

ACTIVITAT AVALUABLE AC3**Mòdul:** MP12- Projectes**UF:** UF1**Professor:** Albert Guardiola, Marc Callejón**Data límit d'entrega:** 26/10/2025 23:59**Mètode d'entrega:** Per mitjà del Clickedu de l'assignatura. Les activitats entregades més enllà de la data límit només podran obtenir una nota de 5.**Instruccions:** S'ha d'entregar **un únic fitxer comprimit** amb el nom:***MP12-PROJ1-ENTR3.zip/rar***

Es valorarà la presentació.

MONSTER ASP.NET**Tasca 1. Desenvolupar el codi de l'API especificada, en llenguatge C#. El codi de l'API ha de complir els següents requisits:*****Requisits funcionals:*****-Adequar-se completament a l'especificació elaborada a l'ENTREGA 1. Qualsevol desviació respecte de l'especificació ha de documentar-se a la Tasca 2.**

A cadascun dels tres endpoints es poden realitzar operacions GET, POST, PUT i DELETE.

Per a aquestes operacions és necessari utilitzar l'identificador (clau primària) corresponent de cada taula:

UFCEVENTS: la clau primària és EVENT. Exemple: "UFC Fight Night: Ulberg vs. Reyes".

UFCFIGHTERS: la clau primària és FIGHTER. Exemple: "Dominick Reyes".

UFCRESULTS: la clau primària és FIGHTERS. Exemple: "Ilia Topuria vs. Charles Oliveira".

Requisits tecnològics:**-El codi de l'API ha d'estar modularitzat, com a mínim, de la següent manera.****-Mòdul principal: instanciació i configuració (incloses les rutes) del servidor (Controller).**

El fitxer Program.cs s'encarrega de la instanciació i configuració del servidor, incloent-hi els controladors, les rutes i Swagger.

-Mòdul de base de dades: crides a la base de dades.

La carpeta Data conté la classe MiDbContext, que gestiona la connexió amb la base de dades SQLite, i la carpeta Models defineix les entitats corresponents a cada taula.

-Mòdul d'autenticació: gestió de l'autenticació d'usuaris.

Aquest mòdul no s'ha implementat.

Tasca 2. Lliurar un checklist dels requisits funcionals especificats (ENTREGA 1) que marqui clarament si s'han implementat o no en el codi lliurat. Els requisits no implementats han de venir acompanyats d'una justificació de per què no s'han implementat.

En l'entrega 1 vam indicar que s'utilitzarien dos endpoints: UFCEVENTS i UFCFIGHTERS.

Finalment vam afegir **un tercer endpoint (UFCRESULTS)**.

Aquest endpoint no estava previst, però s'ha incorporat per tal de proporcionar informació més completa sobre els resultats dels esdeveniments, millorant així la funcionalitat global de l'API.

Les **consultes GET** es poden realitzar **sense cap requeriment d'autenticació**, tal com es va indicar a l'entrega 1. No obstant això, **els mètodes POST, PUT i DELETE no inclouen el sistema d'autenticació via API_KEY que inicialment es preveia**. La justificació és que, per simplificar el projecte, no s'ha implementat la gestió d'usuaris ni el control d'autenticació. Tot i això, les operacions funcionen correctament quan s'accedeix sense autenticació, complint amb els requisits mínims de funcionalitat.

Pel que fa al filtratge de dades, inicialment vam pensar fer-ho per diversos criteris. Per evitar complicacions innecessàries i mantenir l'API coherent i fàcil de mantenir, s'ha decidit que només es pugui identificar i accedir als registres mitjançant la clau primària de cada taula (EVENT, FIGHTER i FIGHTERS), tal com es descriu a la Tasca 1.

Tasca 3. Allotjar l'API a un servidor de C#

No s'ha pogut allotjar l'API a cap hosting, per fer el testing dels mètodes, hem utilitzat Swagger UI.

Ufcevents		^
GET	/api/Ufcevents	▼
POST	/api/Ufcevents	▼
GET	/api/Ufcevents/{id}	▼
PUT	/api/Ufcevents/{id}	▼
DELETE	/api/Ufcevents/{id}	▼
Ufcfighters		^
GET	/api/Ufcfighters	▼
POST	/api/Ufcfighters	▼
GET	/api/Ufcfighters/{id}	▼
PUT	/api/Ufcfighters/{id}	▼
DELETE	/api/Ufcfighters/{id}	▼
Ufcresults		^
GET	/api/Ufcresults	▼
POST	/api/Ufcresults	▼
GET	/api/Ufcresults/{id}	▼
PUT	/api/Ufcresults/{id}	▼
DELETE	/api/Ufcresults/{id}	▼