Platforma pentru gestionarea fișierelor

cu **API REST** și arhitectura bazată pe **micro-servicii**

Pentru proiectare, urmăm o serie de pași

1. Stabilirea cerințelor funcționale și nefuncționale
2. Identificarea resurselor
3. Definirea micro-serviciilor
4. Stabilirea comunicării între servicii (REST API)
5. Modelarea datelor
6. Definirea API-urilor
7. Securitate, autentificare, monitorizare, logging, scalare, deploy, testare, validare

Platforma pentru gestionarea fișierelor

Cerințe funcționale

Utilizatorul poate:

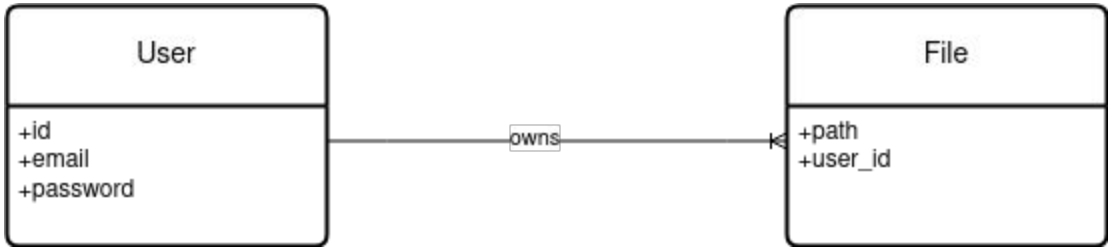
- CRUD fișiere tip X de dimensiuni între Y-Z
- Organiza fișierelor în directoare
- Partaja fișierelor cu alți utilizatori
- ...

Cerințe nefuncționale

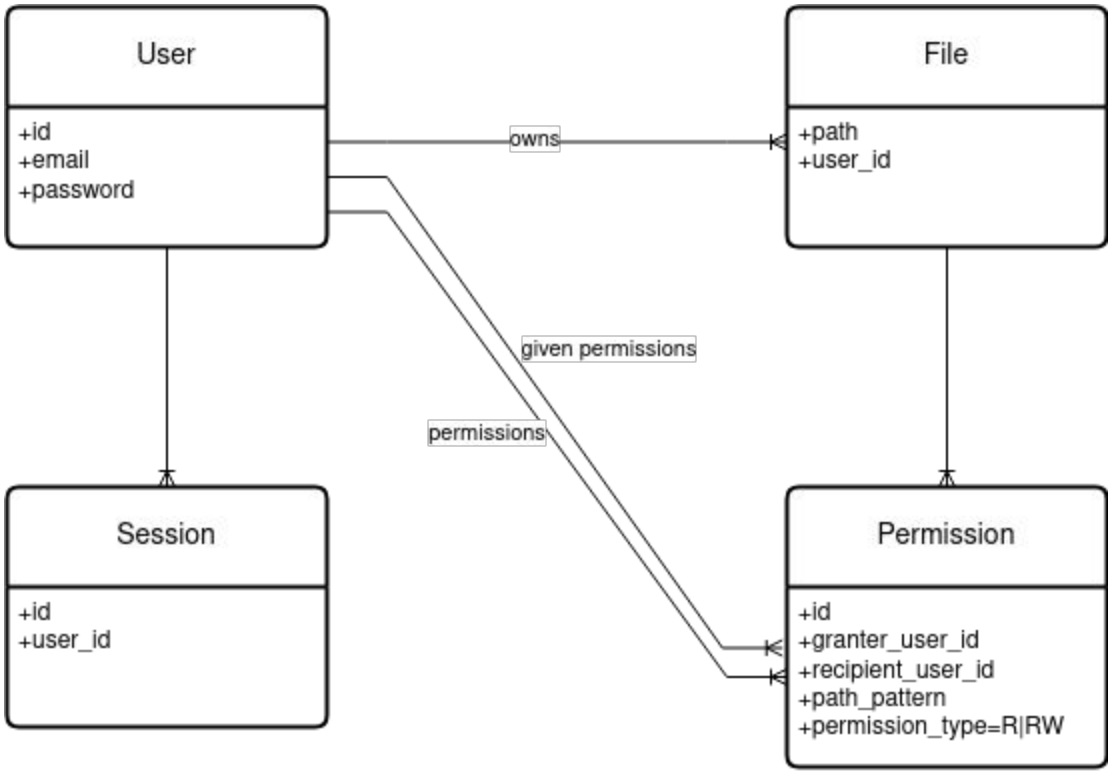
Platforma trebuie să:

- Permită prelucrarea a X requesturi pe secundă, care sunt în proporție de Y% read sau write
- Aibă un latency mic
- ...

