

PRÀCTICA P4

DTD:

Exercici 1:

The screenshot shows the Apache NetBeans IDE interface. The main window displays an XML document with the following DTD code:

```
<!DOCTYPE llibreria [
    <!ELEMENT llibreria (llibres, revistes)*>
    <!ELEMENT llibres (llibre*)>
    <!ELEMENT llibre EMPTY>
    <!ATTLIST llibre
        titol CDATA #REQUIRED
        autor CDATA #IMPLIED>
    <!ELEMENT revistes (revista*)*>
    <!ELEMENT revista EMPTY*>
    <!ATTLIST revista
        nom CDATA #REQUIRED
        periodicitat (diari | setmanal | mensual) #IMPLIED>
]
```

The code editor has syntax highlighting and line numbers. Below the editor is an "Output - XML check" panel showing validation results:

```
XML validation started.
Checking file:/C:/Users/Bemen3/llenguatge%20de%20marques/exercici_01.xml...
Referenced entity at "file:/C:/Users/Bemen3/llenguatge%20de%20marques/llibreria.dtd".
C:\Users\Bemen3\llenguatge de marques\llibreria.dtd [El sistema no puede encontrar el archivo especificado.] [2]
XML validation finished.
```

The desktop taskbar at the bottom shows various application icons and system status.

dtd

<!DOCTYPE llibreria [

Aquest document XML fa referència a una DTD incorporada que diu com ha de ser l'estructura de l'element principal.

<!ELEMENT llibreria (llibres, revistes)>

Defineix com ha de ser l'estructura de l'array dins del DTD, dient que ha de tenir només dos elements: **<llibres>** i **<revistes>**.

<!ELEMENT llibres (llibre*)>

Dins del DTD, es defineix com ha de ser l'element **<llibres>**. (*) vol dir que l'element es pot repetir o pot no sortir.

<!ELEMENT llibre EMPTY>

Especifica que l'element <llibre> és buit, o sigui, no té contingut ni elements dins, i tota la informació es posa en atributs.

<!ATTLIST llibre> defineix els atributs que porta l'element a la DTD. Això indica quins atributs són obligatoris, opcionals o tenen valors per defecte.

CDATA #REQUIRED

Dins de la definició <!ATTLIST llibre>, es diu que l'atribut titol és obligatori per a cada element i que ha de ser de tipus CDATA.

CDATA #IMPLIED

Vol dir que és un text opcional. Es pot deixar en blanc sense que l'XML sigui incorrecte.

Després hem creat atributs semblants pero en comptes de per l'element llibre per l'element revista.

L'atribut periodicitat, que es pot posar com (diari | setmanal | mensual) #IMPLIED, és opcional. Si el poses, ha de ser un d'aquests tres valors: diari, setmanal o mensual. Així, només es poden fer servir aquests valors.

The screenshot shows the Apache NetBeans IDE interface. The main window displays an XML file named 'exercici_01.xml'. The code is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE llibreria SYSTEM "exercici_01.dtd">
<llibreria>
    <llibre>
        <titol>El Senyor dels Anells</titol>
        <autor>J.R.R. Tolkien</autor>
    <llibre>
        <titol>1984</titol>
        <autor>George Orwell</autor>
    <llibre>
        <titol>El Quixot</titol>
    </llibres>
    <revistes>
        <revista nome="National Geographic" periodicitat="mensual"/>
        <revista nome="La Vanguardia" periodicitat="diari"/>
        <revista nome="Time" periodicitat="setmanal"/>
    </revistes>
</llibreria>
```

The bottom right corner of the IDE shows the system tray with the date and time (15/01/2025, 11:58), battery level (8%), and network status (Soleado).

The 'Output - XML check' window shows the XML validation results:

```
XML validation started.
Checking file:/C:/Users/Bemen3/lienguatge%20de%20margues/exercici_01.xml...
Referenced entity at "file:/C:/Users/Bemen3/lienguatge%20de%20margues/exercici_01.dtd".
The markup declarations contained or pointed to by the document type declaration must be well-formed. [2]
XML validation finished.
```

xml

El document XML és vàlid segons la DTD. Llibreria conté llibres i revistes, amb atributs obligatoris i opcionals

Exercici 2:

The screenshot shows the Apache NetBeans IDE 24 interface. The main window displays an XML file named 'exercici_02.xml' with the following content:

```
<!ELEMENT universitat (estudiant*, professor*)>
<!ELEMENT estudiant (nom, cognom)>
<!ELEMENT professor (nom, cognom)>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT cognom (#PCDATA)>
<!ATTLIST professor
  categoria CDATA #REQUIRED>
<!ATTLIST estudiant
  rol (delegat | subdelegat) #IMPLIED>
```

The output window titled 'Output - XML check' shows the results of an XML validation:

```
XML validation started.
Checking file:/C:/Users/Bemen3/llenguatge%20de%20marques/exercici_02.xml...
Referenced entity at "file:/C:/Users/Bemen3/llenguatge%20de%20marques/exercici_02.dtd".
XML validation finished.
```

dtd

```
<!ELEMENT universitat (estudiant*, professor*)>
<!ELEMENT estudiant (nom, cognom)>
<!ELEMENT professor (nom, cognom)>
```

Aquests fragments defineixen com ha de ser un document XML on l'element principal és `<universitat>`, que pot tenir 0 o més elements `<estudiant>` i `<professor>`. Tant els `<estudiant>` com els `<professor>` han de tenir sí o sí dos elements dins: `<nom>` i `<cognom>`.

```
<!ELEMENT nom (#PCDATA)>
<!ELEMENT cognom (#PCDATA)>
```

Aquests fragments diuen que tant `<nom>` , `<cognom>` només poden tenir text i no poden incloure altres elements ni atributs..

<!ATTLIST professor categoria CDATA #REQUIRED>

Aquesta part diu que l'element <professor> ha de tenir un atribut anomenat categoria. Aquest atribut és de tipus text (CDATA) i és obligatori perquè està marcat com a #REQUIRED.

<!ATTLIST estudiant rol (delegat | subdelegat) #IMPLIED>

Aquesta secció diu que l'element <estudiant> pot tenir un atribut opcional anomenat rol. Aquest atribut només pot ser "delegat" o "subdelegat". Com que està marcat com #IMPLIED, no cal posar-lo.

The screenshot shows the Apache NetBeans IDE interface. The main window displays an XML file named 'exercici_02.xml'. The code is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE universitat PUBLIC "-//EXAMPLE//DTD Universitat//EN" "exercici_02.dtd">
<universitat>
    <estudiant rol="delegat">
        <nom>Joan</nom>
        <cognom>Pérez</cognom>
    </estudiant>
    <estudiant>
        <nom>Maria</nom>
        <cognom>López</cognom>
    </estudiant>
    <professor categoria="Matemàtiques">
        <nom>Carles</nom>
        <cognom>Garcia</cognom>
    </professor>
    <professor categoria="Història">
        <nom>Carles</nom>
        <cognom>Martí</cognom>
    </professor>
</universitat>
```

Below the code, there are two tabs: 'universitat' and 'estudiant'. An output window titled 'Output - XML check' shows the validation process:

```
XML validation started.
Checking file:///C:/Users/Bemen3/llenguatge%20de%20marques/exercici_02.xml...
Referenced entity at "file:///C:/Users/Bemen3/llenguatge%20de%20marques/exercici_02.dtd".
XML validation finished.
```

xml

Aquest document XML organitza una universitat. Té un element principal <universitat> que inclou <estudiant> i <professor>. Els estudiants poden tenir un atribut opcional rol i els professors han de tenir un atribut obligatori categoria. Tots han de tenir <nom> i <cognom>.

Exercici 3

```
<!ELEMENT vehicles (vehicle+)>
<!ELEMENT vehicle ((cotxe | moto), comentari?)>
<!ELEMENT cotxe (#PCDATA)>
<!ELEMENT moto (#PCDATA)>
<!ELEMENT comentari (#PCDATA)>
```

Output - XML check

```
XML validation started.
Checking file:///C:/Users/Bemen3/lenguatge%20de%20marques/exercici_02.xml...
Referenced entity at "file:///C:/Users/Bemen3/lenguatge%20de%20marques/exercici_02.dtd".
XML validation finished.
```

dtd

<!ELEMENT vehicles (vehicle+)>

L'element <vehicles> ha de tenir almenys un <vehicle> (1 o més).

