

Gestionare anunțuri auto

TEMA: L1. Proiectarea studiului de caz

Indicativ_echipa: SIA_07 (Casian Larisa, Ciuc Tiberiu, Nicsan Raluca)

Surse de date:

1) DS_1: PostgreSQL:

Tip sursă de date:

- Tip model de date: relațional
- Tip format de acces: SQL

Tabelul **posts.anunt**

- Descriere: Conține anunțuri auto postate de utilizatori.
- Câmpuri:
 - **id** – identificator unic (**SERIAL**, cheie primară)
 - **date** – dată publicare (**TIMESTAMP**, implicit data curentă)
 - **description** – descrierea anunțului (**TEXT**)
 - **kilometers** – număr de kilometri ai mașinii (**INTEGER**)
 - **userid** – identificatorul utilizatorului care a postat anunțul (**TEXT**)
 - **carid** – identificator al mașinii (**INTEGER**)

Tabelul **posts.comentarii**

- Descriere: Comentarii lăsate de utilizatori la anunțuri.
- Câmpuri:
 - **id** – identificator unic al comentariului (**SERIAL**, cheie primară)
 - **userid** – utilizatorul care a lăsat comentariul (**TEXT**)
 - **commenttext** – textul comentariului (**TEXT**)
 - **anuntid** – legătură către anunțul comentat (**INTEGER**, cheie externă către **posts.anunt(id)**, cu **ON DELETE CASCADE**)

Relații între tabele:

- 1:N între **posts.anunt** și **posts.comentarii**, prin **anuntid**.

```

DROP TABLE IF EXISTS posts.anunt;
DROP TABLE IF EXISTS posts.comentarii;

CREATE TABLE posts.anunt (
    id SERIAL PRIMARY KEY,
    date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    description TEXT NOT NULL,
    kilometers INTEGER NOT NULL,
    userid TEXT NOT NULL,
    carid INTEGER NOT NULL
);

CREATE TABLE posts.comentarii (
    id SERIAL PRIMARY KEY,
    userid TEXT NOT NULL,
    commenttext TEXT NOT NULL,
    anuntid INTEGER,
    CONSTRAINT fk_anuntid FOREIGN KEY (anuntid)
        REFERENCES posts.anunt(id)
        ON DELETE CASCADE
);

```

2) DS_2: Oracle:

Tip sursă de date:

- Tip model de date: Relațional
- Tip format de acces: SQL

Tabelul **car**

- Descriere: Reprezintă autoturismele disponibile în sistem. Fiecare înregistrare reprezintă o mașină cu detalii tehnice.
- Câmpuri:
 - **id** – identificator unic al mașinii (**NUMBER**, generat automat, cheie primară)
 - **manufacturer** – producătorul mașinii (**VARCHAR2(50)**)
 - **model** – modelul mașinii (**VARCHAR2(50)**)
 - **year** – anul de fabricație (**NUMBER**)
 - **engine_capacity** – capacitatea motorului în litri (**NUMBER(3,1)**)
 -

Relații:

- Nu există relații explicite în această sursă de date (tabel standalone).
- Această tabelă poate fi utilizată în legătură cu alte surse (ex: legătură logică cu `posts.anunt.carid` din PostgreSQL).

```
drop table car;
CREATE TABLE car (
  id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
  manufacturer VARCHAR2(50) NOT NULL,
  model VARCHAR2(50) NOT NULL,
  year NUMBER NOT NULL,
  engine_capacity NUMBER(3,1) NOT NULL
);
```

3) DS_3: MongoDB:

Tip sursă de date:

- Tip model de date: Document (NoSQL)
- Tip format de acces: JSON (interogare prin MongoDB)

Colecția `utilizatori`

- Descriere: Colecția stochează informații despre utilizatorii platformei.
- Structură document (schema implicită):
 - `_id` – identificator unic generat de MongoDB (`ObjectId`)
 - `id` – identificator numeric al utilizatorului (folosit pentru referință logică în alte sisteme)
 - `username` – numele de utilizator (`string`)
 - `birthday` – data nașterii a utilizatorului (`string` în format `YYYY-MM-DD`)

Observații:

