

MindLink

фаза II – Модел података и перзистенције

Чланови тима:

Душан Марковић, 19212
Јован Станишић, 19369

Назив тима:

OG10D1

Датум: 11.2.2026.

1. Увод

Фаза 2 пројекта MindLink била је фокусирана на имплементацију читавог Data Access Layer-a, где је циљ био да се имплементирају Business модели, Entity модели и Repository pattern са AutoMapper интеграцијом.

2. Business Model

Главна класа нашег Business модела јесте GameSession и она представља једну партију која се тренутно игра. Као таква, она има своје име, и у себи чува све оно што је потребно за одигравање партије.

То подразумева да у себи садржи референцу на Board која се састоји од поља карата (у класи представљеног као низ карата) која означава једно игриво поље које има своју позицију на том пољу, реч коју представља као и то да ли припада неком од тимова, да ли је неутрално или је ипак поље које представља крај партије ако буде откривено.

Такође у истој класи се чувају и тимови, од којих сваки има своју боју и назив и у себи садржи играче, који пак у себи чувају своја имена.

Игра такође има и додатне податке о томе када је почела и када се завршила уколико јесте и наравно статус који одређује у ком се стању налази игра у бази.

Од могућих потеза које играчи могу начинити, у зависности од својих улога, имамо класе Guess, која садржи референцу на играча који је то одиграо, као и то коју карту је изабрао и који је то покушај по реду, и Hint која такође садржи референцу на играча, а чува и то која реч или краћа синтагма, представља вербалну смерницу и број речи на које се та смерница односи.

3. Entity Model

Све класе које су наведене изнад, имају скоро идентичну репрезентацију у односу на прошли модел што се тиче информације које носе у себи. Додата је апстрактна класа BaseEntity коју имплементирају све ентитетске класе и која у себи садржи поље за јединствени идентификатор који служи свим класама као примарни кључ у табелама за те класе као и то када су ти објекти креирани у бази и када је последњи пут дошло до њихове измене.

Сви ентитети се складиште у бази података у посебним табелама и то на начин да сваки ентитет има своју табелу, осим ентитета који представља карте које се играју у некој партији, које се чувају у json документу као вредност једне колоне у табели boards. Таква одлука је донета из разлога што те речи једино постоје у оквиру те партије и тог конкретног поља на коме се игра и не доводе се у вези ни са једним другим ентитетом, а такође, речи које се бирају и које се налазе на картицама, се добијају из одређеног речника па је непотребно чувати их све посебно у бази.

4. Repository Pattern

Repository Pattern је архитектурски шаблон који представља апстракцију за приступ бази података. Свака класа има своју репозиторијумски класу које садрже методе за основне CRUD операције, али и неке специјализоване упите, као што је упит за учитавање партије са свим њеним везаним ентитетима.

Сви репозиторијуми наслеђују генеричку базну класу RepositoryBase која имплементира све те основне операције које не зависе од класе, а такође и имплементирају посебан репозиторијумски интерфејс који садржи све те специјализоване упите који зависе од класе до класе. Оваква имплементација овог шаблона омогућава лакше тестирање и јаснију поделу одговорности, а повећава прегледност самог кода.

5. Мапирање (AutoMapper)

AutoMapper је класа која аутоматски конвертује из једног типа модела у други и обрнуто, и за сваки од модела има посебна правила за мапирање, као што јесу на пример правила за мапирање ентитет вредности или мапирање у json објекте када је реч о објектима који представљају картице. Осигурано је и то да се избегну циклуси који могу настати код мапирања, па се поједине ствари игноришу или одлажу.