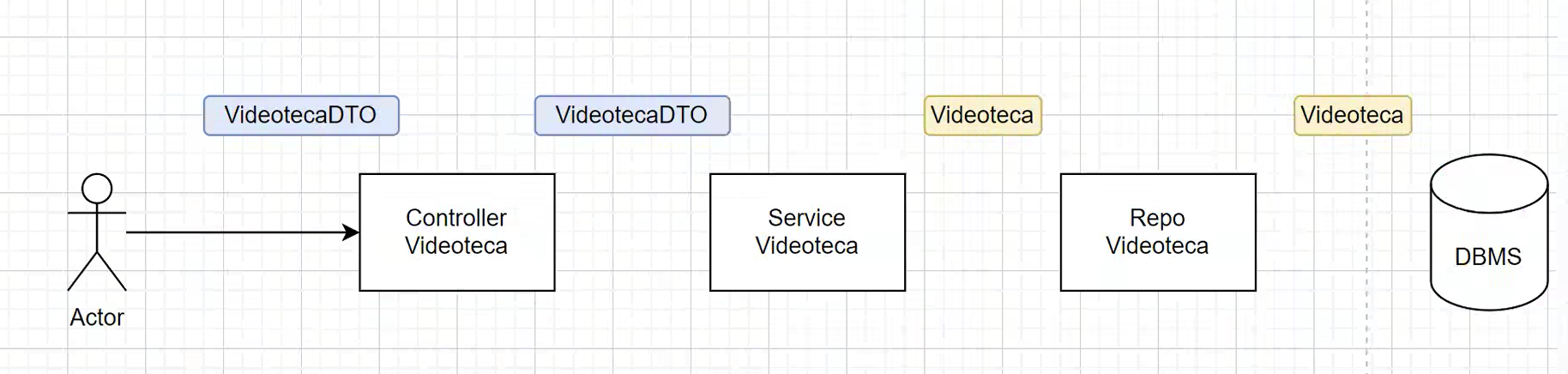
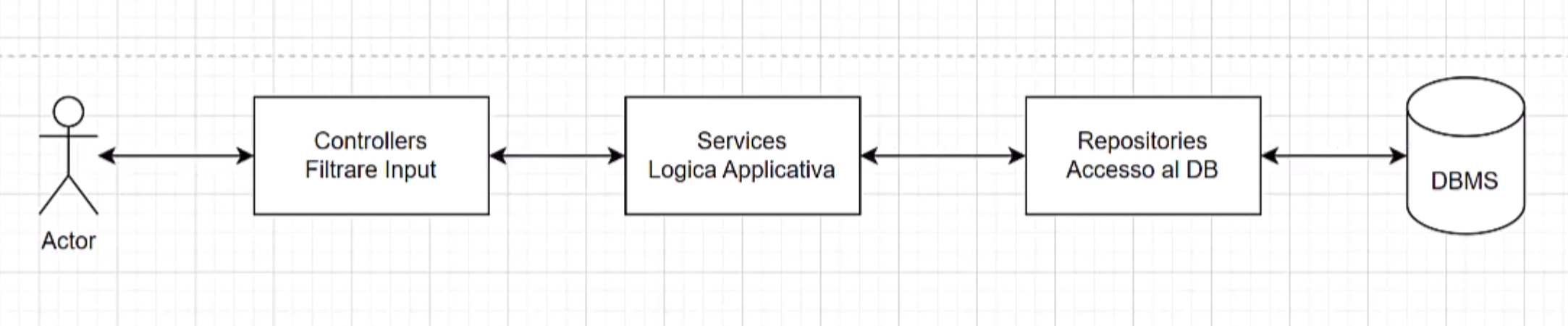
EF API

Pino => Controller => Service => Repo=> Database





CONTROLLER: si occupa di effettuare i controlli immessi dall’utente (in maniera tale da non scomodare il service in caso di input errati) , probabilmente con delle regex. Quando l’utente effettua una richiesta i suoi dati passano per la validazione del controller.

SERVICE: si occupa di effettuare la conversione da oggetto a oggetto DTO . Inoltre alleggerisce il controller in quanto gli fa vedere solo quello che vuole: solo alcuni attributi e alcuni metodi

DTO: data transfer object , è l’oggetto restituito dal service al controller e dal controller all’utente, ma in una forma con meno informazioni, ovvero tutte quelle info che può vedere l’utente

IActionResult la si utilizza quando il tipo di ritorno è una http response. ActionResult si utilizza quando devo tipizzare la response, ad esempio quando una chiamata ci restituisce un oggetto oppure una lista di oggetti.

