end in React si m-am complicat prea tare. Totodata am priedut timp si la partea de securitate care mai mult m-a incurcat, stiu ca nu trebuia facuta, dar m-am incapatanat sa o fac. Macar putem miza pe logica.