```
pkg VMSApi.Test

pkg VMSApi.Test.TestingUtils

PostTaskItemModelBuilder
- _item: PostTaskItemModel
+ «create» PostTaskItemModelBuilder ()
+ WithDueDate (dueDate : DateTime): PostTaskItemModelBuilder
+ WithStartDate (startDate : DateTime): PostTaskItemModelBuilder
+ WithTaskTitle (title : string): PostTaskItemModelBuilder
+ WithDescription (description : string): PostTaskItemModelBuilder
+ WithRepeatingIntervalDays (repeatingIntervalDays : int): PostTaskItemModelBuilder
+ WithCompleted (completed : bool): PostTaskItemModelBuilder
+ Build (): PostTaskItemModel

TaskItemBuilder
- _item: TaskItem
+ «create» TaskItemBuilder ()
+ WithId (id : Guid): TaskItemBuilder
+ WithDueDate (dueDate : DateTime): TaskItemBuilder
+ WithStartDate (startDate : DateTime): TaskItemBuilder
+ WithTaskTitle (title : string): TaskItemBuilder
+ WithDescription (description : string): TaskItemBuilder
+ WithRepeatingIntervalDays (repeatingIntervalDays : int):
TaskItemBuilder
+ WithCompleted (completed : bool): TaskItemBuilder
+ Build (): TaskItem

pkg VMSApi.Test.StubClasses

FakeSignInManager
+ «create» FakeSignInManager ()

FakeUserManager
+ «create» FakeUserManager ()

pkg VMSApi.Test.Controllers

TaskItemControllerTest
- _repoMock: Mock<ITaskItemRepository>
- _mapperMock: Mock<IMapper>
- _sut: TaskItemsController
+ SetUp ()
{Annotation = SetUp}
+ «async» Should_Get_All_TaskItems (): Task
+ «async» GetByIdAsync_Should_Return_TaskItem_By_Id_If_Exists (): Task
+ «async» GetByIdAsync_Should_Return_NotFound_When_No_TaskItem_Is_Found (): Task
+ «async» PostAsync_Should_Return_OkObjectResult_When_Adding_TaskItem_To_Database_Is_Successful (): Task
+ «async» PostAsync_Should_Return_BadRequest_Result_When_ModelState_Is_Invalid (): Task
+ «async» PostAsync_Should_Return_BadRequest_When_Exception_Thrown (): Task
+ «async» PutAsync_Should_Return_BadRequest_When_DbUpdateConcurrencyException_Thrown (): Task
+ «async» PutAsync_Should_Return_NotFound_When_IdException_Thrown (): Task
+ «async» PutAsync_Should_Return_BadRequest_Result_When_ModelState_Is_Invalid (): Task
+ «async» PutAsync_Should_Return_OkObjectResult_When_TaskItem_Correctly_Updated (): Task
+ «async» DeleteAsync_Should_Return_OkObjectResult_When_TaskItem_Successfully_Deleted (): Task
+ «async» DeleteAsync_Should_Return_NotFoundObjectResult_When_No_TaskItem_With_Given_Id_Has_Been_Found (): Task
+ «async» DeleteAsync_Should_Return_BadRequestResult_When_Exception_Has_Been_Thrown (): Task
+ «async» AssignUserToTaskPostAsync_Should_Return_BadRequestResult_When_ModelState_Is_Invalid (): Task
+ «async» AssignUserToTaskPostAsync_Should_Return_BadRequestResult_When_Route_TaskId_And_TaskId_From_Body_Do_Not_Match (): Task
+ «async» AssignUserToTaskPostAsync_Should_Return_NotFoundObjectResult_When_Task_Not_Found (): Task
+ «async» AssignUserToTaskPostAsync_Should_Return_NotFoundObjectResult_When_User_Not_Found (): Task
+ «async» AssignUserToTaskPostAsync_Should_Return_BadRequestResult_When_User_Already_Assigned_To_Task (): Task
+ «async» AssignUserToTaskPostAsync_Should_Return_BadRequest_When_Exception_Thrown (): Task
+ «async» AssignUserToTaskPostAsync_Should_Return_NoContent_When_User_Successful_Assigned_User_To_Task (): Task
{Annotation = Test}
- CreateDefaultListOfTaskItems (): IList<TaskItem>
- CreateDefaultPostTaskItemModel (): PostTaskItemModel
- CreateDefaultTaskItem (): TaskItem

FoodPurchaseControllerTest
- _mappingsMock: Mock<IMappings>
- _repoMock: Mock<IFoodPurchasesRepo>
- _sut: FoodPurchaseController
+ SetUp ()
{Annotation = SetUp}
+ «async» GetAll_Request_Returns_BadRequest_When_Exception_Is_Thrown (): Task
+ «async» GetAll_Returns_List_Of_FoodPurchases (): Task
+ «async» GetByIdAsync_Should_Return_FoodPurchase_By_Id_If_Exists (): Task
+ «async» GetByIdAsync_Should_Return_NotFound_When_No_TaskItem_Is_Found (): Task