- Nu există relații explicite între colecții în MongoDB, dar câmpul `id` poate fi utilizat pentru mapări logice către alte surse (ex: `userid` în PostgreSQL sau Oracle).
- Structura este flexibilă și poate fi extinsă ușor cu alte câmpuri (ex: email, telefon, adresă)

```

db.utilizatori.insertMany([
  { "_id": ObjectId(), "id": 1, "username": "john_doe", "birthday": "1990-05-15" },
  { "_id": ObjectId(), "id": 2, "username": "jane_smith", "birthday": "1988-09-22" },
  { "_id": ObjectId(), "id": 3, "username": "mike_jones", "birthday": "1995-07-10" },
  { "_id": ObjectId(), "id": 4, "username": "anna_brown", "birthday": "1993-03-05" },
  { "_id": ObjectId(), "id": 5, "username": "sam_wilson", "birthday": "1985-12-30" }
]);
|

```

TEMA: L2. Arhitectura sistemului federativ de integrare

1) DS_1: PostgreSQL

- Format / Tip de acces: **JSON** (prin REST API expus cu PostgREST)
- Mecanism de acces:
 - Datele din PostgreSQL sunt expuse prin **PostgREST**, care generează un API RESTful peste schema [posts](#).
 - Oracle consumă acest API folosind [HTTPURITYPE](#) pentru a prelua datele în format JSON și [JSON_TABLE](#) pentru a le transforma într-o structură relațională locală.

```

DROP VIEW anunt_view;

CREATE OR REPLACE VIEW anunt_view AS
WITH rest_doc AS (
  SELECT HTTPURITYPE.createuri('http://localhost:3000/anunt').getclob() AS doc
  FROM dual
)
SELECT
  id,
  TO_DATE(date_str, 'YYYY-MM-DD') AS date_posted,
  description,
  kilometers,
  userid,
  carid
FROM JSON_TABLE(
  (SELECT doc FROM rest_doc), '$[*]'
  COLUMNS (
    id          NUMBER          PATH '$.id',
    date_str    VARCHAR2(20)    PATH '$.date',
    description  VARCHAR2(400)  PATH '$.description',
    kilometers   NUMBER         PATH '$.kilometers',
    userid      VARCHAR2(50)    PATH '$.userid',
    carid       NUMBER          PATH '$.carid'
  )
);

SELECT * FROM anunt_view;

```

2) DS_2: MongoDB

- Format / Tip de acces: JSON (primit prin REST API, server RESTHeart)
- Mecanism de acces:
 - Se utilizează un serviciu REST expus de **RESTHeart** peste colecția **users** din MongoDB
 - Accesul este realizat din Oracle cu ajutorul unei funcții personalizate (**get_restheart_data_media**), care gestionează autentificarea și returnează conținutul JSON
 - Datele JSON sunt convertite într-o structură relațională Oracle prin **JSON_TABLE**.

```
drop function get_restheart_data_media;
CREATE OR REPLACE FUNCTION get_restheart_data_media(pURL VARCHAR2, pUserPass VARCHAR2)
RETURN clob IS
    l_req UTL_HTTP.req;
    l_resp UTL_HTTP.resp;
    l_buffer clob;
begin
    l_req := UTL_HTTP.begin_request(pURL);
    UTL_HTTP.set_header(l_req, 'Authorization', 'Basic ' ||
        UTL_RAW.cast_to_varchar2(UTL_ENCODE.base64_encode(UTL_I18N.string_to_raw(pUserPass, 'AL32UTF8'))));
    l_resp := UTL_HTTP.get_response(l_req);
    UTL_HTTP.READ_TEXT(l_resp, l_buffer);
    UTL_HTTP.end_response(l_resp);
    return l_buffer;
end;
/
--SELECT get_restheart_data_media('http://localhost:8080/utilizatori', 'admin:secret') from dual;
--SELECT HTTPURITYPE.createuri('http://admin:secret@localhost:8080/utilizatori').getclob() as doc from dual;

drop view utilizatori_view_mongodb;
CREATE OR REPLACE VIEW utilizatori_view_mongodb AS
WITH json AS
    (SELECT get_restheart_data_media('http://localhost:9090/users', 'admin:secret') doc FROM dual)
SELECT DISTINCT
    user_id, username, birthday
FROM JSON_TABLE(
    (SELECT doc FROM json),
    '$[*]'
    COLUMNS (
        user_id PATH '$.id'
        , username PATH '$.username'
        , birthday PATH '$.birthday' |
    )
);

SELECT * FROM utilizatori_view_mongodb;
```

