

Costruisci le tue API con l'AI



Alberto Acerbis





CONSORZIO
UNIVERSITARIO
DI PORDENONE
MOLTIPLICATORE DI VALORE

OVERNET.
upgrade your digital skills

[stesi]
Powered by Innovation

Q Search^K

Introduction

SpiedoBresciano

/healthzGET

/readyGET

Macelleria

Trattoria

/v1/trattoriaGET

/v1/trattoria/startPOST

/v1/trattoria/{id}/suspendPOST

/v1/trattoria/{id}/resumePOST

/v1/trattoria/{id}/greasePOST

/v1/trattoria/{id}/renew-embersPOST

/v1/trattoria/{id}/evaluatePOST

/v1/trattoria/{id}/finishPOST

Models

Open API Client

Powered by Scalar

v1.0OAS 3.0.4

SpiedoBresciano API

Download OpenAPI Document

SpiedoBresciano API

SpiedoBresciano

/healthz

Server

http://localhost:5000

Client Libraries

>_ Shell

Ruby

Node.js

PHP

Python

...

ShellCurl

Operations

GET /healthz

GET /ready

PirOverflow



La nuova Community a Brescia.

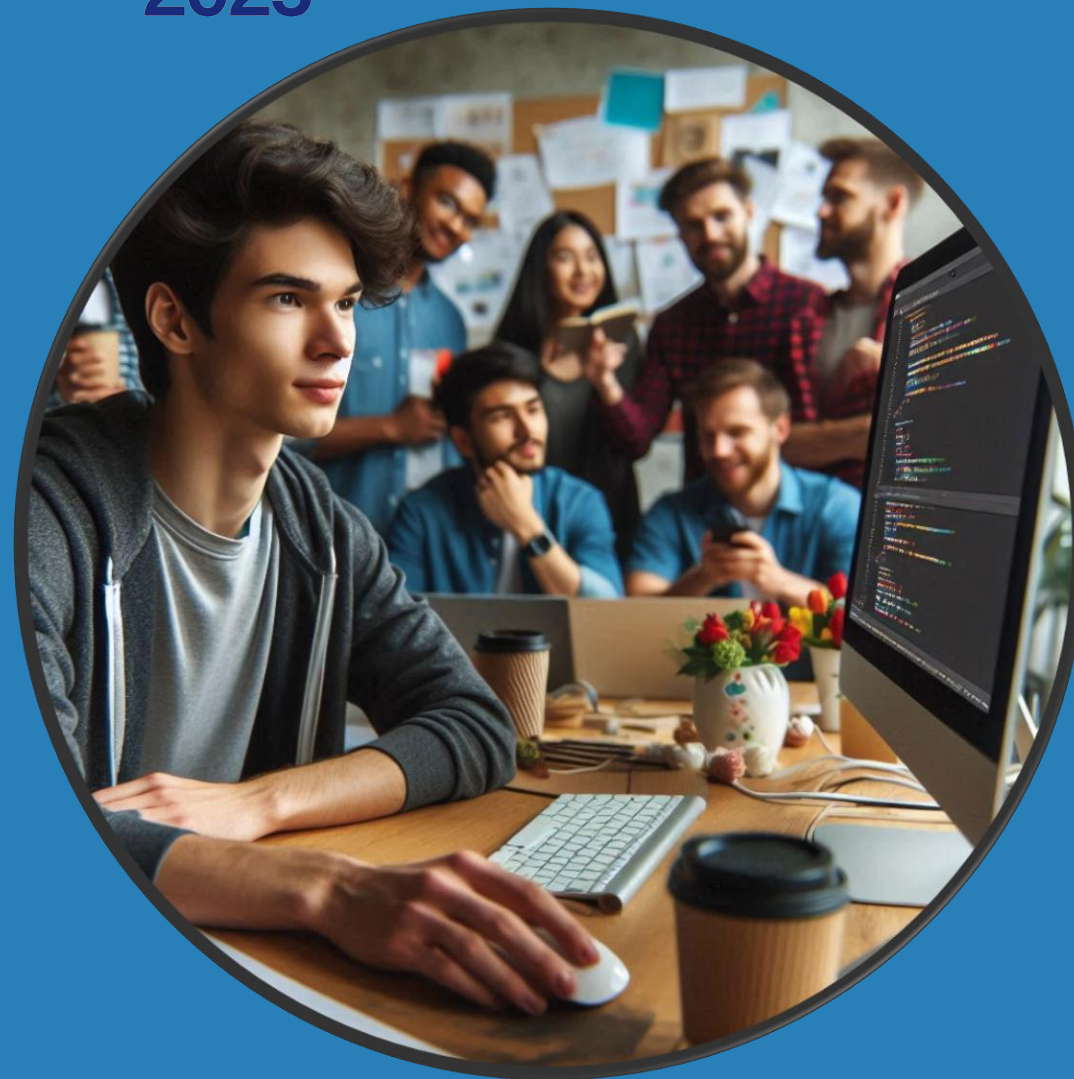


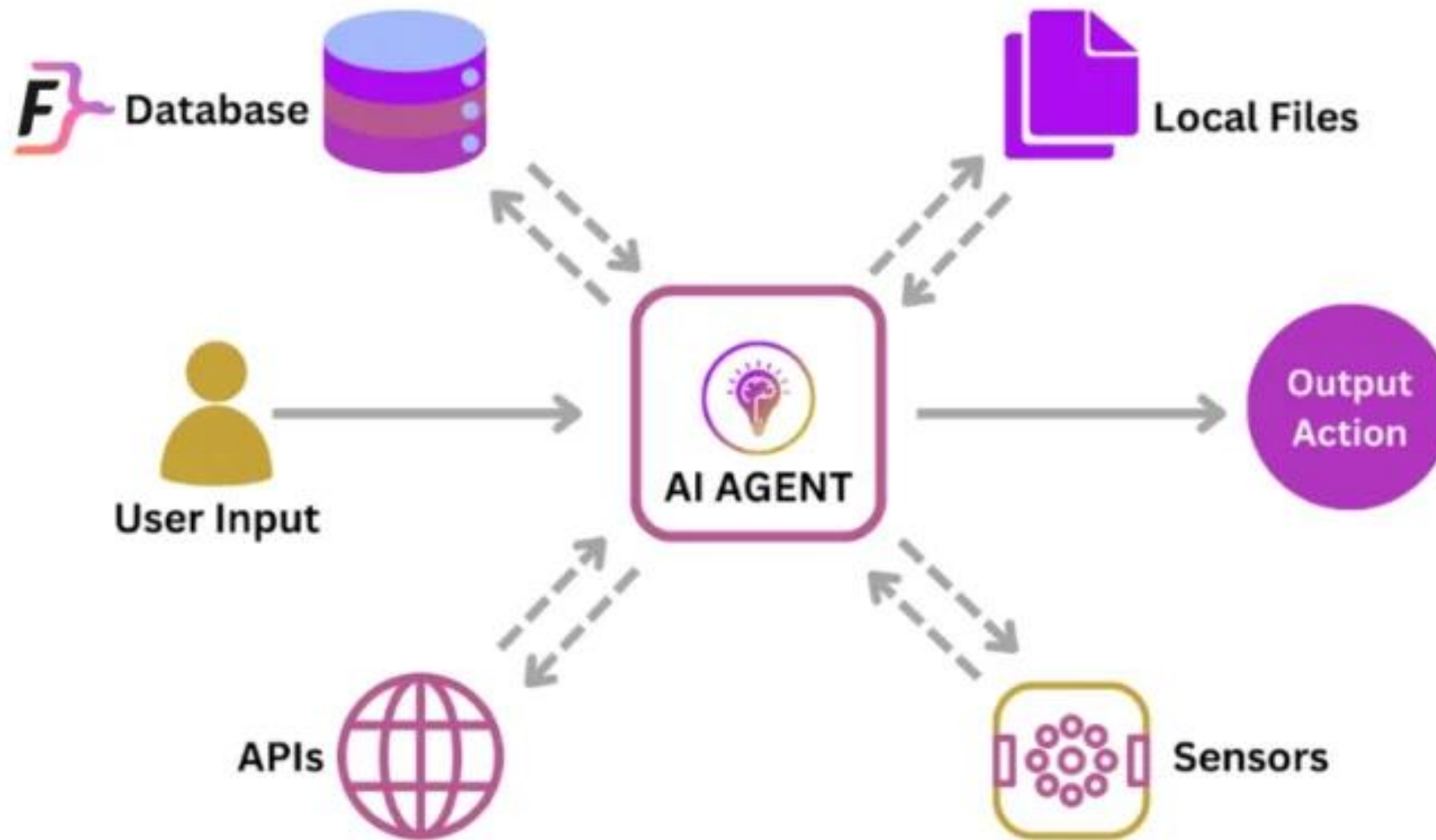
Gruppo Telegram



Proposte

Once upon a time ...





AI Agent Architecture



AI – Protesi Cognitiva



- Se noi siamo confusi, anche la nostra estensione (GenAI) lo sarà. Ma in modo più veloce e sistematico.
- Se abbiamo le idee chiare, l'AI amplificherà quella chiarezza.
- L'AI non pensa al posto nostro! *Pensa di più quello che stiamo già pensando!*

Bias di Autorità



- Un agente AI non sa! Indovina bene.
- Non cerca la verità! Cerca la verosomiglianza.
- Risponde sempre. Non sempre nel modo corretto!!!