+ «async» GetByIdAsync_Should_Return_BadRequest_When_Exception_Is_Thrown (): Task
+ «async» CreateAsync_Should_Return_OkObjectResult_When_Adding_FoodPurchase_To_Database
_Is_Successful (): Task
+ «async» CreateAsync_Should_Return_BadRequest_Result_When_ModelState_Is_Invalid (): Task
+ «async» CreateAsync_Should_Return_BadRequest_When_Exception_Thrown (): Task
+ «async» Update_Should_Return_BadRequest_When_Exception_Is_Thrown (): Task
+ «async» Update_Should_Return_NotFound_When_IdException_Thrown (): Task
+ «async» Update_Should_Return_BadRequest_Result_When_ModelState_Is_Invalid (): Task
+ «async» Update_Should_Return_OkObjectResult_When_FoodPurchase_Correctly_Updated (): Task
- GetDefaultFoodList (): List<FoodPurchase>
- GetDefaultPostModel (): PostMapperFoodPurchase

MeasurePointControllerTests
- _sut: MeasurePointController
- _mappingsMock: Mock<IMappings>
- _repoMock: Mock<IMeasurePointRepo>
+ SetUp ()
{Annotation = SetUp}
+ «async» GetAll_Request_Returns_BadRequest_When_Exception_Is_Thrown (): Task
+ «async» GetAll_Returns_OKResult_With_List_Of_MeasurePoints_When_No_Exception (): Task
+ «async» GetById_Should_Return_MeasurePoint_By_Id_If_Exists (): Task
+ «async» GetById_Should_Return_NotFound_When_No_MeasurePoint_Is_Found (): Task
+ «async» GetById_Should_Return_BadRequest_When_Exception_Is_Thrown_While_Searching_For_MeasurePoint_With_Id ():
Task
+ «async» Create_Should_Return_OkObjectResult_When_Adding_New_MeasurePoint_To_Database_Is_Successful (): Task
+ «async» Create_Should_Return_BadRequest_Result_When_ModelState_Is_Invalid_Of_PostMapperMeasurePoint (): Task
+ «async» Create_Should_Return_BadRequest_When_Exception_Thrown_While_Adding_New_MeasurePoint (): Task
+ «async» Update_Should_Return_BadRequest_When_Exception_Is_Thrown_While_Updating_MeasurePoint (): Task
+ «async» Update_Should_Return_NotFound_When_IdException_Thrown_While_Updating_MeasurePoint (): Task
+ «async» Update_Should_Return_BadRequest_Result_When_ModelState_Is_Invalid_While_Updating_MeasurePoint (): Task
+ «async» Update_Should_Return_OkObjectResult_When_MeasurePoint_Correctly_Updated (): Task
- CreateDefaultMeasurePoint (id : Guid = new Guid()): MeasurePoint
- CreateDefaultListOfMeasurePoints (): List<MeasurePoint>
- CreateDefaultModel (): PostMapperMeasurePoint

AccountsControllerTests
- _mapperMock: Mock<IMapper>
- _userManagerMock: Mock<FakeUserManager>
- _signInManagerMock: Mock<FakeSignInManager>
- _tokenGeneratorMock: Mock<ITokenGenerator>
- _sut: AccountsController
+ SetUp ()
{Annotation = SetUp}
+ «async» Should_Return__BadRequest_When__RegisterModel_Not_Valid (): Task
+ «async» Should_Not_Register_When_Model_Is_Valid (): Task
+ «async» Should_Login_When_Credentials_Correct (): Task
+ «async» Should_Return_BadRequest_When_Credentials_Incorrect (): Task
+ «async» Should_Delete_User_When_UserID_Is_Found (): Task
+ «async» Should_Return_404_NotFound_When_No_User_With_Given_Id_Was_Found (): Task
+ «async» Should_Return_400_BadRequest_When_Something_In_Method_Throws_An_Exception_When
_Deleting_User (): Task
+ Deleting_User_Should_Succeed_When_Valid_Authorized_Role (role : string)
+ Deleting_User_Should_Fail_When_Valid_Authorized_Role (role : string)
+ AssignRolesToUser_Should_Succeed_When_Administrator_Calls_Action ()
{Annotation = Test}
+ AssignRolesToUser_Should_Fail_When_Not_An_Administrator_Calls_Action ()
+ «async» AssignRolesToUser_Should_Return_BadRequestObjectResult_When_ModelState_Invalid (): Task
+ «async» AssignRolesToUser_Should_Return_NotFoundObjectResult_When_No_Matching_User_wasFound
(): Task
+ «async» GetAll_Should_Return_OkResult_With_All_Users (): Task
+ «async» GetAll_Should_Return_BadRequestResult_When_Exception_Has_Been_Thrown (): Task
{Annotation = Test}
- SetUpUserMangerMock ()
- GetMockRegistrationModel (): UserRegistrationModel
- GetMockUser (): User
```