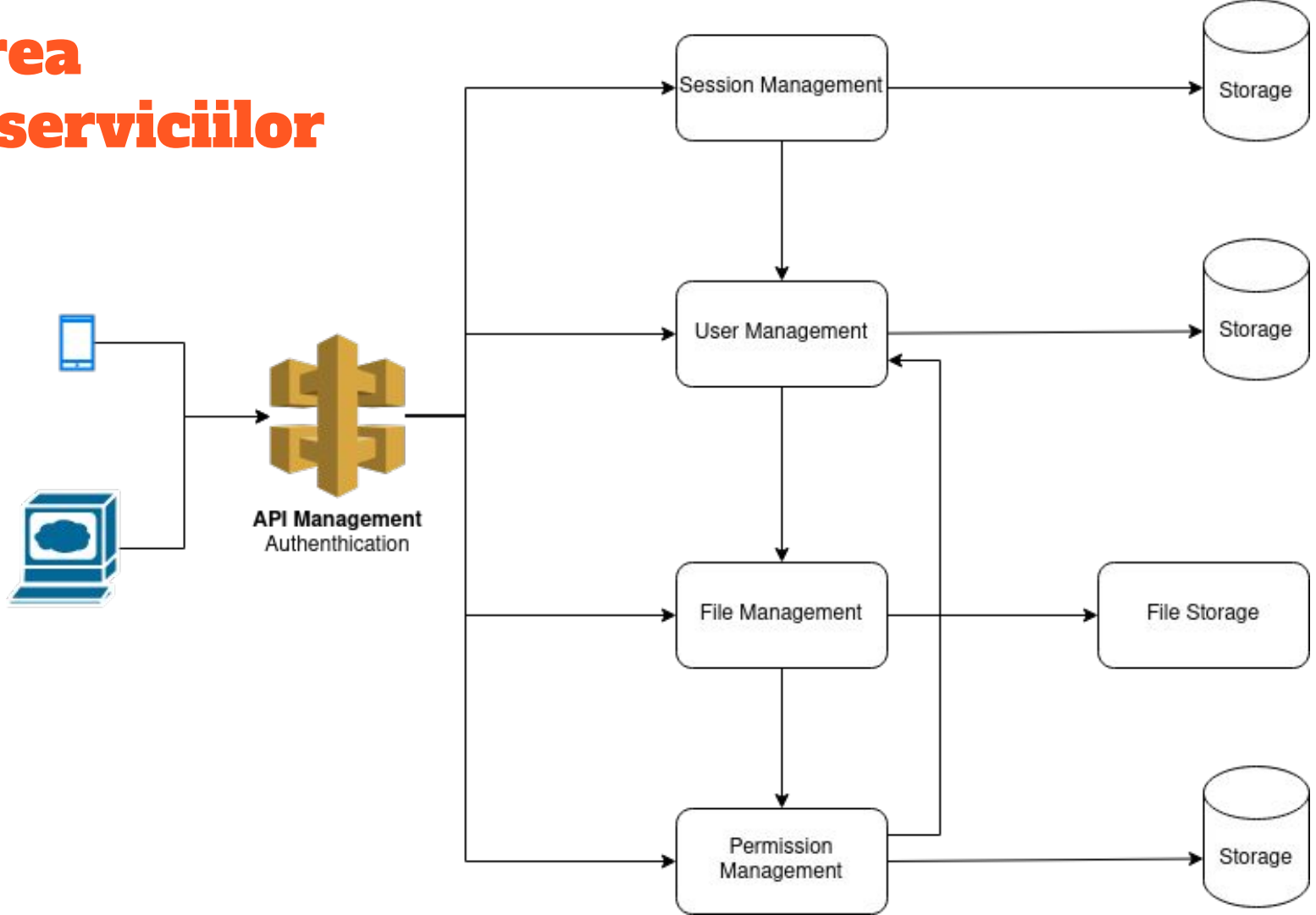
Identificarea Resurselor



Identificarea Resurselor



Definirea micro-serviciilor



Question Time #1

Trebuie un micro-serviciu să folosească stocare persistentă (ex, o bază de date)?