File Edit Selection View Go Run Terminal Help

rest-api.prompt.md SpiedoBresciano.Trattoria.SharedKernel.csproj SpiedoBresciano.Trattoria.Domain.csproj

EXPLORER

PORDENONE

.github

prompts

- domain-driven-design.prompt.md
- dot-net-expert.prompt.md
- eventstorming-software-design.prompt.md
- glossario.prompt.md
- improvements.prompt.md
- macelleria-module.prompt.md
- muflone-core.prompt.md
- rest-api.prompt.md
- trattoria-module.prompt.md

workflows

- copilot-instructions.md

src

- Macelleria
- SpiedoBresciano.Rest
- Trattoria
- SpiedoBresciano.Sln
- temp-template

rest-api.prompt.md

```
.github > prompts > rest-api.prompt.md > # Istruzioni per implementare una nuova API .NET utilizzando un template
1 # Istruzioni per implementare una nuova API .NET utilizzando un template
3 ## Feature iniziale
4 TrattoriaFacade .
45 - `SpiedoBresciano.Trattoria.Domain`: per le entità e le logiche di dominio.
46 - `SpiedoBresciano.Trattoria.SharedKernel`: per le classi, i tipi custom e le
interfacce condivise tra i progetti del modulo Trattoria.
47 - `SpiedoBresciano.Trattoria.ReadModel`: per le classi di ReadModel e le query.
48 - `SpiedoBresciano.Trattoria.Infrastructure`: per l'accesso ai dati e le
implementazioni dei repository del modulo Trattoria.
49 - `SpiedoBresciano.Trattoria.Tests`: per i test architetturali del modulo Trattoria.
50 - Solution Folder `30 Shared`:
51 - `SpiedoBresciano.Shared`: per le classi, i tipi custom e le interfacce condivise tra
i progetti della solution.
52
53 7. Non creare Entità, Repository o servizi specifici per i moduli Macelleria e Trattoria.
Concentrati solo sulla struttura della solution e sull'implementazione dei Facade e dei
moduli.
54 8. Implementa i test di architettura utilizzando la libreria NetArchTest, presente nei
progetti `CrastuArrustutu.Carnizzaro.Tests` e `CrastuArrustutu.Tannura.Tests`, per
verificare il corretto isolamento dei moduli.
55
56 9. Questa fase è considerata DONE quando:
57 - La solution compila (Debug/Release) senza warning critici (idealmente treat warnings as
errors).
58 - Tutti i test architetturali (NetArchTest) passano.
59 - Endpoint di base esposti: `/v1/macelleria`, `/v1/trattoria` (anche vuoti) → HTTP 200/204.
60 - Swagger/OpenAPI accessibile (JSON + UI Scalar) e versione valorizzata.
61 - Telemetria configurabile (OpenTelemetry module disattivato finché non configurato).
62 - README aggiornato con istruzioni run locali.
63
64 10. Regole per i Test Architetturali
65 - Non ci devono essere dipendenze fra i moduli (es. Nessun progetto di un modulo deve fare
riferimento a un progetto di un altro modulo).
66 - Il progetto `{SpiedoBresciano.Rest}` deve essere l'unico punto di accesso per le API
esterne.
67 - Il progetto `{SpiedoBresciano.Rest}` deve avere dipendenze solo con i progetti `Facade`
```

OUTPUT

Filter C#

```
BuildHostProcessManager] .NET BuildHost started from C:\Users\lacr\vscode\extensions\ms-dotnettools.csnaip-2.90.
60-win32-x64\.roslyn\Microsoft.CodeAnalysis.LanguageServer.exe reloading to start from C:\Program
Files\dotnet\dotnet.exe to match necessary SDK location.
2025-09-24 18:28:54.650 [warning] [textDocument/diagnostic] [LanguageServerProjectLoader] Project
c:\Sviluppo\Ace68\Pordenone\src\Trattoria\SpiedoBresciano.Trattoria.Facade\ITrattoriaFacade.csproj has
unresolved dependencies
2025-09-24 18:28:54.651 [info] [textDocument/diagnostic] [LanguageServerProjectLoader] Successfully completed
load of c:\Sviluppo\Ace68\Pordenone\src\Trattoria\SpiedoBresciano.Trattoria.Facade\ITrattoriaFacade.cs
2025-09-24 18:28:55.159 [error] [workspace/_roslyn_restore] [Microsoft.CodeAnalysis.LanguageServer.Handler.
RestoreHandler] Restore completed with errors.
2025-09-24 18:28:55.160 [info] [textDocument/diagnostic] [LanguageServerProjectLoader] Completed (re)load of all
```



