PW

- MVC = model View controller

- **model** - označuje množinu tříd, obsahuje logiku aplikace, udržuje stav systému, obsahuje data pro view, o vytvoření se starají controllery

- **view** = to, s čím uživatel interaguje

- **controller** = přijímá vstup od uživatele, manipuluje se modelem, volí vhodné View
- více View může používat stejný controller
- akční metody (výchozí = GET)

- **Areas** = menší funkční jednotek sdružené do 1 modulu. organizují velké projekty do malých celků
- [Area(“products”)]
- public void HomeController { … }

Razor

@Datetime.now
escapnutí = @@a.b

explicitní zápis = celý výrraz v závorkách
@ {var answer = 42}
@answer , sdílí

Sdílení dat
return View(new ProductViewModel())
ViewData[“Name”] = new p();
ViewBag.Name = “filip”; // nevyžaduje přetypování

Helpers
@Html.raw(Model.carousels[i].CarouselHTMLContent)
@Html.Label(“First”, new), DisplayNameFor, displayFor

Entita

[Table(“Product”)]
public class Product {
[Key][Required][Column(“ID”)]
public int ID {get; set;}

}

key = primary key
required = not null
table v SQL

závislá a prim. entita, cizí a závislý klíč
migrace dat = Migrate-all

Liskov Substition

3D deska dědí z 2D desky (ThreeDBoard: 2DBoard)

Interface segregation principle
(třída by neměla být nucena implementovat část rozhraní, které nepotřebuje)

DIP (Dependency inversion principle)
High level modules by neměly být závislé na Low level modules
služby by měla třída dostat zvenčí

IoC = Inversion of control
řeší jak docílit DIP, higl level závislé na abstrakci low level

AJAX

XMLHttpRequest XHR, dotaz na server bez opětovného načtení stránky
XML => Internet => Server => Browser
metody: open, send, abort
události: abort, error, load, onOpenStateChange, Progress, timeout

Promise = objekt reprezentující dokončení/selhání operace

Client vs server side
Je nutné server side!, ohrožení dat na serveru

Development (lokální prostredí pro vývoj)
Staging (pred-produkční verze pro testování na soukromých serverech)
Production (produkeni web pouzivany klienty, uzivateli apod.)
env.isStaging

Vlastní Exception Page

- app.UseExceptionHandler("/Home/Error")
- Z HttpContext lze vycist TraceIdentifier,
- property ExceptionHandlerPathFeature.Error/Path
- V základu neposkytuje aplikace odpovèd' pro Status kody 400-599
- Lze definovat pomoci app.UseStatusCodePages ()

```
if (env. IsDevelopment () ){
        app. UseDeveloperExceptionPage()
} else {
        app.UseExceptionHandler(“/Home/Error!
}
```

HttpContext
když je nutné přistoupit k requestu, response, Identity, session, atd.
Controllery a Views primo obsahuji instanci HttpContext
HttpContext v Controlleru
Context ve View

Úrovně logů

trace - trasovat a logovat konr. f
debug - vývojáře, adminy
info - obecná informace
warning - zvl. chování
error - nedokončení operace, není kritické pro aplikaci
critical - krit. pro apl.

Autentizace = ověření uživatele
Autorizace = co má uživatel povoleno vykonávat

Vlastní definice třídy User musí dědit od IdentityUser<int>
Vlastní definice třídy Role musi dědit od IdentitvRole<int>

UserManager<TUser>
SignInManager<TUser>

DDL

zvýšení znovupoužitelnosti kódu
1) **Client Applications** - SPA (Single Page Application)
2) **Presentation Layer** - interface pro uživatele (ASP.NET MVC)
3) **Distributed Service Layer** - API controllery, Data controllery
4) **Application Layer** - aplikační sluzby (services). DTOs
5) **Domain Laye**r - entity, doménové sluzby, rozhraní pro repozitare,..
6) **Infrastructure Layer** - rozhraní repozitářů, ORM (Entity Framework)

Repository pattern = abstrakce strategie pro uchování dat
využívá dependency injection
při požití s ORM (EF Core = antipatern)

Unit of Work
Aplikování změn entit by nemělo být v rep.
umožňuje sloučení všech transakci v rámci Repository do jedné transakce

Modifikátory přístupu

public
private - jen ve stejné třídě
protected - stejná/odvození
internal - jen uvnitř assembly
private protected: je přístupný jen uvnitr assembly, kde je deklarován

protected internal: je pristupny kdekoliv uvnitr assembly, kde je deklarován, ale také v odvozené tridè deklarované v jiné assembly.

Testing

Použijí se jak platné, tak neplatné vstupy a testuje se výstup/odpověď z objektu

xUnit, Moq, Selenium
vlastnosti testu: automatizovaný, opakovatelný, jednoduše implementovatelný

1) **integrační testy**: 2 nebo více dohromady
např. přidání předmětu do košíku
2) **systémové** - celek z pohledu zákazníka správně?, test všech fcí
3) **akceptační testy** - u zákazníka

Podezřelý kód = mnoho členských prom.
dlouhý seznam parametrů
cizí data
shluky dat - různé části syst. obsh. skupinu stejných prom., spíše malé objekty, než mnoho členských prom.

Cloud

výhody,
nižší náklady
rychlost, výkon,
spolehlivost, zabez.

nevýhody - možnost úniku
náročná změna poskytovale
AlwaysOn Availability SQL Server (replikace dat)
primární server přepne v řádu sekund na sekundární
health check
load balancer
Autoscaler
Google Cloud, Azure