



Nom i Cognoms:	Joel Martínez Villa
URL Repositori Github:	https://github.com/JoelMartinezVilla/PR4.2.git

ACTIVITAT
Objectius: <ul style="list-style-type: none">• Familiaritzar-se amb el desenvolupament d'APIs REST utilitzant Express.js• Aprendre a integrar serveis de processament de llenguatge natural i visió artificial• Practicar la implementació de patrons d'accés a dades i gestió de bases de dades• Desenvolupar habilitats en documentació d'APIs i logging• Treballar amb formats JSON i processament de dades estructurades
Criteris d'avaluació: <ul style="list-style-type: none">- Cada pregunta indica la puntuació corresponent
Entrega: <ul style="list-style-type: none">- Repositori git que contingui el codi que resol els exercicis i, en el directori "doc", aquesta memòria resposta amb nom "memoria.pdf"

Punt de partida

<https://github.com/jpala4-ieti/DAM-M06-UF04-PR4.2-24-25-Punt-Partida.git>



Preparació de l'activitat

- Clonar el repositori de punt de partida
- Llegir els fitxers README.md que trobaràs en els diferents directoris
- Assegurar-te de tenir una instància de MySQL/MariaDB funcionant
- Tenir accés a una instància d'Ollama
- Completar els quatre exercicis proposats
- Lliurar el codi segons les instruccions d'entrega



Exercicis

Exercici 1 (2.5 punts)

L'objectiu de l'exercici és familiaritzar-te amb **xat-api**. Respon la les preguntes dins el requadre que trobaràs al final de l'exercici.

Configuració i Estructura Bàsica:

1. Per què és important organitzar el codi en una estructura de directoris com controllers/, routes/, models/, etc.? Quins avantatges ofereix aquesta organització?

Perque és una bona pràctica. Fer això ajuda a tenir tota la estructura més clara i , per tant a desenvolupar codi més ràpid. Qualsevol problema que tinguis indica l'arxiu on passa i si l'arxiu està dedicat a una sola cosa i no té un mix d'altres arxius ajuda molt.

2. Analitzant el fitxer server.js, quina és la seqüència correcta per inicialitzar una aplicació Express? Per què és important l'ordre dels middlewares?

Primer es declara totes les coses que utilitzarà l'aplicació, ja siguin protocols o rutes de fitxers estatics. Després els endpoints per a cada cossa de la api. Finalment s'inicia l'aplicació per a poder accedir. És important l'ordre dels middlewares perquè es perd molta eficiència fent comprovacions, aleshores si no ordenes bé els middlewares y es fan moltes més comprovacions que no s'han de fer es perd rendiment de l'aplicació.

3. Com gestiona el projecte les variables d'entorn? Quins avantatges ofereix usar dotenv respecte a hardcodejar els valors?

Amb arxius .env. Sobretot privacitat i simplicitat, es a dir, si tens un arxiu ple de codi i tens les variables d'entorn pel mig del codi, serà molt difícil trobar les variables. I sobre la privacitat si tens la clau d'una api com a una variable d'entorn, qualsevol persona que pugui veure el codi la veurà, però si la tens en un .env ningú podrà veure-la.



API REST i Express:

1. Observant `chatRoutes.js`, com s'implementa el routing en Express? Quina és la diferència entre els mètodes HTTP GET i POST i quan s'hauria d'usar cadascun?

Has de crear un router on es declara cada funcionalitat que vols amb l'endpoint que vulguis donar-li.

La diferencia entre get y post es que get es una petició que no te un request body y l'objectiu és rebre un recurs del servidor, mentre que el post si que disposa d'un request body i l'objectiu es enviar unes dades al servidor, independentment de si rebrà dades o no.

2. En el fitxer `chatController.js`, per què és important separar la lògica del controlador de les rutes? Quins principis de disseny s'apliquen?

Perquè facilita el desenvolupament i manteniment del codi.

Els principis aplicats son:

- Separation of Concerns (SoC)
- Modularitat
- Single Responsibility Principle (SRP)
- Facilitat de manteniment
- Principi DRY (Don't Repeat Yourself)

3. Com gestiona el projecte els errors HTTP? Analitza el middleware `errorHandler.js` i explica com centralitza la gestió d'errors.

Quan surt un error el fica a la request i el middleware fa el return del error `http.en` comptes de fer-ho al mateix arxiu on passa.