Prepariamo la solution

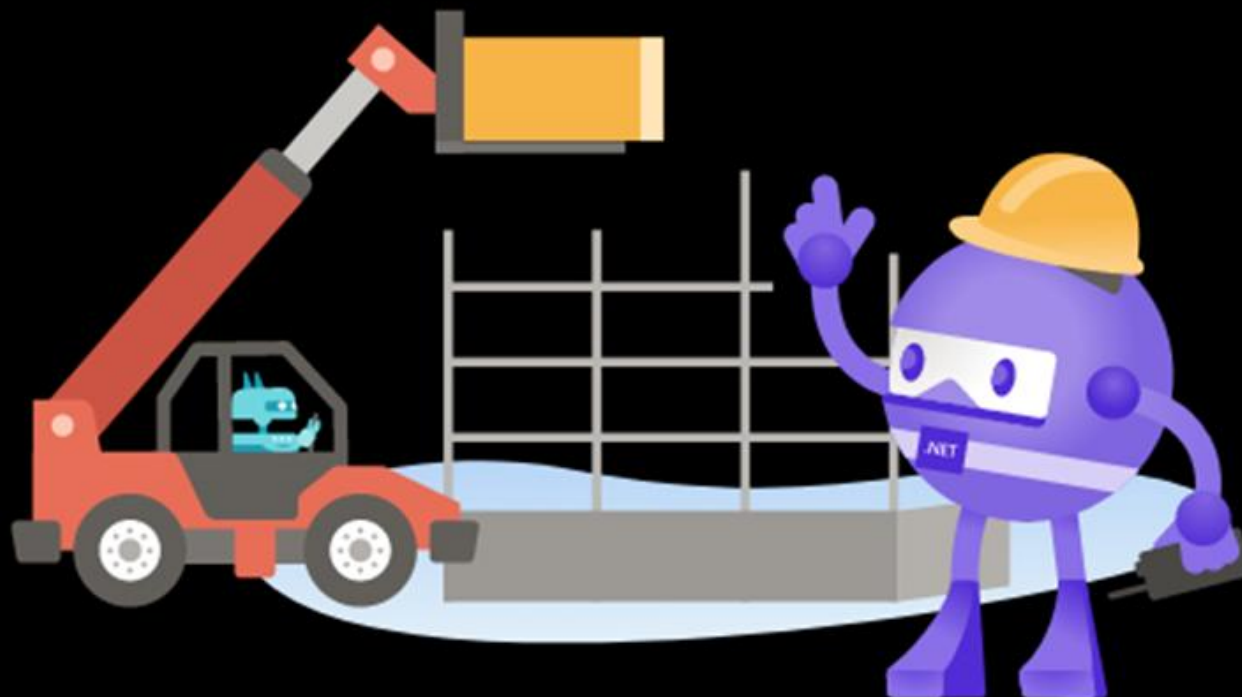
Partendo dal contesto `#file:rest-api.prompt.md`
implementa una solution .NET che ne rispetti i
requisiti nella cartella src.