TEMA: L3. Implementare views

(1) Nivel CONSOLIDARE date

1) View_Consolidare_1: Anunțuri cu utilizatori și mașini

Surse de date integrate:

- DS_1: PostgreSQL (view-ul `anunt_view` cu date despre anunțuri)
- DS_2: Oracle (tabela `car` cu detalii despre mașini)
- DS_3: MongoDB (view-ul `utilizatori_view_mongodb` cu date despre utilizatori)

```
drop view anunturi_cu_useri_si_masini;  
CREATE OR REPLACE VIEW anunturi_cu_useri_si_masini AS  
SELECT  
    c.manufacturer AS car_manufacturer,  
    c.model        AS car_model,  
    a.kilometers   AS milage,  
    c.engine_capacity AS engine_capacity,  
    us.username    AS posted_by,  
    a.description AS description  
FROM  
    anunt_view a  
JOIN  
    car c ON a.carid = c.id  
JOIN  
    utilizatori_view_mongodb us ON a.userid = us.user_id;  
  
select * from anunturi_cu_useri_si_masini;
```

Este realizat un JOIN între aceste surse pentru a crea un view care oferă o imagine completă a anunțurilor, inclusiv detaliile despre mașină și utilizatorul care a postat anunțul.

2) View_Consolidare_2: Activitate utilizatori

Surse de date integrate:

- DS_1: PostgreSQL (view-ul comentarii_view)
- DS_1: PostgreSQL (view-ul anunt_view)
- DS_3: MongoDB (view-ul utilizatori_view_mongodb)

```
drop view user_activity;
CREATE OR REPLACE VIEW user_activity AS
WITH user_comments AS (
    SELECT
        c.userid AS user_id,
        COUNT(c.id) AS num_comments
    FROM comentarii_view c
    GROUP BY c.userid
),
user_anunturi AS (
    SELECT
        a.userid AS user_id,
        COUNT(a.id) AS num_anunturi
    FROM anunt_view a
    GROUP BY a.userid
),
usernames AS (
    SELECT
        user_id,
        username
    FROM utilizatori_view_mongodb
)
SELECT
    u.user_id, un.username,
    COALESCE(uc.num_comments, 0) AS num_comments,
    COALESCE(ua.num_anunturi, 0) AS num_anunturi
FROM (
    SELECT DISTINCT userid AS user_id FROM comentarii_view
    UNION
    SELECT DISTINCT userid AS user_id FROM anunt_view
) u
LEFT JOIN user_comments uc ON u.user_id = uc.user_id
LEFT JOIN user_anunturi ua ON u.user_id = ua.user_id
LEFT JOIN usernames un ON u.user_id = un.user_id;

select * from user_activity;
```

View-ul `user_activity` agregă numărul de comentarii și anunțuri postate de fiecare utilizator.

3) Tabela_de_fapte_1: Anunțuri per producător

Surse de date integrate:

- DS_2: Oracle (tabela `car` cu detalii despre mașini)
- DS_1: PostgreSQL (view-ul `anunt_view` cu anunțuri)

```
delete CAR_DET;  
CREATE OR REPLACE VIEW CAR_DET AS  
SELECT  
    c.manufacturer,  
    COUNT(a.id) AS total_anunturi,  
    AVG(a.kilometers) AS avg_kilometers,  
    MIN(a.kilometers) AS min_kilometers,  
    MAX(a.kilometers) AS max_kilometers  
FROM  
    car c  
JOIN  
    anunt_view a ON c.id = a.carid  
GROUP BY  
    c.manufacturer;  
  
select * from car_manufacturer_summary;
```

Acest view agregă datele despre anunțurile de vânzare a mașinilor în funcție de producător, calculând numărul total de anunțuri, media, minimumul și maximumul de kilometri pentru fiecare marcă de mașină.

4) Tabela_de_fapte_2: Activitate utilizatori per mașină

Surse de date integrate:

- DS_3: MongoDB (view-ul `utilizatori_view_mongodb`)
- DS_1: PostgreSQL (view-ul `anunt_view`)
- DS_2: Oracle (tabela `car`)


```

drop view USER_CAR_DET;
CREATE OR REPLACE VIEW USER_CAR_DET AS
SELECT
    u.username,
    COUNT(a.id) AS num_anunturi,
    AVG(a.kilometers) AS avg_kilometers,
    MAX(c.year) AS latest_car_year
FROM
    utilizatori_view_mongodb u
JOIN
    anunt_VIEW a ON u.user_id = a.userid
JOIN
    car c ON a.carid = c.id
GROUP BY
    u.username;

select * from USER_CAR_DET;

```

Acest view analitic sumarizează activitatea utilizatorilor în legătură cu anunțurile lor auto, inclusiv numărul de anunțuri, kilometrajul mediu al acestora și anul celei mai noi mașini postate de utilizator.