Documentació amb Swagger:

1. Observant la configuració de Swagger a `swagger.js` i els comentaris a `chatRoutes.js`, com s'integra la documentació amb el codi? Quins beneficis aporta aquesta aproximació?
2. Com es documenten els diferents endpoints amb els decoradors de Swagger? Per què és important documentar els paràmetres d'entrada i sortida?
3. Com podem provar els endpoints directament des de la interfície de Swagger? Quins avantatges ofereix això durant el desenvolupament?

Base de Dades i Models:

1. Analitzant els models `Conversation.js` i `Prompt.js`, com s'implementen les relacions entre models utilitzant Sequelize? Per què s'utilitza UUID com a clau primària?
2. Com gestiona el projecte les migracions i sincronització de la base de dades? Quins riscos té usar `sync()` en producció?
3. Quins avantatges ofereix usar un ORM com Sequelize respecte a fer consultes SQL directes?

Logging i Monitorització:

1. Observant `logger.js`, com s'implementa el logging estructurat? Quins nivells de logging existeixen i quan s'hauria d'usar cadascun?
2. Per què és important tenir diferents transports de logging (consola, fitxer)? Com es configuren en el projecte?
3. Com ajuda el logging a debugar problemes en producció? Quina informació crítica s'hauria de loguejar?



Exercici 2 (2.5 punts)

Dins de **practica-codi** trobaràs **src/exercici2.js**

Modifica el codi per tal que, pels dos primers jocs i les 2 primeres reviews de cada jocs crei una estadística que indiqui el nombre de reviews positives, negatives o neutres.

Modifica el prompt si cal.

Guarda la sortida en el directori data amb el nom **exercici2_resposta.json**

Exemple de sortida

```
{
  "timestamp": "2025-01-09T12:30:45.678Z",
  "games": [
    {
      "appid": "730",
      "name": "Counter-Strike 2",
      "statistics": {
        "positive": 1,
        "negative": 0,
        "neutral": 1,
        "error": 0
      }
    },
    {
      "appid": "570",
      "name": "Dota 2",
      "statistics": {
        "positive": 1,
        "negative": 1,
        "neutral": 0,
        "error": 0
      }
    }
  ]
}
```



Exercici 3 (2.5 punts)

Dins de **practica-codi** trobaràs **src/exercici3.js**

Modifica el codi per tal que retorni un anàlisi detallat sobre l'animal.

Modifica el prompt si cal.

La informació que volem obtenir és:

- Nom de l'animal.
- Classificació taxonòmica (mamífer, au, rèptil, etc.)
- Hàbitat natural
- Dieta
- Característiques físiques (mida, color, trets distintius)
- Estat de conservació

Guarda la sortida en el directori **data** amb el nom **exercici3_resposta.json**

```
{
  "analisi": [
    {
      "imatge": {
        "nom_fitxer": "nom_del_fitxer.jpg",
      },
      "analisi": {
        "nom_comu": "nom comú de l'animal",
        "nom_cientific": "nom científic si és conegut",
        "taxonomia": {
          "classe": "mamífer/au/rèptil/amfibi/peix",
          "ordre": "ordre taxonòmic",
          "familia": "família taxonòmica"
        },
        "habitat": {
          "tipus": ["tipus d'hàbitats"],
          "regioGeografica": ["regions on viu"],
          "clima": ["tipus de climes"]
        },
        "dieta": {
          "tipus": "carnívor/herbívor/omnívor",
          "aliments_principals": ["llista d'aliments"]
        },
        "caracteristiques_fisiques": {
          "mida": {
            "altura_mitjana_cm": "altura mitjana",
            "pes_mitja_kg": "pes mitjà"
          },
          "colors_predominants": ["colors"],
          "trets_distintius": ["característiques"]
        },
        "estat_conservacio": {
          "classificacio_IUCN": "estat",
          "amenaces_principals": ["amenaces"]
        }
      }
    }
  ]
}
```



Exercici 4 (2.5 punts)

Implementa un nou endpoint a xat-api per realitzar anàlisi de sentiment

Haurà de complir els següents requisits

- Estar disponible a l'endpoint POST /api/chat/sentiment-analysis
- Disposar de documentació swagger
- Emmagatzemar informació a la base de dades
- Usar el logger a fitxer

Abans d'implementar la tasca, explica en el requadre com pla plantejaràs i fes una proposta de json d'entrada, de sortida i de com emmagatzemaràs la informació a la base de dades.