- ❏ Da, altfel există riscul să se piardă date și să existe incoerență între răspunsuri.
- ❏ Nu.

https://PollEv.com/cezarandrici171

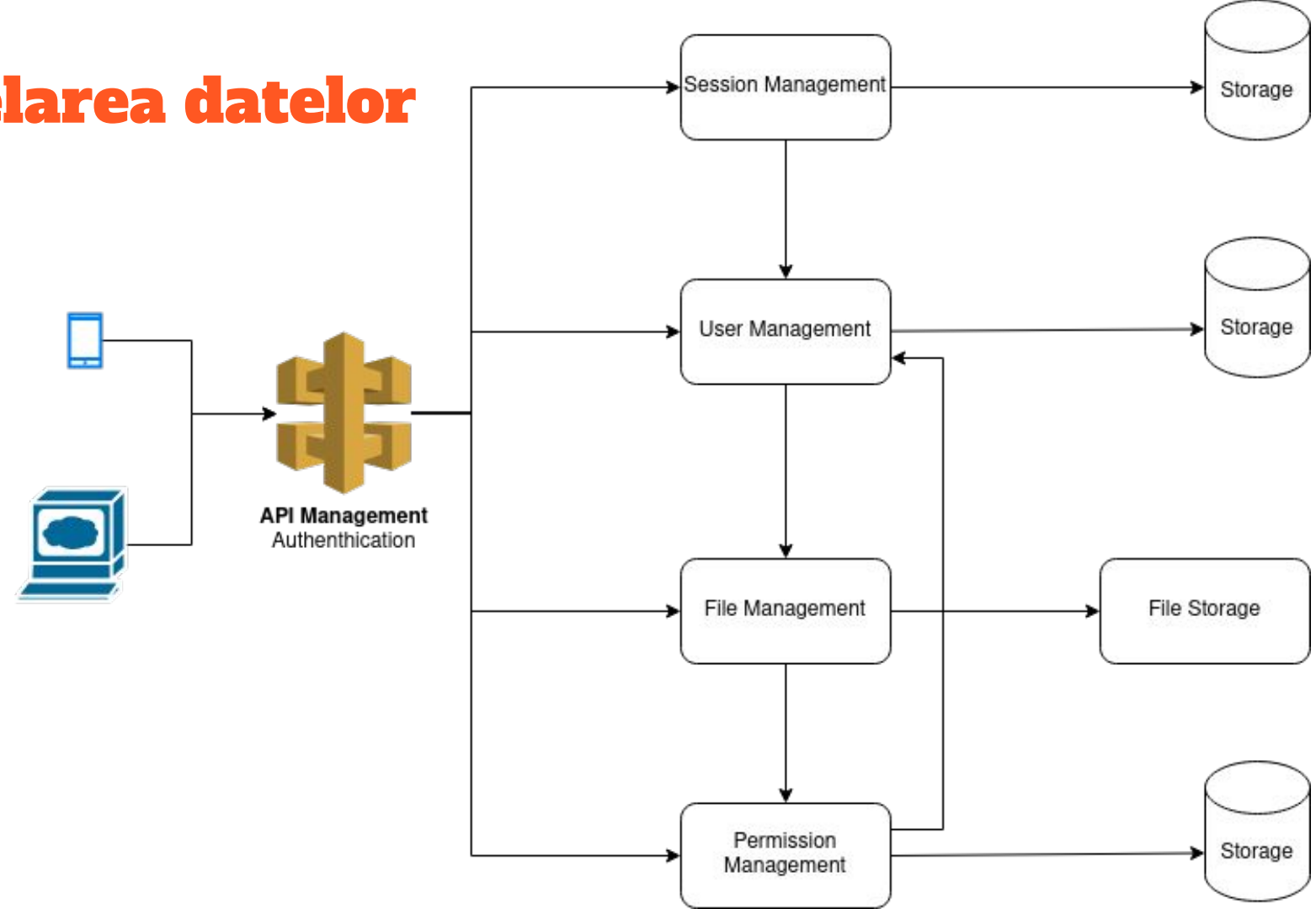
Question Time #2

Este recomandat ca un micro-serviciu să citească/modifice datele stocate a altui micro-serviciu?

- ❏ Da, este mai rapid decât un apel al unui API.