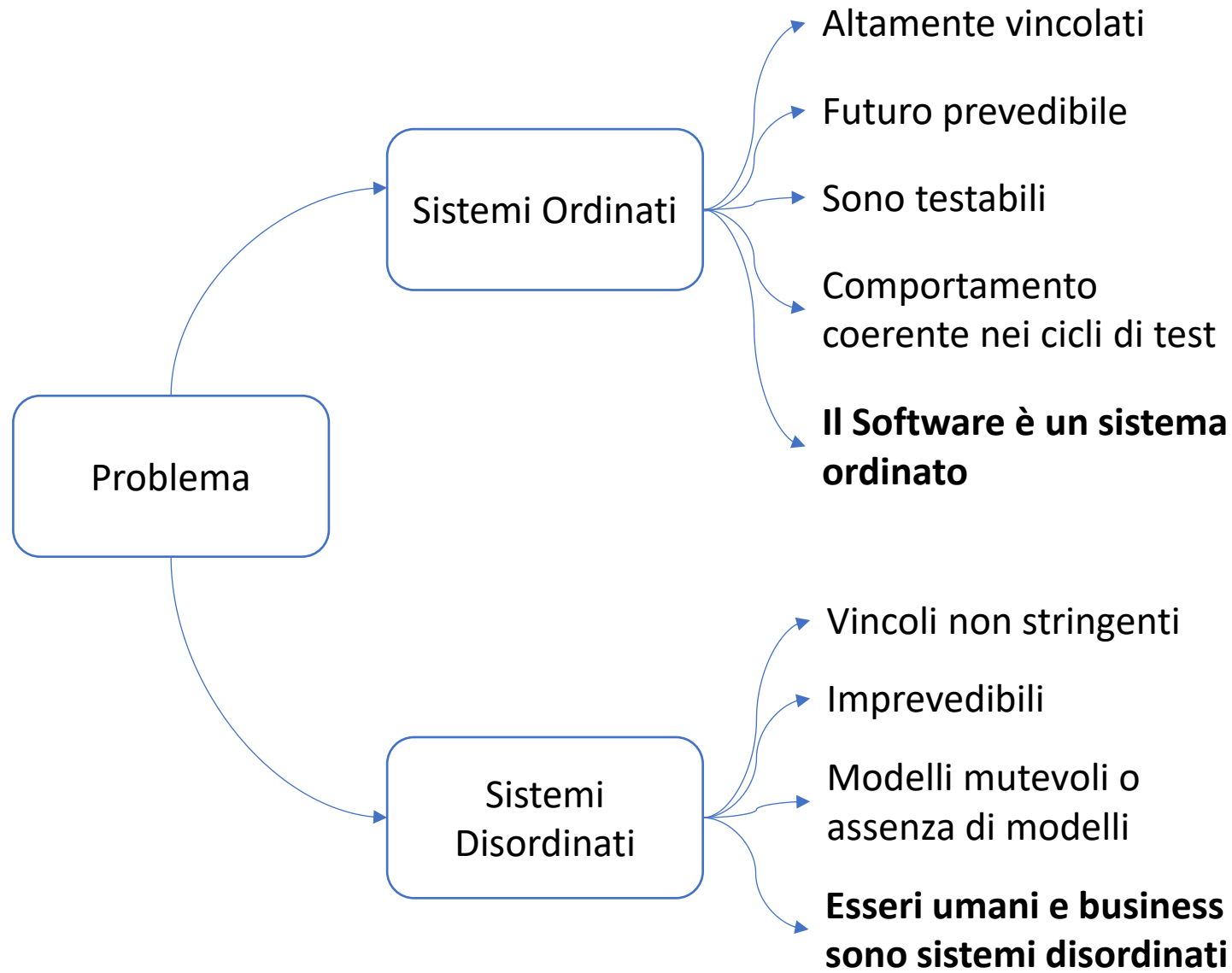
Funziona?