TEMA: P3. REST and Web Model

1) View_Resursa_REST_1: anunturi_cu_useri_si_masini

- Tabela/View :
 - ANUNTURI_CU_USERI_SI_MASINI
- Detalii de implementare:
 - Acest view combină anunțurile cu informațiile despre utilizatori și mașini.
 - Datele sunt accesibile printr-un API RESTful.

```

BEGIN
    ORDS.ENABLE_SCHEMA(p_enabled => TRUE,
        p_schema => 'auto_data',
        p_url_mapping_type => 'BASE_PATH',
        p_url_mapping_pattern => 'auto',
        p_auto_rest_auth => FALSE);

    COMMIT;
END;
/

```

```

BEGIN
ORDS.ENABLE_OBJECT(
  p_enabled      => TRUE,
  p_schema       => 'auto_data',
  p_object        => 'ANUNTURI_CU_USERI_SI_MASINI',
  p_object_type   => 'VIEW',
  p_object_alias  => 'anunturi_cu_useri_si_masini',
  p_auto_rest_auth => FALSE
);
COMMIT;
END;
/

```

URL resursa RESTful: localhost:8080/ords/auto/utilizatori/

```

{
  "items": [
    {
      "car_manufacturer": "Honda",
      "car_model": "Civic",
      "mileage": 150000,
      "engine_capacity": 1.6,
      "posted by": "john doe",
      "description": "Selling a Toyota Corolla, well maintained."
    },
    {
      "car_manufacturer": "Toyota",
      "car_model": "Corolla",
      "mileage": 90000,
      "engine_capacity": 1.8,
      "posted by": "john doe",
      "description": "Audi A4 2015 available, diesel engine."
    },
    {
      "car_manufacturer": "Audi",
      "car_model": "A4",
      "mileage": 120000,
      "engine_capacity": 2,
      "posted by": "jane smith",
      "description": "Honda Civic first owner, good condition."
    },
    {
      "car_manufacturer": "Honda",
      "car_model": "Civic",
      "mileage": 50000,
      "engine_capacity": 1.6,
      "posted by": "anna brown",
      "description": "BMW X5 2018, luxury SUV."
    },
    {
      "car_manufacturer": "BMW",
      "car_model": "X5",
      "mileage": 80000,
      "engine_capacity": 3,
      "posted by": "sam wilson",
      "description": "Ford Fiesta, reliable and economic."
    }
  ],
  "hasMore": false,
  "limit": 25,
  "offset": 0,
  "count": 5,
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:8080/ords/auto/anunturi_cu_useri_si_masini/"
    },
    {
      "rel": "describedby",
      "href": "http://localhost:8080/ords/auto/metadata-catalog/anunturi_cu_useri_si_masini/"
    },
    {
      "rel": "first",
      "href": "http://localhost:8080/ords/auto/anunturi_cu_useri_si_masini/"
    }
  ]
}

```

2) View_Resursa_REST_2: user_activity

- Tabela/View: **USER_ACTIVITY**
- (Acest view agregă activitatea utilizatorilor, calculând numărul de anunțuri și comentarii)

```

BEGIN
ORDS.ENABLE_OBJECT(
  p_enabled      => TRUE,
  p_schema       => 'auto_data',
  p_object       => 'USER_ACTIVITY',
  p_object_type  => 'VIEW',
  p_object_alias => 'activity',
  p_auto_rest_auth => FALSE
);
COMMIT;
END;
/