<!ELEMENT vehicle ((cotxe | moto), comentari?)>

Cada <vehicle> ha de tenir un <cotxe> o una <moto> i pot incloure opcionalment un <comentari>.

<!ELEMENT cotxe (#PCDATA)>

L'element <cotxe> només pot contenir text.

<!ELEMENT moto (#PCDATA)>

L'element <moto> només pot contenir text.

<!ELEMENT comentari (#PCDATA)>

L'element <comentari> només pot contenir text.

The screenshot shows the Apache NetBeans IDE interface. In the top menu bar, the following items are visible: File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help, and Apache NetBeans IDE 24. Below the menu is a toolbar with various icons. The main workspace contains several tabs: Exercici_03.dtd, Exercici_03.xml (selected), Exercici_04.dtd, Exercici_04.xml, Exercici_05.dtd, Exercici_05.xml, Exercici_06.dtd, Exercici_06.xml, Exercici_07.dtd, Exercici_07.xml, Exercici_08.dtd, Exercici_08.xml, and Exercici_09.dtd. The code editor displays the following XML content:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE vehicles SYSTEM "Exercici_03.dtd">
<vehicles>
    <vehicle>
        <cotxe>BMW</cotxe>
    </vehicle>
    <vehicle>
        <moto>Yamaha</moto>
        <comentari>Nou</comentari>
    </vehicle>
</vehicles>
```

Below the code editor is a status bar with the letter 'I'. To the right of the code editor is a vertical scroll bar. At the bottom of the screen is a taskbar with various application icons. An 'Output - XML check' window is open, showing the following validation results:

```
XML validation started.
Checking file:/C:/Users/Bemen3/llenguatge%20de%20marques/exercici_02.xml...
Referenced entity at "file:/C:/Users/Bemen3/llenguatge%20de%20marques/exercici_02.dtd".
XML validation finished.
```

The desktop background features a large watermark reading "Edim Batalla".

xml

Aquest és l'XML que ens dona l'enunciat de l'exercici. Al codi es defineix una llista de `<vehicle>` dins de `<vehicles>`. Cada `<vehicle>` pot ser un `<cotxe>` o una `<moto>`, i pot tenir opcionalment un `<comentari>`.

Exercici 4

The screenshot shows the Apache NetBeans IDE 24 interface. The main window displays an XML DTD file named 'Exercici_03.dtd' with the following content:

```
<!ELEMENT biblioteca (llibre+)>
<!ELEMENT llibre (titol, autor)>
<!ATTLIST llibre
  isbn ID #REQUIRED
  any #CDATA #IMPLIED>
<!ELEMENT titol (#PCDATA)>
<!ELEMENT autor (#PCDATA)>
```

The 'Output - XML check' window at the bottom shows the results of an XML validation:

```
XML validation started.
Checking file:/C:/Users/Bemen3/llenguatge%20de%20marques/exercici_02.xml...
Referenced entity at "file:/C:/Users/Bemen3/llenguatge%20de%20marques/exercici_02.dtd".
XML validation finished.
```

dtd

<biblioteca> té una llista d'element <llibre>.

<llibre> té un atribut obligatori, l'isbn, i un atribut opcional, l'any. També ha de tenir els elements <titol> i <autor>, que només poden tenir text.

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE biblioteca SYSTEM "Exercici_04.dtd">
<biblioteca>
    <llibre isbn="id0001" any="2020">
        <titol>El nom del vent</titol>
        <autor>Patrick Rothfuss</autor>
    </llibre>
    <llibre isbn="id0002">
        <titol>1984</titol>
        <autor>George Orwell</autor>
    </llibre>
    <llibre isbn="id003" any="2018">
        <titol>Origen</titol>
        <autor>Dan Brown</autor>
    </llibre>
</biblioteca>

```

Output - XML check

```

XML validation started.
Checking file:/C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_04.xml...
Referenced entity at "file:/C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_04.dtd".
XML validation finished.

```

xml

- **Atributs:**

- **isbn:** obligatori i únic per a cada llibre (de tipus ID).
- **any:** opcional, indica l'any de publicació del llibre.

- **Elements fills:**

- **<titol>:** conté el títol del llibre en text.
- **<autor>:** conté el nom de l'autor, també en text.

Exercici 5

The screenshot shows the Apache NetBeans IDE 24 interface. The main window displays a DTD file named 'Exercici_05.dtd' with the following content:

```
<!ELEMENT noticia (#PCDATA | cita | lloc | data | autor)*>
<!ELEMENT cita (#PCDATA)>
<!ATTLIST cita
  font CDATA #REQUIRED>
<!ELEMENT lloc (#PCDATA)>
<!ATTLIST lloc
  pais CDATA #REQUIRED>
<!ELEMENT data (#PCDATA)>
<!ELEMENT autor (#PCDATA)>
<!ATTLIST autor
  rol CDATA #REQUIRED|
```

The 'Output - XML check' window below shows the results of XML validation:

```
XML validation started.
Checking file:/C:/Users/Bemen3/llienguatge%20de%20marques/exercici_02.xml...
Referenced entity at "file:/C:/Users/Bemen3/llienguatge%20de%20marques/exercici_02.dtd".
XML validation finished.
```

dtd

<!ELEMENT noticia (#PCDATA | cita | lloc | data | autor)*>

Aquesta part defineix l'element `<noticia>`, que pot tenir text (#PCDATA) i qualsevol nombre d'elements `<cita>`, `<lloc>`, `<data>` i `<autor>`, en qualsevol ordre. El símbol * permet que aquests elements es repeteixin o no apareguin, però sempre amb text entre ells. Això fa que `<noticia>` sigui un element mixt.

<!ELEMENT cita (#PCDATA)>

l'element `<cita>` com un element que només pot contenir text.

<!ATTLIST cita font CDATA #REQUIRED>

Defineix que l'element `<cita>` té un atribut obligatori `font`, que conté text (tipus CDATA) i indica la font de la cita.

<!ELEMENT lloc (#PCDATA)>
<!ATTLIST lloc pais CDATA #REQUIRED>

```

<!ELEMENT data (#PCDATA)>
<!ELEMENT autor (#PCDATA)>
<!ATTLIST autor rol CDATA #REQUIRED>

```

L'element <lloc> només té text i un atribut obligatori, pais, que indica el país. L'element <data> també té només text, que és la data. L'element <autor> té text i un atribut obligatori, rol, que diu quin és el paper de l'autor a la notícia.

The screenshot shows the Apache NetBeans IDE interface. The main window displays an XML file named 'Exercici_05.xml'. The code is as follows:

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE noticia SYSTEM "Exercici_05.dtd">
<noticia>
    El president va dir:
    <cita font="Agència Internacional">Aquest és un gran dia per a la nostra
    organització.</cita>
    <lloc pais="Espanya">Barcelona</lloc>
    Publicat el <data>2025-01-03</data> per
    <autor rol="periodista">Laura Martínez</autor>
</noticia>

```

Below the code editor, there is an 'Output - XML check' window showing the validation results:

```

XML validation started.
Checking file:/C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_05.xml...
Referenced entity at "file:/C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_05.dtd".
XML validation finished.

```

The desktop taskbar at the bottom shows various application icons, including a weather widget indicating 10°C and a date/time indicator showing 12:11 on 16/01/2025.

xml

El XML descriu una notícia amb text i elements organitzats. <cita> inclou la cita amb l'atribut font. <lloc> indica el lloc amb l'atribut pais. <data> mostra la data de publicació, i <autor> identifica l'autor amb l'atribut rol. És una estructura clara per a una notícia.

Exercici 6

```

<!ELEMENT classe (nom, alumnus)*>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT alumnus (alumne+)>
<!ELEMENT alumne EMPTY>
<!ATTLIST alumne
    id ID #REQUIRED
    nom CDATA #IMPLIED>

```

Output - XML check ×

XML validation started.
Checking file:/C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_06.xml...
Referenced entity at "file:/C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_06.dtd".
XML validation finished.

dtd

Hem afegit la declaració de l'atribut nom com a opcional (#IMPLIED).

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE classe SYSTEM "Exercici_06.dtd">
<classe>
    <nom:Matemàtiques></nom>
    <alumnus>
        <alumne nom="Joan" id="a1"/>
        <alumne nom="Anna" id="a2"/>
    </alumnus>
</classe>

```

Output - XML check ×

XML Validation started.
Checking file:/C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_06.xml...
Referenced entity at "file:/C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_06.dtd".
XML validation finished.

xml

Hem posat l'atribut id al segon <alumne> i hem canviat els valors d'id perquè comencin amb una lletra (a1, a2), seguint les regles de NCName.