WEB API




FITNESS TEST USING NETARCHTEST





Panta Rei

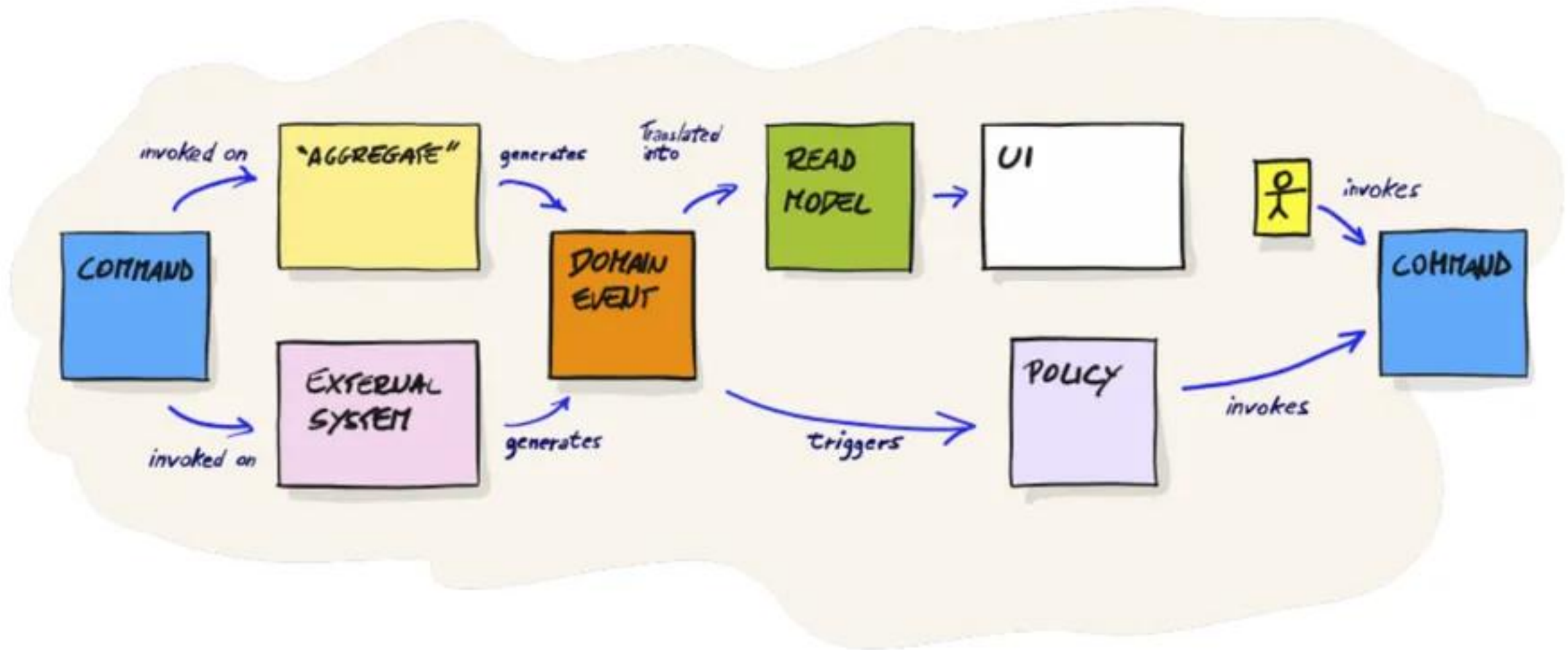


A schizophrenic out for a walk is
a better model than a neurotic
lying on the analyst's couch.

Deleuze Guattari

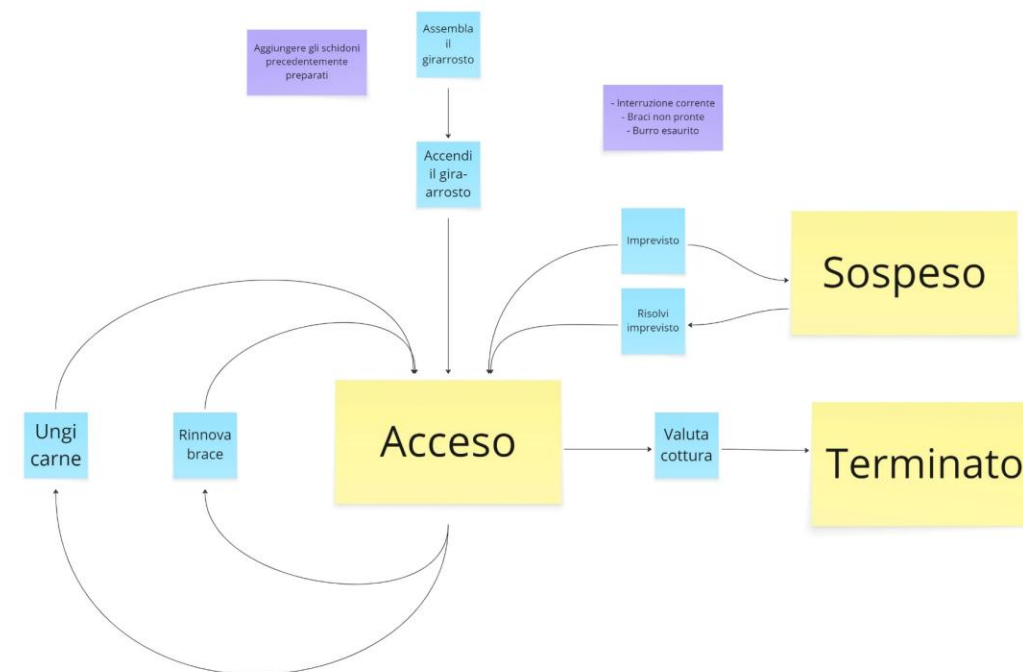
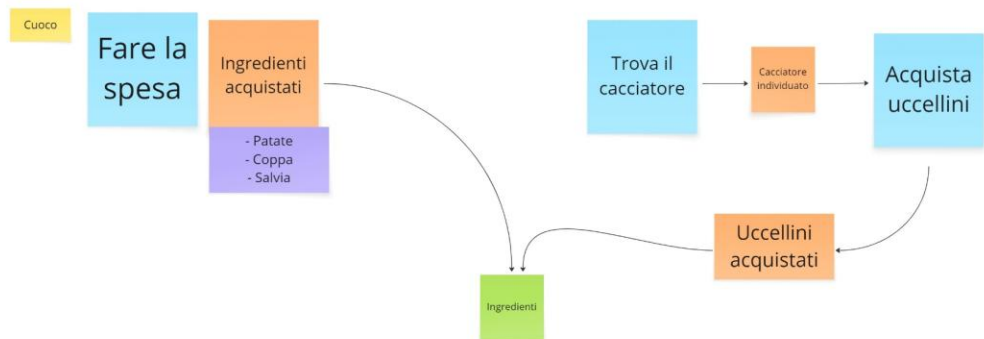