CREAZIONE DEL DBCONTEXT A PARTIRE DA C#

Creo il context che deve ereditare da dbcontext.

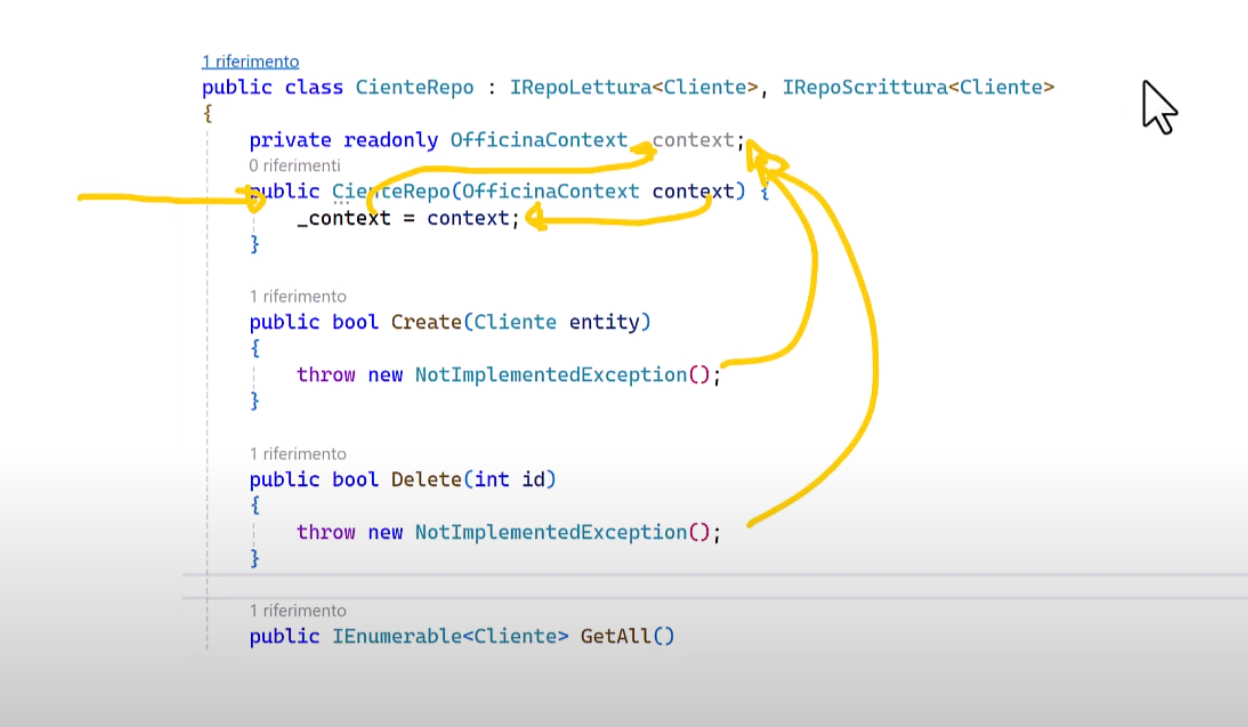
Quando creo il costruttore passo tra i suoi parametri le options (approfondire options)

Nel file della classe utilizzo un decoratore per collegare la classe alla tabella del database [Table(“Videoteca”)]

RIVEDERE: DBCONTEXT-DBCONTEXT OPTIONS

L’ORM si occupa di creare una corrispondenza tra le classi e le tabelle del database.

DEPENDENCY INJECTION



CLIENT-SERVER

Il client utilizza un protocollo http per effettuare una richiesta.

Il server è colui a cui inviamo una richiesta e ci risponde con le info richieste.

Una richiesta http è stateless, ovvero non ha memoria delle richieste precedenti.

Ad ogni singola richiesta deve essere associata una http response.

La request GET è composta da: - Header (mittente, destinatario, payload)

La request POST è composta da - Header e Body , che conterrà il payload.

Se il body ha una dimensione elevata viene fatto a pezzi e ricomposto al suo arrivo.

La response è composta da header e body, con la differenza che l’header ha lo status code. (http status code)

Le richieste 1xx sono informative,

le richieste 2xx sono di success, ovvero quando una richiesta va a buon fine

le richieste 3xx sono di redirect

le richieste 4xx sono errori da parte del client

le richieste 5xx sono errori da parte del server

I nostri browser sono in grado di fare solo richieste GET.

**ASPNET WEB**

REST: metodologia di comunicazione tramite protocollo http

STATELESS: non conserva le modifiche (lo stato)

RAZOR: template engine (un mix di cs e html)

**APPROCCIO CODE FIRST – MIGRATIONS**

Durante l’approccio code first il contenitore, ovvero il database deve essere già creato (non lo creiamo con il codice).