Exercici 7

The screenshot shows the Apache NetBeans IDE 24 interface. The main window displays an XML file named 'Exercici_07.dtd'. The code is as follows:

```
<!ELEMENT projecte (nom, equips)*>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT equips (equip+)*>
<!ELEMENT equip (nom, membres)*>
<!ELEMENT membres (membre+)*>
<!ELEMENT membre (#PCDATA)>
<!ATTLIST membre nom CDATA #REQUIRED rol (developer | tester | manager) #REQUIRED>
```

Below the code editor is an 'Output - XML check' panel showing the results of an XML validation:

```
XML validation started.
Checking file:/C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_07.xml...
Referenced entity at "file:/C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_07.dtd".
XML Validation finished.
```

The desktop taskbar at the bottom shows various application icons, including a search bar and system status indicators like weather and date.

dtd

L'arxiu DTD que ens dona l'enunciat sembla estar bé, el hem creat dins de NetBeans i ara modificarem l'arxiu XML.

The screenshot shows the Apache NetBeans IDE 24 interface. The main window displays an XML file named 'Exercici_07.xml' with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE projecte SYSTEM "Exercici_07.dtd">
<projecte>
    <nom>Desenvolupament de Software</nom>
    <equips>
        <equip>
            <nom>Backend</nom>
            <membres>
                <membre nom="Joan" rol="developer"/>
                <membre nom="Maria" rol="manager"/>
            </membres>
        </equip>
        <equip>
            <nom>QA</nom>
            <membres>
                <membre nom="Anna" rol="tester"/>
            </membres>
        </equip>
    </equips>
</projecte>
```

The status bar at the bottom indicates the XML validation process:

```
XML validation started.
Checking file:/C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_07.xml...
Referenced entity at "file:/C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_07.dtd".
XML validation finished.
```

xml

A l'arxiu XML, hem fet dues modificacions per fer-lo vàlid segons el DTD. Al primer <equip>, hem afegit l'atribut nom="Maria" al segon <membre> perquè és obligatori. Al segon <equip>, hem afegit <nom> amb el valor QA i hem creat un element <membres> que inclou el <membre>, complint amb el DTD.

Exercici 8

```

<!ELEMENT empresa (# empleat+)>
<!ELEMENT empleat (# projecte*)>
<!ATTLIST empleat
  id ID #REQUIRED
  nom CDATA #REQUIRED
  posicio CDATA #REQUIRED>
<!ELEMENT projecte EMPTY>
<!ATTLIST projecte
  id CDATA #REQUIRED
  nom CDATA #REQUIRED>

```

Output - XML check ×

```

XML validation started.
Checking file:///C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_08.xml...
Referenced entity at "file:///C:/Users/Bemen3/llenguatge%20de%20marques/Exercici_08.dtd".
XML validation finished.

```

dtd

L'element <empresa> té una llista d'empleats (<empleat+>). Cada <empleat> pot tenir zero o més projectes assignats (<projecte*>). Els atributs d'<empleat> són:

ID: Obligatori i únic (de tipus ID).

Nom: Obligatori i de tipus text.

Posició: Obligatori i de tipus text.

Els atributs de <projecte> són:

ID: Obligatori i únic (de tipus ID).

Nom: Obligatori i de tipus text.

The screenshot shows the Apache NetBeans IDE 24 interface. The main window displays an XML file named 'Exercici_08.xml' with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE empresa SYSTEM "Exercici_08.dtd">
<empresa>
    <empleat id="e1" nom="Joan" posicio="Developer">
        <projecte id="p1" nom="WebApp"/>
        <projecte id="p2" nom="MobileApp"/>
    </empleat>
    <empleat id="e2" nom="Anna" posicio="Manager">
        <projecte id="p1" nom="WebApp"/>
    </empleat>
    <empleat id="e3" nom="Carlos" posicio="Tester">
        <projecte id="p3" nom="API"/>
    </empleat>
</empresa>
```

The code editor has tabs for multiple files: Exercici_04.xml, Exercici_05.dtd, Exercici_06.xml, Exercici_06.dtd, Exercici_07.xml, Exercici_07.dtd, Exercici_08.xml, Exercici_09.xml, and exercici_01.dtd... .

The bottom right corner of the IDE window shows the system tray with icons for battery, signal, and date/time (16/01/2025).

Below the IDE, the Windows taskbar is visible with various application icons.

A separate window titled 'Output - XML check' is open, showing the results of an XML validation:

```
XML validation started.
Checking file:///C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_08.xml...
Referenced entity at "file:///C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_08.dtd".
XML validation finished.
```

xml

Es defineixen tres empleats amb els atributs id, nom i posició. Cada empleat té projectes assignats amb els atributs id i nom. Els projectes poden ser compartits entre empleats.

Exercici 9

```
<!ELEMENT empresa (departament+)>
<!ELEMENT departament (empleat+)>
<!ATTLIST departament
      nom CDATA #REQUIRED>
<!ELEMENT empleat (nom, email?)>
<!ATTLIST empleat
      id ID #REQUIRED
      posicio CDATA #REQUIRED>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT email (#PCDATA)>
```

Output - XML check

```
XML validation started.
Checking file:///C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_09.xml...
Referenced entity at "file:///C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_09.dtd".
XML validation finished.
```

dtd

<empresa> té un o més <departament>, com a mínim un.
<departament> té l'atribut obligatori nom i conté un o més <empleat>.
<empleat> té dos atributs obligatoris: id (únic per a cada empleat) i posicio.
Conté <nom> (obligatori) i <email> (opcional).
<nom> i <email> només tenen text.

The screenshot shows the Apache NetBeans IDE 24 interface. The main window displays an XML file named 'Exercici_09.xml' with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE empresa SYSTEM "Exercici_09.dtd">
<empresa>
    <departament nom="HR">
        <empleat id="e1" posicio="Manager">
            <nom>Maria</nom>
            <email>maria@empresa.com</email>
        </empleat>
    </departament>
    <departament nom="IT">
        <empleat id="e2" posicio="Developer">
            <nom>Juan</nom>
        </empleat>
    </departament>
</empresa>
```

The 'Output - XML check' panel shows the results of XML validation:

```
XML validation started.
Checking file:/C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_09.xml...
Referenced entity at "file:/C:/Users/Bemen3/lenguatge%20de%20marques/Exercici_09.dtd".
XML validation finished.
```

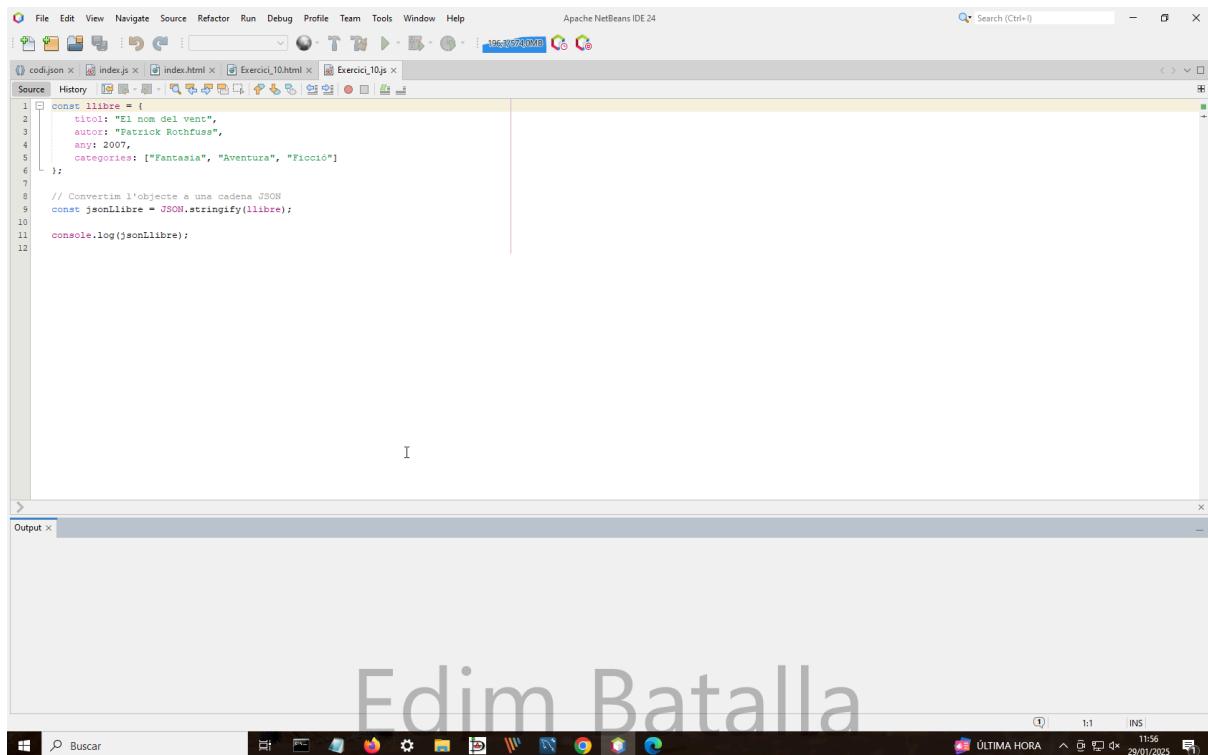
xml

L'exercici ens proporciona aquest arxiu xml, l'hem implementat utilitzant el NetBeans.

JSON:

Exercici 10:

Captura javascript:



```
const llibre = {
    titol: "El nom del vent",
    autor: "Patrick Rothfuss",
    any: 2007,
    categories: ["Fantasia", "Aventura", "Ficció"]
};

// Convertim l'objecte a una cadena JSON
const jsonLlibre = JSON.stringify(llibre);

console.log(jsonLlibre);
```

Aquí es declara una variable constant **const llibre ={}** que conté un objecte JavaScript.

“titol”: És una cadena de text que representa el nom del llibre.

“autor”: És una altra cadena de text que especifica qui ha escrit el llibre.

“any”: Es tracta d'un número enter que indica l'any de publicació del llibre.

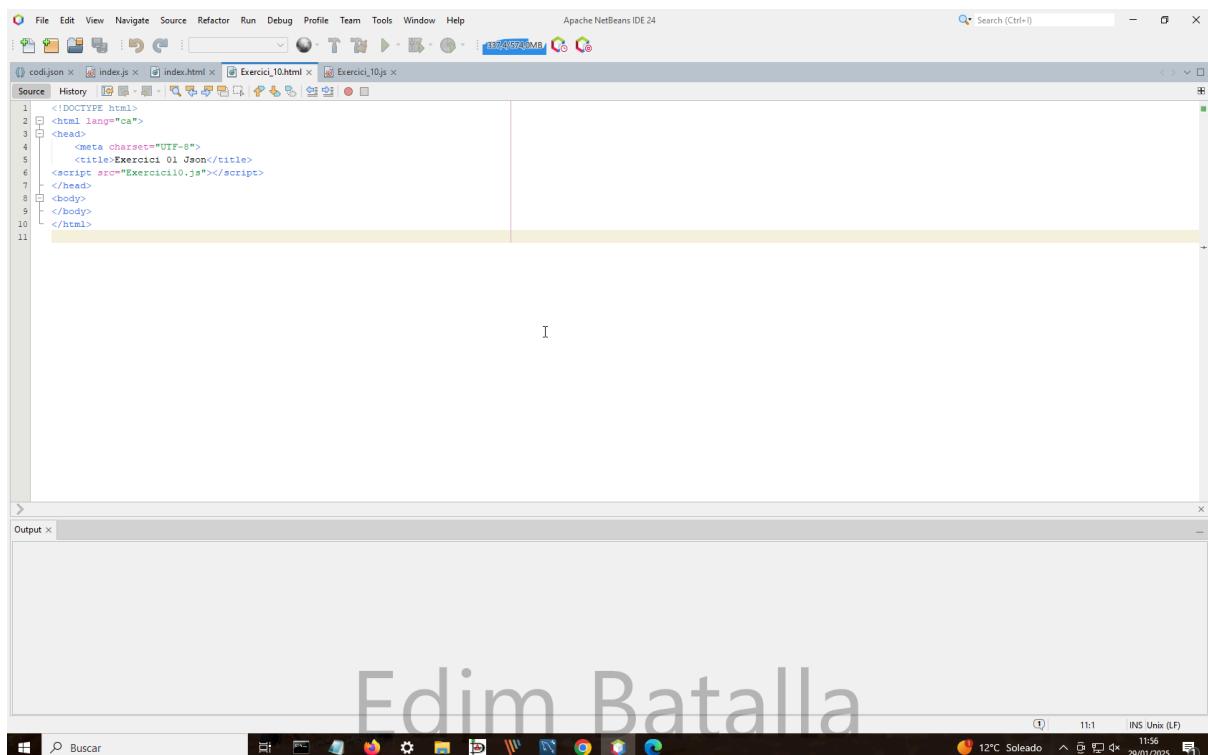
“categories”: És un array de cadenes de text que categoritza el llibre segons el seu gènere.

JSON.stringify(llibre); transforma l'objecte llibre en una cadena JSON.

jsonLlibre ara conté un string en format JSON.

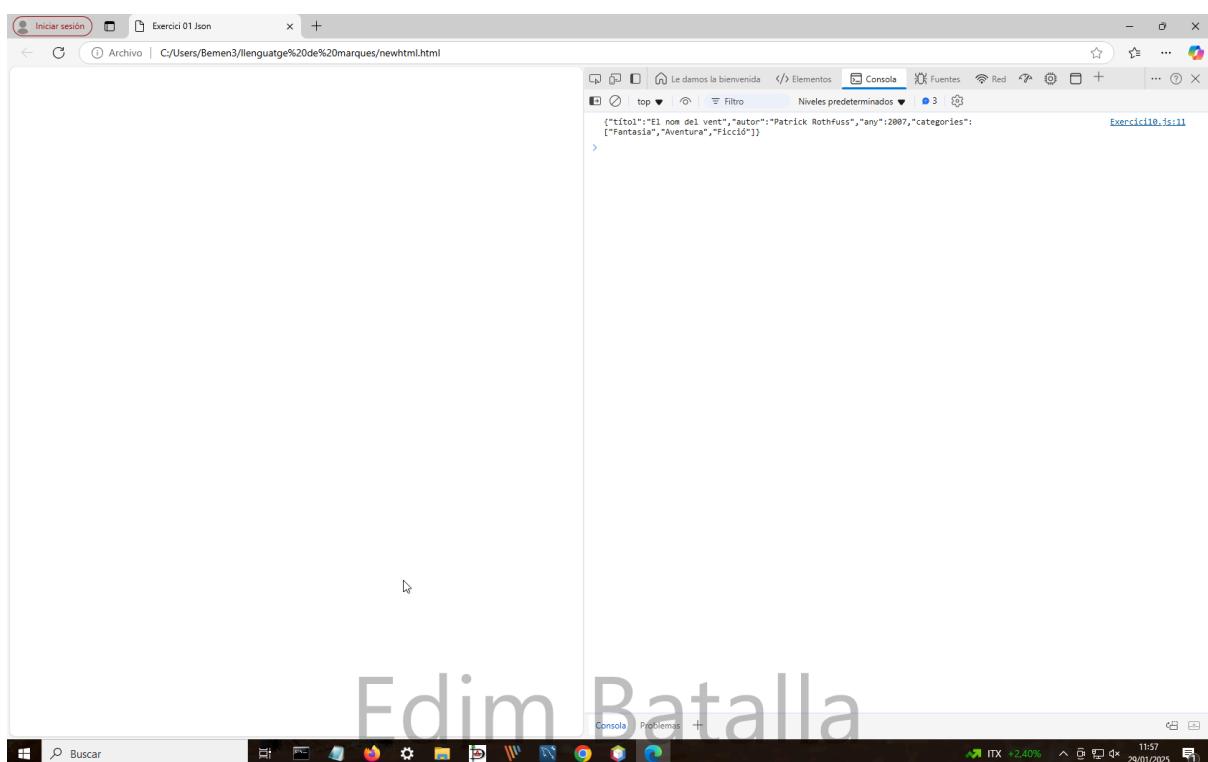
console.log(jsonLlibre); imprimeix la cadena JSON a la consola del navegador o terminal.

Captura html



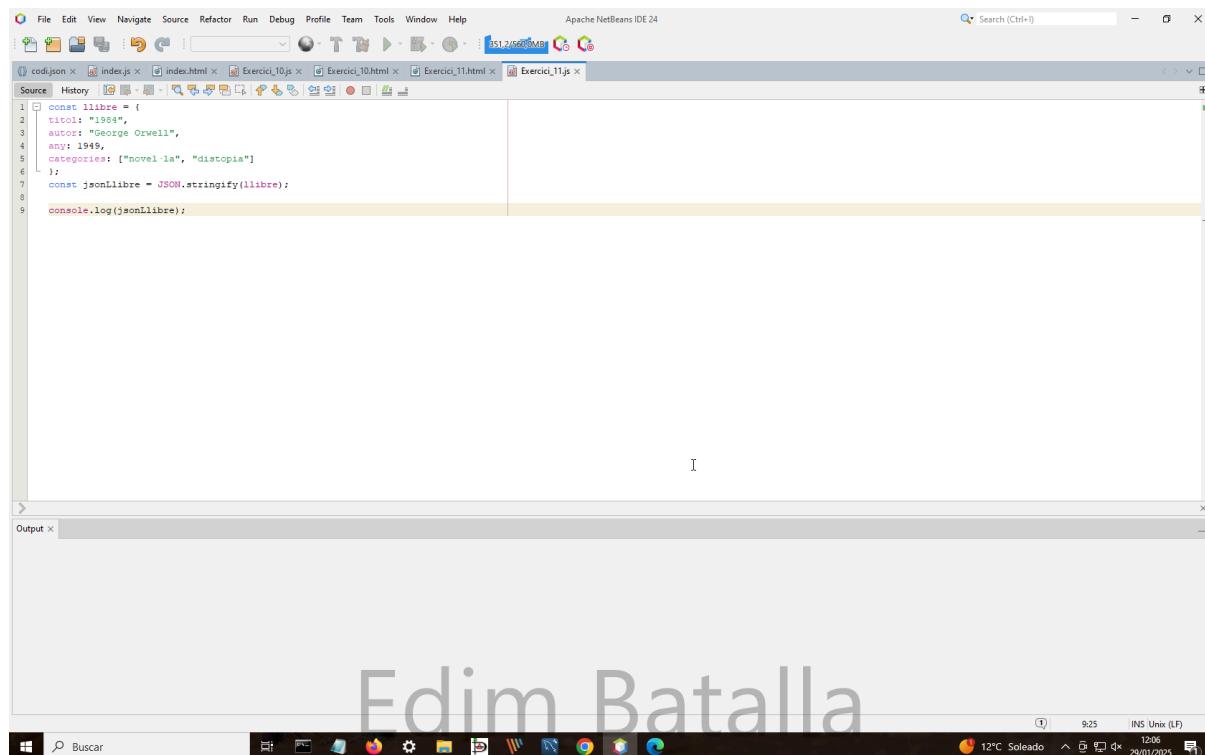
Creem un arxiu html bàsic on vinculem l'arxiu javascript amb la linea:
<script src="Exercici10.js"></script>

Captura consola



Exercici 11:

Arxiu JavaScript:



```
const llibre = {
    titol: "1984",
    autor: "George Orwell",
    any: 1949,
    categories: ["novel·la", "distopia"]
};
const jsonLlibre = JSON.stringify(llibre);

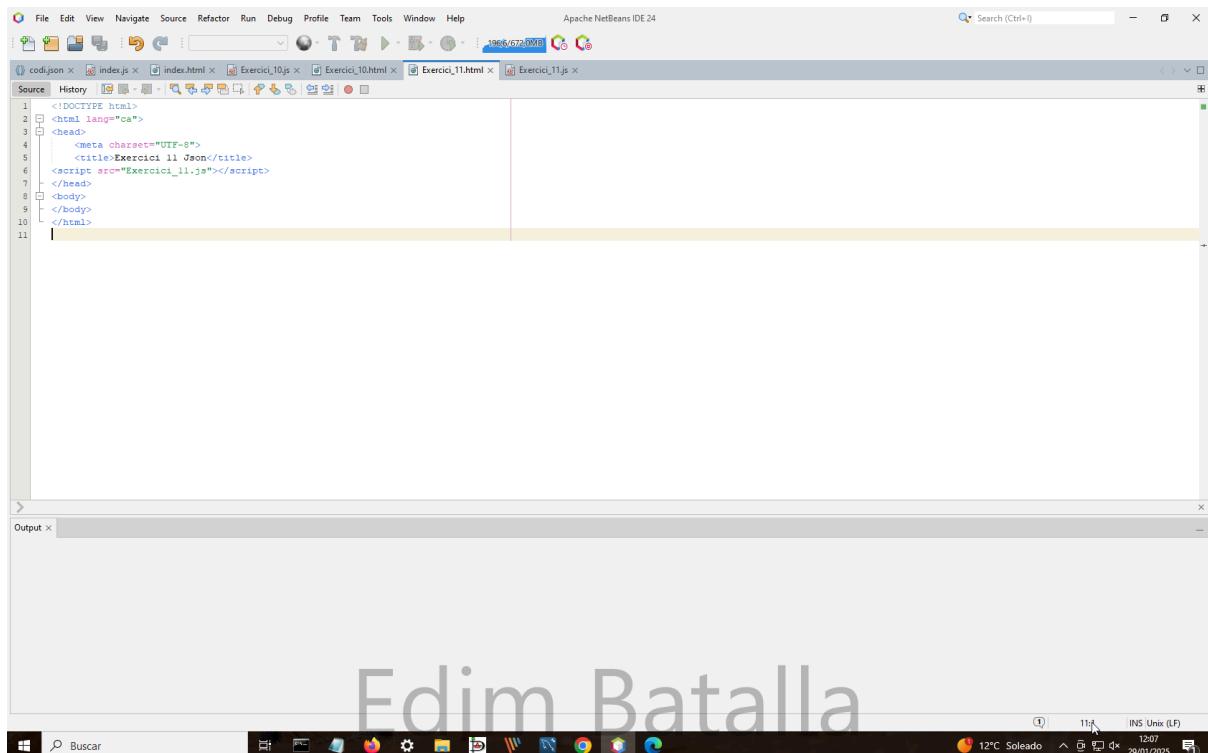
console.log(jsonLlibre);
```

L'exercici ens dona el seguent codi del arxiu JavaScript, l'implementem:

```
const llibre = {
    titol: "1984",
    autor: "George Orwell",
    any: 1949,
    categories: ["novel·la", "distopia"]
};
const jsonLlibre = JSON.stringify(llibre);

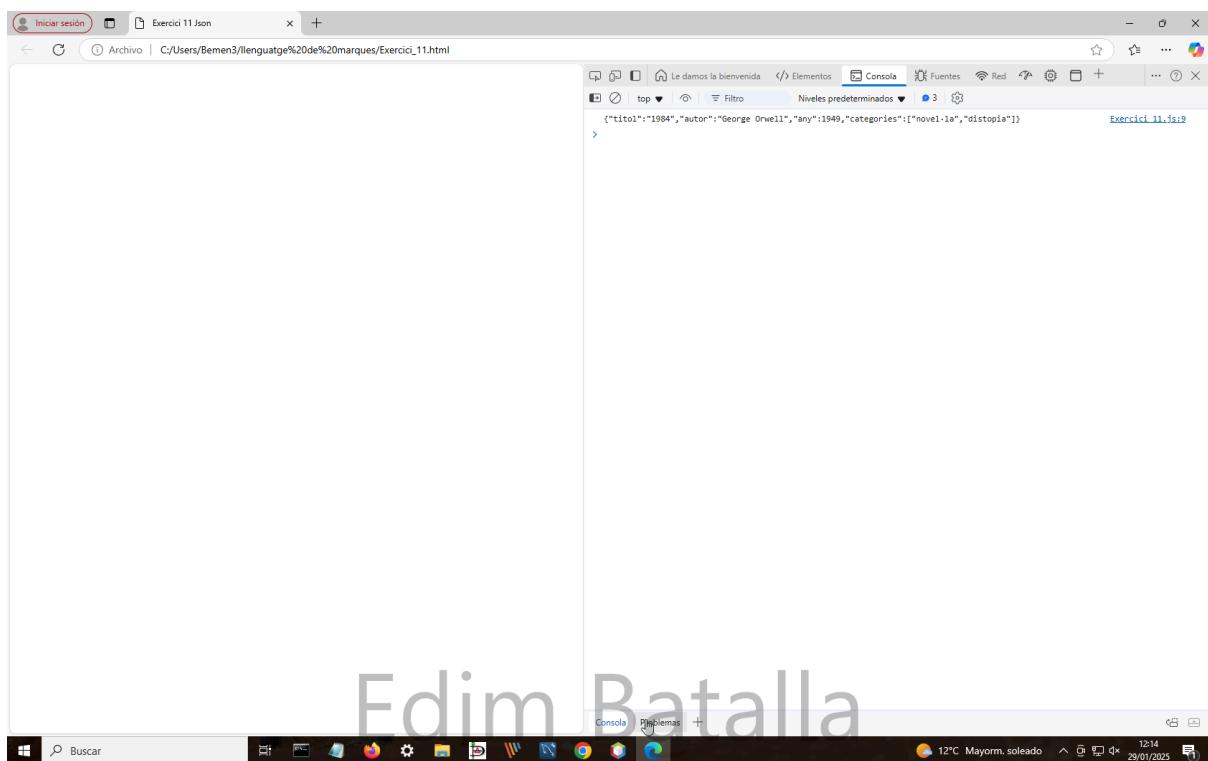
console.log(jsonLlibre);
```

Arxiu HTML



Hem tingut que crear un arxiu HTML i vincular l'arxiu javascript utilitzant la ineia:
<script src="Exercici_11.js"></script>

Captura Consola



Exercici 12:

Aquesta és la cadena JSON amb informació sobre una pel·lícula:

```
{  
  "titol": "Interstellar",  
  "director": "Christopher Nolan",  
  "any": 2014,  
  "actors": ["Matthew McConaughey", "Anne Hathaway", "Jessica  
Chastain"],  
  "duracio": 169  
}
```

Tasques:

1. Converteix aquesta cadena JSON a un objecte JavaScript utilitzant **JSON.parse**.
2. Mostra el valor de la propietat **director** a la consola.
3. Accedeix a la llista d'actors i mostra-la a la consola.

Arxiu JavaScript:

```
// 1. Definim correctament la cadena JSON  
const jsonPelicula = '{  
  "titol": "Interstellar",  
  "director": "Christopher Nolan",  
  "any": 2014,  
  "actors": ["Matthew McConaughey", "Anne Hathaway", "Jessica Chastain"],  
  "duracio": 169  
}';  
  
// 2. Convertim la cadena JSON en un objecte JavaScript  
const objectePelicula = JSON.parse(jsonPelicula);  
  
// 3. Mostrem el valor de la propietat "director" a la consola  
console.log("Director:", objectePelicula.director);  
  
// 4. Mostrem la llista d'actors a la consola  
console.log("Actors:", objectePelicula.actors);
```

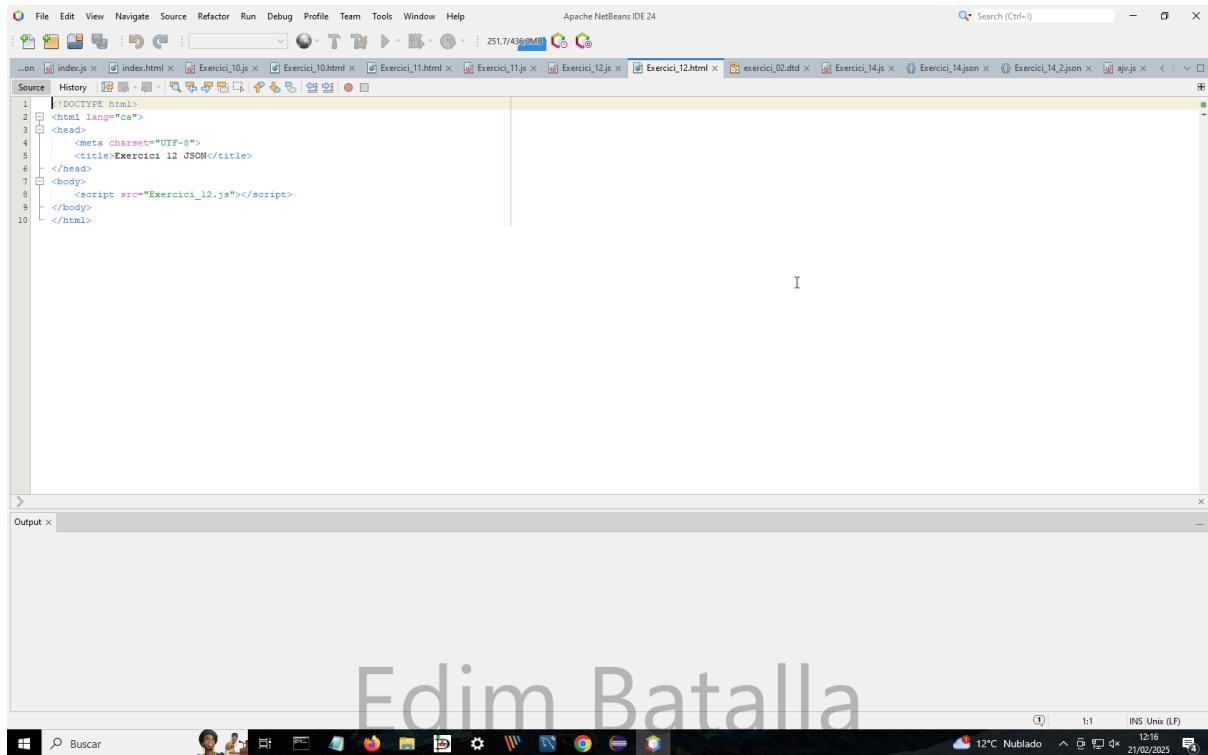
Es crea una variable **jsonPelicula** que emmagatzema la informació de la pel·lícula en format JSON. Conté detalls com el títol, el director, l'any d'estrena, la llista d'actors i la durada.

S'usa **JSON.parse(jsonPelicula)** per a transformar la cadena JSON en un objecte JavaScript anomenat **objectePelicula**.

S'accedeix a la propietat **director** de l'objecte i s'imprimeix en la consola.

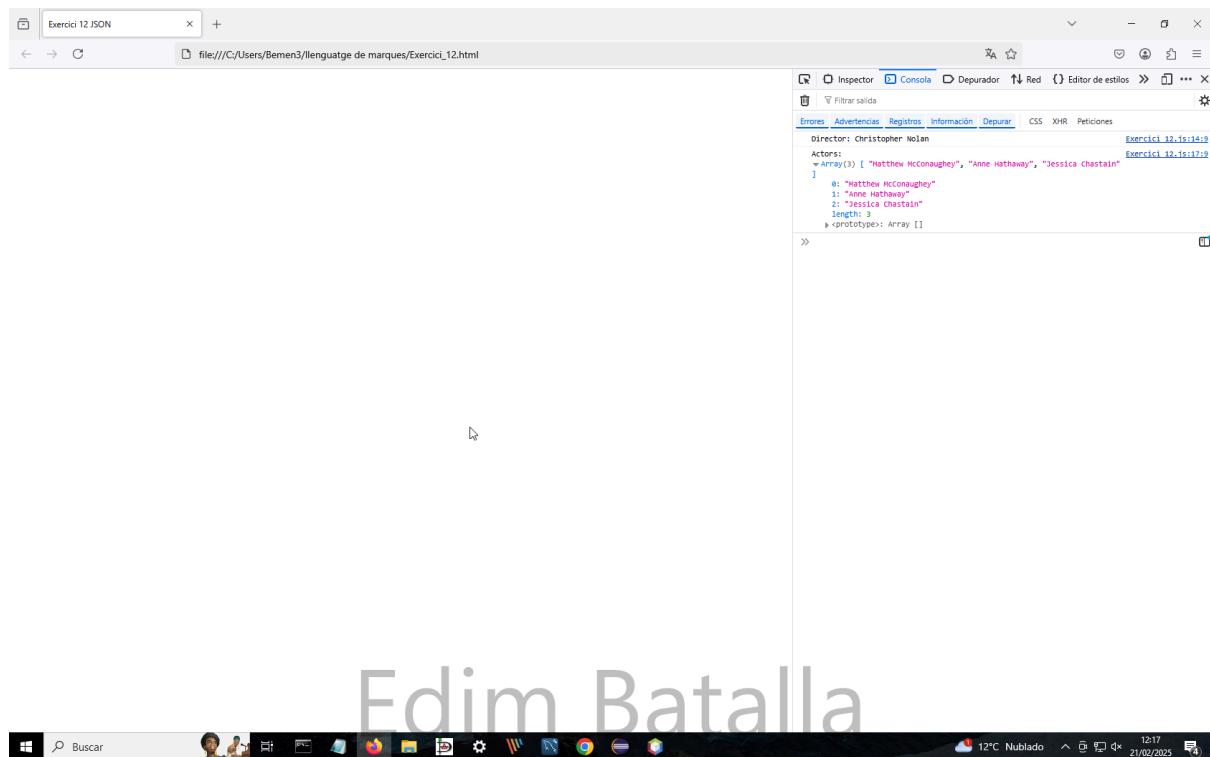
S'accedeix a la propietat **actors**, que és un array, i s'imprimeix el seu contingut en la consola.

Arxiu HTML