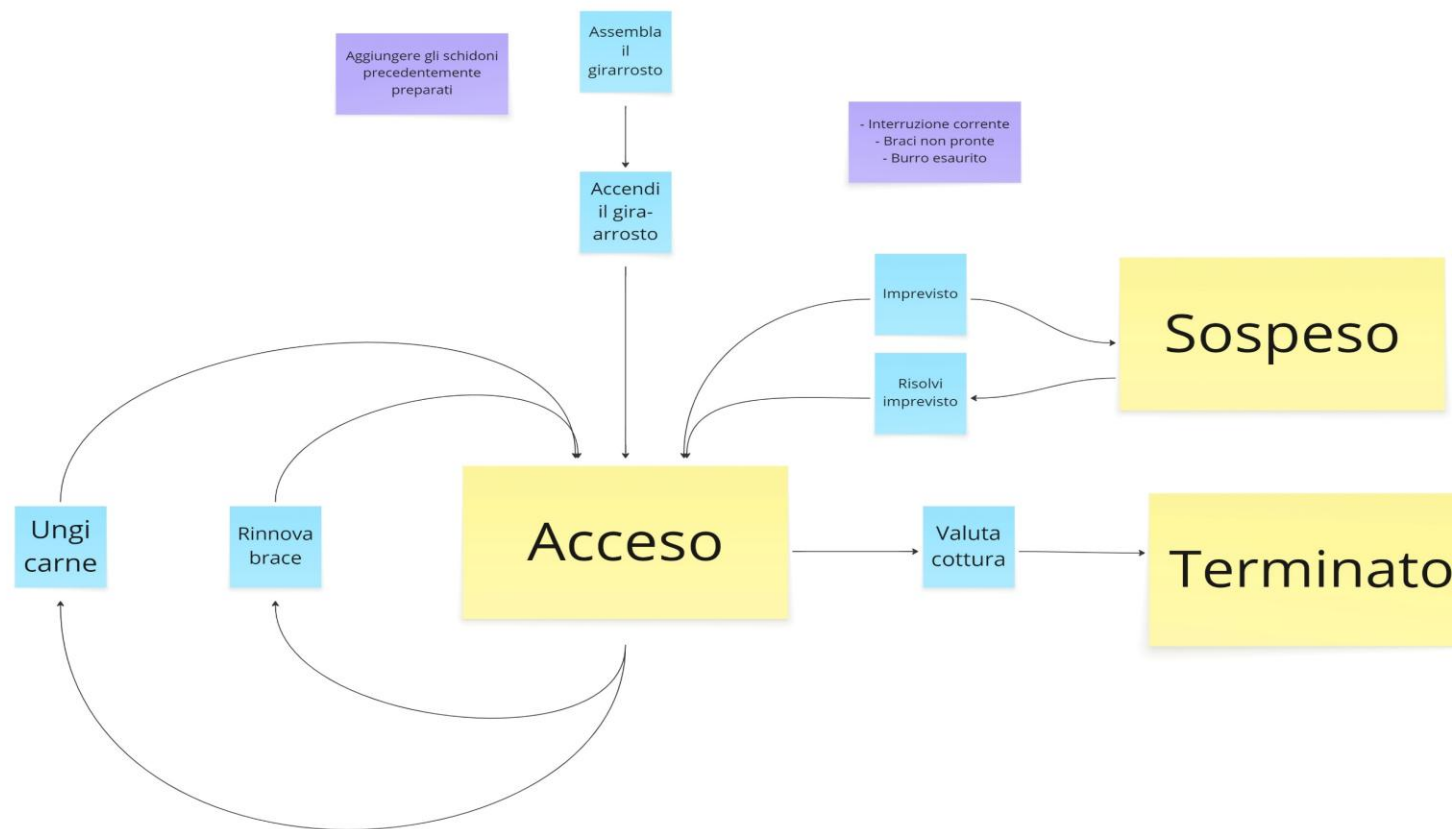
Esploriamo il Dominio



[EventStorming](#)

[David Hume](#)