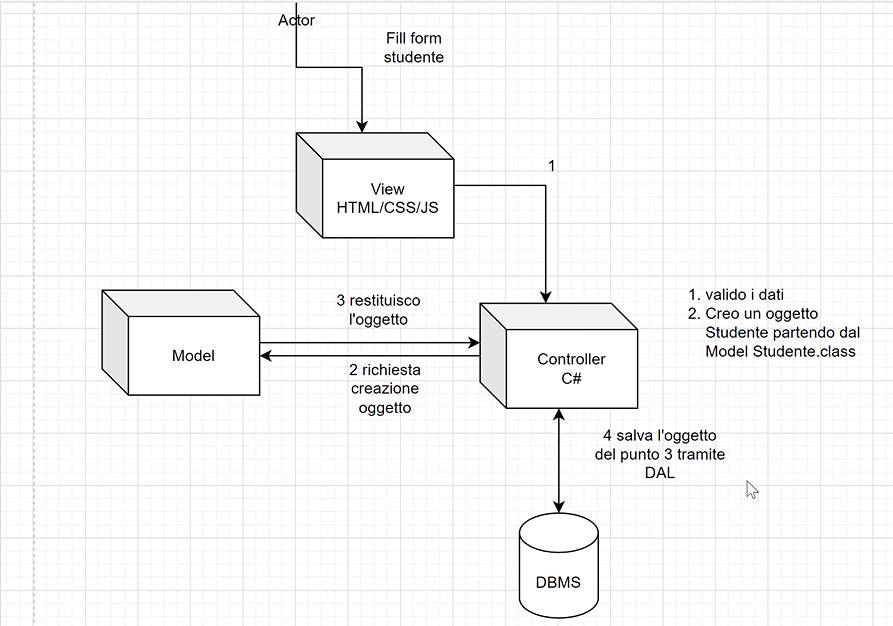
MVC – Model View Controller

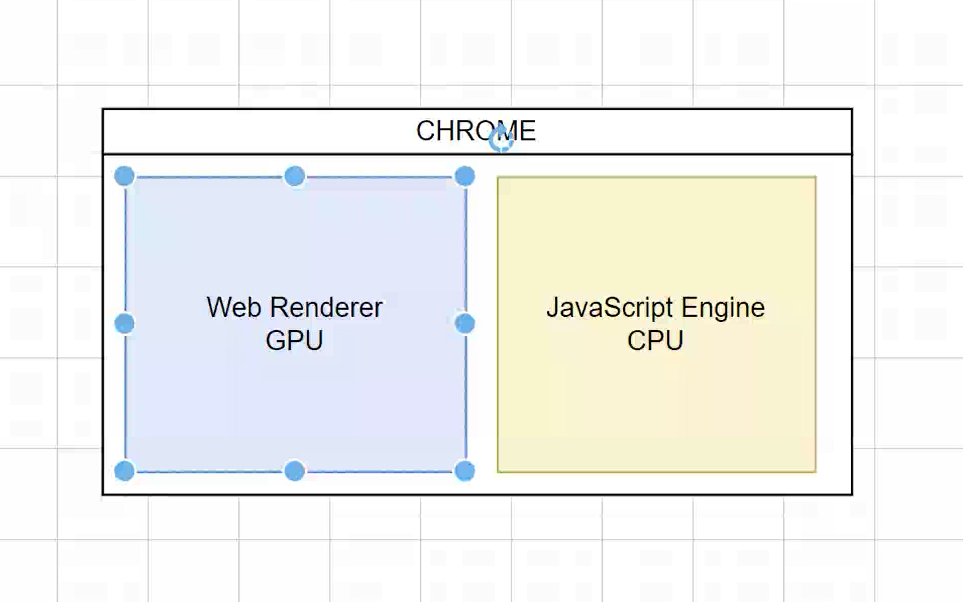
L’utente entra in contatto con la VIEW (html) , la view trasferisce tutte le informazioni al CONTROLLER che creerà l’oggetto grazie ai MODELS.

Nella VIEW fillo il form.

Nel CONTROLLER valido i dati e creo un oggetto partendo dal Model Class-Studente

Il MODEL quindi restituisce l’oggetto creato a partire dalla classe





Ollama run llama3

26/09

EF: ENTITY FRAMEWORK

E’ un fw contenente un set di funzionalitù utili per l’interazione con il database.

INSTALLARE NUGET:

* EntityFrameworkCore
* EntityFrameworkCore.SqlServer
* EntityFrameworkCore.Tools