```

URL resursa RESTful: localhost:8080/ords/auto/activity/

The screenshot shows a web browser window with the address bar displaying `localhost:8080/ords/auto/activity/`. The browser's developer tools are open, showing the response of a REST API call. The response is a JSON object with the following structure:

```

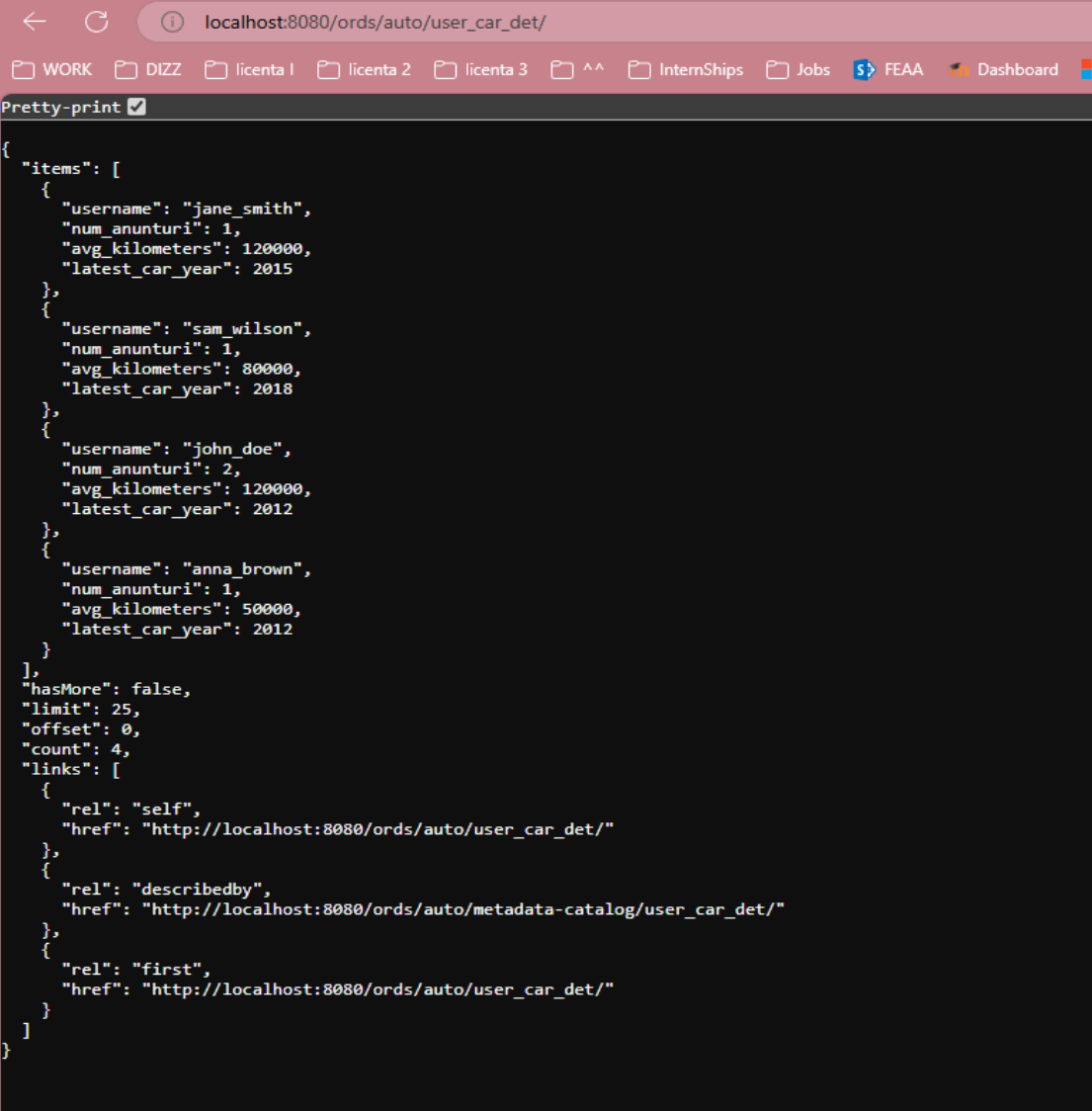
{
  "items": [
    {
      "user_id": "1",
      "username": "john_doe",
      "num_comments": 2,
      "num_anunturi": 2
    },
    {
      "user_id": "2",
      "username": "jane_smith",
      "num_comments": 2,
      "num_anunturi": 1
    },
    {
      "user_id": "3",
      "username": "mike_jones",
      "num_comments": 1,
      "num_anunturi": 0
    },
    {
      "user_id": "4",
      "username": "anna_brown",
      "num_comments": 0,
      "num_anunturi": 1
    },
    {
      "user_id": "5",
      "username": "sam_wilson",
      "num_comments": 0,
      "num_anunturi": 1
    }
  ],
  "hasMore": false,
  "limit": 25,
  "offset": 0,
  "count": 5,
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:8080/ords/auto/activity/"
    },
    {
      "rel": "describedby",
      "href": "http://localhost:8080/ords/auto/metadata-catalog/activity/"
    },
    {
      "rel": "first",
      "href": "http://localhost:8080/ords/auto/activity/"
    }
  ]
}