Facciamo lavorare Copilot

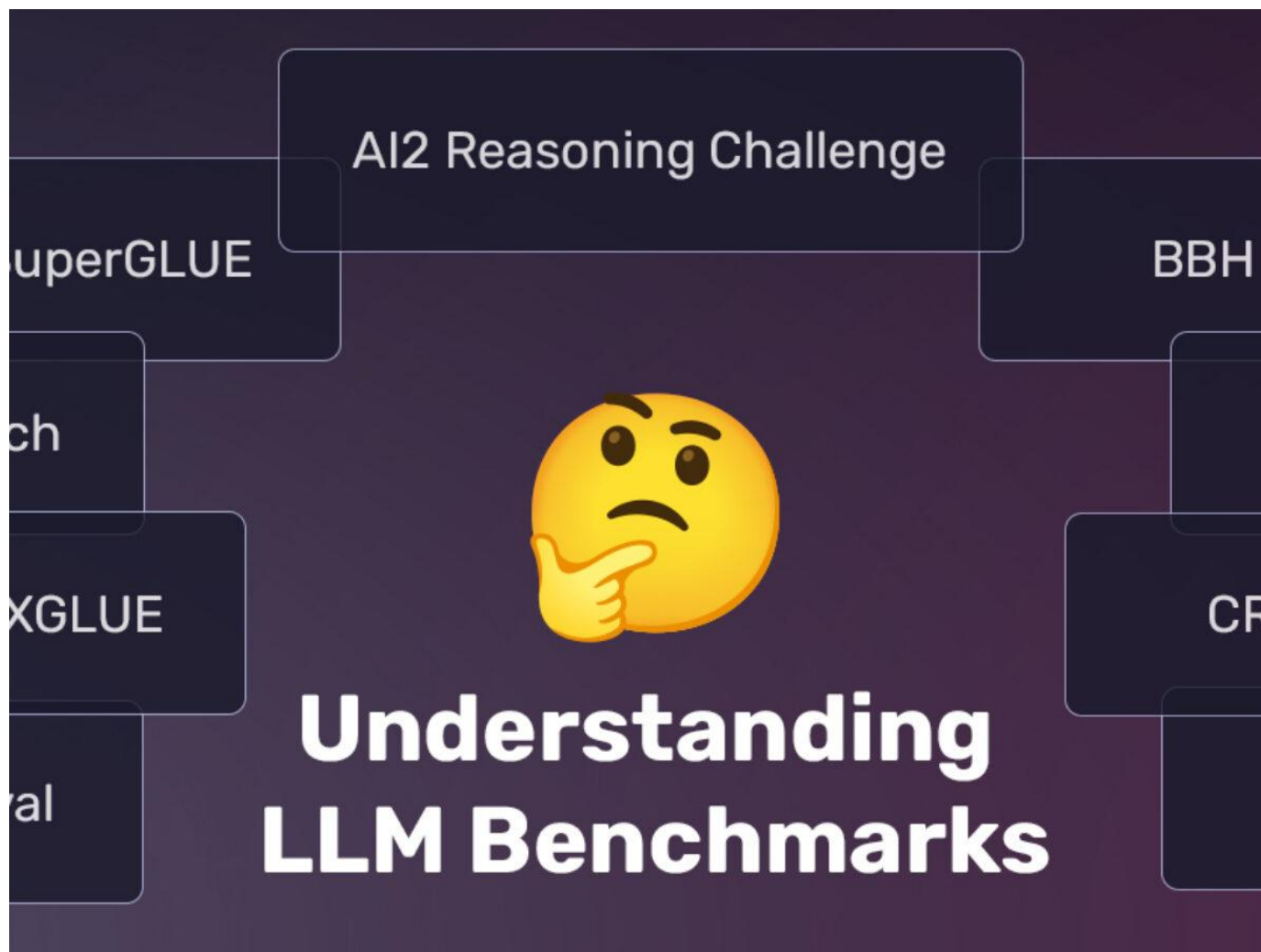
Utilizzando le istruzioni del file `#file:eventstorming-softwaredesign.prompt.md` e l'immagine seguente, prova ad implementare la parte di dominio.

Utilizza la libreria Muflone per la parte di CQRS-ES. Le istruzioni per farlo le trovi in `#file:muflone-core.prompt.md`. Il dominio da Implementare è quello descritto in `#file:trattoria-module.prompt.md`



Ci sono almeno due tipi di giochi:

- I giochi finiti
 - e quelli infiniti.
-
- Un gioco finito si gioca per vincerlo.
 - Un gioco infinito per continuare a giocare.
 - I partecipanti a un gioco finito giocano entro certi confini ben precisi.
 - I partecipanti a un gioco infinito giocano con i confini.



Codility's Multi-dimensional Programming ASsessment

- Correttezza.
- Efficienza.
- Qualità.

[COMPASS: A Multi-Dimensional Benchmark for Evaluating Code Generation in Large Language Models](#)



Alberto Acerbis



alberto.acerbis@intre.it



<https://github.com/Ace68/1nn0vaAI-2025>



<https://github.com/BrewUp/CrastuArrustutu>