```
<!DOCTYPE html>
<html lang="ca">
<head>
    <meta charset="UTF-8">
    <title>Exercici 12 JSON</title>
</head>
<body>
    <script src="Exercici_12.js"></script>
</body>
</html>
```

Captura Consola



Exercici 13:

Utilitza la API JSONPlaceholder per obtenir una llista d'usuaris.

1. Fes una petició GET a l'URL següent:

<https://jsonplaceholder.typicode.com/users>.

2. Mostra a la consola els noms (name) de tots els usuaris.

Arxiu JavaScript:

```
// 1. Far una petició GET a l'API JSONPlaceholder
fetch("https://jsonplaceholder.typicode.com/users")
  .then(response => response.json()) // Convertim la resposta a JSON
  .then(data => {
    // 2. Mostrem a la consola els noms dels usuaris
    data.forEach(user => console.log(user.name));
  })
  .catch(error => console.error("Error en la peticció:", error));
```

Edim Batalla

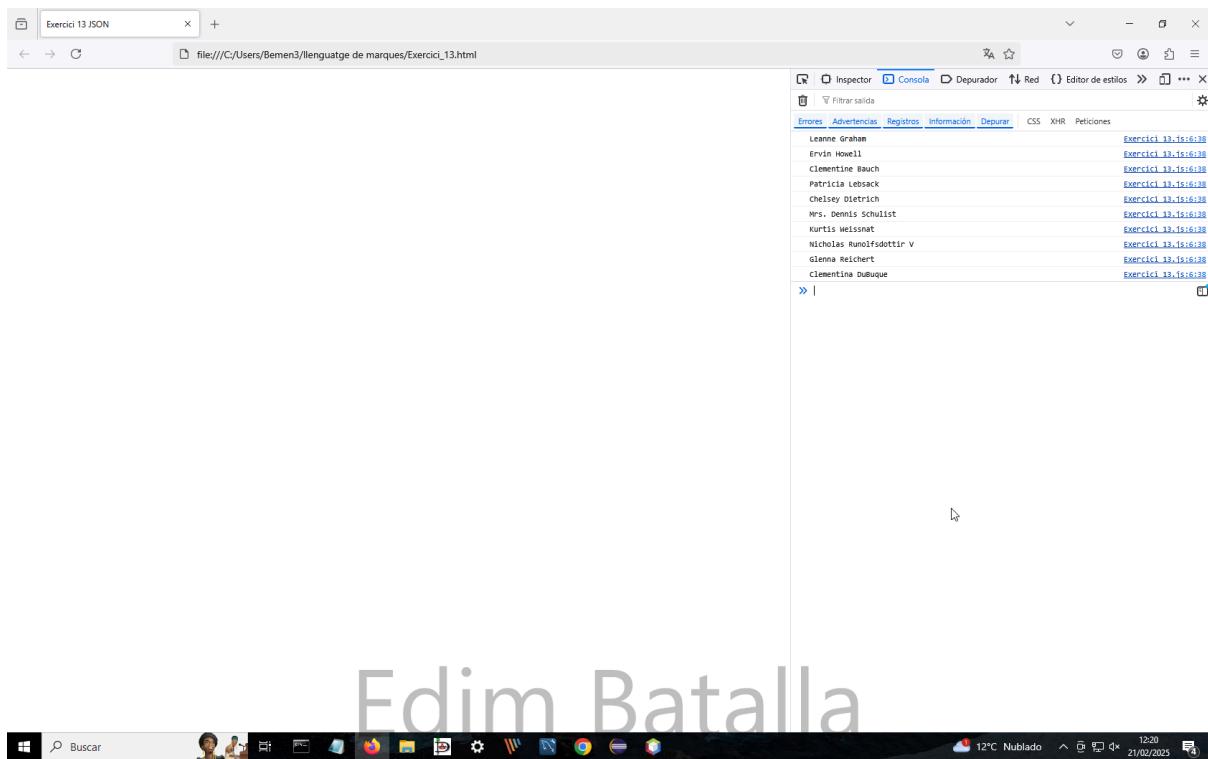
El codi fa una petició a l'API de **JSONPlaceholder** per obtenir una llista d'usuaris. El JS (**Exercici_13.js**) fa el **fetch()**, converteix la resposta a JSON i mostra els noms a la consola. El HTML (**Exercici_13.html**) només carrega el script.

Arxius Javascript

```
<!DOCTYPE html>
<html lang="ca">
  <head>
    <meta charset="UTF-8">
    <title>Exercici 13 JSON</title>
  </head>
  <body>
    <script src="Exercici_13.js"></script>
  </body>
</html>
```

Edim Batalla

Captura Consola



Exercici 14:

Crea un esquema per validar un producte d'una botiga en línia. El producte ha de tenir les següents propietats:

- **id: un número enter (obligatori).**
- **nom: una cadena de text (obligatori).**
- **preu: un número positiu (obligatori).**
- **categories: un array de cadenes de text, com ara ["Electrònica", "Llar"] (opcional).**
- **enEstoc: un booleà que indica si el producte està disponible (obligatori).**

Captura Javascript

```

File Edit Selection View Go Run Terminal Help
RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug, open a folder and create a launch.json file.
JavaScript Debug Terminal
You can use the JavaScript Debug Terminal to debug Node.js processes run on the command line.
Debug URL

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
C:\>Program Files\nodejs\node.exe .\Exercici_14.js
El JSON es válido!

```

L'exercici valida un JSON de producte d'una botiga en línia amb JSON Schema i Ajv a Node.js. Es defineix un esquema (**Exercici_14.json**) que exigeix un **id** enter, **nom** text, **preu** número positiu, **categories** (opcional) com array de textos i **enEstoc** com a booleà. Després, un JSON (**Exercici_14_2.json**) amb un producte es valida amb un codi (**Exercici_14.js**). Si compleix les regles, es mostra "*El JSON és vàlid!*", si no, es llisten errors.

Captura Json Schema

```

File Edit Selection View Go Run Terminal Help
RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug, open a folder and create a launch.json file.
JavaScript Debug Terminal
You can use the JavaScript Debug Terminal to debug Node.js processes run on the command line.
Debug URL

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
C:\>Program Files\nodejs\node.exe .\Exercici_14.js
El JSON es válido!

```

Captura Json

The screenshot shows a Windows desktop environment. In the foreground, there is a code editor window titled "Exercici_14.js" which contains JSON code. The code defines an object with properties like id, name, price, categories, and endstock. Below the code editor is a terminal window titled "powershell" showing command-line history related to the file. The desktop taskbar at the bottom includes icons for various applications like File Explorer, Edge, and FileZilla. The system tray shows the date and time as 21/02/2025 and 12:22.