```

3) View_Resursa_REST_3: user_car_det

- Tabela/View: `USER_CAR_DET` (view care detaliază anunțurile postate de utilizatori, incluzând informații despre mașinile lor).

```
BEGIN
ORDS.ENABLE_OBJECT (
  p_enabled      => TRUE,
  p_schema       => 'auto_data',
  p_object        => 'CAR_DET',
  p_object_type   => 'VIEW',
  p_object_alias  => 'car_det',
  p_auto_rest_auth => FALSE
);
COMMIT;
END;
```



The screenshot shows a web browser window with the address bar displaying `localhost:8080/ords/auto/user_car_det/`. The browser's developer tools are open, showing a JSON response from the RESTful API. The response is formatted with "Pretty-print" checked. The JSON data includes a list of users with their usernames, the number of announcements, average kilometers, and the latest car year. It also includes pagination information like "hasMore", "limit", "offset", and "count", as well as a "links" section with "self", "describedby", and "first" links.

```
{
  "items": [
    {
      "username": "jane_smith",
      "num_anunturi": 1,
      "avg_kilometers": 120000,
      "latest_car_year": 2015
    },
    {
      "username": "sam_wilson",
      "num_anunturi": 1,
      "avg_kilometers": 80000,
      "latest_car_year": 2018
    },
    {
      "username": "john_doe",
      "num_anunturi": 2,
      "avg_kilometers": 120000,
      "latest_car_year": 2012
    },
    {
      "username": "anna_brown",
      "num_anunturi": 1,
      "avg_kilometers": 50000,
      "latest_car_year": 2012
    }
  ],
  "hasMore": false,
  "limit": 25,
  "offset": 0,
  "count": 4,
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:8080/ords/auto/user_car_det/"
    },
    {
      "rel": "describedby",
      "href": "http://localhost:8080/ords/auto/metadata-catalog/user_car_det/"
    },
    {
      "rel": "first",
      "href": "http://localhost:8080/ords/auto/user_car_det/"
    }
  ]
}
```

URL resursa RESTfull: localhost:8080/ords/auto/user_car_det/

4) View_Resursa_REST_4: car_det

- Tabela/View: [CAR_DET](#) (view care agregă informațiile legate de mașini, incluzând totalul anunțurilor și diverse statistici despre kilometri)

```
BEGIN
ORDS.ENABLE_OBJECT(
  p_enabled      => TRUE,
  p_schema       => 'auto_data',
  p_object       => 'USER_CAR_DET',
  p_object_type  => 'VIEW',
  p_object_alias => 'user_car_det',
  p_auto_rest_auth => FALSE
);
COMMIT;
END;
/
```

URL resursa RESTfull: localhost:8080/ords/auto/car_det/

```
localhost:8080/ords/auto/car_det/

WORK DIZZ licenta 1 licenta 2 licenta 3 ^^ InternShips Jobs FEAA Dashb

Pretty-print ☒

{
  "items": [
    {
      "manufacturer": "Honda",
      "total_anunturi": 2,
      "avg_kilometers": 100000,
      "min_kilometers": 50000,
      "max_kilometers": 150000
    },
    {
      "manufacturer": "Toyota",
      "total_anunturi": 1,
      "avg_kilometers": 90000,
      "min_kilometers": 90000,
      "max_kilometers": 90000
    },
    {
      "manufacturer": "Audi",
      "total_anunturi": 1,
      "avg_kilometers": 120000,
      "min_kilometers": 120000,
      "max_kilometers": 120000
    },
    {
      "manufacturer": "BMW",
      "total_anunturi": 1,
      "avg_kilometers": 80000,
      "min_kilometers": 80000,
      "max_kilometers": 80000
    }
  ],
  "hasMore": false,
  "limit": 25,
  "offset": 0,
  "count": 4,
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:8080/ords/auto/car_det/"
    },
    {
      "rel": "describedby",
      "href": "http://localhost:8080/ords/auto/metadata-catalog/car_det/"
    },
    {
      "rel": "first",
      "href": "http://localhost:8080/ords/auto/car_det/"
    }
  ]
}
```

Implementare Interfață Web cu HTML și JavaScript pentru Accesarea API-urilor

Pagină web cu HTML și JavaScript pentru a permite utilizatorilor să acceseze și să vizualizeze date din API-urile RESTful expuse prin ORDS, incluzând anunțuri, activitatea utilizatorilor, și detalii despre mașini.