- ❏ Nu, un micro-serviciu trebuie sa fie independent de modul de stocare a altor micro-servicii.

https://PollEv.com/cezarandrici171

Modelarea datelor



Modelarea datelor

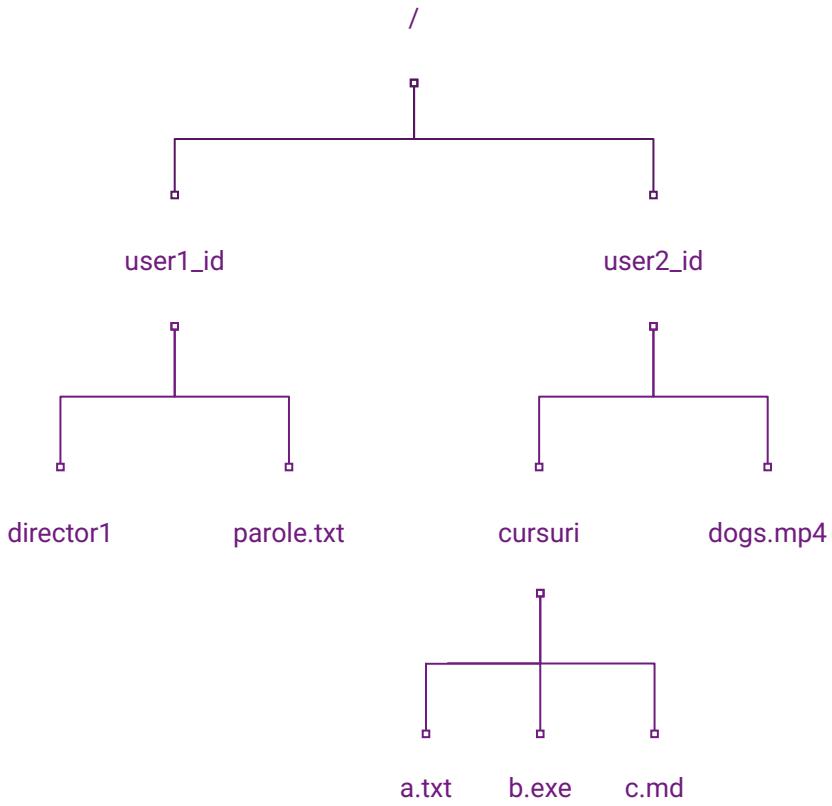
Identificator unic pentru fișiere - soluție simplă:

Fișierele sunt identificate printr-o cale (virtuală).