```
Exercici_14.js
C:\Users\Bemem3\lenguatge de marques > Exercici_14_2.json > ...
1
2
3
4
5
6
7
8
9

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
at Function._load (node:internal/modules/cjs/loader:1128:12)
at TracingChannel.traceSync (node:diagnostics_channel:1322:14)

Node.js v22.12.0
PS C:\Users\Bemem3\lenguatge de marques> node Exercici_14.js
PS C:\Users\Bemem3\lenguatge de marques>
  • History restored
PS C:\Users\Bemem3\lenguatge de marques>
PS C:\Users\Bemem3>
PS C:\Users\Bemem3> Buscar
12°C Nublado 12:22 21/02/2025
```

Exercici 15:

Crea un login amb un formulari per introduir l'usuari i la contrasenya.

Quan l'usuari es loguegi correctament:

- 1. Genera un token JWT.**
- 2. Mostra el token a la pantalla.**
- 3. Fes una petició a <https://jsonplaceholder.typicode.com/comments> i mostra els primers 10 comentaris a la pantalla.**

Comandos CMD

```

[dependencies] {
  "dependencies": {
    "ajv": "^6.17.1",
    "ajv-formats": "^3.0.1"
  },
  "name": "dtde-i-json",
  "version": "1.0.0",
  "main": "Exercici10.js",
  "scripts": {
    "test": "echo \'Error: no test specified\' && exit 1"
  },
  "keywords": [],
  "repository": {},
  "license": "ISC",
  "description": ""
}

C:\Users\Pantera 2.0\Desktop\DTD i JSON>npm install express jsonwebtoken axios open
added 107 packages, and audited 108 packages in 3s
27 packages are looking for funding
  run 'npm fund' for details
Found 0 vulnerabilities

C:\Users\Pantera 2.0\Desktop\DTD i JSON>

```

Edim Batalla

Primer, executem **npm init -y** per inicialitzar un projecte de Node.js amb un arxiu **package.json**. Després, instal·lem les dependències necessàries amb **npm install express jsonwebtoken axios open**, que ens permeten crear un servidor web amb Express, generar un token JWT, fer peticions HTTP i obrir el navegador automàticament. Finalment, executem **node Exercici15.js** per iniciar l'aplicació, que inclou un formulari de login, la generació del token JWT en autenticar l'usuari i una petició a l'API de comentaris per mostrar els primers 10 comentaris.

Arxiu JS

```

const express = require('express');
const jwt = require('jsonwebtoken');
const axios = require('axios');
const open = require('open').default;

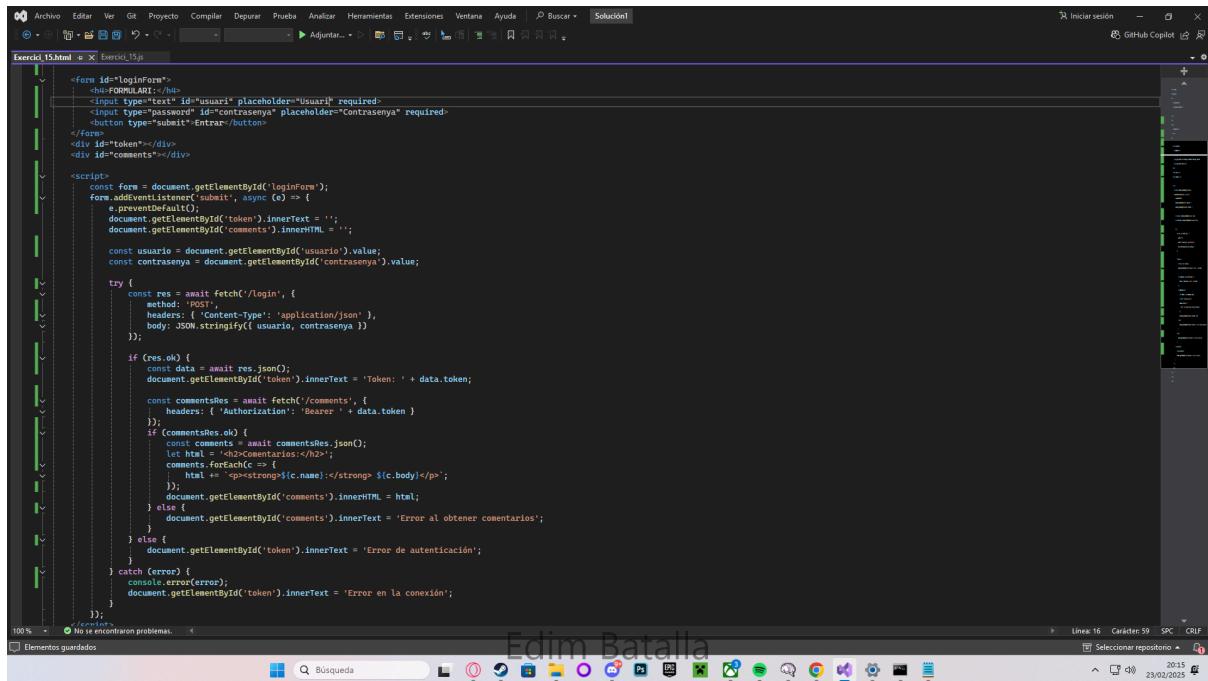
const app = express();
const secretKey = 'ClaveSecreta';
app.use(express.json());
app.use(express.static('.'));
app.post('/login', (req, res) => {
  const { usuario, contraseña } = req.body;
  if (usuario === 'Admin' && contraseña === '1234') {
    const token = jwt.sign({ user: 'Admin' }, secretKey, { expiresIn: '1h' });
    res.json({ token });
  } else {
    res.status(401).json({ error: 'Credenciales incorrectas' });
  }
});
const authenticateJWT = (req, res, next) => {
  const authHeader = req.headers.authorization;
  if (authHeader) {
    const token = authHeader.split(' ')[1];
    jwt.verify(token, secretKey, (err, decoded) => {
      if (err) return res.status(403).json({ error: 'Token inválido' });
      req.user = decoded;
      next();
    });
  } else {
    res.status(401).json({ error: 'Token no proporcionado' });
  }
};
app.get('/comments', authenticateJWT, async (req, res) => {
  try {
    const response = await axios.get('https://jsonplaceholder.typicode.com/comments');
    res.json(response.data.slice(0, 10));
  } catch (error) {
    res.status(500).json({ error: 'Error al obtener comentarios' });
  }
});
app.listen(3000, () => {
  console.log('Servidor ejecutándose en el puerto 3000');
  open('http://localhost:3000');
});

```

Edim Batalla

Primer, obtenim el formulari de login i afegim un esdeveniment que capture quan s'envia. Quan el formulari s'envia, evitem que la pàgina es recarregui i netegem els missatges previs. Agafem els valors de l'usuari i la contrasenya, i fem una petició **POST** al servidor per autenticar l'usuari. Si el login és correcte, rebem un token JWT que mostrem a la pàgina. Després, fem una petició **GET** per obtenir els comentaris, enviant el token en l'encapçalament **Authorization**. Si la petició de comentaris és exitosa, mostrem els comentaris al navegador; si no, mostrem un missatge d'error. Si alguna de les peticions falla o hi ha un error de connexió, capturem l'error i mostrem un missatge adequat.

Arxiu HTML



```

<form id="loginForm">
    <div><input type="text" id="usuario" placeholder="Usuari" required></div>
    <div><input type="password" id="contrasenya" placeholder="Contrasenya" required></div>
    <div><button type="submit" value="Entrar"></button></div>
</form>
<div id="token"></div>
<div id="comments"></div>

<script>
    const form = document.getElementById('loginForm');
    form.addEventListener('submit', async (e) => {
        e.preventDefault();
        document.getElementById('token').innerText = '';
        document.getElementById('comments').innerHTML = '';

        const usuario = document.getElementById('usuario').value;
        const contrasenya = document.getElementById('contrasenya').value;

        try {
            const res = await fetch('/login', {
                method: 'POST',
                headers: { 'Content-Type': 'application/json' },
                body: JSON.stringify({ usuario, contrasenya })
            });

            if (res.ok) {
                const data = await res.json();
                document.getElementById('token').innerText = `Token: ${data.token}`;

                const commentsRes = await fetch('/comments', {
                    headers: { 'Authorization': `Bearer ${data.token}` }
                });
                if (commentsRes.ok) {
                    const commentsResJson = await commentsRes.json();
                    let html = '<h2>Comentaris</h2>';
                    commentsResJson.forEach(c => {
                        html += `<p><strong>${c.name}</strong> ${c.body}</p>`;
                    });
                    document.getElementById('comments').innerHTML = html;
                } else {
                    document.getElementById('comments').innerText = 'Error al obtener comentarios';
                }
            } else {
                document.getElementById('token').innerText = 'Error de autenticación';
            }
        } catch (error) {
            console.error(error);
            document.getElementById('token').innerText = 'Error en la conexión';
        }
    });
</script>

```

Importem els mòduls necessaris per crear el servidor web amb Express, generar tokens JWT amb jsonwebtoken, fer peticions HTTP amb axios i obrir el navegador amb open. Creem una ruta **/login** per autenticar l'usuari i generar un token JWT si les credencials són correctes. Per protegir rutes, utilitzem un middleware que verifica el token. A la ruta **/comments**, obtenim els primers 10 comentaris d'una API externa després de verificar el token. Finalment, iniciem el servidor al port 3000 i obrim el navegador automàticament.