Structura HTML și CSS:

- Pagină de bază HTML cu titlul "Auto API".
- Patru butoane pentru a accesa diverse resurse: Anunțuri, Statistica user activity, Mașini și Kilometraj, Detalii auto per user.
- Pagină cu tabele ce vor conține datele returnate de la API.

JavaScript:

- Patru funcționalități pe fiecare buton:
 1. **Butoanele** fac apeluri GET la adresele API specificate.
 2. **Datele returnate** din API sunt transformate în tabele HTML și sunt adăugate în pagină.
 3. **Erori**: Orice eroare în apelurile fetch sunt logate în consolă.

Apeluri la API:

- Fiecare buton face o cerere la o adresă diferită (de exemplu, http://localhost:8080/ords/auto/anunturi_cu_useri_si_masini/).
- API-ul returnează un obiect JSON, iar datele sunt afișate într-un tabel HTML pe pagină.

Rulare server local:

- Se utilizează `python -m http.server` pentru a rula pagina local pe server, iar browser-ul va putea accesa pagina prin <http://localhost:8000>

localhost:8000/index.html

WORKDZZlicenta 1licenta 2licenta 3InterShipsJobsFEAADashboardMail - NICSAN I. RA...LeetCode - The Wor...netX editFeed | LinkedInAI, Niccan Raluca, SL...Knowledge - 148 cl...Other favorites

Auto API

AnunturiStatistica user activityMasini si KilometrajDetalii auto per user

Producător	Model	Kilometraj	Capacitate motor	Postat de	Descriere
Honda	Civic	150000	1.6	john_doe	Selling a Toyota Corolla, well maintained.
Toyota	Corolla	90000	1.8	john_doe	Audi A4 2015 available, diesel engine.
Audi	A4	120000	2	jane_smith	Honda Civic first owner, good condition.
Honda	Civic	50000	1.6	anna_brown	BMW X5 2018, luxury SUV.
BMW	X5	80000	3	sam_wilson	Ford Fiesta, reliable and economic.

localhost:8000/index.html

WORKDZZlicenta 1licenta 2licenta 3InterShipsJobsFEAADashboardMail - NICSAN I. RA...LeetCode - The Wor...netX editFeed | LinkedInAI, Niccan Raluca, SL...Knowledge - 148 cl...Other favorites

Auto API

AnunturiStatistica user activityMasini si KilometrajDetalii auto per user

User ID	Username	Numar comentarii	Numar anunturi
1	john_doe	2	2
2	jane_smith	2	1
3	mike_jones	1	0
4	anna_brown	0	1
5	sam_wilson	0	1

localhost:8000/index.html

WORKDZZlicenta 1licenta 2licenta 3InterShipsJobsFEAADashboardMail - NICSAN I. RA...LeetCode - The Wor...netX editFeed | LinkedInAI, Niccan Raluca, SL...Knowledge - 148 cl...Other favorites

Auto API

AnunturiStatistica user activityMasini si KilometrajDetalii auto per user

Producător	Total anunțuri	Kilometraj mediu	Kilometraj minim	Kilometraj maxim
Honda	2	100000	50000	150000
Toyota	1	90000	90000	90000
Audi	1	120000	120000	120000
BMW	1	80000	80000	80000

localhost:8000/index.html

WORKDZZlicenta 1licenta 2licenta 3InterShipsJobsFEAADashboardMail - NICSAN I. RA...LeetCode - The Wor...netX editFeed | LinkedInAI, Niccan Raluca, SL...Knowledge - 148 cl...Other favorites

Auto API

AnunturiStatistica user activityMasini si KilometrajDetalii auto per user

Utilizator	Total anunțuri	Kilometraj mediu	Anul mașinii
jane_smith	1	120000	2015
sam_wilson	1	80000	2018
john_doe	2	120000	2012
anna_brown	1	50000	2012