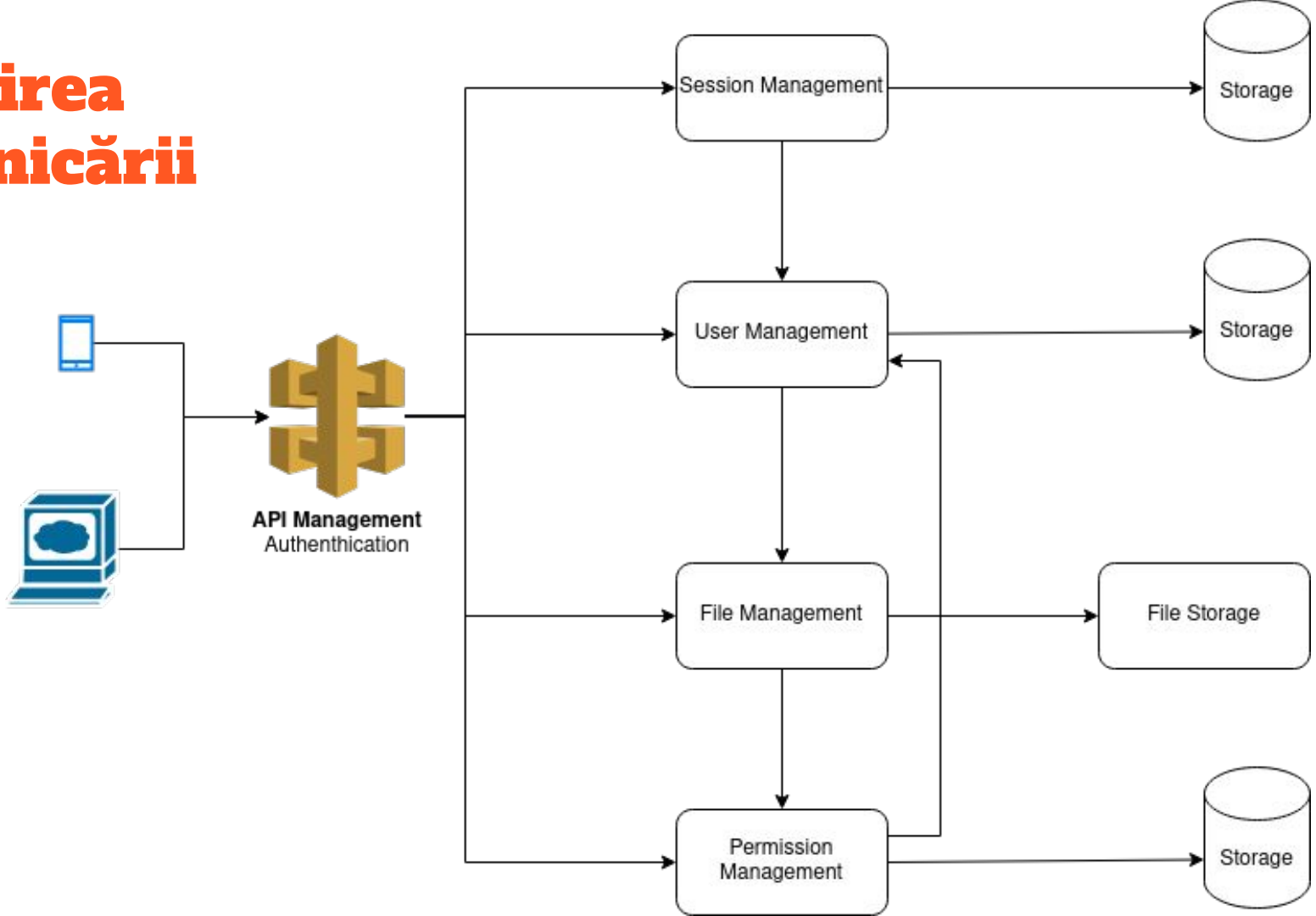
Aranjăm fișierele într-un arbore de directoare ierarhic, începând cu directorul rădăcină (/), care se ramifică în subdirectoare și fișiere.

Primul nivel este format din câte un director pentru fiecare utilizator.

Organizarea fișierelor în directorul utilizatorului este responsabilitatea utilizatorului.



Stabilirea comunicării



API Endpoints: file management

Create file
`POST /files/new`

Access path
`GET /files/:path_to_file_or_folder`

Update file
`PATCH /files/:path_to_file`

Delete file
`DELETE /files/:path_to_file`

Check if file exists
`HEAD /files/:path_to_file`

Index files readable by a user
`GET /files/shared_with/:user_id`

User management

Create new user
`POST /users/new`

Read user
`GET /users/:user_id`

Update user
`PATCH /users/:user_id`

Delete user
`DELETE /users/:user_id`

Session management

Create session (login)
`POST /sessions/new`

Read session
`GET /sessions/:session_id`

Update session
`PATCH /sessions/:session_id`

Delete session (logout)
`DELETE /sessions/:session_id`

API Endpoints: permissions management

Create permission
`POST /permissions`

Update permission
`PATCH /permissions/:permission_id`

Delete permission
`DELETE /permissions/:permission_id`

Index permissions granted by an user
`GET /permissions?granted_by=:user_id`

Index permissions granted to an user
`GET /permissions?to=:user_id`

Check if user is allowed to do an operation
`POST /permissions/check`

POST /permissions/check

Verifică dacă un utilizator are voie să acceseze/modifice un fișier/folder.

