Mariusz Wieczorek

mariusz.wieczorek@kabat.pl

Streszczenie

Tworzymy serwisy

asp.net Core

Tworzenie Serwisów

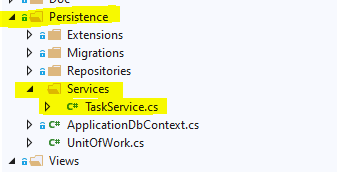
**Tworzymy Modele**

Nie ma obecnie w naszym kontrolerze jakieś zaawansowanej logiki biznesowej.  
Na ten moment kontroler jest napisany jak najbardziej w porządku, co jednak, gdy będziemy musieli umieścić jakąś dodatkową logikę. Nie umieszczamy jej w kontrolerze, ponieważ kontrolery powinny być chude, czyli powinny zawierać jakąś walidację, wywołanie jakiegoś modelu, i zwracanie widoku z przekazaniem do niego ViewModelu, albo przekierowania do innej akcji.

W UnitOfWork może być logika odpowiedzialna za operację na bazie danych, nic więcej.

Dodatkową warstwę biznesową powinniśmy umieścić w osobnej warstwie np. jakiś serwisach.  
Nazwa Serwis jest umowna, mogą to być równie dobrze komendy, managery itp. w kazgej firmie wygląda to inaczej.

Tworzymy w folderze Persistence folder Services, a w nim klasę TaskService.



|  |
| --- |
| namespace MyTasks.Persistence.Services  {  public class TaskService  {  private readonly UnitOfWork \_unitOfWork;  public TaskService(UnitOfWork unitOfWork)  {  \_unitOfWork = unitOfWork;  }  }  } |

Kopiujemy do niego sygnatury metod z repozytoriów i przerabiamy jak poniżej.

|  |
| --- |
| public Task Get(int id, string userId)  {  return \_unitOfWork.Task.Get(id, userId);  }  public void Add(Task task)  {  \_unitOfWork.Task.Add(task);  \_unitOfWork.Complete();  } |

Następnie ponownie przerabiamy kontroler

|  |
| --- |
| namespace MyTasks.Controllers  {  [Authorize]  public class TaskController : Controller  {  private readonly UnitOfWork \_unitOfWork;  private readonly TaskService \_taskService;  public TaskController(ApplicationDbContext context)  {  \_unitOfWork = new UnitOfWork(context);  \_taskService = new TaskService(new UnitOfWork(context));  } |

|  |
| --- |
| [HttpPost]  public IActionResult Delete(int id)  {  try  {  var userId = User.GetUserId();  \_taskService.Delete(id, userId);  }  catch (Exception ex)  {  // logowanie do pliku  return Json(new { success = false, message = ex.Message });  }  return Json(new { success = true });  } |