**Web API Cheat Sheet**

1. **New -> ASP.NET Web Application -> Web API with Individual User Accounts + Unit Tests**
2. **App\_Start -> IdentityConfig -> PasswordValidator => false everywhere**
3. **Registering: api/Account/Register POST**- No content type  
   - Headers -> x-www-form-urlencoded =>Email, Password, ConfirmPassword
4. **Login: /Token POST (configurable from App\_Start -> StartupAuth -> OAuthOptions**- x-www-form-urlencoded => grant\_type = password; username (email); password  
   - response contains access\_token
5. **[Authorize]** attribute specifies authorization requirement for the current controller OR method
6. **Authentication:** Headers -> Authorization = Bearer + access\_token
7. **Models** class library
8. **Data** class library
9. **Manage NuGet Packages =>   
   EntityFramework + Microsoft ASP.NET Identity EntityFramework**
10. **Move** **ApplicationUser** from Web API IdentityModels to Models
11. **Move ApplicationDbContext** from Web API IdentityModels to Data
12. **Add references** in Web API to Data and Models, and include required namespaces, so that the solution can successfully build
13. **Enable migrations** for Data
14. **Modify connection string** in Web.config
15. **Build models**
16. **Repository pattern**
17. **Unit of work**
18. **Controllers creation**
19. **Install Ninject.Web.WebApi.OwinHost**
20. **In Startup.cs:**using Ninject;  
    using Ninject.Web.Common.OwinHost;  
    using Ninject.Web.WebApi.OwinHost;  
      
    public void Configuration(IAppBuilder app)

{

app

.UseNinjectMiddleware(CreateKernel)

.UseNinjectWebApi(GlobalConfiguration.Configuration);

}

private static StandardKernel CreateKernel()

{

var kernel = new StandardKernel();

kernel.Load(Assembly.GetExecutingAssembly());

RegisterMappings(kernel);

return kernel;

}  
  
private static RegisterMappings(StandardKernel kernel)  
{

kernel

.Bind<Interface>

.To<Class>()

.WithConstructorArgument(“context”, c => new BlablaDbContext());

}

1. **!!!** **routeTemplate** => api/{controller}/{action}/{id}, if required by the endpoints specifications
2. **DataModels** in the Web project
3. if(request == null || this.ModelState.IsValid)  
   {  
    return this.BadRequest();   
   }
4. using Microsoft.AspNet.Identity;  
   this.User.Identity.GetUserId();
5. Thread.CurrentPricipal ~= this.User; for Unit Testing