Request Body (JSON)

```
{
  "user_id": "string",
  "path": "string",
  "operation": "string"
}
```

Response Body (JSON)

```
{
  "permitted": bool,
  "message": "string"
}
```

Status codes:

- **200 OK**:
- **400 Bad Request**: dacă corpul cererii este incorect (de exemplu, lipsește ID-ul utilizatorului, calea sau operațiunea sau dacă conține valori invalidă/neacceptată).

Time to code

Implementați endpoint-ul:

```
POST /permissions/check
```

În fișierul `server.php`.

https://github.com/andricicezar/web-php

Extra Question

Când folosim HTTPS, este URI-ul vizibil altor dispozitive din rețea (cum ar fi alte computere, routere intermediare sau furnizorul de internet)?

Exemplu URI: `/files/5/poze/mydog.png`

- ❏ Da
- ❏ Nu

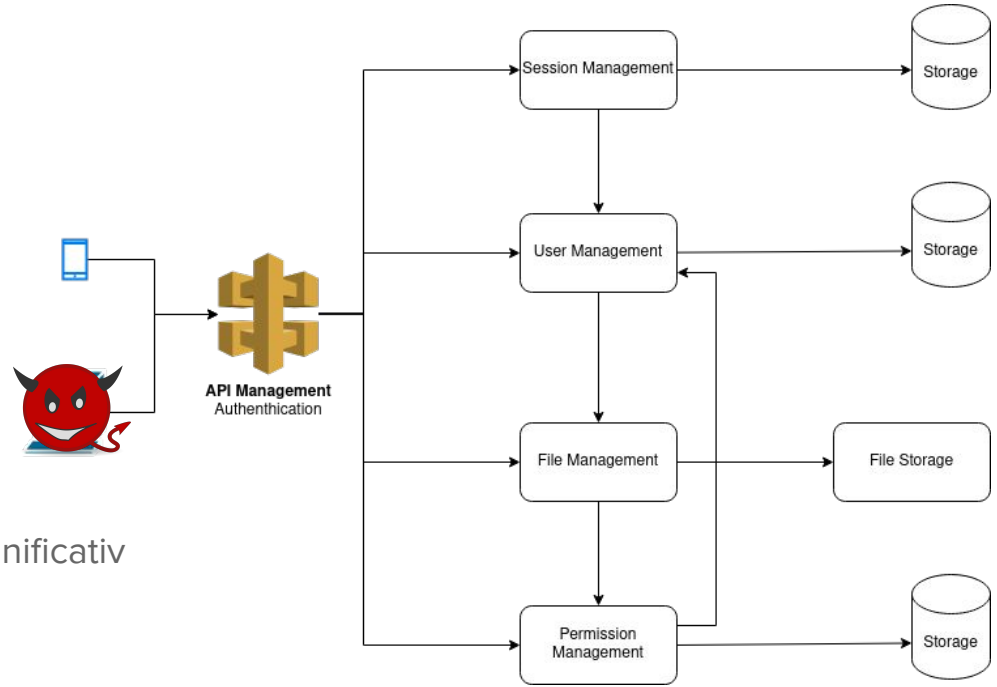
https://PollEv.com/cezarandrici171

Securitate și confidențialitate

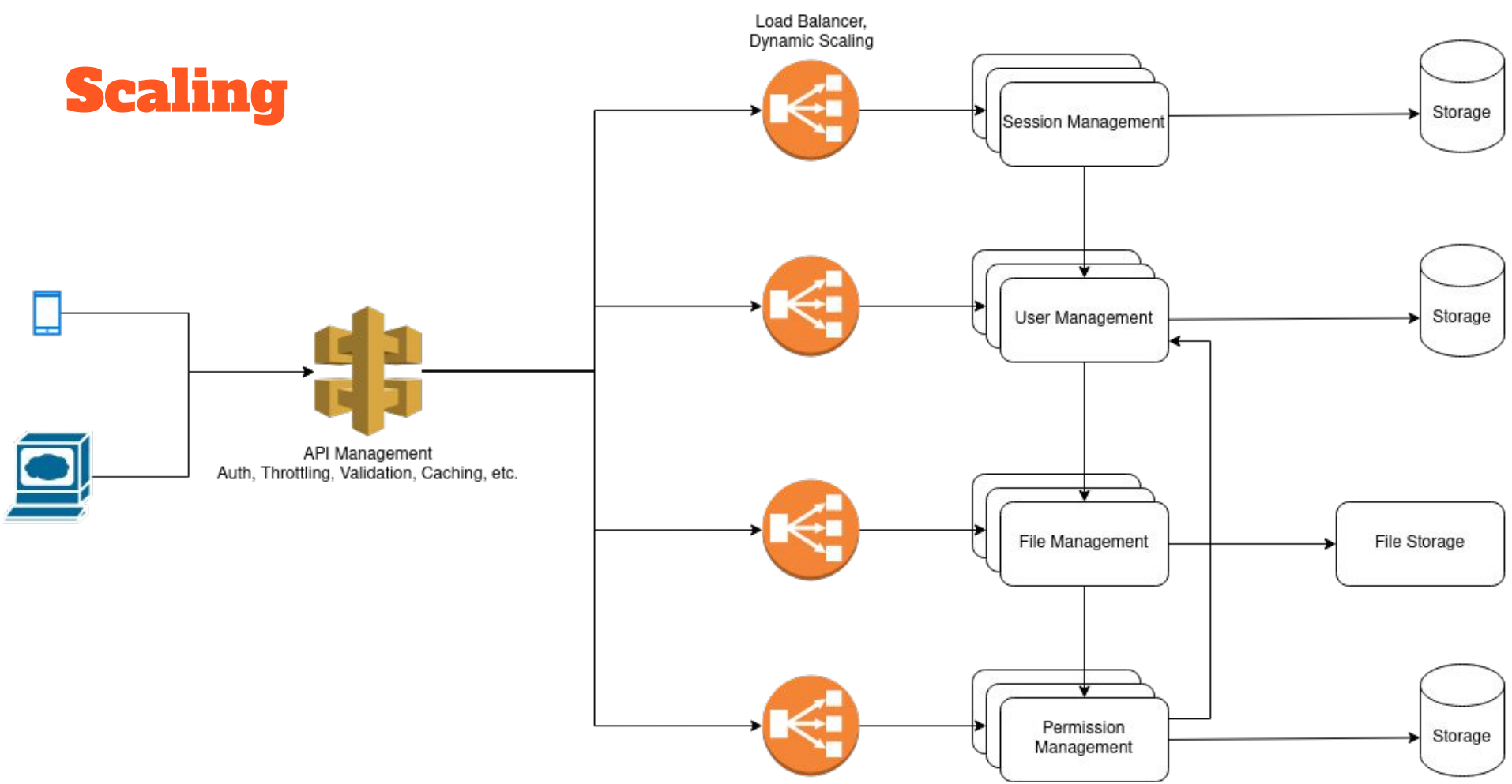
Token-uri imposibil de falsificat
- Risc: uzurparea identității (impersonate)
- Soluții sigure: JWT token

Calcularea permisiunilor corect
- Risc: acces/schimbări necontrolate ale fișierelor
- Soluții sigure: Cedar Language

Identificator pentru fișiere
- Risc: asocierea fișierelor cu utilizatorii
- Soluții sigure: atribuirea unui identificator nesemnificativ fiecărui fișier



Scaling


Load Balancer,
Dynamic Scaling
Session Management
Storage
User Management
Storage
File Management
File Storage
Permission
Management
Storage
API Management
Auth, Throttling, Validation, Caching, etc.